Innovations to fundamental stock valuations: Estimating future earnings per share and free cash flows using statistical and machine learning methods

Ivan Evdokimov

A thesis submitted for the degree of

Doctor of Philosophy

at the

School of Computer Science and Electronic Engineering

University of Essex

May 2025

Abstract

This study proposes innovations to financial valuation models. Fundamental valuation is used by investors to make buy/sell decisions regarding stock issues. Valuation is the process of determining the intrinsic value — the price reasonable to pay for a stock given its future prospects, which are measured with cash flows, given the level of risk an investor bears when buying stock. These cash flows are measured with two key series: Earnings-Per-Share (EPS) and Free Cash Flows (FCF). Correspondingly, the two series are used as inputs in common valuation models: the Forward Price-Earnings and Discounted Cash Flows. It is required that investors estimate the future cash flows — a sensitive process whereby under- or overstating future cash flows is prone to the risk of losing invested equity. Hence, being able to accurately capture the next quarter's value is of utmost importance for investors active in financial markets: it guides the stock selection process.

We propose to formulate this as a regression problem, where the target variable is the next quarter's Earnings-Per-Share or Free Cash Flow value. The input features are their respective lags. The main challenge in this problem is the fact that the series are sparse and limited in the number of observations. This is because the fundamental financial data is published every quarter of the year, as required by law. Hence, our estimators have limited training/validation data to learn from. We approach this problem with the selection of 8 Machine Learning (ML)

and 5 Statistical (SE) estimators, conducting experiments on a representative sample of 100 U.S. publicly traded companies.

Our study contributes in several ways. First, we show that the quantile transformer and the PCHIP interpolation improves model generalization by making more blatant the linear relationship of the target variable with its features and artificially increasing the number of data observations, respectively. Second, we demonstrate that while certain ML estimators do overfit to the small data sets, others perform at the same or better rate than the statistical estimators. Third, building on top of our observations about data patterns and behaviour of a diverse range of estimators, we propose the transfer learning methodology that allows to combine the predictive capabilities of the ML and SE. We demonstrate the effectiveness of our approach based on the reduction across the range of regression error measures, and the improvement in portfolio performance, over a fixed backtesting simulation period.

Acknowledgements

Thanks to my supervisor, Michael Kampouridis, and to Tasos Papastylianou. Special thanks to my wife, parents, and friends. Last but not least, thanks to the UK Data Archive, and the University of Essex.

Contents

List of Figures

List of Tables

LIST OF ACRONYMS

AIC – Akaike Information Criterion

ARD – Automatic Relevance Determination

ARIMA – Auto-Regressive Integrated Moving Average

BR – Bayesian Ridge

BWA – Bayesian Weighted Average

DCF – Discounted Cash Flows

DDM – Dividend Discount Model

DT – Decision Tree

EPS – Earnings Per Share

EV – Enterprise Value

FCF – Free Cash Flows

HR – Huber Regressor

KNN – K-Nearest Neighbour

LASSO – Least Absolute Shrinkage And Selection Operator

MAE – Mean Absolute Error

MAPE – Mean Absolute Percentage Error

ML – Machine Learning

MLP – Multi-Layer Perceptron

OLS – Ordinary Least Square

PE – Price-Earnings Ratio

PS – Price-Sales Ratio

PCHIP – Piecewise Cubic Hermite Interpolation

RF – Random Forest

RLM – Robust Linear Model

ROE – Return On Equity

SE – Statistical Estimator

SES – Simple Exponential Smoothing

sMAPE – Symmetrical Mean Absolute Percentage Error

SPS – Sales Per Share

TL – Transfer Learning

WACC – Weighted Average Cost Of Capital

WLS – Weighted Least Squares

List of Publications

Published works

- Ivan Evdokimov, Michael Kampouridis, Tasos Papastylianou. "Application Of Machine Learning Algorithms to Free Cash Flows Growth Rate Estimation." Procedia Computer Science, Volume 222, 2023, 529-538

Works Under Review

- Ivan Evdokimov, Michael Kampouridis, Tasos Papastylianou. "Deriving fundamental stock value using transfer learning and earnings-per-share." 2025

Chapter 1

Introduction

Financial data tend to be modeled as a random walk process, with a constant mean and variance over the observed period. This is usually the case in both pricing and fundamental data. However, in reality, the 'random walk' assumption is often violated, as financial data tend to have varying drift. With increasing interest in applications of machine learning algorithms in the area of finance, most researchers tend to focus on pricing data [1], that is, the market price of a financial security (e.g., common stock, bond), due to the open-source availability and abundance of such data: 252 data points are created every year (assuming daily frequency). Fundamental data, such as earnings per share and free cash flows, usually used by creditors, auditors, suppliers, and investors to perform their analyses, on the other hand, is in short supply, as companies only disclose such figures on a quarterly basis, resulting in four data points per annum.

Free cash flow (FCF) is a measure of cash available to shareholders after all necessary expenses and investments have been made. Earnings per share (EPS) is a measure of net profit made by a company at the end of the accounting period. The key distinction between

the two series is that FCF allows the addition of non-cash expenses back. These non-cash expenses include the depreciation of tangible assets such as buildings, manufacturing facilities, machinery, etc. Conversely, EPS reports profits net of all cash and non-cash expenses and other allowances alike.

Fundamental investors aim to predict the FCF and EPS in order to estimate a business's intrinsic value. Compared to the price of a financial security, intrinsic value is a measure of how much a business is worth [2], given its future profits to investors, discounted at an appropriate rate. The two models are widely used to establish intrinsic value: Discounted Free Cash Flows (DCF) and Forward Price-Earnings (Forward PE). Specifically, the DCF model takes the next period's FCF and discount rates as inputs, while the forward PE method multiplies the quotient of the current stock price and EPS by the estimated next period's EPS value. Both economic models output the intrinsic value.

This study aims to model financial time-series data. Specifically, we seek to predict the next quarter's EPS and FCF values (the targets) given their past lags, mean, and standard deviation (feature set). We utilize historical EPS and FCF data from 100 publicly traded U.S. companies. These datasets exhibit distributional shifts, high skewness, and excess kurtosis.

In this part of the work, we specify the motivation and the layout of the study. We start with the motivation.

1.1 Motivation and Contributions

As mentioned previously, the two target variables in this study are used by fundamental investors to determine the intrinsic value of a stock. Compared to the market price, the intrinsic value gives an estimate of a price worth paying for a stock, given future income and

the risk an investor bears when buying into the issue [2]. On the other hand, price is how much one must pay for a stock in order to buy it.

While there is a consensus among investment practitioners and researchers on how to derive the discount rate [3], the best way to approximate next period's EPS and FCF values is still an open question. This is partly due to the fact that the future value at a particular point in time tends to be highly dependent on a company's financial and operating conditions and other external factors, which are generally not constant.

Therefore, our motivation is to propose a state-of-the-art approach to modeling EPS and FCF values. These values could then be used in their respective models: forward PE and DCF, to estimate intrinsic values. Given the number of data augmentation, transformation, and regression approaches used in this study, we seek to outline a methodology that yields the lowest error between the genuine and estimated target data points.

Importantly, under- or over-stating either of the target values leads to financial losses, as it results in unrealistic estimates of intrinsic values. However, we earlier emphasized the fact that these series display distributional shifts and high excess kurtosis, in addition to the main challenge — the lack of data observations.

Over the course of this research, we make several contributions. First, our study compares the performance of machine learning (ML) and statistical estimators (SE) in the sparse time-series context. Further, we investigate how the two approaches perform on datasets augmented with mathematical interpolation and on those transformed to a different space using quantiles. Second, we outline reasons why certain regression methodologies and estimators fit better than others, investigating properties of datasets and their influence on the outcome. Thirdly, we propose a transfer learning methodology that allows combining the predictive performance of a subset of both ML and SE estimators, while transferring the knowledge within the same

data domain: the fundamental financial time-series.

Finally, our main contribution lies in the innovation of established valuation methods commonly used in finance. Specifically, we incorporate advanced quantitative techniques to estimate more precise expectations for next quarter's EPS and FCF values. As a result, our method enables more accurate computation of a stock's fundamental intrinsic value. Our work follows the structure outlined in the next section.

1.2 Structure

This work is structured as follows. We first give a detailed explanation of the concepts used in this study in Chapter 2. There, we outline the mathematics used by the feature transformation and scaling approaches tested, the interpolation methods utilized, and the ML and SE approaches used. Following this is the Literature Review in Chapter 3. This chapter gives an overview of existing methods for forecasting not only FCF and EPS variables but also modeling sparse time-series data from financial and other domains.

Our first contribution is presented in Chapter 4. This chapter gives a technical overview of the statistical properties and outlines the data preprocessing pipeline for both FCF and EPS series, which will be used by the end of this study. Experiments in this chapter are conducted using Ordinary Least Squares Regression on a subset of 50 companies.

Our second contribution is the subject of Chapter 5. There, we investigate how various regression algorithms from both the ML and SE branches perform on sparse time-series data. We also describe why certain estimates were made and how the data affected the decisions made by the model-types. This chapter is the first to use all 100 datasets and the ML estimators.

Our third contribution is presented in Chapter 6, where we propose a transfer learning-

based methodology that allows transferring knowledge from both ML and SE estimators within the same data domain. This chapter makes use of a subset of estimators that meet a specific criterion outlined in the previous Chapter 5.

The final contribution is presented in Chapter 7. Here, we conduct a series of experiments on the out-of-sample simulation (not previously used anywhere in this work), forming series of portfolios based on various estimation methodologies from Chapters 5 and 6. In this chapter, we seek to find the relationship between regression error and the resulting portfolio performance. Specifically, we perform both Forward PE and DCF valuation for all 100 stocks, program buy/sell rules for them, and check how the resulting basket of assets performs against the commonly used market benchmark.

Results and contributions of this work are summarized in Chapter 8. We start with a background review in the next section.

Chapter 2

Background Review

2.1 Introduction

In finance, data can be split into two categories: pricing data and fundamental data. Pricing data is a collection of stock or other security prices that an investor must pay in order to purchase a security [4]. The fundamental data, on the other hand, gives an investor an understanding of how well the management leads the business [5]. For the purposes of this study, the key distinction between the two types of series is the frequency of their publication. Specifically, pricing data is recorded every second of the trading day, while fundamental data is only revealed on a quarterly basis. As such, this difference is one of the reasons the research community pays vast attention to pricing data and less to fundamental data [1]. Given that the organized digital collection and distribution of fundamental stock data began in the 1980s [6], a typical dataset with fundamental financial data is sparse and limited in the number of observations.

Fundamental analysis refers to the process of researching a company's financial data in order to determine whether and how a company increases sales volume and profits, if it

is financed in a way that minimizes investors' risks, and whether its market price is lower than the estimated 'intrinsic' value. Typically, fundamental investors — i.e. those undertaking to perform a fundamental analysis — conduct their due diligence on the basis of financial reports filed by companies every quarter of the year. Specifically, publicly traded companies are obligated by law to submit three '10-Q' and one '10-K' report. The '10-Q' report is known as the 'quarterly statement' and provides all related parties, including investors, customers, suppliers, regulators, etc., with information on what the company owns and owes (Balance Sheet section), how much money the company managed to make from selling products/services (Income/Loss section), and finally, how the company decided to allocate available cash (Cash Flow section). The '10-K' report is the annual report, which, in addition to the information supplied in the '10-Q', also provides a more detailed discussion of the business and other relevant matters (Management Discussion and Analysis section) [7]. This constitutes the essence of fundamental analysis — a crucial step in deciding whether a company is a good investment [8].

In this chapter, we provide background information on the two fundamental financial series and valuation methods used in this study. Financial valuation is a function that approximates the intrinsic value of a stock [8]. The idea of valuation originates from the notion of present value, which is simply the value of a financial security given its expected duration, the level of risk, and the future expected cash flows an investor receives [8]. The present value itself was useful for computing the market price of a bond — the debt security issued by a company and sold to investors [9]. With bonds, intrinsic value is easy to compute: it is the present value of the coupon — the interest payment an investor receives while holding a company debt contract and before the debt is paid off [10].

And while the coupon payment is fixed with bonds in most cases, there is no such fixed

amount paid by companies when it comes to corporate stocks. The exception is the dividend payment which may not be paid by a company. Therefore, whenever an investor aims to purchase stock in a company's shares, she needs to evaluate the future potential cash flows a company generates [2]. She then needs to first, pick the cash flows relevant to her and then forecast the future direction of these cash flows. Under- or over-stating the future cash flows places the investor in financial jeopardy, as it can lead to the over- or under-estimation of the intrinsic value of a stock. The estimation of future cash flows can be formulated as a regression problem, where the target is to predict the next period's cash flow, given its past values — the lags' features' set.

In this study, we consider two types of such cash flows: Earnings-Per-Share (EPS) and Free Cash Flow (FCF). The two series are used in two different valuation methods, respectively: the Price-Earnings (PE) and the Discounted Cash Flow (DCF) methods. The two series and respective valuation methods are described in Section 2.2.

We mentioned that the series we model are sparse and limited in the number of observations. In an attempt to provide regression estimators with more training data, we carry out a series of experiments to select the transformations and interpolation techniques. These methods are described in Section 2.3.

Additionally, we mentioned that the problem of this study is of a regression type, which we attempt to solve with the use of statistical and machine learning estimators. The two branches of regression model types are described in Section 2.4.

When data is in limited supply, quantitative estimators tend to be less reliable and prone to poor generalization. In such cases, an estimator can be trained on similar data or data from the same domain, thereby transferring knowledge about the target and feature relationships to the key series to be predicted. This constitutes the transfer learning approach, which we

outline in Section 2.4.3.

We begin with Section 2.2, which provides a description of the two target variables.

2.2 Financial Series and Valuation Methods

In this section, we provide an explanation of the Earnings-Per-Share (EPS) series and the Price-Earnings (PE) valuation model in Subsection 2.2.1, followed by a similar description of Free Cash Flows (FCF) and the Discounted Cash Flow (DCF) method. We then provide a description of other valuation methods, outlining reasons for not including those into our study.

2.2.1 Earnings-per-Share and the Price-Earnings Multiples model

In accounting, EPS is the measure of net profit a company made at a point in time, divided by the number of shares outstanding [8], and is expressed as in Equation 2.1.

$$\text{EPS}_q = \frac{\text{Net Income}_q}{\text{Number of Shares Outstanding}_q} \tag{2.1}$$

where Net Income$_q$ denotes a company's profit, net of all accounting expenses, at the end of quarter q, and Number of Shares Outstanding$_q$ is the physical number of shares a company issued, as reported at quarter q, traded on the public markets [5]. The two components required for EPS computation can be found in the '10-K' and '10-Q' annual and quarterly financial reports.

There are two reasons for interest in EPS. First, an increase in EPS means a company can invest more in its production, implying higher future profits [11]. Second, as mentioned earlier, EPS can be used to determine the 'intrinsic value' of a stock. For a stock to be considered

purchasable, the 'intrinsic value' should be greater than the current 'market value' [2].

In this study, we consider the 'forward' P/E method for determining the 'intrinsic value.' 'Forward' in this context means the expected or predicted next quarter's EPS value. P/E refers to the ratio of the current price to earnings per share, expressed in Equation 2.2.

$$P/E = \frac{\text{Price}_t}{\text{EPS}_q} \tag{2.2}$$

where $Price_t$ denotes the price of a stock on day t and EPS_q refers to the most recent quarter's (q) EPS value. The result of this computation is used for valuation as in Equation 2.3.

$$V_t = P/E \times \text{EPS}_{q+1} \tag{2.3}$$

where V_t refers to the 'fair value', P/E is the result of Equation 2.2, and EPS_{q+1} denotes the next quarter's EPS. EPS_{q+1} from Equation 2.3 is the target variable in our regression problem. Observe that over-/under-stating the future EPS figure results in inflated or deflated V_t estimates, which is why getting the forward EPS is important. Another series utilized over the course of this study is the FCF, which is used as the input to the DCF valuation model, which we explain next.

2.2.2 Free-Cash-Flows and the Discounted Cash Flows model

From economic theory, the value of any asset should be equal to the future cash flows this asset generates, discounted at an appropriate rate, corresponding to the risk an investor takes when purchasing the asset in question [8]. This constitutes the 'intrinsic value' [2], which is an estimate of a reasonable price worth paying for the stock, given its future cash flows, discounted at the appropriate rate. The most popular method for estimating the intrinsic value

is the DCF model [3]. The DCF model takes the Free Cash Flows (FCF) as input, with FCF defined as follows:

$$FCF_q = \frac{(\text{CFO}_q - \text{CUI}_q)}{\text{Number of Shares Outstanding}_q}, \tag{2.4}$$

where CFO_q denotes the 'Cash From Operations', and CUI_q denotes the 'Cash Used Investing', scaled by the 'Number of Shares Outstanding'. These two variables are reported quarterly by a company and can be found in the 'Cash Flow Statement'. In Equation 2.4, q denotes the financial quarter, for which the FCF value is computed.

In addition to Free Cash Flows, DCF model requires the weighted average cost of capital (WACC), commonly accepted as an approximation of the risk embedded in common stock [3]. The WACC takes into account the default risk given the amount of leverage at a firm, as well as the risk related to stock market shocks, given the amount of equity capital a firm possesses [5]. More formally:

$$WACC = r_E \frac{E}{D+E} + r_D \frac{D}{D+E}, \tag{2.5}$$

where we let D denote the total debt a firm possesses, E denote the total equity of the firm, r_D denote the required return on debt, and r_E the required return on equity. The values for D and E are displayed on the Balance Sheet of a firm, usually labeled as 'Total Liabilities' and 'Total Shareholders' Equity', respectively.

The cost of debt, r_D, is usually assigned by credit rating agencies, such as Moody's, Standard & Poor's, and Fitch, and is disclosed in 10-K/Q financial reports. The calculation of r_E is different; researchers usually apply the Capital Asset Pricing Model (CAPM) formula, as:

$$r_E = R_f + \beta(R_m - R_f), \tag{2.6}$$

where R_f denotes the risk-free rate, which is typically estimated via the 3-month U.S. Treasury bills rate (i.e., the yield on short-term U.S. government bonds with a 3-month maturity); R_m denotes the expected (average) return on a broader market index, such as the S&P 500; and β denotes the slope coefficient from an Ordinary Least Squares regression, performed on 3 years' worth of monthly returns on the stock, against the financial index's price returns. Specifically, the required rate of return on stock R_s is derived from:

$$R_s = \beta R_{idx} + \epsilon, \tag{2.7}$$

where R_{idx} is the vector of market returns; the slope coefficient β is the quantity required for Equation 2.6; and finally, ϵ denotes the regression error [12].

Together, the WACC and FCF are an essential part of the DCF valuation model, formally defined as:

$$V_q = \sum_{q=1}^{Q} \frac{FCF_{q+1}}{(1 + WACC)^q} + \frac{FCF_q * (1 + g)}{(WACC - g)}, \tag{2.8}$$

where $q = 1, ..., Q$ is the time index, with $q = 1$ corresponding to the first quarter; Q denotes a sufficiently distant point of interest in the future; V_q denotes the intrinsic value of a stock, estimated at quarter q; FCF_q denotes the Free Cash Flows at quarter q; and $WACC$ is the weighted average cost of capital. The right-hand side of Equation 2.8 is the terminal rate — the long-term expected FCF growth, where g denotes some long-term growth rate, typically set at the expected inflation rate [5].

Notice that the variable FCF_{q+1} is the quarter-ahead FCF value, which is the target variable

of the regression problem in this study. In other words, to arrive at the intrinsic value with respect to a future point in time, researchers predict the future Free Cash Flows Q periods into the future, scale each future FCF by $(1 + WACC)^q$ and sum up the resulting values. We calculate the Free Cash Flows as:

As mentioned earlier, intrinsic value estimation is typically performed using the DCF model, and there is broad agreement on calculating the WACC, often through the CAPM and detailed financial statements. However, there is no universally accepted approach for deriving future FCF. Therefore, this research seeks to address this gap by formulating the derivation process as a regression problem. Specifically, we aim to predict future FCF (FCF_{q+1}) using past lags of this variable.

In this study, we only consider a limited subset of valuation approaches: DCF and PE valuation methods. We chose these two primarily because of their popularity within the investment community. The popularity of these arises due to the input series being used: FCF and EPS, respectively. These input series are more reflective of how a business performed in the most recent quarter. Specifically, a portion of these values, both the EPS and FCF that net the necessary business expenses, will be retained by a company to finance further growth, while another portion will be distributed to shareholders. In the next section, we consider other valuation models, their series, and the reasons for not including them as part of the experiments in this document.

2.2.3 Other Valuation Methods

For this study, we focus on two widely used valuation methods—DCF and PE valuation—which will be applied throughout the subsequent chapters [13]. In this section, we present an overview of other notable valuation models, which typically rely on different types of financial

data and have certain limitations that we will discuss. The following subsection is structured as follows: we begin by examining models that utilize data from the 'Income/Loss' statements, before moving on to those based on a typical 'Balance Sheet.'

Price-to-Sales Multiple Model

The Price-Earnings (PE) valuation method is often referred to as the "multiples" approach. In this method, the price of a stock is divided by a key financial figure and then multiplied by the expected value of that same figure. As a result, companies also publish various other financial metrics that can be used for valuation purposes.

The 'Income/Loss' statement in any financial filing begins with the 'Revenues' or 'Net Sales' figure, which represents the total volume of a product or service sold, multiplied by its price, after accounting for any discounts and value-added taxes [14]. For simplicity, we will refer to this figure as 'Sales'. Similar to the EPS method, the sales figure can also be used for valuations. To do so, we first divide the total sales by the number of shares outstanding, as shown in Equation 2.9.

$$SPS_q = \frac{\text{Total Sales}_q}{\text{Number of Shares Outstanding}_q} \tag{2.9}$$

where Total Sales$_q$ represents a company's revenue at the end of quarter q, and Number of Shares Outstanding$_q$ is the total number of shares, as discussed earlier. The resulting value is the Sales Per Share (SPS) ratio. To convert this into a financial multiple, the stock price is divided by the SPS ratio, as shown in Equation 2.10.

$$P/S_q = \frac{\text{Price}_t}{SPS_q} \tag{2.10}$$

where P/S_q represents the Price-To-Sales ratio, with Price_t being the price of a stock on day t during quarter q, and SPS_q is the result from Equation 2.9, the Sales per Share ratio. To estimate the intrinsic value, the P/S ratio is multiplied by the expected (forecasted) Sales for the upcoming quarter, as shown in Equation 2.11.

$$V_t = P/S \times \text{Sales}_{q+1} \tag{2.11}$$

where, as in the previous equations, V_t represents the intrinsic value, P/S is the Price-To-Sales Ratio, and Sales_{q+1} denotes the projected sales for the next quarter [15]. The Price-to-Sales (P/S) method is particularly useful when a company reports negative earnings, as its sales can never be negative — a negative sales figure is legally prohibited [16]. This method is especially relevant for new, fast-growing businesses that have developed a popular product with high sales growth, even though their expenses remain high.

However, this model overlooks any expenses a company may have incurred or will incur in the future, focusing solely on the demand for the company's product or service. As a result, it fails to account for the investments a company must make to reach the projected sales level, as well as essential accounting deductions such as salaries, taxes, and other operational costs.

This limitation of the current valuation method is addressed by the next model we will describe.

Enterprise Value to Earnings Before Interest, Taxes, Depreciation and Amortization Multiple Model

The essence of investing lies in identifying stocks that not only exhibit strong growth potential but also offer protection against bankruptcy. For shareholders, bankruptcy can result in the

permanent loss of capital.

In most cases, when a company declares bankruptcy, it faces two options: either complete liquidation or continuation under restructuring. In both scenarios, shareholders have claims on the company's assets — everything the company owns [14].

This is what the Enterprise Value to Earnings Before Interest, Taxes, Depreciation, and Amortization (EV/EBITDA) multiple captures: the value of a company, considering its debt, relative to the profits it generates from selling its product, paying its employees, and covering operating expenses.

Often, liquidation is caused by high levels of debt a company owes—more than it can service. Hence, Enterprise Value (EV) is a financial measure that considers the value of a business based on its total debt. It is mathematically represented in Equation 2.12.

$$EV_q = \text{Mkt Cap}_q + \text{Total Debt}_q - \text{Cash and Cash Equivalents}_q \tag{2.12}$$

where EV_q is the Enterprise Value, Mkt Cap$_q$ is the market capitalization, defined as the number of shares outstanding multiplied by the price per share, and Cash and Cash Equivalents$_q$ represent the immediate funds and their equivalents—assets that can be quickly sold or converted to cash, available to a business in quarter q. EV_q is the numerator in the EV/EBITDA multiple.

The denominator in this multiple is EBITDA—the profit a company made from selling its product, net of operating expenses. Operating expenses are those that a business must pay to keep operating—such as manufacturing, promoting, and distributing its goods. These include salaries, marketing, research, and manufacturing expenses. This figure can be found in the 'Income/Loss' section of quarterly and annual reports.

The EV/EBITDA multiple is obtained by dividing the Enterprise Value by the EBITDA figure. Hence, deriving the intrinsic value using this multiple is similar to the multiples we presented earlier and is outlined in Equation 2.13.

$$V_q = \text{EV/EBITDA}_q * \text{EBITDA}_{q+1} \tag{2.13}$$

where V_q denotes the intrinsic value, EV/EBITDA_q is the Enterprise Value to Earnings Before Interest, Taxes, Depreciation, and Amortization multiple for quarter q, and EBITDA_{q+1} is the EBITDA value for the next quarter.

The benefit of this valuation method lies in the fact that EBITDA is used as the input series. Unlike a company's sales, this figure takes into account the necessary expenses required to keep the business operating. At the same time, this is a significant drawback of using this model as the basis for a valuation decision.

EBITDA cannot be used as a replacement for Earnings Per Share or Free Cash Flow. This is because EBITDA can be misleading — it does not represent the value a shareholder will receive. In fact, it represents the cash flow available to the debtor, as interest on debt is paid first. Then, the government receives its portion in the form of taxes [14].

Depending on where and how the business is conducted, a company must recognize the depreciation and amortization of its tangible and intangible assets, respectively. Depreciation and amortization allow a business that has recently purchased property or a patent to expand and reduce taxable income [14]. This is done by regulators to encourage business activities and investments.

After deductions for amortization and depreciation, a company pays taxes, the rate of which varies depending on the regulator to which the company is subject. Finally, after all

previous deductions, shareholders receive the amount they are entitled to: the net profit, which is explained in Section 2.2.1.

Therefore, EBITDA is not the figure that represents cash flows to a shareholder; rather, it indicates whether the operations conducted by a business are generally profitable. Hence, this is why we did not use this model in the present study. The next model we describe has similar mechanics, but it uses a different type of series as input.

Price-To-Book Multiple

An essential part of the annual and quarterly report is the 'Balance Sheet', which shows the position of a company in terms of what it owns and owes. A company owns assets—cash, buildings, and everything else it has purchased over the course of its existence. These purchases can be made using debt, or funds borrowed.

In the case of liquidation of a business, the company must sell its assets, pay all its debts, and the remainder is given to the shareholders. This remainder is called the 'Book Value of Equity' or 'Total Equity,' as these funds represent everything gathered and owned by the company's shareholders. Since every shareholder has claims, the total book value can be divided by the number of shares, as indicated in Equation 2.14.

$$BPS_q = \frac{Eq_q}{\text{Number of Shares Outstanding}_q} \tag{2.14}$$

where BPS_q is the Total Equity per Share (or Book Value per Share), Eq_q is the book value of equity for quarter q, and Number of Shares Outstanding is the total number of shares issued by the company.

The derivation of intrinsic value is similar to the Price-to-Sales and Price-to-Earnings

models: we divide the price of a stock by the BPS_q value and multiply the result by the expected total equity for the next quarter, as illustrated in Equation 2.15.

$$V_q = \frac{\text{Price}_q}{BPS_q} \times BPS_{q+1} \tag{2.15}$$

where V_t denotes the value of a stock, Price_q is the price of the company's stock during quarter q, and BPS is the book value of equity per share for quarter q and the following quarter $q+1$.

The book value of equity, while better reflecting what belongs to a shareholder in a company compared to figures such as Sales, Earnings, and EBITDA, has one important flaw due to how the company recognizes debt and assets [17]. Specifically, for an established and mature company, the book value could be very small relative to its stock price; thus, the intrinsic value computed using this method may not be accurate: it could be too small and not useful for comparison with the market price.

This is the last multiple we consider in this study. Next, we consider the final model included as part of the Background review.

Dividend Discount Model

When an investor purchases a stock, she is entitled to a dividend. The dividend represents the cash flow to a shareholder and can therefore be used to derive the value of a business.

The Dividend Discount Model (DDM) is similar to the Discounted Free Cash Flow model: it takes the dividend as input and computes the intrinsic value based on the required rate of return and the risk the investor assumes when buying a stock [18].

The dividend is a portion of Earnings-Per-Share that is distributed by the company to investors. This is not required by law, and the company's management is free to choose any

amount they deem appropriate. However, there is a trade-off: the higher the portion of the dividend relative to EPS, the lower the amount of earnings available to reinvest in the business. Hence, it is rare for dynamically growing companies to pay a large portion of their income as a dividend [8]. The model that uses the dividend payment is mathematically represented in Equation 2.16.

$$V_q = \frac{D_q * (1 + g)}{r - g} \tag{2.16}$$

where V_q is the intrinsic value of a stock in quarter q, D_q is the dividend in quarter q, r is the required rate of return, computed using the Capital Asset Pricing Model, similar to the DCF model in Equation 2.6, and g is the dividend growth rate.

It is proposed that the growth rate g can be computed using the Return on Equity (ROE) ratio, which measures how much money a company generates per unit of currency invested, in percentage terms, as depicted in Equation 2.17.

$$ROE_q = \frac{NI_q}{Eq_q} \tag{2.17}$$

where ROE is the return on equity ratio in quarter q, NI_q is the net income, and Eq is the total shareholders' equity in quarter q, respectively.

Originally, it is proposed that a reasonable assumption for the growth rate g is computed using the 'bare-minimum' required growth. Hence, the growth rate implied in the Dividend Discount Model is calculated as $g = ROE_q * r_q$, the expected return on equity proportional to the required return. However, one may use any other growth rate they deem appropriate [19].

There are two reasons this model is not used in our study. First, not every firm pays a dividend, as it is not an obligation. Therefore, this limits the number of companies on which

we could theoretically perform our experiments. Additionally, if a company reports a negative EPS value, it is possible that it cancels dividend payments. For our study, this implies that there could be extended gaps between dividend payments, further complicating estimator training and inference.

Second, firms may declare dividend proceeds at different frequencies. While some companies pay dividends quarterly, others may opt for annual payments. This difference in payment frequencies would further complicate the forecasting pipeline, as data would arrive at different intervals.

Finally, a dividend is not an obligation—some businesses, especially those with rapid increases in sales volume [8], prefer to retain more money within the company rather than distributing funds to shareholders. For our study, this imposes a limitation on the selection of stocks for inclusion in our experiments.

In the next section, we summarize information about the financial valuation models we use.

2.2.4 Summary of Valuation Methods

In this part of the Background Review, we present a number of financial valuation approaches considered for this study. The models presented here are among the most popular with the investment community.

The multiples outlined in this section can be used in two ways. The first, as presented in this study, involves multiplying by the expected value of a key series (sales, earnings, EBITDA, etc.). The second method is to multiply by the average multiple for the comparable company or industry.

This method is outside the scope of our study for several reasons. First, the method of

comparables requires selecting companies to which the business under consideration will be compared.

Specifically, it is important to choose a company or set of companies with similar size, region, and product. Picking the wrong companies leads to elevated multiples and, consequently, to an inaccurate intrinsic value. In particular, a fast-growing start-up is hardly comparable to a mature, established business: the two have different risk tolerances and different fund sizes to allocate for production.

Technically, the comparable method would eliminate the need for forecasting, thereby imposing a different kind of research, one that does not propose innovations to valuation models (or innovations of a different nature).

From the selection of multiples, we identified the Price-Earnings method as the most appropriate for our study. Unlike the Price-Sales and EV/EBITDA methods, it considers Earnings-Per-Share as money attributable to shareholders and, therefore, is more suitable for determining the intrinsic value of a stock.

Alternatively to the multiples, there are the Discounted Cash Flow and Dividend Discount Models, which we also describe in this part of the document. Since dividends are not required by law, some companies may choose not to pay them at all. Additionally, gaps in dividend payments could limit the number of available data points for model training and validation purposes. Therefore, we chose the Discounted Cash Flow model, as its primary input series (Free Cash Flows) are reported by companies each quarter.

Therefore, in this study, we aim to model FCF and EPS data. Both series, along with other types of fundamental financial time-series data, are sparse due to their quarterly publication frequency and the fact that organized data collection began in the 1980s [6]. However, mathematical methods exist that allow us to assume the data is generated at a higher frequency.

One such method is interpolation, which is discussed along with feature transformations in the next section.

2.3 Series Augmentation and Transformation Methods

As mentioned earlier, we aim to solve a regression problem, where the objective is to minimize the distance between the target variable (EPS or FCF in our case) and the value predicted by a regression model.

To solve these types of problems, researchers define a cost function — the objective function to be minimized by a regression algorithm [20]. This cost function can be sensitive to the distribution and units of the data. For this reason, it is common to apply feature scaling and transformation methods to both the target and feature variables.

In our case, both the target and feature variables are in the same units. However, due to adverse external or internal factors, companies could report a negative FCF/EPS value after a quarter of positive results, creating a discrepancy between the target and feature values. Therefore, several scaling and transformation methods were applied to identify the most appropriate methodology for our series. These methods are described in the next Subsection 2.3.1.

2.3.1 Scaling and Transformation Methods

Scaling, normalization, and transformation are commonly applied to regression-type problems [21]. This is typically done to minimize the impact that the distribution or unit measure might have on the estimation process [22].

One way to address this challenge is to enforce that both the target and feature vectors

follow a Gaussian (normal) distribution. This is the purpose of the normalization function in Equation 2.18.

$$x\prime_t = \frac{(x_t - \mu)}{\sigma} \tag{2.18}$$

where $x\prime_t$ is the output, a scaled variable, x_t denotes an observation at time t of variable x, and μ and σ represent the sample mean and standard deviation, respectively.

Normalization in Equation 2.18, while imposing a Gaussian distribution on a vector, does not scale it to the 0-1 range. This is important because, if the features and target are in different units, the estimator may struggle to minimize the regression error [23].

There are several methods that allow scaling a vector to the 0-1 range. The simplest is the 'MinMax' scaling, defined in Equation 2.19.

$$x\prime_t = \frac{x_t - x_{min}}{x_{max} - x_{min}} \tag{2.19}$$

where, again, $x\prime_t$ is the output, x_t denotes the t-th observation of vector x, x_{min} is the minimum of vector x, and x_{max} is the maximum of vector x. When the original data ranges from a high positive to a high negative number, 'MinMax' scaling has the effect of compressing the data into a narrow range [24]. In such cases, 'MaxAbs' scaling is applied, as shown in Equation 2.20.

$$x\prime_t = \frac{x_t}{|x_{max}|} \tag{2.20}$$

where $x\prime_t$ is the scaled output variable, x_t is the value of variable x at index t, and $||$ is the absolute value operator, i.e., $|-1| = 1$, with x_{max} being the maximum value of vector x.

The aforementioned scaling methods are not without flaws: they are all sensitive to a small

number of marginal outliers. This challenge is overcome with the robust scaler, a technique where values in a vector are scaled into the -3 to 3 range, according to the percentiles. The formula for the robust scaler is outlined in Equation 2.21.

$$x\prime_t = \frac{x_t - Q1}{Q3 - Q1} \tag{2.21}$$

where $x\prime_t$ is the output scaled variable of x at index t, and $Q1$ and $Q3$ are the first and third quartiles of the distribution of vector x. The denominator in Equation 2.21 is therefore the inter-quartile range.

The 'MaxAbs', 'MinMax', and 'Robust' methods are used solely to scale data within a specific range. In contrast, Normalization is applied to enforce data to follow a predefined Gaussian distribution. Alternatively to the Normal (Gaussian) distribution, we use the Quantile Transformer (QT), which, makes the transformed data to follow the Uniform distribution, in order to minimize the potential impact of outliers [25].

The Quantile Transformer (QT) works as follows. First, each observation is assigned a specific rank based on its position in a sorted (in descending order) array:

$$r_t = rank(x_t) \tag{2.22}$$

where r_t is the output of the function $rank$, which ranks the input value of x_t at index t. Consequently, all values of r_t are scaled to the 0-1 range, as shown in Equation 2.23.

$$q_t = \frac{r_t - 1}{T - 1} \tag{2.23}$$

where q_t is the transformed value at index t, r_t is the ranked value of the original vector x according to Equation 2.22, and T is the total number of data points in the vector to be

transformed.

Apart from scaling and transforming our datasets, we also make an assumption about the data frequency. Specifically, the interpolation techniques described in the next section allow us to assume a more frequent data stream.

2.3.2 Augmentation Methods

Interpolation is a mathematical technique used to approximate values between two known data points [26]. This technique allows us to assume a frequency different from the quarterly one for our target series by approximating a new EPS/FCF value between two quarters.

Different approximation techniques exist for performing interpolation. The simplest is Linear Interpolation, which is formulated for a data point y as defined in Equation 2.24.

$$\tilde{y} = y_1 + (x - x_1)\frac{y_2 - y_1}{x_2 - x_1} \tag{2.24}$$

where $\tilde{y}$ is the approximated data point, y_1 and y_2 are the two known data points, and x, x_1 and x_2 are the x-axis coordinates of the data points. Since the interpolation occurs along a 1-dimensional vector, the x-values are simply the indexes at which the interpolation takes place. However, this interpolation method has one flaw: the resulting series tends to be smoother than the original, thus lacking the volatility present in the known data in our setting [27].

A more advanced interpolation technique we used involves polynomials. Specifically, consider the polynomial function defined in Equation 2.25.

$$P(x) = \sum_{i=0}^{n} y_i L_i(x) \tag{2.25}$$

where $P(x)$ is the polynomial function output, n is the maximum degree of the polynomial,

and $L_i(x)$ is the Lagrange multiplier corresponding to the x-coordinate of i-th order. The Lagrange multiplier is defined in Equation 2.26.

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{2.26}$$

where n is the degree of the polynomial and x is the x-coordinate of the data point to which we attempt to interpolate.

Spline interpolation builds upon polynomial interpolation [28] by also utilizing polynomials. The key difference is that spline interpolation is a piecewise method: it operates on intervals $[x_i, x_{i+1}]$ and is defined as shown in Equation 2.27.

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \tag{2.27}$$

where $S_i(x)$ is the spline interpolation function of degree i with input x. The coefficients a_i, b_i, c_i and d_i are determined by the system of equations in Equation 2.28.

$$c_i = \frac{M_i}{2}; d_i = \frac{M_{i+1} - M_i}{6(x_{i+1} - x_i)}; b_i = \frac{y_{i+1} - y_i}{x_{i+1} - xi} - \frac{x_{i+1} - x_i}{6}(2M_i + M_{i+1}); a_i = y_i \tag{2.28}$$

where i is the degree of polynomial and M is the second derivative of function $S_i(x)$.

The final interpolation method used is the Piecewise Cubic Hermite Interpolating Polynomial (PCHIP). This method is particularly useful for preserving the local properties of the series [29]. The PCHIP interpolation is defined as follows.

$$P_i(x) = h_{00}(t)y_i + h_{10}(t)m_i + h_{01}(t)y_{i+1} + h_{11}(t)m_{i+1} \tag{2.29}$$

Equation 2.29 defines a PCHIP polynomial, where $t = \frac{x - x_i}{h_i}$ is the normalized parameter

with h_i The Hermite basis functions, defined in Equation 2.30, and m_i and m_{i+1} represent the slopes (first derivatives) at x_i and x_{i+1}, respectively.

$$h_{00}(t) = 2t^3 - 3t^2 + 1; h_{10}(t) = t^3 - 2t^2 + t; h_{01}(t) = -2t^3 + 3t^2; h_{11}(t) = t^3 - t^2; \quad (2.30)$$

Notice that index values of function h: $00, 10, 01, 11$ are used to denote the specific functions associated with the cubic Hermite interpolation. The function h_{00} is used to weight the function at a first point, y_i. Function h_{10} is associated with the slope and derivative at the first point m_i. Function h_{01} weights the second function value y_{i+1}. Finally, h_{11} is associated with the second slope of m_{i+1}.

Computation of m_i and m_{i+1} terms is given in Equation 2.31.

$$m_i = \frac{d_{i-1} + d_i}{2} \text{min}(\frac{|d_{i-1}|}{w_1}, \frac{|d_i|}{w_2}) \quad (2.31)$$

where $d_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$represents the slope between two points to be interpolated. In Equation 2.31, the terms $w_1 = 2h_i + h_{i-1}$ and $w_2 = h_i + 2h_{i-1}$ are scaling factors.

The interpolation methods described above enable us to generate additional data points between the existing ones. Combined with the scaling and transformation techniques, the goal is to provide our estimators with more information to better address the regression problem, which is outlined in the next section.

2.4 Regression Problem

In this section, we define the regression problem that we aim to solve in this study. Specifically, our goal is to predict the EPS/FCF value for the next quarter using the past lags, along with

the mean and standard deviation of these series. This implies a supervised regression setting, where the estimator aims to minimize the distance between the predicted values and the true values (labels) [20]. Therefore, we employ several machine learning (ML) and statistical (SE) regression estimators, each with different cost functions. For the purposes of this study, we define the distinction between a ML and a SE estimator as follows: the former utilizes a form of regularization during the estimation process.

We begin with a description of the statistical estimators in the next subsection.

2.4.1 Statistical Estimators

As mentioned in the previous section, we aim to model a regression problem where the goal is to minimize the cost function between the target and the inferred values of the features. In this study, we consider several regression estimators, each with varying cost functions.

We start with a description of the 'Ordinary Least Squares' estimators.

Ordinary Least Squares

One of the most common cost functions in regression problems is the Mean Squared Error (MSE) [30], as formulated in Equation 2.32.

$$\epsilon_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\tilde{y_i} - y_i)^2 \tag{2.32}$$

where N is the total number of observations, and $\tilde{y_i}$ is the estimated value of the target variable y_i, at observation $i = 1, \ldots, N$. The estimated value $\tilde{y}$ can be formulated as in Equation 2.33.

$$\tilde{y} = \sum_{k=1}^{K}\theta_k x_k \tag{2.33}$$

where θ_k is a weight assigned to each feature vector x_k — a parameter to be optimized during the training procedure, with $k = 1, \ldots, K$ the total number if input features, K in total.

Equation 2.33 is also known as 'Ordinary Least Squares' (OLS) and is one of the statistical methods we will use in this study. Specifically, in this method, we aim to minimize the regression error (Equation 2.32) by optimizing the coefficient θ. In matrix notation, the solution for the optimal θ is expressed in Equation 2.34.

$$\theta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y \tag{2.34}$$

where $\mathbf{X}$ represents the matrix of input features, T denotes the matrix transpose operation, $\mathbf{X}^{-1}$ indicates the inverse of matrix $\mathbf{X}$, and y represents the target variables.

The benefit of this estimator lies in its simplicity. The interpretation of the assigned weights signals a higher correlation between the target and a feature. However, several assumptions are imposed on the data, and the violation of these assumptions can lead to poor results [30]. These assumptions are as follows:

1. Linearity — relation between target and features must be strictly linear;
2. No Multicollinearity — features matrix should be invertible, otherwise weights are impossible to estimate;
3. Homoscedasticity — the difference between target and estimated values, known as residuals, has a constant variance;
4. Independence — features are independent of each other;
5. Normality — errors are assumed to be normally distributed between 0 and a constant variance;

The violation of any of these assumptions could lead to an inconsistent estimation procedure, resulting in overfitting or poor generalization to out-of-sample data. However, in the context of time series, it is common for errors to exhibit non-constant variance, and for the data to violate other assumptions.

Due to this, we also employ the Weighted Least Squares regression, which is described next.

Weighted Least Squares

The 'Weighted Least Squares' (WLS) is an extension of the OLS regression that addresses violations of the homoscedasticity assumption. Specifically, for each data point, a weight is assigned such that observations with smaller variance receive higher weights [31].

Specifically, the objective function is an extension of the MSE error, with an additional term that accounts for weighting, as shown in Equation 2.35.

$$\epsilon_{WMSE} = \frac{1}{N}\sum_{i=1}^{N} w_i(\tilde{y}_i - y_i)^2 \tag{2.35}$$

where N is the total number of observations, $\tilde{y}_i$ is the i-th estimated value of the target variable y, and w_i is the weight, formulated as $w_i = \frac{1}{\sigma_i^2}$, with $\sigma_i^2 = \text{Var}(\epsilon_i)$, where ϵ_i is the regression error, defined in Equation 2.32. Hence, the WLS regression cost function extends the OLS method by bypassing some of its limitations related to strict assumptions about the data. Consequently, this alters the optimization of the regression θ parameters, which is represented in Equation 2.36.

$$\theta = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}y \tag{2.36}$$

where $\mathbf{X}$ represents the input feature matrix, $\mathbf{X}^T$ denotes the transpose, and $\mathbf{X} - 1$ represents the inverse of matrix $\mathbf{X}$. The vector y represents the target values, and $\mathbf{W}$ is the weighting matrix, optimized using Equation 2.35. The diagonal elements of $\mathbf{W}$ represent the inverse of the variance terms, so that $\mathbf{W} = \text{diag}\left(\frac{1}{\sigma_1^2}, \frac{1}{\sigma_2^2}, \frac{1}{\sigma_3^2}, \ldots, \frac{1}{\sigma_N^2}\right)$ for N observations.

One limitation of the WLS estimator is that the structure of the variance of residuals (regression errors) needs to be known in advance [30]. If the variance is not known, the estimation may yield inconsistent results due to the bias introduced by the regression procedure. Additionally, since the data used in this study is time-series in nature, the variance of the error term is subject to changes over time, further complicating the estimation process. Additionally, the WLS estimator shares some of the assumptions of the OLS estimator, particularly the assumption of feature independence. In our study, the features represent past lags of the target variables, which may lead to a high correlation between them. Moreover, both the OLS and WLS estimators are sensitive to outliers, which makes them less effective for datasets that exhibit dynamic or volatile behavior.

The next estimator we present combines both OLS and WLS: the Robust Linear Model.

Robust Linear Models

The Robust Linear Model (RLM) is an extension of both OLS and WLS methods [32]. Specifically, it is designed to be less sensitive to outliers by replacing the MSE cost function with the Huber Loss Function [33]. The Huber loss is given in Equation 2.37.

$$L(\theta) = \begin{cases} \frac{1}{2}(y_i - x_i^T\theta_i)^2, \text{if} |y_i - x_i\theta_i| \leq c \\ c|y_i - x_i^T\theta_i| - \frac{1}{2}c^2 \text{otherwise} \end{cases} \tag{2.37}$$

where θ represents a set of regression parameters, x_i represents the i-th observation of input

vector x, c is the threshold parameter, commonly set at 1.35 [33]. The Huber Loss function represents a kernel function, where the loss is computed as either an MSE or absolute difference between a target and predicted observations, depending on the threshold.

Similar to the WSL, we aim to update regression coefficients, using the total error as in Equation 2.38.

$$\theta^{t+1} = (\mathbf{X}^T \mathbf{W}^{(t)} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}^{(t)} y \tag{2.38}$$

where, again, $\mathbf{X}$ is the input matrix, T and $\mathbf{X}^{-1}$ are transpose and inverse of matrix $\mathbf{X}$, and $\mathbf{W} = \text{diag}(w_1, w_2, \ldots, w_N)$, where each $w = L(\theta)$, with parameters θ derived from the previous iteration of optimization.

By switching from squared to absolute loss, the RLM estimator becomes more robust to outliers. However, this requires a choice of the threshold c, which we keep fixed at 1.35, the commonly accepted value [34]. The major flaw of this regressor is its computational efficiency, as it requires several iterations for optimization of parameters θ.

Following the footsteps of the previous two estimators, the RLM model also assumes a constant variance term, therefore being prone to errors in a time-series context. The next estimator, however, focuses more on time-series properties.

Auto-Regressive Integrated Moving Average

The Auto-Regressive Integrated Moving Average is a model type specifically designed to work with time-series data. Specifically, in such a setting, the target variable often displays high auto-correlation — a correlation with its own lags [35].

The ARIMA regression is performed in several steps. Generally, this model is expressed in

Equation 2.39.

$$\phi_p(B)(1-B)^d y_t = \theta_q(B)\epsilon_t \tag{2.39}$$

where B is the lag-shift operator, i.e. $By_t = y_{t-1}$ for time-step index t, d is the number of differencing steps, which is necessary to assure that the series is stationary. The whole estimation process is split into two steps [30].

The first step is the Auto-Regressive step, formulated as a function $\phi_p(B)$ in Equation 2.39. This step is therefore formulated in Equation 2.40.

$$y_t = \phi_p(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p \tag{2.40}$$

where we assume the value of the target variable y at time-step t depends on its past lags — the function B. ARIMA allows integrating the moving average component of residuals, computed as in Equation 2.41.

$$y_t = \theta_q(B)\epsilon_t = \epsilon_t + \theta_1\epsilon_1 + ... + \theta_q\epsilon_q \tag{2.41}$$

where y_t depends on the moving average moments of error ϵ up to the q-th moment.

The final step is the integration, which is given with the use of Equations 2.40 and 2.41, and, after reorganization and integration, yields the results in Equation 2.42 for the next quarter.

$$y_{t+1} = \phi_1 y_t + \phi_2 y_{t-1} + ... + \phi_p y_{t-p+1} + \theta_1\epsilon_t + \theta_2\epsilon_{t-1} + ... + \theta_q\epsilon_{t-q+1} \tag{2.42}$$

where ϕ and θ are weights optimized for the lags of y and errors ϵ.

ARIMA is generally used for time-series settings as it combines the Auto-Regressive and Moving Average components, both of which are used to estimate series trend and volatility. On the other hand, this forecasting model type requires data to be stationary, meaning it needs to be de-trended and de-seasoned before fitting the estimator.

The final estimation methodology used in this study accounts for the dynamic nature of the data.

Exponential Smoothing

Exponential Smoothing (SES) estimates future values by assigning higher weights to the more recent data observations, and lower weights to the older ones, making it a more appropriate choice of estimator for time-series data with no clear structure [36]. Specifically, we use the Holt-Winters Seasonal method that accounts for trends and seasonality [37].

The prediction is done via the specification outlined in Equation 2.43.

$$\tilde{y}_{t+h} = (S_t + hb_t)I_{t+h-m(k)} \tag{2.43}$$

where $\tilde{y}$ is the predicted value at time t, with a forecasting horizon of h, m is the seasonality period, and $k = \lfloor \frac{(h-1)}{m} \rfloor$ is the result of floor division to control the indexing of seasonal periods. In Equation 2.43, the term S_t is the level of smoothing of the series, defined in Equation 2.44.

$$S_t = \alpha \frac{y_t}{I_{t-m}} + (1-\alpha)(S_{t-1} + b_{t-1}) \tag{2.44}$$

where α is the smoothing parameter, where $0 < \alpha \leq 1$, y_t is the observed target variable value at time t, and b_t is the smoothed trend, defined in Equation 2.45.

$$b_t = \theta(S_t - S_{t-1}) + (1 - \theta)b_{t-1} \tag{2.45}$$

with θ being the smoothing weight. The last component that we have not yet defined in Equation 2.43 is I_t, which controls the seasonality smoothing and is defined in Equation 2.46.

$$I_t = \gamma\frac{y_t}{S_t} + (1 - \gamma)I_{t-m} \tag{2.46}$$

where γ is the seasonality smoothing components.

The SES estimator is effective at predicting short-term series, effectively smoothing the observed series. However, it, as well as all the previous estimators, requires the relation between features and target variables to be strictly linear. It is more effectively handled by the Machine Learning Estimators, present in the next section.

2.4.2 Machine Learning Estimators

We mentioned earlier that for the purposes of this study, we use a series of Machine Learning (ML) estimators. The key distinction we draw between ML and SE estimators is that the former imposes regularization on the parameters of regression. As such, one of the first estimators we consider from an ML branch is Lasso.

Least Absolute Shrinkage and Selection Operator

The LASSO estimator is an extension of the OLS estimators by imposing restrictions on the weights [38]. The estimator attempts to minimize an MSE cost function, with a modification to simplify the derivation of gradients, as shown in Equation 2.47.

$$MSE_{lasso} = \frac{1}{2N}\sum_{i=1}^{N}(y_i - \mathbf{x}_i\theta_i)^2 \tag{2.47}$$

where $\frac{1}{2N}$ is the scaling factor used to simplify the derivation procedure with respect to θ, as the factor of 2 in the fraction and the power will cancel each other out. In a similar to the previous equations manner, the y_i represents the i-th observation of the target variable y (the scalar value), $\mathbf{x}_i$ is, respectively i-th observation of the input features vector, with respective weights vector θ One of the key properties of Lasso regression is the fact that it maximizes the impact of the most relevant features on an estimated value. This is accomplished via the use of L1-regularization [39], expressed in Equation 2.48.

$$\|\theta\|_1 = \sum_{j=1}^{p}|\theta_j| \tag{2.48}$$

where $\|\theta\|_1$ denotes L1-regularization, which is the sum of the absolute values of the regression coefficients. Note that the number of coefficients p will coincide with the number of vectors in the input data. The regularization term in Equation 2.48 adds to the total loss function, as expressed in Equation 2.47. This creates the minimization objective: to select the set of coefficients θ that minimizes the distance between observed and estimated target values, as expressed in Equation 2.49.

$$\hat{\theta} = \arg\min_{\theta}\left\{\frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \mathbf{x}_i^T\theta\right)^2 + \lambda\sum_{j=1}^{p}|\theta_j|\right\} \tag{2.49}$$

where the estimated set of coefficients is the one that minimizes the MSE, as expressed in Equation 2.47, subject to regularization as per Equation 2.48, with parameter λ. Here, $\lambda \geq 0$ is a tunable parameter that controls the strength of regularization: the larger λ, the more coefficients will shrink to zero. Conversely, with small λ, coefficients will be closer to those

estimated with OLS.

One of the flaws of this estimator is the fact that when features are highly correlated, it may maximize a weight for a single feature, effectively ignoring the rest [38]. Additionally, a high weight imposed on the regularization term may introduce bias in the estimation process, leading to overfitting.

A similar idea is to pick a subset of features and assign higher weights to them, while replacing the standard MSE with the more robust Huber Loss function. As mentioned earlier, this cost function is robust to outliers. The ML estimator that uses it is the Huber Regressor, outlined next.

Huber Regressor

The Huber Regressor (HR) is an extension of the Robust Linear Models (RLM) statistical estimator that uses the Huber Loss function. Effectively, this cost function represents a kernel that adjusts depending on the size of the regression residuals, which are defined as the squared error between predictions and observed target values [34]. For convenience, the Huber loss function is repeated in this subsection in Equation 2.50.

$$L_c(\theta) = \begin{cases} \frac{1}{2}(y_i - \mathbf{x}_i^T\theta_i)^2, \text{if } \ |y_i - \mathbf{x}_i^T\theta_i| \leq c \\ \\ c|y_i - \mathbf{x}_i^T\theta_i| - \frac{1}{2}c^2 \ \ \text{otherwise} \end{cases} \tag{2.50}$$

where y_i is the target observation, $\mathbf{x}_i$ is the set of features at observation i, θ_i is the set of regression parameters, and c is a constant threshold, commonly set to 1.35 [34].

As with other loss functions, we aim to minimize the average Huber Loss by adjusting the θ regression parameters, as mathematically expressed in Equation 2.51.

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} L_c(y_i - \mathbf{x}_i^T \theta) \tag{2.51}$$

where we seek to find the set of optimal parameters $\hat{\theta}$ that yields the lowest Huber loss error L_c, with respect to the difference between the true target observation y_i and the model estimate $\mathbf{x}_i^T \hat{\theta}$.

The major extension is due to the fact that Huber loss makes use of L2-regularization of the regression coefficients, an extension of the type of regularization used by Lasso regression. Hence, our final version of the regression loss function is provided in Equation 2.52.

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} L_c(y_i - \mathbf{x}_i^T \theta) + \frac{\lambda}{2} \sum_{j=1}^{p} \theta_j^2 \tag{2.52}$$

where the additional term denotes the regularization, i.e., the sum of squared regression coefficients, where p coincides with the number of features in the input matrix, and the λ term controls the strength of regularization. The major difference between L1 and L2 regularization is that with the latter, coefficients will never become exactly zero for any feature — all features contribute at least partially to the regression output [40].

The Huber loss is not the only regression model with a loss function different from the standard MSE. We next consider Automatic Relevance Determination, which makes use of Bayes' theorem for optimizing regression coefficients.

Automatic Relevance Determination

The Automatic Relevance Determination (ARD) estimator makes use of certain Bayesian properties of the estimation process. Specifically, it places independent Gaussian priors on each regression coefficient [38]. Mathematically, this is expressed in Equation 2.53.

$$\theta_j = \theta_j \sim \mathcal{N}(0, \alpha_j^{-1}) \tag{2.53}$$

where each coefficient θ_j with $j = 1, ..., p$, and p is the total number of coefficients, which coincides with the number of features. Each coefficient thus comes from a Gaussian (Normal) distribution with mean 0 and inverse variance α_j^{-1}. This term is the precision parameter for the j-th coefficient.

This suggests that the probability of the coefficients set, given the precision parameter, is expressed in Equation 2.54.

$$p(\theta|\alpha) = \prod_{j=1}^{p} \mathcal{N}(\theta_j|0, \alpha_j^{-1}) \tag{2.54}$$

where $\alpha = [\alpha_1, \alpha_2, ..., \alpha_p]^T$ is the vector of precision terms, independent for each coefficient θ. The $p(\theta|\alpha)$ denotes the probability of the set of coefficients θ, given the vector of precision terms α.

This implies that the likelihood of the observed data y given the input matrix $\mathbf{X}$ and the set of parameters θ is expressed in Equation 2.55.

$$p(y|\mathbf{X}, \theta, \sigma^2) = \mathcal{N}(y|\mathbf{X}\theta, \sigma^2\mathbf{I}) \tag{2.55}$$

which states that the likelihood $p(\cdot)$ of y given the feature matrix $\mathbf{X}$, parameters θ, and the variance of the target variable σ^2, with I as the identity matrix, necessary to convert the scalar value σ^2 to the corresponding matrix form.

Using Bayes' theorem [20], we derive the posterior distribution of the parameterization θ, given in Equation 2.56.

$$p(\theta \mid y, \mathbf{X}, \boldsymbol{\alpha}, \sigma^2) \propto p(y \mid \mathbf{X}, \theta, \sigma^2) p(\theta \mid \boldsymbol{\alpha}) \tag{2.56}$$

also following the Gaussian distribution: $p(\theta \mid y, \mathbf{X}, \boldsymbol{\alpha}, \sigma^2) = \mathcal{N}(\theta|\mu, \boldsymbol{\Sigma})$, where the distribution is centered around the mean, defined as $\mu = \sigma^{-2}\boldsymbol{\Sigma}\mathbf{X}^T y$, and the variance is defined as $\boldsymbol{\Sigma} = (\sigma^{-2}\mathbf{X}^T\mathbf{X} + \text{diag}(\alpha))^{-1}$. This implies that, for the estimation process, we need to derive the precision terms $\boldsymbol{\alpha}$ and the noise variance σ^2. This can be done by maximizing the marginal log-likelihood of the data [41], represented in Equation 2.57.

$$\log p(y|\mathbf{X}, \alpha, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{C}| - \frac{1}{2}y^T\mathbf{C}^{-1}y \tag{2.57}$$

where $\mathbf{C} = \sigma^2\mathbf{I} + \mathbf{X}, \text{diag}(\alpha)^{-1}\mathbf{X}^T$ represents the combination of uncertainty due to the noise ($\sigma^2\mathbf{I}$) and the uncertainty due to the regression coefficients ($\mathbf{X}, \text{diag}(\alpha)^{-1}\mathbf{X}^T$). Generally, the parameters $\boldsymbol{\alpha}$ and σ^2 are the tunable hyperparameters of this regression estimator.

The main advantage of the model is that it selects the most relevant features given the uncertainty of the regression parameters. However, it assumes Gaussian priors, which may not be suitable in all cases [38].

In our study, we utilize another estimator that makes use of Bayes' theorem — the Bayesian Ridge, which we describe next.

Bayesian Ridge

The Bayesian Ridge (BR) estimator is similar in nature to the ARD regression. Both estimators are designed to estimate the value of the target variable y by selecting the most relevant input features from the matrix of all features $\mathbf{X}$. The key difference is that BR imposes a single precision parameter λ, meaning that the same prior variance is imposed on all regression

coefficients θ [41].

In particular, the prior distribution of the regression coefficients is assumed to follow a zero-mean multivariate Gaussian prior, expressed mathematically in Equation 2.58.

$$p(\theta|\lambda) = \mathcal{N}(\theta|0, \lambda^{-1}\mathbf{I}) \tag{2.58}$$

where λ denotes the precision of the prior on θ. The conjugate Gamma prior controls the prior distribution of the precision, given by: $p(\sigma^{-2}) = \text{Gamma}(\alpha, \beta)$, where α and β are hyperparameters that can be tuned.

Again, following Bayes' theorem, we derive the posterior distribution of θ, which is proportional to the product of the likelihood and the priors [42], as shown in Equation 2.59.

$$p(\theta|y, \mathbf{X}, \lambda, \sigma^2) \propto p(y|\mathbf{X}, \theta, \sigma^2)p(\theta|\lambda) \tag{2.59}$$

with $p(y|\mathbf{X}, \theta, \sigma^2) = \mathcal{N}(y|\mathbf{X}\theta, \sigma^2\mathbf{I})$ denoting the likelihood of the target variable, conditioned on θ and the noise term σ^2. Therefore, the regression coefficients θ are assumed to follow a Gaussian distribution, given in Equation 2.60.

$$p(\theta \mid y, \mathbf{X}, \lambda, \sigma^2) = \mathcal{N}(\theta \mid \boldsymbol{\mu}, \mathbf{S}), \tag{2.60}$$

where the mean is given by $\boldsymbol{\mu} = \sigma^{-2}\mathbf{S}\mathbf{X}^T y$ with the covariance defined as $\mathbf{S} = (\lambda\mathbf{I} + \sigma^{-2}\mathbf{X}^T\mathbf{X})^{-1}$.

The posterior allows us to estimate the marginal log-likelihood of the target variable, given the input feature matrix, precision, and noise, as shown in Equation 2.61.

$$\log p(y \mid \mathbf{X}, \lambda, \sigma^2) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log\left|\sigma^2\mathbf{I} + \lambda^{-1}\mathbf{X}^T\mathbf{X}\right| - \frac{1}{2}y^T\left(\sigma^2\mathbf{I} + \lambda^{-1}\mathbf{X}^T\mathbf{X}\right)^{-1}y \tag{2.61}$$

where π is the mathematical constant, which is part of the Gaussian distribution. Therefore, in Bayesian Ridge, the regression parameters are estimated by the mean of the posterior distribution, as shown in Equation 2.62.

$$\hat{\theta} = \boldsymbol{\mu} = \sigma^{-2}(\lambda \mathbf{I} + \sigma^{-2}\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y \tag{2.62}$$

which means that the BR estimator assumes equal contribution of each feature to the target variable [41]. This suggests that the BR estimator may not perform well when only a subset of features is informative of the target variable.

Our subset of Machine Learning estimators also includes non-parametric estimators, of which the K-Nearest Neighbours is the example, described next.

K-Nearest Neighbours

K-Nearest Neighbours (KNN) is one of the non-parametric regression methods we used throughout this study. Non-parametric means that it does not rely on coefficients and makes predictions based on distances between data points [43].

The KNN estimator has several important hyperparameters. One of them is p, which affects the type of distance measure in Equation 2.63 [44].

$$L^p(x_j, x_q) = (\sum_i |x_{j,i} - x_{q,i}|^p)^{1/p} \tag{2.63}$$

where we measure the distance between two data points, x_i and x_j, with $i \neq j$, raised to the power of p. This function is often referred to as the Minkowski distance [44], as depending on the value of p, we get the Manhattan distance for $p = 1$, the Euclidean distance for $p = 2$, or other types of distances as the value of p increases.

Therefore, for a regression problem, for a data point in the target vector y_i, we compute the distance from observation i to every data point in the feature matrix $\mathbf{x}_i = [x_1, x_2, ..., x_N]^T$ and store these distances. This is done using a function $\mathcal{N}_k(x)$, where k denotes the number of nearest neighbours to consider. In other words, $\mathcal{N}_k(x)$ is a function that returns the indices of the k closest data points to y in the L^p space. Consequently, we can approximate the target vector y using the function given in Equation 2.64.

$$\hat{f(x)} = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i \tag{2.64}$$

where k is a tunable hyperparameter. To avoid pairing features with a target at specific observations, the value of k is typically odd, i.e., $k = 1, 3, 5, \ldots$. This prevents a regression from assigning equal distances to each data points. In other words, the odd number will always result in a clear winner. Notice that k controls the fit: a smaller value of k provides more flexibility, as it will look up distances between fewer features, while a larger value of k makes the estimator less flexible. For instance, if $k = n$, where n is the total number of observations available for training, the model would essentially predict the mean of the target values. In such a setting, the estimator becomes a constant, where $\hat{f(x)} = \frac{1}{n} \sum_{i=1}^{n} y_i$, representing the mathematical mean of all y_i values at each x [45].

To prevent such behavior, we typically apply a distance-based weighting during the estimation process. Thus, our estimation procedure takes the form outlined in Equation 2.65.

$$\hat{f(x)} = \frac{\sum_{i \in \mathcal{N}(x) w_i(x) y_i}}{\sum_{i \in \mathcal{N} w_i(x)}} \tag{2.65}$$

where $w_i(x)$ is the weight function, which also represents a tunable hyper-parameter. The weighting can be done in two ways: uniform or distance-based. 'Uniform' means that we scale

the appropriate data point x with the number of k, as illustrated in Equation 2.66.

$$w_i(x) = \begin{cases} \frac{1}{k}, & \text{if } \ i \in \mathcal{N}_k(x), \ \ i = 1, ..., n, \\ 0, & \text{otherwise} \end{cases} \tag{2.66}$$

The second option is to assign a weight to a value of x based on the resulting distance measure, L^p, as provided in Equation 2.67.

$$w_i(x) = \frac{1}{L^p(y_i, x_i) + \epsilon} \tag{2.67}$$

with ϵ being a small value necessary to avoid zero-division.

KNN estimator is effective in data sets characterized by the presence of noise [44]. At the same time, because the number of k-neighbours is tunable, one might set it too high, leading to poor generalization. We next consider a branch of tree-based estimators, starting with Decision Trees outlined next.

Decision Trees

Tree-based structures are one of the fundamental structures in Computer Science. These are commonly used to partition the data in various types of problems [46]. As such, we can utilize tree structures in order to solve regression problems of the kind we study in this research. Specifically, the Decision Tree (DT) estimator is commonly applied for these tasks.

Specifically, our input feature matrix $\mathbf{X}$ and output target values y can be split into subsets based on some threshold t, specifically:

$$S_{\text{left}} = \{(x, y) | x_j \leq t\}; \quad S_{\text{right}} = \{(x, y) | x_j > t\} \tag{2.68}$$

for a data point x_j. In other words, we recursively divide the feature space into subspaces conditioned on the threshold parameter t. For regression-type problems, t is estimated as the impurity of the split [47]. For a subset S, we first compute the variance, defined in Equation 2.69.

$$Var(S) = \frac{1}{|S|} \sum_{i \in S} (y_i - \bar{y_S})^2 \tag{2.69}$$

where $\bar{y_S} = \frac{1}{|S|} \sum_{i \in S} y_i$ is the mean of the target values in subset S. This can then be used to calculate the MSE of the split, weighted by the variance Var of S, defined in 2.70.

$$MSE(S_{left}, S_{right}) = \frac{|S_{left}|}{|S|} Var(S_{left}) + \frac{|S_{right}|}{|S|} Var(S_{right}) \tag{2.70}$$

which is the objective function of the DT estimator. For the unseen set of features, x^*, the DT estimator will traverse the tree constructed during training to find a node with the minimum MSE result. Therefore, a prediction made by the DT estimator is the average target value at that node of the tree closest to the unseen input data. This is mathematically depicted in Equation 2.71.

$$\hat{y*} = \frac{1}{|S_{leaf}|} \sum_{i \in S_{leaf}} y_i \tag{2.71}$$

The overfitting is typically prevented by the maximum depth and the minimum number of data points/features per node. These are the tunable hyperparameters of the DT estimator. In problems where data is in limited supply, any tree could potentially overfit to the training data [48]. One of the possible ways to prevent it is to fit several trees — a procedure accomplished by the Random Forest estimator, considered next.

Random Forest

As its name implies, Random Forest (RF) is a collection of Decision Trees, each fit on different subsets of data [49]. It makes use of bootstrapping — a technique used for generating multiple data set samples: $D = x_{i,1}, x_{i,2}, ..., x_{i,n}$ with uniform probability $i_j \sim \text{Uniform}(1, 2, ..., n)$. For the purposes of regression, each underlying tree is fit with a subsample D, with the objective function outlined in the previous subsubsection.

The prediction in RF is the average of the predictions from all trees (thus from all respective subsamples), as illustrated in Equation 2.72.

$$\hat{y*} = \frac{1}{T} \sum_{t=1}^{T} \hat{y}_t \tag{2.72}$$

where $t = 1, \ldots, T$ is the total number of trees constructed during the training stage, and $\hat{y}_t$ is the t-th tree's prediction regarding the target variable.

To further increase the robustness of the forest, the Out-Of-Bag (OOB) error is used. This is a technique where a model is evaluated on a sample not used during the training stage [44]. This consists of a few steps. First, for each (x_i, y_i) sample, yield an estimate only using trees that were fit without x_i in their sample, as per Equation 2.73.

$$\hat{y}_{OOB} = \frac{1}{T_i} \sum_{t \in T_i} \hat{y}_t, T_i = \{t | x_i \notin X_b^{(t)}\} \tag{2.73}$$

where T_i denotes the i-th tree in a forest T, $D := X_b^{(t)}$ is a bootstrap sub-sample of the training matrix X split into b parts for every t-th tree.

The second step involves the calculation of the OOB score across every sample, defined as the MSE for regression-type problems, as indicated in Equation 2.74.

$$\text{OOB Error} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_{\text{OOB},i})^2 \tag{2.74}$$

While the idea is simple, it is not without a flaw: our sparse data sets are split into further (and smaller) samples, therefore each tree is constructed with an even fewer number of data observations. This could lead to overfitting. Next, we move to the final estimator used in our study: the Multi-Layer Perceptron regression.

Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is the final estimator we use for the purposes of this study. Generally, the MLP estimator is an artificial neural network, which consists of several layers of interconnected nodes, with each performing a weighted summation with an activation function [23].

MLP is characterized by the input layer, hidden layers, and the output layer. Each layer has a certain number of interconnected nodes. The input layer simply accepts the features matrix and propagates it to the next layers. The hidden layer performs the computation of the form outlined in Equation 2.75.

$$\mathbf{h} = f_1(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) \tag{2.75}$$

where $\mathbf{h}$ denotes the output of a hidden layer, $\mathbf{W}_1$ denotes a matrix of weights assigned to each feature in the input matrix $\mathbf{x}$, $\mathbf{b}_1$ is the vector of biases computed at the hidden layer, and f_1 denotes an element-wise activation function.

Activation function is a tunable parameter. In this study, we choose the appropriate activation function for a specific data set during the hyper-parameter tuning. Appropriate,

in the context of our problem, is the activation function that yields the lowest error on the validation subset of data. Specifically, one of the several options is chosen. The Rectified Linear Unit (ReLU): $f_1(z) = \max(0, z)$, which outputs either the input or zero. Another option is the Sigmoid activation function: $f_1(z) = \frac{1}{1+\exp^{-z}}$, which maps the input to a small number in the zero-one range. The final activation function is the Tanh: $f_1(z) = \frac{\exp^{z} - \exp^{-z}}{\exp^{z} + \exp^{-z}}$, which introduces the sigmoid-like activation but in the -1 to 1 range. These activation functions allow the MLP estimator to derive non-linear patterns as data is being propagated through the network [23].

In the MLP estimator, a prediction can mathematically be described as in Equation 2.76.

$$\hat{y} = \mathbf{W}_2\mathbf{h} + b_2 \tag{2.76}$$

where $\hat{y}$ is the output prediction, and the rest of the variables are the same as represented in Equation 2.75. Neural networks generally consist of several layers. Equation 2.76 then illustrates the final (output) layer in the network. Because this is an estimator, it aims to minimize the regression error, which we set to be the MSE. The training process takes several steps (epochs), where at each step, we minimize the MSE loss with respect to the regression parameters. Specifically, at every epoch, we update the parameters as outlined in Equation 2.77.

$$\begin{aligned}
\mathbf{W}_1 &\leftarrow \mathbf{W}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}, \\
\mathbf{W}_2 &\leftarrow \mathbf{W}_2 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}, \\
\mathbf{b}_1 &\leftarrow \mathbf{b}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{b}_1}, \\
b_2 &\leftarrow b_2 - \eta\frac{\partial\mathcal{L}}{\partial b_2}.
\end{aligned} \tag{2.77}$$

where the operation $\leftarrow$ denotes an update step, and η is the learning rate — a tunable hyper-parameter. According to Equation 2.77, we take the derivative of the loss function $\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, MSE in our study, with respect to each element over the course of training. Generally, as long as the this function is differentiable and provides the convergence to the target problem, any other loss function is possible. All derivatives are outlined in Equation 2.78.

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{W}_1} &= -2\sum_{i=1}^{n}(y_i - \hat{y}_i)\,\mathbf{W}_2^T\sigma_1'(\mathbf{W}_1\mathbf{x}_i + \mathbf{b}_1)\mathbf{x}_i^T, \\
\frac{\partial \mathcal{L}}{\partial \mathbf{W}_2} &= -2\sum_{i=1}^{n}(y_i - \hat{y}_i)\,\mathbf{h}_i^T, \\
\frac{\partial \mathcal{L}}{\partial \mathbf{b}_1} &= -2\sum_{i=1}^{n}(y_i - \hat{y}_i)\,\mathbf{W}_2^T\sigma_1'(\mathbf{W}_1\mathbf{x}_i + \mathbf{b}_1), \\
\frac{\partial \mathcal{L}}{\partial b_2} &= -2\sum_{i=1}^{n}(y_i - \hat{y}_i)\,.
\end{aligned}
\tag{2.78}
$$

This process is also known as the backpropagation [23] and is used to minimize the regression error. The process of error minimization is repeated for several epochs — until convergence or some other stopping criterion.

Because of the number of computational steps, the MLP estimator is probably the most complex in our selection of estimators. One of its benefits is the embedded non-linearity in the form of the activation function. At the same time, because of the continuous minimization, it requires a high number of observations to avoid overfitting the training data.

The final methodology we consider in this study — Transfer Learning — allows us to apply inferences gained from one domain (or one data set) to another domain. We consider this methodology in more depth in the next section.

2.4.3 Transfer Learning

For the statistical inference to be successful, it is important that the training data follows the same distribution as the unseen data [20]. This assumption is, however, frequently violated in the real world [50]. This is especially important when the training series is limited in the number of observations.

Transfer Learning helps to prevent this problem. Effectively, knowledge transfer tackles two issues. First, it is effective when labeled data is in limited supply. In this case, one could fit a model on the data domain that has a similar pattern and apply it to the problem where the number of observations is limited. Second, it is useful when the distribution is easily outdated. In this case, learning from different distributions is effective at minimizing errors [50].

Transfer Learning is defined by several key components. The first is the source task: the data set with a sufficient number of data points: $\mathcal{D}S = (xi^S, y_i^S)_{i=1}^N$. The purpose of the source task is to minimize the regression error using the parameters from the regression fit in the source domain, f_θ, defined in Equation 2.79.

$$\mathcal{L}_S = \frac{1}{N_S} \sum_{i=1}^{N_S} \ell(f_\theta(\mathbf{x}_i^S), y_i^S) \tag{2.79}$$

where x is the vector of observations for i, y is the target variable, and N denotes the number of observations available for training in the source domain S. The objective at this stage is to pick a set of parameters that minimize the loss function: $\theta_S = \arg\min_\theta \mathcal{L}_S$.

The optimized parameters are then used in the target task domain. Specifically, for a target domain data set $\mathcal{D}T = (xj^T, y_j^T)j = 1^K$, we apply a model $f\theta_S$ fitted with the domain data set $\mathcal{D}S$. In order to perform the knowledge transfer — the key feature of the transfer learning approach — we fine-tune a model, keeping at least a portion of the parameters θ_S untouched.

Our objective at this stage is to minimize the loss function in the target domain, mathematically expressed in Equation 2.80.

$$\mathcal{L}_T = \frac{1}{K} \sum_{i=1}^{K} \ell(f_\phi(\mathbf{x}_i^K), y_i^K) \tag{2.80}$$

where K denotes the number of observations available for the target space and ϕ denotes the tuned coefficients for the target domain θ. Our objective is to minimize the loss: $\phi = \arg\min_\phi \mathcal{L}_T$.

The benefit of transfer learning, for the purposes of this study, is that it allows for the transfer of knowledge from one domain to another, bypassing limitations caused by the lack of training data observations. At the same time, the source domain could differ too much from the target domain; therefore, the knowledge gained in one domain is not transferrable to the other [51].

Because in multiple parts of this document we compare transformation, data augmentation and estimators comparative analysis, we need error measurements that will reveal how a particular methodology performs in comparison to other. These error measurements are considered in the next section

2.5 Error Measurements

In this study, we use several error measurements, commonly applied to problems of regression type. The choice of several metrics is necessary by the fact that every error measurement has its flaws and benefits. We therefore try to cover these limitations and get an objective measure about the results of our experiments. The first metric is the Mean Absolute Error (MAE), depicted in Equation 2.81.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.81}$$

where MAE is the output error for a regression, $i = 1, ..., n$ is the index for a data point with n denoting the total number of data points an error is measured with, y_i is the true, genuine data point and $\hat{y}_i$ is the estimate of the genuine data point inferred by a model-type.

MAE measures errors in the same units as the data. Specifically, it does not differentiate between cases when estimated value was much larger/smaller than the actual data: all errors make equal contribution to the final measure, regardless of their sign.

While it is convenient to measure the error in the same units as data, it may lead to inconsistent and therefore incomparable results, as some companies may have very large EPS/FCF values (in 1000s). Visually, it will appear as if the error is very large in these data sets, while it may not be, in relative terms. To account for these defects of the MAE error measurement, we use the Mean Absolute Percentage Error (MAPE), provided in Equation 2.82.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i - \hat{y}_i}{y_i}| \tag{2.82}$$

where MAPE is the output we measure, i is the index of total n number of data points in a measurement set, y_i is the genuine i-th observation with $\hat{y}_i$ being the value of i-th data point inferred by an estimator.

Unlike MAE, MAPE errors are relative. The relativity is due to the fact that each residual (difference between a genuine and an estimated data value) is scaled by the actual data observation. However, in some cases, the actual data observation may be near or very close to zero; therefore, the final result could explode, leading to a high output value, not representative of the actual difference between the actual and estimated data values. Therefore, the output

of the MAPE computation is not bounded and can be misleading if not incomputable, in cases the division by zero involved.

The third error measurement we use is the symmetric Mean Absolute Percentage Error (sMAPE), represented in Equation 2.83.

$$sMAPE = \frac{100}{n} \sum_{i=1}^{n} \frac{|\hat{y_i} - y_i|}{|y_i| + |\hat{y_i}|} \tag{2.83}$$

where, as previously, y_i and $\hat{y_i}$ are the actual and inferred i-th data observations with $i = 1, \ldots, n$ the index, where n is the total number of data points used for computing of the sMAPE score.

In our study, the sMAPE is the only error measurement, bounded between zero and 100. Thus, this error measurement is helpful in evaluating overall results regardless of size of data points values.

Besides investigative work of looking at how different ML and SE regression model-types perform in sparse time-series data sets, we also aim to outline pipelines and algorithmic regression approaches for EPS and FCF forecasting to be used in the financial valuation models. Thus, we seek to compare data preprocessing techniques and regression algorithmic approaches.

In order to do this comparison, we used the Friedman test statistical procedure. The Friedman test is a non-parametric test, in a sense that it does not assume a specific probability distribution for the population from which the data is. The Friedman test works in the several steps. First, for every method under consideration, the average ranking is computed, where the lower ranking is indicative of a better performance. A methodology with the lowest average rank is then considered to be the 'control' methodology. To derive statistical significance, we

apply the Hommel's post-hoc test [52, 53]. Hommel's post-hoc test represents the pair-wise comparison of distribution of error metrics the 'control' (i.e. best) methodology against the rest. Particularly, for each pairwise comparison — and for each type of error separately, i.e. MAE, MAPE, and sMAPE — the null hypothesis to be rejected at a 5% level ($\alpha = 0.05$), states that the observed error ranks obtained during the Friedman ranking process are all sampled from the same continuous distribution.

Next section summarizes the background information outlined in this chapter.

2.6 Conclusion

When making a buy/sell decision, it is vital for a fundamental investor to determine the intrinsic value of a company. The intrinsic value is the approximate price worth paying for a stock given its future expected cash flows and risk. If the resulting intrinsic value is greater than the market price, the stock is worth buying, suggesting a buy decision; otherwise, sell.

To estimate intrinsic value, two valuation techniques are commonly used: the Discounted Cash Flows (DCF) and the Price-Earnings-Per-Share (PE) multiple. The former makes use of the Free Cash Flows (FCF) series, and the latter uses the Earnings-Per Share (EPS) series. Both are measures of cash flows to investors, with the key difference being the computation methodology: EPS subtracts all the accounting expenses, while FCF adds the non-cash expenses back.

Valuation methods, such as DCF and PE multiple, require an estimate of the input series for the upcoming quarter (FCF and EPS, respectively). Underestimating or overestimating the quarter-ahead value could lead to financial losses. Hence, we propose to formulate this as a regression problem where the target variable is the EPS and FCF values.

To solve this regression problem, we use a set of statistical and machine learning estimators. To tackle this challenge, we make use of the predictive power of Machine Learning (ML) and Statistical Estimator (SE) methods for these series. For our purposes, the operational distinction between ML and SE methods here is that the former make use of regularization to improve model generalization where appropriate, whereas the latter do not. Regularization is the process of favouring solutions that are more 'regular' in some useful sense over 'irregular' ones; the particular choice of regularization (i.e., what it means for a solution to be more 'regular') reflects, to some extent, prior knowledge regarding the problem domain, which is directly injected into the training process of the model in question. As a result, the use of regularization prevents ML models, to some extent, from producing solutions that are 'too irregular' in a manner that would make the model more prone to overfitting; however, this often comes at the expense of requiring a somewhat higher volume of data to train adequately. SE models, on the other hand, are completely data-driven, making no assumptions about the 'regularity' of potential solutions (other than the models themselves being expressed in some useful parametric form), and thus do not involve any regularization coefficients, which might require the model to undergo further training to account for their effect.

The technical challenge of our study is due to the sparsity of the series under consideration. Because the data is published on a quarterly basis, as required by law, the target values of both EPS and FCF series are limited in the number of observations. To mitigate the effects of sparsity, we propose applying interpolation techniques that allow us to approximate a value between two existing data points. This enables us to assume a different data stream frequency: we assume weekly and monthly arrivals in both EPS and FCF series. Another way to address this challenge is to use the transfer learning methodology. Transfer learning allows us to train a model in the source domain — one with a sufficient number of labeled observations — and

apply it to another domain — one with a lack of data observations.

To evaluate the validity of our results we use three regression error measures: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and symmetrical Mean Absolute Percentage Error (sMAPE). These three metrics measure the absolute distance between the genuine and inferred with an estimator data points, reporting results in different units. The MAE reports in the same units as the data, the MAPE reports in percentages, without any upper boundary and sMAPE reports error in the 0-100 range. For all three of those measures, the lower value indicates a better fitness of estimator to data.

The next chapter gives an overview of state-of-the-art approaches to the regression problem outlined in this chapter.

Chapter 3

Literature Review

3.1 Introduction

In this chapter, we conduct a literature review of the state-of-the-art methodologies for approximating financial time-series data. While this study primarily focuses on the EPS and FCF series, other types of financial data have been the subject of numerous studies that we consider in this chapter. Our interest in the EPS and FCF series stems from the fact that the two are used to estimate the intrinsic value of a stock, which, compared to the stock price, outlines the reasonable price an investor should pay for a stock based on its future prospects and risk.

From this definition, our study aims to outline a method for forecasting future prospects. We define this forecasting task as a regression estimation problem, where the objective is to minimize the distance measure between the target values and the model's predicted values. Specifically, we assume that both target series depend on their past lags, mean, and standard deviation.

The finance literature emphasizes the usefulness of machine learning (ML) algorithms in

addressing classification and regression problems [1, 54]. Typical regression tasks involve predicting revenues and earnings, as these variables are indicative of future stock price performance. Classification tasks, on the other hand, often focus on estimating the likelihood of significant events, such as the probability of bankruptcy for companies.

The purpose of this chapter is to outline the approaches and challenges associated with forecasting financial time-series data.

To achieve this, we first introduce novel techniques used in modeling financial and economic time-series, including both pricing and fundamental data in Section 3.2, followed by Section 3.3 which narrows the review down to the fundamental financial time series forecasting. Subsequent Section 3.4 gives an overview of techniques utilized to model sparse series, including those outside of financial domain. Finally, Section 3.5 summarizes the information given in this chapter.

3.2 Financial and Economic Forecasting

Time-series modeling has been a central topic in numerous studies [55], spanning applications such as electricity demand forecasting [56], retail decision-making [57], and finance [58]. As noted earlier, financial time-series data can generally be divided into two categories: pricing data and fundamental data. The primary distinction between these categories lies in their publication frequency, with pricing data typically reported daily and fundamental data reported quarterly. Both types of data have been subjects of scientific interest [59], [60].

The surveyed publications differ in the algorithmic approaches applied. For instance, Box et al. [61] provides the thorough description of the widely used quantitative methods, such as Maximum Likelihood Estimation methods, examples of which are the stochastic models,

including univariate and multivariate settings. Typical examples of such models are Integrated Moving Average (IMA), Auto-Regressive Integrated Moving Average (ARIMA), Vector Auto-Regressive Models (VAR), etc. The book outlines the common approaches to time-series analysis and modeling. A study by [62] showed the ARIMA method can be successfully applied to model Netflix Inc., stock price. In their study, they examined how the number of past lags and moving average components affects the Mean Absolute Percentage Error (MAPE) score. In particular, their study achieved 99.75% accuracy, showing almost a perfect fit to the data, successfully predicting the next three months of stock movements (daily percent returns) on the handout data set. Paper by [63] did a similar study, introducing the Back-Propogation Neural Network and also expanding the number of data sets up to two European Stocks (JD inc., and PDD Holdings Inc.). In particular, their study which utilized around 600 training data points and 10 test data observations, showed that ARIMA and the Neural Network produce similar results, with ARIMA showing a slightly better out-of-sample performance.

Exponential smoothing is another approach utilized by scholars. In particular, study by Gardner et al., [37] introduces the Exponential Smoothing model and outlines common cases for its use. Specifically, it is acknowledged that the model is more complicated than the standard ARIMA in terms of explainability and the number of computational steps, however, it is noted that the Holt-Winters variant of exponential smoothing is more effective at capturing seasonality and other recurring patterns in the time-series data. Nevertheless, a book by Andrew C. Harvey [64] examined a broad range of exponential smoothing and other structured models, outlining a wide spectrum of time-series forecasting tasks, mostly focusing on macroeconomic data. This illustrates that by capturing the general trend in data (which the majority of smoothing models aim to accomplish) it is possible to accurately form expectations regarding the future Gross Domestic Product growth in the UK and US economies, rainfalls in Brazil and even the

number of car crashes in Chicago, US.

A more recent set of experiments documented by Shukor et al., [65] demonstrated how various types of exponential smoothing, linear trend and random walk models can be applied to estimate the market price of gold, crude oil and other precious metals. Of the proposed methods, the Holt's Linear Trend exponential smoothing variant demonstrated the lowest regression error, measured in their study as the Root Mean Squared Error, Sum Squared Error and Mean Squared Error, when predicting the crude oil and platinum prices. In their study, they used approximately 200 data points for model training, predicting the next 10 data points. They noted that the data is remarkable in high number of fluctuations, hence smoothing these fluctuations is beneficial for the estimation of the future prices of these commodities.

The interest in financial time-series modeling extends beyond the use of a single estimator or the comparison between various estimators. Using the Bayesian Model averaging which allows approach to fit different variants of LASSO estimator to the pricing data of a set of financial firms, over the time span of 1990s to 2017. Additionally, a number of external indicators typically used to detect the economic recession were used as the input data to the LASSO estimators. It is concluded that by examining the weights in the Bayesian Weighted average approach across different models, it is possible to establish which factors (input vectors representing signals of recession) contribute to the volatility the most.

One of the foundations of economic theory is the notion of uncertainty [66]. Broadly, the uncertainty is the spread of potential outcomes. In technical terms, we can formulate the past stock returns as the distribution, where, on most future days, the return will be within a predefined, measurable spread. The uncertainty can also be modeled, commonly with Bayesian methods.

As an example of such, the paper by Bertschinger et al., [67] proposes to model volatility

in the agent-based economic simulation, with the use of GARCH and Bayesian estimation methods. This paper states that the simulation performs in par with the economietric methods, such as GARCH and its variants, albeit there are cases where the simulation requires a more sophisticated simulation techniques to properly estimate the volatility levels.

Further, Bayesian methods were applied for investigative purposes. A notable publication by Takyi et al., [68] utilized the Bayesian structural models for examination of the impact of COVID-19 on African stock returns. As expected, the majority of stock markets were significantly and negatively affected by the pandemic, with only 3 out of 10 countries not experiencing significant impact. No markets gained positive returns with the declaration of the global pandemic.

The price of selected financial instruments, such as derivatives depends on the future price of an underlying commodity. A derivative is a type of contract that allows the future sale or purchase of a financial asset (such as a stock) at an agreed-upon price [69]. Therefore, to determine whether the price for a derivative contract is 'fair' it is necessary to approximate the future price of such goods. Number of studies have advanced this field. For instance, Krzysztof Drachal [70] attempts to estimate the future spot price of crude oil. In their study, they use the Bayesian Symbolic regression methodology compared against the LASSO, Ridge, ARIMA, dynamic, genetic and Bayesian model averaging techniques. Their objective is not only to propose a more accurate solution to the problem of estimating the future crude oil price, but also to outline the method that will incorporate more complicated, non-linear features set by experimenting with the symbolic regression in a Bayesian way. Their study concludes that the Bayesian Symbolic Regression, while not yielding most accurate results at a statistical significance, is still computationally faster than genetic benchmark methods.

Derivative instruments also exist for common stocks: these are called stock options, which

allow the owner of a option the right to buy or sell a stock in the future, at agreed upon price [71]. The study by Carverhill et al., [72], attempts to model the stock price dynamics and associated options pricing via means of Monte Carlo Markov Chains (MCMC) simulations. Their paper demonstrated that simultaneous increases in volatility are useful for explaining the time-series of the stock returns [72].

Another study, conducted by Gao et al., [73] an investigative work on the sensitivity of stock options to the underlying stock liquidity constraints. Liquidity in stock markets is defined as the ease of quickly converting a stock to cash [71]. The study shows that there is a significant impact of market liquidity on options' prices [73], clarifying the previously established assumptions imposed by the Black-Scholes Model, commonly used for stock options pricing [74].

A collection of stocks can be used as an index, which is representative of how the stock market performs in a particular geographic region. These indexes are often used to assess whether an economy or a particular segment is in a recession or experiencing economic upheaval. This has generated interest in estimating the future values of stock indexes. In this respect, focused on the long-term price levels of the global financial indexes, a study by Beniwal et al., [75] use a series of machine learning, namely Support Vector Machines, Long-Short Term Memory models and Genetic Algorithm to predict year ahead prices in the US, Japanese, German and Shanghai indexes. The study finds the SVM regression to outperform other regression methods.

Another study by Abraham et al., [76] also attempts to predict the daily trends of the global stock market indexes. In particular, they employ a Genetic Algorithm to select the set of international indexes and select a stock to be forecasted. A Genetic Algorithm is then employed to find which indexes are significant in forecasting the stock's trend. Finally, the

Random Forest regressor is applied to estimate the hidden relationships between the selected features and the stock's trend. It is shown that such approach yields up to 80% accuracy for the stock trend forecasting.

A paper by Arashi et al., [77] employs the ARMA and GARCH models to estimate the future NASDAQ stock market index returns. In particular, combining a model (ARMA) to estimate the trend and another (GARCH) to estimate the volatility yields a 1% error rate.

Generally, various machine learning methods show robust performance in predicting stocks and stock indexes returns in different geographic regions. As such, a number of studies aims to predict stock returns in China [78], [79], Europe [80], [81], and the Americas [82], [83].

Researchers also highlight the usefulness of machine learning estimators in high-dimensional data sets [84], [85]. High-dimensional data sets are characterized by the number of input variables exceeding the number of observations [84]. This is especially useful since modern capabilities allow to collect data from multiple sources and combine these entries into a single data set, as well as to use complex mathematical functions and transformations to exploit non-linear settings in regression problems. Thus, researchers focus on predicting options returns conditioned on their non-linear patterns [86], extracting useful features that minimize prediction error by transforming high-dimensional data sets [87], and exploring market efficiency with a large number of input variables [88].

Another way to extract information from the input features matrix is through transformations [22]. There are still doubts about whether transformers bring any benefits to the modeling process or if they cause model overfitting [89]. Nevertheless, transformers enable the modeling of pricing data with Convolutional Neural Networks [90] and the use of multi-attention mechanisms with linear models [91], both of which report more accurate prediction done with this approach, as compared to state-of-the-art approaches. There are cases where

transformers allow for faster data processing [92], thus minimizing the time it takes to predict a data point. The estimation speed is essential for high-frequency trading.

Importantly, modern computational techniques also enable the inclusion of non-quantitative data in the estimation process [93] as part of features set. Specifically, multiple studies report increased accuracy in the Long-Short Term Memory Gating model when sentiment from multiple text-based data sources is included as part of the features in the estimation process [94], [95], [96], compared to models fit with more traditional features, such as technical indicators and past lags.

Despite the successes mentioned earlier, predicting longer-term horizons remains a challenge for quantitative estimators [97], both statistical and machine learning [98]. Nevertheless, researchers have discovered ways to improve the reliability of longer-horizon financial predictions, such as using the extended Long-Short Term Memory algorithm [99] which outperforms the classical LSTM model when the prediction horizon widens. Additionally, data transformations, such as wavelet transformations, have also proven useful in increasing estimation accuracy under the extended forecasting horizons, against the collection of neural networks fit with data without the wavelet transformations [100].

Finally, Reinforcement Learning has been successfully applied to pricing data predictions [101], [102]. Specifically, it has been demonstrated that Reinforcement Learning can also be used for stock market forecasting tasks [103], volatility predictions [104], trading strategy optimization [105], and automation [106], [107], [108], options pricing [109], and hedging [110], [111].

Departing from the financial forecasting task, a text by Valle-Cruz et al., [112] explored how Twitter data impacted stock market performance during H1N1 and COVID-19 pandemics. They discovered that markets reacted to the news about pandemics within 10-15 days.

A more complex task of analyzing financial documents is accomplished by Xie et al., [113]. In particular, they propose the Large Language model tailored for performing crucial financial tasks such as financial documents analysis and interpretation.

From this section, we see that there is an increased interest in applying the various machine learning approaches to the financial data, not only to predict a value in the future but also to understand the potential volatility levels, understand a relation between various quantitative and qualitative variables in the financial markets, and automate trading strategies. As we mentioned earlier, the pricing data, which was the subject of studies mentioned in this section, is published on the daily and more frequent basis. This differs from the fundamental financial data, published quarterly, subject of studies explored in the next section.

3.3 Fundamental Financial Variables Forecasting

It is acknowledged that the reports disclosure affects the stock prices in a large way. This data is vastly used for financial valuations. However, due to the fact that the financial data is published quarterly, and the organized data collection started in the 1980s [6] the typical data set will be limited in the number of records. Because of the limited data size, investors tend to look for ways that do not involve quantitative modeling. For instance, investors usually refer to the projections made by financial institutions and investment banks. The benefit of using these estimates is that analysts tend to have more insights into the state of a business [5]. A study by [114] showed that these estimates indeed outperform quantitative time-series models, due to information that cannot be incorporated into the model.

Nevertheless, such projections are akin to a black box and are often biased [115]. Specifically, the source of bias is the agency-type problem: pessimistic reports tend to worsen the

client-analyst relationship, and analysts also tend to lose interest in companies that have seen several quarters of earnings below expectations [114].

Another popular approach is to use financial variables or the past historical average growth rate [8]. However, this ignores external factors such as economic recessions or seasonal drops in consumer demand. Moreover, since companies evolve over time, their growth tends to decrease as a company matures [8], and hence, the past growth rate can end up being a poor estimator of future performance.

Numerous studies have reported robust performance from quantitative forecasting models, such as ARIMA [116]. However, such models usually make predictions based on a set of assumptions, which financial data tends to violate over longer periods of time [30]. And while there has been increasing interest in the use of ML algorithms, which avoid the need for stationarity and linearity assumptions [23], such research has mainly focused on forecasting earnings per share [117], revenues [118], and cash flows [119].

The predictability of EPS is a popular question in scientific and professional communities, given the fact that earnings announcements (i.e., the disclosure of a company's EPS) tend to have a significant impact on stock returns [120]. Several studies have attempted to predict future changes in EPS using other accounting variables. Specifically, [121] used the Economic Value Added (EVA) financial measure to fit an ordinary least squares (OLS) model. The focus of their study is on the relationship between EVA and EPS changes, rather than the performance of the OLS model. The study reveals that out-of-sample OLS predictions tend to be more accurate than analysts' estimates.

Additionally, an earlier paper by Brown et al., [122] showed that four different variations of the Box-Jenkins model, varying in the number of lags, outperformed analysts' estimates on a quarter-ahead basis, even without the inclusion of other accounting variables.

More recently, a study by Kureljusi et al., [118] evaluated a selection of machine learning algorithms — namely Decision Trees, Random Forests, Neural Networks, XGBoost, Gradient Descent Linear Regression, LightGBM, and CatBoost — to check if these algorithms are capable of producing more accurate results over time, benchmarking them against expectations data available at I/B/E/S (Institutional Brokers' Estimate System). Their conclusion was that ML models provided better or comparable results to those of analysts. A study by Easton et al., [123] showed that K-Nearest Neighbours regression successfully captures a year and three-years ahead EPS values, as compared to market analysts.

Another study, conducted by Cao et al., [124] showed that Backpropagation Neural Networks (MLP) and Genetic Algorithm models outperform the OLS and Auto-Regressive Integrated Moving Average (ARIMA) regressions, in both univariate and multivariate cases. Additionally, Fisher et al., [125] compared the ARIMA and Support Vector Machine (SVM) regression models under different scenarios, including multiple prediction horizons (two and three quarters ahead) and the limited availability of historical data for model training. They demonstrated that, in most cases, the SVM model performs better than ARIMA in a univariate setting. A study by Jadhav et al., [126] showed that OLS regression generally performs well across their sample of companies, but fails when the data exhibits highly non-linear patterns, where MLP outperforms.

The forecasting of business fundamentals and earnings per share has also been approached as a classification problem. For instance, paper by Chen et al., [127] used Random Forests and Stochastic Gradient Descent Boosting to predict year-ahead directional changes in corporate earnings. Ensemble learning models demonstrate robust performance on out-of-sample data, primarily due to their ability to automatically select significant variables from high-dimensional data. Another study by Jiang et al., [128] developed a tree-based technique called TreeNet

to estimate the probability of bankruptcy in Chinese public companies using financial ratios, valuation multiples, macroeconomic variables, and other factors. The model achieved an accuracy of over 90% in predicting bankruptcy.

The applicability of ML algorithms to fundamental company valuation was studied in paper by Vayas-Ortega et al., [3]. In their work, they examined two components of the DCF: WACC and the FCF growth rate. To predict the latter, they used a linear regression model with detailed data from balance sheets and income/loss statements, polynomials denoting past FCF lags, and selected macroeconomic variables. Although the study did not consider benchmark models, their research shows that linear regression-based valuation models produce less biased estimates of intrinsic value compared to market analysts. Study by Anand et al., [129] focused on the directional changes of several enterprise profitability measures, including the FCF growth rate. In their study, they trained multiple Random Forest Classification algorithms, adding one feature for each successive training iteration. The results show that ML algorithms can predict the directional changes of future FCF growth rates with 55-67% accuracy, compared to 50% accuracy from the Random Walk model on out-of-sample data.

We acknowledge that studies cited in this section performed experiments similar to the ones conducted in this thesis, albeit they pursue rather different purposes. For example, [3] focus on performing fundamental valuation with predictions made for both the FCF and WACC. Additionally, they experimented with different input variables using only Linear Regression as the forecasting model. Other studies, including Kureljusic et al., [118], Easton et al., [123] and Fisher et al., [125] are focused on forecasting with a limited selection of regression models, benchmarking those against market analysts.

Therefore, in this study, we address a gap in the fundamental financial forecasting literature by examining how the accuracy of Earnings Per Share (EPS) and Free Cash Flow (FCF) forecasts

impacts firm valuation. Specifically, we investigate the relationship between the forecasting of these two key financial variables, fundamental valuation, and portfolio performance. We assess whether predictions generated using transfer learning, machine learning (ML), and statistical estimators (SE) — each with different regression errors—provide measurable benefits to the fundamental investment decision-making process.

Because the series we predict in this study are characterized by limited number of observations, rather than increasing the number of regression input variables as was done in papers by Cao et al., [124], Easton et al., [123], Vayas-Ortega et al., [3], we tackle the problem of sparsity by augmenting data sets with interpolation and linearizing those with transformation. Similar methods were successfully applied to other types of sparse series in studies summarized in the next Section.

3.4 Other Sparse Series Forecasting

A number of transfer learning techniques have been applied to different kinds of financial data. Specifically, experiments by He et al., [130] showed that training an Artificial Neural Network to predict the price of a stock, using data from another, positively impacts the overall predictive accuracy. A further study by Cao et al., [131] revealed that transfer learning techniques can be successfully applied to portfolio optimization problems.

Study by Zhang et al., [17] showed that transfer learning can be applied to stock recommendation systems, tailored for different products, where part of the feature set includes fundamental stock data. Another, done by Laptev et al., [132] proposed a pre-trained LSTM-based Neural Network with a new type of loss function. Their methodology shows robust performance in sparse time-series data over state-of-the-art time-series forecasting methods,

based on Pacific Gas & Electric energy demand data sets. The paper by Christinaki et al., [133] successfully applied Transfer Learning and Bayesian Model Averaging to predict the well-being of patients through wearable devices. Particularly, they successfully used this methodology to forecast a patient's well-being even over longer-term horizons. Similarly, study by Zhang et al., [134] used Bayesian Model Averaging to average predictions made with several ResNet pre-trained models, with different hyperparameters. Their study, which proposes a new way of forecasting residential load data, shows that Bayesian Model Averaging with incorporated Transfer Learning outperforms other methods, given the low availability of training data.

Interpolation is one additional way to generate more time-series data. A study by Oh et al., [135] evaluates a number of popular techniques for augmenting time-series data, including Jittering, Cropping, Scaling, Rotation, etc. [136]. The study concludes that cubic splines were helpful in training a deep learning neural network, regardless of the fact that the data showed changes in seasonality, trends, and other patterns that commonly affect the estimation process [135].

In particular, paper by Wang et al., [137] studied how different interpolation methods affect clustering outcomes in the task of predicting electric bus driving cycles. Their results show that a mix of linear and cubic interpolation yields better results over benchmark methods, as it generates synthetic data that is closest to the genuine series. Additionally, the text by Challu et al., [138] proposed an interpolation technique that improves the accuracy of longer prediction horizons. This is achieved through multi-rate sampling and hierarchical interpolation. Their results indicate a 20% accuracy increase across various time-series datasets, including currency exchange rates, San Francisco traffic load, and electricity consumption.

More recent advances in Generative Artificial Intelligence allow researchers to generate synthetic data using Generative Adversarial Networks (GANs). For instance, paper by Smith

et al., [139] propose the Time-Series GAN, which successfully generates health-related data that closely resembles genuine samples. They demonstrated the benefits of their approach for classification tasks. Similarly, paper by Liu et al., [140] proposed several GAN models to generate synthetic data. The generator model is then used to make predictions on the series it has just synthesized. Several models they tested showed a 15-20% reduction in error rate compared to state-of-the-art predictions made with publicly available datasets.

From this section, it is observed that the two primary methods for training ML estimators are interpolation and transfer learning. Interpolation is primarily used to generate synthetic data, which is especially useful in the time-series context. In this context, slight deformations of a data observation, similar to those performed on images [22], can significantly impact the estimation process [135]. While the use of GANs is gaining popularity, it is acknowledged that these models require large volumes of data to generate realistically looking samples [23]. In the studies mentioned in this chapter, researchers had access to more than 3,000 data points, which is more than what is available on EPS and FCF data sets as there are only four data points produced annually, due to the fact that companies publish their reports on quarterly basis.

3.5 Conclusion

This chapter explores the financial time-series forecasting literature, with some papers revealing important findings on time-series modeling outside of the financial domain.

Our findings indicate an increased interest in the applications of various ML and statistical estimators to financial time-series forecasting problems. Predominantly, most studies are concerned with forecasting EPS values, as these have a direct impact on stock returns. Several

studies also consider the use of forecasting models for FCF series in order to perform financial valuations. In both cases, the lack of data observations is cited as a significant challenge in the estimation process.

Studies outside the financial domain, propose the use of transfer learning: a methodology whereby the information gained in one (source) domain is applied to problems in similar or other domains (tasks). Researchers dealing with this methodology cite the improved performance over state-of-the-art methods and propose it as a way around the sparsity challenge. Interpolation is another way to solve the lack of data observations. This allows generating synthetic data between the real observations. Researchers found it useful, especially when fitting deep learning models.

There are only limited papers that discuss the forecasting of the financial time series data with the aim of using those predictions as inputs to established fundamental valuation models: the Price-Earnings and the Discounted Free Cash Flows. In such papers, researchers have limited discussion on the impact of prediction accuracy on intrinsic value computation and further economic benefits to financial markets. Our study therefore attempts to cover this gap: we first explore the key financial series, EPS and FCF, using a diverse range of machine learning and statistical estimators to model those. Consequently, we study the impact that the proposed forecasting methods have on established valuation methods.

In the next chapter, we conduct analysis of our time-series data and the first set of experiments.

Chapter 4

Exploratory Analysis

4.1 Introduction

In this chapter, we conduct a comprehensive analysis of both types of our financial time-series. The purpose of this analysis is to select the appropriate data augmentation or transformation technique. First, we perform a set of statistical tests to better understand the nature of the series used in this study. Specifically, we use the Shapiro-Wilk (SW) normality test, which determines whether the series follows a normal (Gaussian) distribution. Second, we run the Augmented Dickey-Fuller (ADF) test to assess how many data sets in our selection are stationary. A data is said to be stationary if it has a constant mean and variance terms [141]. The final statistical test used is the Mann-Kendall (MK) Trend test, which identifies any monotonic upward or downward trends in the series.

Statistical tests are crucial for recognizing and understanding distributional moments of our data sets, as such tests inform the preprocessing steps required for effective data modeling. In our study, modeling is performed with a selection of statistical and machine learning regression estimators. Since both types of estimators are prone to overfitting when the number of

training observations is in limited supply, we also conduct experiments involving interpolation techniques. Interpolation allows to artificially increase the sample size while preserving the shape of the original data thus giving an estimator more information about the target-feature dynamics. While interpolation can help increase sample size, it may also introduce biases by smoothing out variance [142].

Another way to provide estimators with more information about the data is to scale or transform it, using feature scaling/transformation techniques. As our features represent past lags of the target series, it is expected that our data will be in the same units. At the same time, since companies tend to experience decreases in revenues and even business closures during economic recessions, seasonal decline in demand for their products, natural disasters, and other adversarial circumstances, it is possible that EPS/FCF values drop below zero. Therefore, in such cases, the scaling/transformation techniques can enhance the suitability of the series for ML/SE modeling [143] by enforcing the data to follow a particular distribution and converting all the data to unit variance.

For this chapter, we use Ordinary Least Squares (OLS) regression as a proxy for other forecasting model types. The OLS estimator was chosen because of the simplicity of interpreting its results. In this chapter, all experiments are carried out on the sample of 50 randomly selected data sets which meet the filtering criteria explained in Section 4.3 of this Chapter. We selected a subset of 50 companies to maintain objectivity. In this chapter, we establish a pipeline that will be used in later parts of the thesis. Using all datasets from the outset could have led to the criticism that we merely optimized the pipeline for the specific sample, thereby attributing any subsequent improvements to this tailored configuration rather than to the robustness of the method Information about these data sets, including the full business name and economic sector it operates in is given in Table A-1 in the Appendix, Section A.

The main contribution of this chapter lies in the outlined pipeline, which will be utilized in future chapters. Specifically, we provide the rationale behind the choice of specific interpolation methods and feature transformations. Additionally, we illustrate, using specific datasets, why certain techniques were effective or ineffective and how interpolation and transformation influenced the estimation process.

This chapter is structured as follows: Section 4.2 outlines the statistical tests and formulates the data preprocessing steps. Section 4.3 demonstrates the experimental setup, followed by Section 4.4 which provides the results of our experiments. Section 4.5 concludes this chapter.

4.2 Methodology

In this section, we outline the methodology pursued throughout this chapter, beginning with a description of the statistical tests used in the following subsection.

4.2.1 Statistical tests

As mentioned in the previous section, we use statistical tests to better characterize the properties of our datasets. The first of these is the Shapiro-Wilk (SW) normality test, which evaluates whether the data follows a normal (Gaussian) distribution. The null hypothesis is that the underlying data follows a normal distribution, with the alternative being a non-normal distribution. We reject the null hypothesis at a 5% significance level ($\alpha = 0.05$) and fail to reject it otherwise. This test will identify how many datasets are normally distributed, as many regression algorithms perform better when this assumption holds [22]. In a normal distribution, most data points cluster around the mean, allowing us to reasonably expect future values to remain near that average.

The second test used in this section is the Augmented Dickey-Fuller (ADF) test. The purpose of this test is to determine if a series in our selection is stationary or follows a deterministic trend. Specifically, the null hypothesis is that a unit root is present in the time-series sample. As with the SW test, we reject the null hypothesis at a 5% significance level ($\alpha = 0.05$) and fail to reject it otherwise. The presence of a unit root in the series means that the data is non-stationary — its mean and variance can change over time. The ADF test complements the SW test by examining whether the statistical properties of the series (mean and variance) remain constant over time. If the series is non-stationary, modeling becomes more difficult, as future values may shift unpredictably [30]. Hence scaling/transformation should mitigate such issues by enforcing the data to unit variance.

The third test used in this section is the Mann-Kendall (MK) test, which detects the presence of a monotonic trend — a consistent upward or downward pattern — in a time series [144]. The null hypothesis, which is rejected at a 5% significance level ($\alpha = 0.05$) is that the data are independent and randomly ordered. The alternative is that a monotonic trend is present.

The results of these statistical tests help explain why interpolation and feature scaling may improve prediction accuracy for certain datasets. For example, interpolation might smooth out heavy-tailed distributions caused by outliers, leading to better model performance. Notably, no preprocessing is applied before running these tests — we aim to observe the natural statistical properties of each series. In contrast, during interpolation and transformation experiments, the data is preprocessed according to the steps outlined in the next subsection.

However, when performing the interpolation and transformation experiments, the data is subject to specified changes. The data preprocessing step is directly linked to the forecasting approach we use in this study, outlined in next section.

4.2.2 Forecasting Approach

Financial time series forecasting involves approximating some future value of a target variable by means of regression. Therefore, in order to make a prediction, a regression model needs to first be fit with some training data, fine-tuned (in the ML context) and finally, a prediction can be gathered with some out-of sample data.

These steps can be performed in several ways. In most financial time-series forecasting studies, researchers use the rolling window. This process is similar to the stack data structure, whereby old data is being popped out of a stack and a new data is pushed. Specifically, a 'window' is the fixed number of data points used to train a regression model-type. 'Rolling' refers to the fact that after making a forecast, the oldest data is dropped and the most recent, future, data is added to the set [145].

This approach, while being widely popular with researchers and professional data modellers, is not appropriate for our experiments. It is due to the training data size: one of the largest, in terms of number of available observations from both Earnings Per Share and Free Cash Flows, data sets we have is no more than 140 data observations. Excluding the test set (20% of the total size) there is roughly 112 data points available for model training (and fine-tuning in case of ML estimators). Tables A-2 and A-3 in the Appendix, Section A, reveal the number of data points available for both EPS and FCF data sets.

Because we aim to utilize as many data points as possible for predicting the next quarter Earnings-Per-Share and Free Cash Flows values, we follow the expanding window approach, pseudo-algorithm for which is outlined in Algorithm 1.

In Algorithm 1, X and y are the input and target data, respectively, with train and test denoting the training and testing subsets, respectively. The Concat denotes the concatenation —

Algorithm 1: *Expanding Window Algorithm*

Input: Training data X_{train}, y_{train}, Testing data X_{test}, y_{test}, Estimator $f(X, y)$

Output: Prediction set $\widehat{y}$

for $i \leftarrow 1$ **to** *length*(y_{test}) **do**

  $f \leftarrow f(X_{\text{train}}, y_{\text{train}})$; `// Fit a regression model`

  $\widehat{y}_i \leftarrow f(X_{\text{test},\,i})$; `// Make a prediction for a test data point`

  $X_{\text{train}} \leftarrow \text{Concat}(X_{\text{train}}, X_{\text{test},i})$; `// Expand the training data`

  $y_{\text{train}} \leftarrow \text{Concat}(y_{\text{train}}, y_{\text{test},i})$

end

return $\widehat{y}$;

the process of including the i-th data observation from the test subset to the training, whereby the latter increases the number of data observations and thus the sample size. In other words, the training matrix continuously increases with the number of iterations through the test data set. Unlike the rolling window, the expanding window does not discard any past data points. As a result, our estimators have access to a longer history of a company's EPS and FCF records. Moreover, in real-world scenarios where data availability is limited, researchers are often inclined to utilize all available data — a behavior that the outlined algorithm is designed to mimic. In summary, our choice of forecasting algorithm is justified by two factors: our intention to replicate real-world forecasting conditions as closely as possible, and the scarcity of available data.

Algorithm 1 is the base version we use in this study. We conduct a number of experiments involving the data transformation and interpolation, hence the base Algorithm 1 is subject to changes, specified for the types of experiments we conduct. The changes we intend to

introduce are specified in the next section.

4.2.3 Data Preprocessing

The data preprocessing pipelines we aim to establish in this chapter are formed with data interpolation and transformation techniques, embedded into above described Algorithm 1. Particularly, we apply the most popular interpolation methods, which include: Linear Interpolation, Polynomial Interpolation, PCHIP Interpolation, and Spline Interpolation [142]. Generally, interpolation methods allow us to assume that data arrives at more frequent intervals rather than quarterly (the original frequency). For our experiments, we assume monthly and weekly frequencies. Assuming more frequent data arrival (daily) means that many artificial data points are used to generate additional artificial data points, placing our estimator at risk of inferring relationships between artificial data points, rather than the real ones. Also, for methods with varying degrees (e.g., polynomials), we limit the maximum degree to 2 in order to preserve the shape of the data and avoid unnecessary oscillations. Oscillation is the phenomenon whereby the value of an interpolated data is higher than the value of either of two data points, between which an interpolation is performed.

In the similar fashion, we use representative transformation/scaling methods, including Quantile Transformation, Max-Absolute Scaling, Min-Max Scaling, Standard Scaling, and Robust Scaling — the most popular for modifying time-series data [143].

Experiments in this chapter are carried out to establish the pipeline that minimizes the error between a true data point and its estimate, generated with the OLS regression model. The objective and evaluation functions are outlined in the next section.

4.2.4 Evaluation of results

In this study, we aim to solve a regression problem where the target variable is dependent on its past lags, mean, and standard deviation. As outlined earlier, the purpose of conducting experiments in this section is to establish a pipeline effective at minimizing the distance between the target value and its OLS estimate. The pipeline, in this context, refers to the expanding window forecasting algorithm augmented with interpolation or transformation that helps an estimator make more precise estimates of a quarter-ahead value. Specifically, we declare a transformation or augmentation as useful if it yields the lowest Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), or symmetric Mean Absolute Percentage Error (sMAPE).

In order to mitigate the effects of randomness when choosing the leading interpolation/transformation method, we apply the Friedman test to the results of our experiments. Specifically the null hypothesis — the distribution of groups is identical, is rejected at $\alpha = 0.05$ or the 5% significance level. In cases where there is no single leading interpolation/transformation technique, we give preference according to majority voting: we pick the technique ranked highest by two of the three accuracy measures. In the next section, we discuss the datasets on which our experiments were carried out.

4.3 Experimental Setup

In this section, we outline the data used for our experiments and the modifications to the base forecasting for both interpolation and transformation techniques. We begin with a description of the data used in these experiments.

4.3.1 Data

For our experiments, we exclude companies from the financial, services, and other non-manufacturing sectors, as well as those with a market capitalization below $1 billion USD. These firms are subject to different tax treatments under U.S. Generally Accepted Accounting Principles (US GAAP), which may cause their EPS and FCF series to behave differently, potentially biasing the estimation process. To minimize such effects, we aim to control for these structural differences.

Additionally, firms with small market capitalizations are typically subject to less stringent disclosure requirements, which can affect data quality. These are therefore excluded as well.

In the case of holding companies—businesses that derive profits primarily from owning shares in other publicly traded firms—we exclude the parent entity, as it is a non-manufacturing unit with distinct tax treatment. Instead, we include the operational (manufacturing) subsidiaries, which are treated as separate entities.

To further ensure that accounting requirements of different jurisdictions do not affect the model estimation process, all the stocks in our selection are from US stock exchanges. Additionally, due to data set size considerations, specifically the number of data points, we excluded all companies that went public (started selling their shares to investors on stock exchanges) after 2008, as this ensures that our ML/statistical models are exposed to at least 50 data observations. Furthermore, all data sets are limited to the 2021-Q4 period for the purpose of evaluating other experiments in subsequent chapters of this study. Note that companies went public at different times, meaning that not all data sets in our selection have the same number of periods for model training. In this chapter, we work with the representative sample of 50 companies.

As discussed earlier, we create univariate models, meaning that the next quarter's value depends on its past lags. Consequently, the appropriate number of lags is determined automatically using the Akaike Information Criterion (AIC): for each company, the number of lags corresponding to the lowest AIC score is included as a feature. Due to data scarcity, we allow the program to automatically select up to 5 past lags (5 past quarters). To account for deviations and trends in the data, we also incorporate the mean and standard deviation of the target variable distribution as part of the feature set. All data sets in our selection were split into 80/20 (train/test) subsets.

Lastly, we would like to mention that in the early stages of this work, we conducted a number of experiments using a different setup. Specifically, our feature set included a number of external variables: financial ratios computed from financial statements and micro-/macro-economic variables, such as gross domestic product (GDP), inflation rate, etc. Further, we conducted a number of experiments with Genetic Algorithms to exploit possible non-linear relationships in our datasets and tried other distributional moments of the target series, such as skewness, kurtosis, and median. Because the inclusion of these variables did not yield any advantage over univariate models, we do not include the results of these experiments in this thesis. The next section explains modifications made to the baseline forecasting algorithm.

4.3.2 Forecasting Algorithm

Because the experiments in this section emphasize the need for interpolation and scaling of our data sets, we modify the baseline forecasting Algorithm 1 explained earlier in this chapter in two ways: to account for interpolation and transformation, respectively. Therefore, our interpolation forecasting algorithm is outlined in Algorithm 2.

Algorithm 2 demonstrates the set of actions for performing the interpolation experiments,

Algorithm 2: Expanding window algorithm for interpolation experiments

Input: Training data X_{train}, y_{train}, Testing data X_{test}, y_{test}, Estimator $f(X, y)$, interpolation function $interp$

Output: Prediction set $\widehat{y}$

for $i \leftarrow 0$ **to** $I := \textit{length}(y_{\textit{test}})$ **do**

- $X_{\text{train}} \leftarrow interp(X_{\text{train}})$; `// interpolate training data`
- $y_{\text{train}} \leftarrow interp(y_{\text{train}})$
- $f \leftarrow f(X_{\text{train}}, y_{\text{train}})$; `// fit estimator with interpolated data`
- $\widehat{y}_i \leftarrow f(X_{\text{test},i})$; `// make a prediction`
- $X_{\text{train}} \leftarrow \text{Concat}(X_{\text{train}}, X_{\text{test},i})$; `// concatenate to training data`
- $y_{\text{train}} \leftarrow \text{Concat}(y_{\text{train}}, y_{\text{test},i})$

end

which yield the estimated data points $\widehat{y}$. Specifically, for each data point $i = 0, \ldots, \text{length}(y_{\text{test}})$ in the test subset, we first perform interpolation of the train set, $interp(X_{\text{train}})$ and $interp(y_{\text{train}})$. Then, we fit the OLS model f with the interpolated data. After that, we make a prediction $\widehat{y}_i$ using the i-th slice of X_{test} and the trained OLS estimator. Following the prediction, we concatenate (Concat) the i-th slice of X_{test} and y_{test} to the training subsets and repeat the process for all observations in the test set: I denotes the total number of observations in the test subset.

A similar pipeline was used for the transformation functions, as illustrated in Algorithm 3.

In a similar fashion, we use the function h to denote a scaling/transformation method, with h^{-1} denoting the inverse process (mapping from the transformed data back to the original space), to obtain the test set predictions $\widehat{y}$. At every iteration through the test set, we first

Algorithm 3: Expanding window algorithm for data transformation/scaling experiments

Input: Training data X_{train}, y_{train}, Testing data X_{test}, y_{test}, Estimator $f(X, y)$, transformation function h

Output: Prediction set $\widehat{y}$

for $i \leftarrow 0$ **to** $I := \mathit{length}(y_{\mathit{test}})$ **do**

- $X_{\text{train}} \leftarrow h(X_{\text{train}})$; `// transform the training data`
- $y_{\text{train}} \leftarrow h(y_{\text{train}})$
- $f \leftarrow f(X_{\text{train}}, y_{\text{train}})$; `// fit the estimator with transformed data`
- $\widehat{y}_i \leftarrow f(h(X_{\text{test},i}))$; `// make and de-transform a prediction`
- $X_{\text{train}} \leftarrow h^{-1}(X_{\text{train}})$; `// de-transform the training data`
- $y_{\text{train}} \leftarrow h^{-1}(y_{\text{train}})$
- $X_{\text{train}} \leftarrow \text{Concat}(X_{\text{train}}, X_{\text{test},\,i})$; `// concatenate to training data`
- $y_{\text{train}} \leftarrow \text{Concat}(y_{\text{train}}, y_{\text{test},\,i})$

end

transform the data, then fit the regression model $f(X, y)$, predicting one test data point $\widehat{y}i$, with $i = 0, \ldots,$ length(y_{test}). We then transform the i-th slice of the input test data $X_{\text{test},i}$ using the function h. Afterward, we transform both the input and target data back to their original format, concatenating the i-th test data point to the train subset. The process is repeated for all data points in the test subset. In the next section, we present the results of the experiments.

4.4 Results

This section presents two sets of results. First, we describe the datasets and report the outcomes of the statistical tests — the Shapiro-Wilk, Augmented Dickey-Fuller, and Mann-Kendall tests — applied to these series. The second set of results focuses on evaluating data preprocessing pipelines — specifically, whether interpolation or scaling/transformation improves forecasting performance relative to a benchmark, which involves no preprocessing. For these experiments, we use Ordinary Least Squares (OLS) regression as a proxy for forecasting algorithms. The performance of each pipeline is benchmarked using three error metrics: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and symmetric Mean Absolute Percentage Error (sMAPE), with lower values indicating better model fit. In all benchmark cases, the OLS estimator was trained on raw, unprocessed data — that is, without interpolation or transformation.

We refer to the benchmark as the 'Baseline' in all tables that present results. When presenting interpolation results, we explicitly state whether the result of the regression is monthly or weekly. Additionally, a number denotes the degree of polynomial of interpolation, i.e. 'Weekly PCHIP2' means that data was subject to PCHIP interpolation of order 2, assuming the weekly frequency.

Importantly, all statistical tests were conducted on raw data — without any interpolation or transformation — to preserve the original characteristics of the series. All statistical test, interpolation and transformation/scaling experiments are conducted on the representative sample of 50 randomly selected companies.

This section is structured as follows. We begin by describing the data and presenting the results of statistical tests in Subsection 4.4.1. This is followed by interpolation results in Subsection 4.4.2, and scaling/transformation results in Subsection 4.4.3. A detailed discussion of all results is provided in Subsection 4.4.4.

4.4.1 Data Characteristics

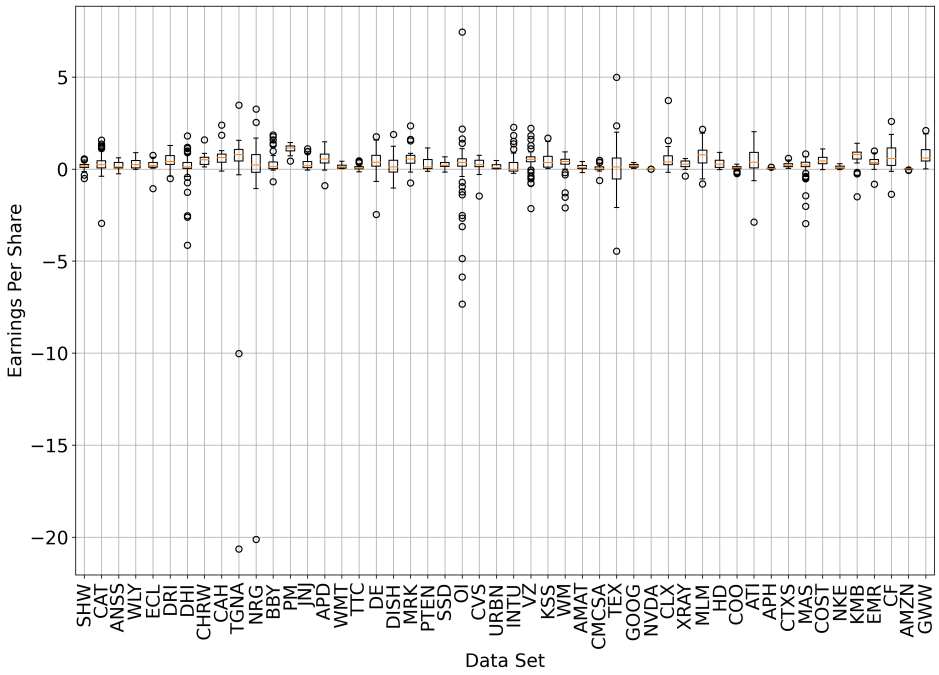
In this subsection, we examine the distributional properties and key statistical characteristics of 50 EPS and FCF data sets, with a particular focusing on normality, stationarity, and trend behavior. First, we present box plots summarizing the distribution of the data sets, starting with the EPS series in Figure 4.1.

Figure 4.1 provides valuable insights. The average EPS value of most data sets is around zero, but the variance exceeds one unit, a finding corroborated by the high skewness. Some data sets exhibit extreme outliers, such as TEGNA Incorporated ('TGNA' in the figure), which shows a value around -21, indicating an extreme outlier. This observation is further supported by the statistical tests, the details of which are summarized in the following bullet points (for full table with p-values for each data set, please refer to Table A-4 in the Appendix, Part A):

1. All 50 data sets are non-normally distributed (SW null hypothesis is rejected);

2. 35 data sets are stationary — the ADF test rejects the null hypothesis of a unit root, indicating constant mean and variance over time;

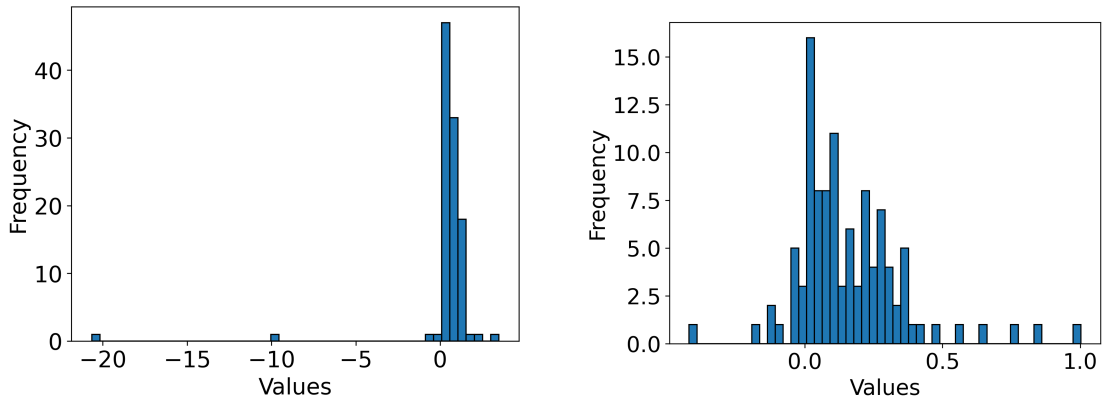
Figure 4.1: *Earnings Per Share data sets box plot*



3. 45 data sets do have the monotonic increasing/decreasing trend (MK hypothesis rejected);

Given this information, we can expect typical data sets to exhibit thick tails in the observed distribution. Furthermore, due to the non-stationary nature of the majority of these data sets, we can infer the presence of distributional shifts. This implies that the statistical properties of the data sets—such as their moments (mean, variance, skewness, and kurtosis)—may vary across time intervals, suggesting dynamic behavior over different periods. This behavior could be attributed to the presence of outliers in the data, which are likely to impact the performance of the regression estimator. We now focus on the distribution of two specific data sets: TEGNA Incorporated ('TGNA' in Figure 4.1) and Applied Materials Incorporated ('AMAT' in Figure 4.1). These data sets were selected as representatives because AMAT exhibited the fewest outliers, while TGNA had the most significant outlier.

Figure 4.2: *EPS: Histograms for representative data sets*



(a) EPS: TGNA Histogram

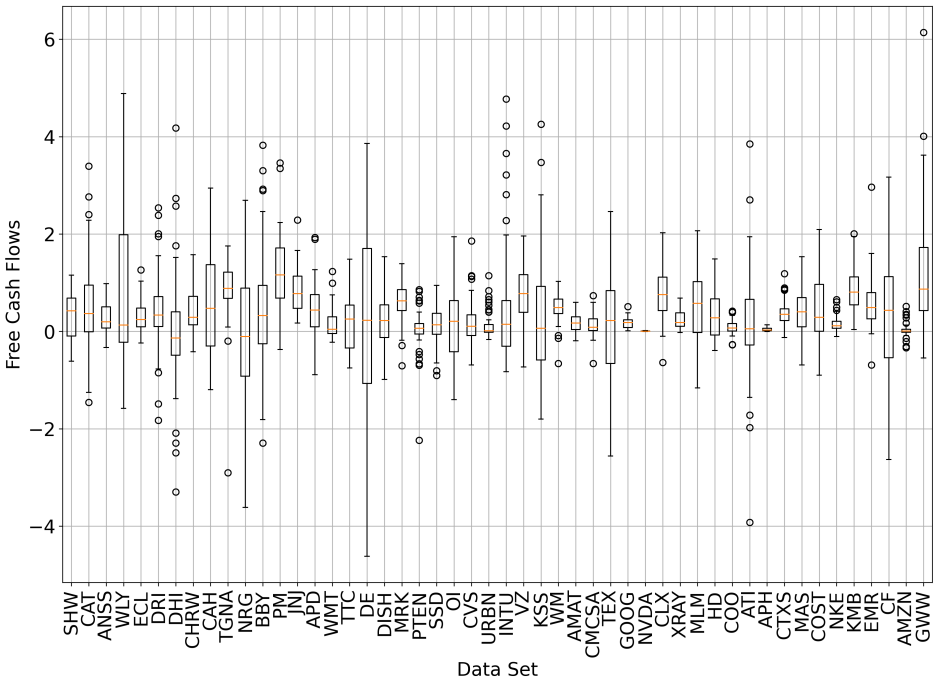
(b) EPS: AMAT Histogram

It is seen that the TGNA data set exhibits a visible negative outlier near -21 and another around -10. Despite this, most values are concentrated near 0. In contrast, the AMAT data set shows values distributed within a variance range of +/-1, indicating a unit variance distribution. Additionally, more values are clustered to the right of the mean, suggesting a positive trend in the series. We now turn to the FCF data sets, summarized by the box plot in Figure 4.3.

Notably, more FCF data sets appear to have outliers, a wider distance between the upper and lower quantiles in most data sets, and longer whiskers in specific data sets, such as DE, TEX, and WLY. The resulting p-values for each of 50 companies are represented in Table A-5. The results of the statistical tests applied to the FCF series are summarized as follows:

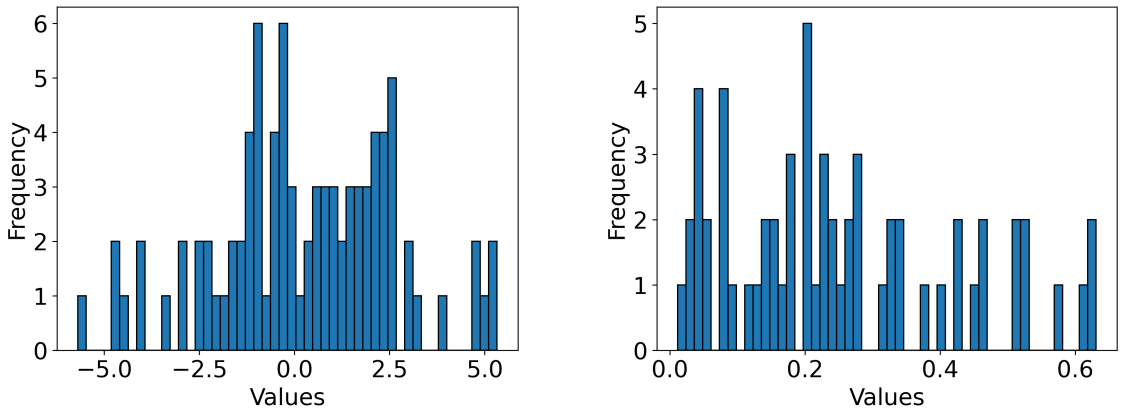
1. 44 data sets are non-normally distributed (SW null hypothesis is rejected);
2. 42 data sets contain at least one unit-root, suggesting non-stationary behaviour (reject the ADF null hypothesis);
3. 43 data sets show monotonic increasing/decreasing trend (MK null hypothesis rejected);

Figure 4.3: *Free Cash Flows data sets box plot*



Therefore, a typical FCF data set exhibits large positive and negative swings, where a positive FCF value can be followed by a significant negative value in the next quarter. These fluctuations lead to outliers that can potentially impact the estimation process. Additionally, the series does not need to be monotonically increasing or decreasing, further supporting the earlier statement about sudden shifts in the data. Evidently, more FCF data sets appear centered around a consistent mean, they also display greater overall variance and volatility, often exceeding a unit variance and exhibiting frequent large swings between positive and negative values. However, the variance is expected to be larger than 1, as the normality assumption does not hold for most of the data sets. To illustrate this, we choose two representative companies: Deere & Company, Incorporated (DE in Figure 4.3), which exhibits the largest dispersion in its distribution, and Google Incorporated (GOOG in Figure 4.3), which has the opposite distributional properties. The histograms for these two series are shown in Figure 4.4.

Figure 4.4: *FCF: Histograms for representative data sets*



(a) FCF: DE Histogram

(b) FCF: GOOG Histogram

The histogram of Deere & Company (left panel in Figure 4.4) data set displays thicker tails in the distribution. Both the extreme right and left tails exhibit a significant distance from the mean (greater than +/- 1). Additionally, many values in this data set are clustered in the -2 to 0 range, with a local peak around 3. In contrast, the Google Inc. (right in Figure 4.4) data set is more concentrated around 0, with values predominantly positive.

In summary, both EPS and FCF series share key characteristics: most are non-normally distributed, with high skewness and excess kurtosis. These distributional challenges, particularly the high concentration of outliers, can worsen the performance of regression estimators. The OLS estimator, in particular, may be sensitive to such extreme values, leading to overfitting and poor generalization. Moreover, limited data availability can further constrain model performance. In the next subsection, we explore whether interpolation helps mitigate these issues.

4.4.2 Interpolation Experiments

As previously discussed, one major challenge in our regression problem is the lack of data observations. To address this, we apply interpolation techniques, which estimate synthetic values between existing data points [142]. Interpolation allows us to assume higher data frequency—such as monthly or weekly observations—thus expanding the dataset while preserving its original structure.

Apart from increasing the number of observations, interpolation can also reduce the impact of abrupt changes or nonlinearity in the series. In some cases, this smoothing effect can help linearize underlying exponential trends, making them more suitable for linear estimators [146].

Consequently, the purpose of interpolation in our study is to improve model generalization to the unseen data, which we assess by the comparing MAE, MAPE and sMAPE scores of OLS estimator fit with and without a set of interpolation methods. To avoid being misled by the cases of extreme overfitting, when the OLS estimator shows a regression error of 0.0, we use the scores derived from the test set — the unseen data. If we see an improvement in error measures, we say that an interpolation method aided in model generalization, which resulted in better model predictions.

We start this section with the analysis of the EPS results. Table 4.1 summarizes the regression results on the test set after applying various interpolation methods.

Table 4.1 presents the mean, standard deviation, minimum, median, and maximum values of the error distribution for the interpolation experiments conducted on the EPS data sets. We comment on several key insights.

Overall, interpolation methods do not yield substantial improvements over the baseline

Table 4.1: *EPS: Interpolation results summary statistics. Values in* ***boldface*** *denote minimum of a measure (mean, standard deviation, etc.) in the respective error measure (MAE, MAPE, sMAPE)*

Method	Error	Mean	Std. Dev.	Minimum	Median	Maximum
Baseline	MAE	0.97895	1.67613	0.12440	**0.50707**	10.76355
	MAPE	1.63271	**3.27743**	0.09778	**0.60601**	**21.25606**
	sMAPE	28.96751	18.87042	5.15181	**22.77532**	79.95304
Monthly Linear 1	MAE	0.99503	1.73630	0.12278	0.52381	11.22503
	MAPE	1.70009	3.45690	0.09845	0.62296	22.27229
	sMAPE	29.052311	18.934526	5.134335	23.425523	80.038339
Monthly Polynomial 1	MAE	0.996035	1.742502	0.122831	0.524122	11.272853
	MAPE	1.70085	3.458871	0.09839	0.623214	22.277953
	sMAPE	29.0493	18.928378	5.131712	23.395255	79.919035
Monthly Spline 1	MAE	**0.973438**	1.643451	0.118837	0.521998	10.80452
	MAPE	**1.614219**	3.338584	0.097928	0.614505	22.159659
	sMAPE	28.994147	**18.656644**	5.152158	23.100348	75.662816
Monthly PCHIP 1	MAE	0.997062	1.730282	0.123651	0.510233	11.138308
	MAPE	1.665886	3.389082	0.098546	0.613627	22.046341
	sMAPE	29.014919	18.821727	5.172756	23.631413	81.047683
Weekly Linear 1	MAE	0.990492	1.710275	0.122808	0.519559	10.963642
	MAPE	1.708696	3.452646	**0.097208**	0.621173	22.142379
	sMAPE	29.066509	18.900958	**5.074706**	23.335044	79.859941
Weekly PCHIP 1	MAE	0.993432	1.752783	0.120236	0.520427	11.510278
	MAPE	1.629862	3.322401	0.097856	0.618049	21.752521
	sMAPE	**28.910007**	18.660071	5.146853	23.55941	81.261767
Weekly Spline 1	MAE	0.985378	1.631284	0.119053	0.524067	10.457265
	MAPE	1.637371	3.352342	0.097947	0.61431	22.331203
	sMAPE	29.221365	18.866999	5.151777	23.124244	83.724062
Weekly Spline 2	MAE	0.97724	**1.598191**	**0.113362**	0.537623	**10.322918**
	MAPE	1.664693	3.56369	0.097997	0.633882	24.035365
	sMAPE	29.590481	18.723452	5.210329	24.079268	**74.661986**

approach (denoted as 'Baseline' in Table 4.1), which makes predictions from non-interpolated training data. In some cases, the baseline even outperforms interpolated models.

Notably, the lowest average MAE is achieved using Monthly Spline interpolation of order 1, followed closely by other interpolation methods (Monthly Polynomial of order 1, Weekly PCHIP of order 1, etc.). The baseline achieves the third-best mean MAE score of 0.978958. Similar is observed for MAPE error measurement, where Monthly Spline of order 1 achieves the lowest average error, followed by Monthly PCHIP of order 1 and the baseline. For sMAPE, the lowest mean error is recorded by Monthly PCHIP of order 1, with the baseline again being second best by this measure.

When considering the median, it is seen that the baseline achieves the lowest MAE, MAPE, and sMAPE scores. Still, some interpolation methods remain competitive—for example, Monthly PCHIP of order 1 ranks second for both median MAE and MAPE, while Monthly Spline of order 1 shows the second-best median sMAPE.

These results suggest that interpolation offers limited benefits when applied to EPS data sets. This conclusion is further supported by the Friedman average ranking test results, presented in Table 4.2.

The Friedman average ranking test is conducted in two steps. In the first step, interpolation methods are ranked for each data set based on their performance (i.e., MAE, MAPE, or sMAPE), with a higher rank indicating lower error. The average rank, referred to as 'Rank' in Table 4.2, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control method (marked as 'control' in Table 4.2) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that all methods (i.e., the OLS regression errors, measured as MAE, MAPE and sMAPE,

Table 4.2: *EPS: Interpolation Friedman Test average ranking results. The Hommel's post-hoc p-value (the 'p-val' column) in* ***boldface*** *indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**p-value**	**Method**	**Rank**	**p-value**	**Method**	**Rank**	**p-value**
Baseline	4.18	control	Baseline	4.12	control	Monthly Polynomial 1	4.38	control
Monthly Polynomial 1	4.6	4.43E-01	Weekly PCHIP 1	4.48	5.11E-01	Weekly Linear 1	4.46	8.84E-01
Weekly PCHIP 1	4.72	4.43E-01	Monthly Spline 1	4.88	3.31E-01	Monthly Linear	4.62	8.84E-01
Weekly Linear 1	4.74	4.43E-01	Monthly PCHIP 1	4.98	2.48E-01	Weekly PCHIP 1	4.88	8.84E-01
Monthly Linear	4.92	4.43E-01	Weekly Spline 2	5.02	2.48E-01	Baseline	4.9	8.84E-01
Monthly PCHIP 1	5.04	4.32E-01	Monthly Polynomial 1	5.3	1.56E-01	Monthly PCHIP 1	4.98	8.20E-01
Monthly Spline 1	5.18	3.49E-01	Weekly Linear 1	5.34	1.30E-01	Monthly Spline 1	5.2	6.02E-01
Weekly Spline 2	5.64	5.38E-02	Monthly Linear	5.36	1.18E-01	Weekly Spline 1	5.78	7.41E-02
Weekly Spline 1	5.98	**8.12E-03**	Weekly Spline 1	5.52	7.28E-02	Weekly Spline 2	5.8	6.67E-02

respectively) are drawn from the same distribution.

Overall, the results from the Friedman test support our earlier conclusions. The baseline (OLS without interpolation) achieved the lowest MAE and MAPE, and thus served as the control method for these two metrics. MAE was the only case in which the baseline showed statistically significant superior over an interpolation method — the Weekly Spline interpolation of order 1, in particular.

For sMAPE, however, the control method was not the baseline but the Monthly Polynomial interpolation of order 1. Still, no statistically significant differences were observed between this method and the baseline or any other method.

These findings reinforce the conclusion that interpolation methods — regardless of type or assumed frequency — yield nearly identical regression errors. Hence, interpolation does not improve forecasting performance for EPS data sets.

Next, we consider the results interpolation methods yield in FCF data sets. The summary statistics for regression errors are presented in Table 4.3.

In contrast to the EPS data sets, interpolation methods show clear benefits in reducing regression errors when applied to the FCF data sets. For two of the three metrics — MAE and MAPE — the average error across data sets is lower when interpolation is applied.

In particular, the lowest mean MAE is achieved by the Monthly PCHIP interpolation of order 1, although the baseline method performs comparably, with only a slightly higher average MAE. A similar pattern is observed for MAPE, where the Weekly PCHIP interpolation of order 1 yields the lowest mean error, again followed closely by the baseline. sMAPE is the only error measure where the baseline method outperforms all interpolation methods in terms of average error.

When considering median errors, the trend remains consistent. The baseline results in

Table 4.3: *FCF: Interpolation results summary statistics. Values in **boldface** denote minimum of a measure (mean, standard deviation, etc.) in the respective error measure (MAE, MAPE, sMAPE)*

Method	Error	Mean	Std. Dev.	Minimum	Median	Maximum
Baseline	MAE	0.857650	0.673946	0.129569	0.620380	2.960124
	MAPE	1.690080	3.270545	0.186151	0.903037	21.92372
	sMAPE	**32.507354**	16.333727	9.916542	**29.730031**	83.982047
Monthly Linear 1	MAE	0.866845	0.680179	0.126789	0.607757	2.903412
	MAPE	1.658828	3.203385	0.185489	0.908472	21.457507
	sMAPE	32.90149	16.27178	9.651893	30.94456	**72.748908**
Monthly PCHIP 1	MAE	**0.854957**	**0.665041**	0.129851	**0.605366**	2.873514
	MAPE	1.591966	3.096013	0.181974	0.898569	20.916913
	sMAPE	32.98231	16.01472	9.943527	30.69275	74.98501
Monthly Spline 1	MAE	0.864368	0.683621	0.128209	0.643145	3.069775
	MAPE	1.579684	**2.929102**	0.189549	0.896556	19.93913
	sMAPE	34.744739	17.059801	11.249774	31.070145	85.473258
Weekly Linear 1	MAE	0.869406	0.680508	**0.126301**	0.611776	2.881614
	MAPE	1.588979	3.119459	**0.180874**	0.880441	21.098495
	sMAPE	33.0355	16.402578	**9.560491**	31.032589	73.793001
Weelky PCHIP 1	MAE	0.868055	0.666317	0.140166	0.611805	**2.862962**
	MAPE	**1.569065**	2.965519	0.182871	**0.872971**	20.177581
	sMAPE	32.510369	**15.997815**	9.619502	30.433799	74.708957
Weekly Spline 1	MAE	0.896610	0.731839	0.127709	0.656187	3.378738
	MAPE	1.658001	2.953568	0.191867	0.900086	19.825926
	sMAPE	35.106848	17.555128	10.978659	31.004538	86.557502
Weekly Spline 2	MAE	0.900582	0.719487	0.127562	0.659008	3.169090
	MAPE	1.661887	2.974079	0.201369	0.931428	**19.80648**
	sMAPE	35.306848	17.555128	11.178659	31.204538	86.757502

Table 4.4: *FCF: Interpolation Friedman Test average ranking results. The Hommel's post-hoc p-value (the 'p-val' column) in* ***boldface*** *indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**p-val**	**Method**	**Rank**	**p-val**	**Method**	**Rank**	**p-val**
Weekly PCHIP 1	3.66	control	Weekly PCHIP 1	3.84	control	Weekly PCHIP 1	2.7	control
Baseline	4.02	4.62E-01	Weekly Linear 1	3.84	1.00E+00	Baseline	2.78	8.70E-01
Monthly Spline 1	4.24	4.62E-01	Monthly PCHIP 1	4.14	1.00E+00	Monthly Linear 1	3.76	6.10E-02
Monthly Linear 1	4.28	4.11E-01	Weekly Spline 1	4.38	8.10E-01	Weekly Linear 1	3.94	$\mathbf{3.41E-02}$
Weekly Linear 1	4.48	3.15E-01	Monthly Linear 1	4.76	2.42E-01	Monthly PCHIP 1	4.38	$\mathbf{2.42E-03}$
Weekly Spline 1	4.6	2.35E-01	Monthly Spline 1	4.88	1.51E-01	Weekly Spline 1	5.5	$\mathbf{5.47E-08}$
Weekly Spline 2	5.04	$\mathbf{2.91E-02}$	Weekly Spline 2	4.98	1.01E-01	Monthly Spline 1	6.26	$\mathbf{2.21E-12}$
Monthly PCHIP 1	5.68	$\mathbf{2.61E-04}$	Baseline	5.18	$\mathbf{4.36E-02}$	Weekly Spline 2	6.68	$\mathbf{3.15E-15}$

the lowest median sMAPE. However, the Weekly PCHIP interpolation of order 1 achieves the lowest median MAPE, while the Monthly PCHIP interpolation of order 1 records the lowest median MAE.

These summary statistics suggest that, unlike with EPS, interpolation can enhance forecasting performance in the FCF data sets. We now examine how these differences are reflected in the Friedman average ranking test, the results of which are shown in Table 4.4.

The Friedman average ranking test for the FCF data sets drastically contrasts with the

EPS results. Across all three error measures—MAE, MAPE, and sMAPE—the Weekly PCHIP interpolation method of order 1 consistently emerges as the control method.

Importantly, some lower-ranked interpolation methods perform statistically significantly worse than the control. For example, in the MAE error group, the Monthly PCHIP interpolation method performs significantly worse than Weekly PCHIP of order 1. In the MAPE group, the baseline method also underperforms the control, making it the only instance where the non-interpolated benchmark is statistically inferior to an interpolation method.

In the sMAPE group, the Weekly PCHIP interpolation method significantly outperforms several data interpolation methods: Monthly PCHIP interpolation of order 1, Monthly Spline interpolation of order 1, and both Weekly Spline interpolations of order 1 and 2.

Given that all three error metrics agree on the top-performing method, we conclude that Weekly PCHIP interpolation of order 1 should be used in all further experiments involving the Free Cash Flows data sets. Conversely, since no such consensus was observed across error metrics in the EPS results, we opt to proceed without interpolation for those data sets.

We now turn to experiments involving scaling and transformation methods, detailed in Section 4.4.3.

4.4.3 Transformations

In both regression and classification tasks, it is common to apply data transformations or scaling techniques—either to enforce a particular distribution or to constrain values within a specific range, or both. This study pursues the transformation/scaling experiments to assess whether such techniques can enhance the performance in our regression problem.

As with the previous experiments, we use the expanding window algorithm described in Algorithm 3 and evaluate model performance on the test set. The benchmark results correspond

Table 4.5: *EPS: Transformations/Scaling results summary statistics. Values in **boldface** denote minimum in the respective statistic (i.e. Average, Minimum, Maximum, etc.)*

Transformer / Scaler	**Error**	**Average**	**Std. Dev.**	**Minimum**	**Median**	**Maximum**
Baseline	MAE	0.978958	1.676134	0.124404	0.507073	10.763558
	MAPE	1.632711	3.277434	0.097785	0.606011	21.256066
	sMAPE	28.967513	18.87042	5.151815	22.775323	79.953044
MaxAbs	MAE	0.978958	1.676134	0.124404	0.507073	10.763558
	MAPE	1.632711	3.277434	0.097785	0.606011	21.256066
	sMAPE	28.967513	18.87042	5.151815	22.775323	79.953044
MinMax	MAE	0.978958	1.676134	0.124404	0.507073	10.763558
	MAPE	1.632711	3.277434	0.097785	0.606011	21.256066
	sMAPE	28.967513	18.87042	5.151815	22.775323	79.953044
Quantile	MAE	**0.67181**	**0.584655**	**0.11811**	**0.457743**	**2.997248**
	MAPE	**0.702621**	**0.614924**	**0.080227**	**0.503758**	**3.39168**
	sMAPE	**26.108303**	**16.850372**	**3.05345**	**22.745026**	**73.187807**
Robust	MAE	0.978958	1.676134	0.124404	0.507073	10.763558
	MAPE	1.632711	3.277434	0.097785	0.606011	21.256066
	sMAPE	28.967513	18.87042	5.151815	22.775323	79.953044
Standard	MAE	0.978958	1.676134	0.124404	0.507073	10.763558
	MAPE	1.632711	3.277434	0.097785	0.606011	21.256066
	sMAPE	28.967513	18.87042	5.151815	22.775323	79.953044

to predictions made using the expanding window algorithm without any transformation or scaling applied.

We begin with presenting regression errors summary statistics depicted in Table 4.5. Since our focus is on incorporating these estimates into valuation models, as discussed in Chapter 2, all results are reported on the de-scaled (or de-transformed) series. This is to mitigate the fact that certain error measures are sensitive to the units of data.

From Table 4.5 we observe that while most transformation and scaling techniques offer

Table 4.6: *EPS: Friedman Test results ranking for Scaling/Transformers. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**p-val**	**Method**	**Rank**	**p-val**	**Method**	**Rank**	**p-val**
Quantile	2.90	control	Quantile	2.40	control	Quantile	2.40	control
Baseline	3.62	5.43E-02	Baseline	3.72	**4.19E-04**	Baseline	3.72	**4.19E-04**
MaxAbs	3.62	5.43E-02	MaxAbs	3.72	**4.19E-04**	MaxAbs	3.72	**4.19E-04**
MinMax	3.62	5.43E-02	MinMax	3.72	**4.19E-04**	MinMax	3.72	**4.19E-04**
Robust	3.62	5.43E-02	Robust	3.72	**4.19E-04**	Robust	3.72	**4.19E-04**
Standard	3.62	5.43E-02	Standard	3.72	**4.19E-04**	Standard	3.72	**4.19E-04**

little to no advantage to the estimator on the EPS data sets, the Quantile Transformer stands out as an exception, consistently outperforming all other methods across all evaluation metrics.

For example, most scaling and transformation techniques result in a median MAE of 0.978959. In contrast, the Quantile Transformer achieves a significantly lower median MAE of 0.507073. This trend holds across other error measures as well, where the Quantile Transformer yields the lowest (i.e., best) scores.

The superiority of the Quantile Transformation on the EPS series is further supported by the Friedman average ranking test, summarized in Table 4.6.

Because the error measures for each score are identical, the MaxAbs, MinMax, Robust, and Standard scalers all received the same average rank of 3.62 and 3.72 for MAE, MAPE and sMAPE errors, respectively. In every error metric group, the Quantile Transformer emerges as the 'control' method, demonstrating statistical significance when compared to the remaining scaling techniques.

Table 4.7: *FCF: Transformations/Scaling results summary statistics. Values in* ***boldface*** *denote minimum in the respective statistic (i.e. Average, Minimum, Maximum, etc.)*

Transformer / Scaler	**Error**	**Average**	**Std. Dev.**	**Minimum**	**Median**	**Maximum**
Baseline	MAE	**0.85765**	**0.673947**	**0.12957**	**0.620381**	**2.960124**
	MAPE	1.69008	3.27055	0.18615	0.90304	21.92372
	sMAPE	**32.50735**	**16.33372**	**9.91654**	**29.73003**	**83.98204**
MaxAbs	MAE	0.85765	0.67395	0.12957	0.62038	2.96012
	MAPE	1.69008	3.27055	0.18615	0.90304	21.92372
	sMAPE	32.50735	16.33373	9.91654	29.73003	83.98205
MinMax	MAE	0.85765	0.67395	0.12957	0.62038	2.96012
	MAPE	1.69008	3.27055	0.18615	0.90304	21.92372
	sMAPE	32.50735	16.33373	9.91654	29.73003	83.98205
Quantile	MAE	0.92515	0.7374	0.11709	0.65452	3.56262
	MAPE	**1.52453**	**3.12619**	**0.17805**	**0.87359**	**21.59568**
	sMAPE	35.84087	18.30355	8.25396	34.38557	83.18649
Robust	MAE	0.85765	0.67395	0.12957	0.62038	2.96012
	MAPE	1.69008	3.27055	0.18615	0.90304	21.92372
	sMAPE	32.50735	16.33373	9.91654	29.73003	83.98205
Standard	MAE	0.85765	0.67395	0.12957	0.62038	2.96012
	MAPE	1.69008	3.27055	0.18615	0.90304	21.92372
	sMAPE	32.50735	16.33373	9.91654	29.73003	83.98205

Notably, in two out of three error groups, the Quantile Transformer also shows statistically significant improvements over the baseline approach, where the OLS was fit on the raw data. These findings for the EPS series contrast sharply with those for the FCF series, the summary statistics of which are presented in Table 4.7.

Table 4.7 indicates that scaling and transformation techniques offer no improvements for the FCF series. Specifically, scaling methods do not help reduce the approximation error (as measured by MAE, MAPE, and sMAPE scores). As with the EPS results, both the scaling

Table 4.8: *FCF: Friedman Test results ranking for Scaling/Transformers. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**p-val**	**Method**	**Rank**	**p-val**	**Method**	**Rank**	**p-val**
Baseline	3.32	control	Quantile	3.00	control	Baseline	3.30	control
MaxAbs	3.32	1.00E+00	Baseline	3.60	1.09E-01	MaxAbs	3.30	1.00E+00
MinMax	3.32	1.00E+00	MaxAbs	3.6	1.09E-01	MinMax	3.30	1.00E+00
Robust	3.32	1.00E+00	MinMax	3.6	1.09E-01	Robust	3.30	1.00E+00
Standard	3.32	1.00E+00	Robust	3.6	1.09E-01	Standard	3.30	1.00E+00
Quantile	4.4	**1.95E-02**	Standard	3.6	1.09E-01	Quantile	4.50	**6.70E-03**

techniques and the baseline approach yield the same outcomes.

However, unlike the EPS results, the Quantile Transformer tends to worsen the forecasting error for the FCF series. In fact, in two of the three error measures, the Quantile Transformer results in higher errors. For example, it produces the highest average and median MAE and sMAPE, while yielding the lowest average and median MAPE.

These findings lead to conflicting results from the Friedman average ranking procedure, as shown in Table 4.8.

Table 4.8 highlights that, in two out of three cases, the baseline approach ranks 'control' among other candidate methods, with average rankings of 3.32, and 3.3 in the MAE and sMAPE groups, respectively. Importantly, the similarly to the EPS values, the Baseline, MaxAbs, MinMax, Robust and Standard data transformation methods result in exact same errors, hence show the same rank of 3.32, 3.50 and 3.30 in MAE, MAPE and sMAPE error measurements, respectively, expectedly displaying no statistical significance. At the same time, the baseline

approach yields statistically significant results compared to the Quantile Transformer, which is the only transformation/scaling method that differs in its mean and median MAE and sMAPE results from the other methods in our selection.

The MAPE error measure is the only case where the Quantile Transformer ranks as the 'control' methodology. However, it is not statistically significant against the benchmark approach.

From this subsection, we draw the following conclusions. Empirically, for the sample of 50 EPS/FCF data sets, we observed that interpolation benefits the FCF series but does not significantly improve the EPS series. In contrast, when the OLS regression is applied to the EPS series with Quantile Transformation, we see improvements across all error measures.

Therefore, in the following sections of this thesis, we will apply weekly PCHIP interpolation of order 1 to the FCF data sets and use Quantile Transformation for the EPS data sets. In the next subsection, we use a representative sample of five data sets to explore the underlying reasons for the results observed in this section.

4.4.4 Results Discussion

The question now arises: why did interpolation work for the FCF series but not for the EPS series, and why did transformation work for the EPS series but not for the FCF series? In this part of the thesis, we investigate the possible reasons for this kind of phenomenon. Our analysis starts with discussion of interpolation effects on data in the next subsection.

Interpolation effects on data

Over the course of this study, we used interpolation to artificially increase the number of data points for subsequent estimator training. As a side effect, the interpolated data tends to be smoother than the original data while preserving its overall shape.

Table 4.9: *Average and Median of the Standard Deviations computed using quarter-to-quarter percentage change (magnitude of changes): original and interpolated*

Series type	Methodology	Average	Median
FCF	Original	15.17357	6.88747
	Interpolated	9.99605	2.41903
EPS	Original	4.856711	2.5588
	Interpolated	1.82774	0.84208

In the case of the EPS series, the interpolation smooths the data too much, causing the estimator to struggle with capturing the potential volatility of the series. In contrast, the FCF data is smoothed just enough to improve the OLS estimator's forecasting accuracy.

To support this conclusion, we measure the percent change of the target series (EPS and FCF) from one quarter to the next, which indicates the magnitude of change between consecutive quarters. This is done in two steps. First, we compute the percent change from one quarter to the next, showing how much the EPS/FCF value increased or decreased compared to the previous quarter, in percentage terms. Second, we compute the standard deviation of the resulting percentage series.

For clarity, we present the average and median of this procedure across the 50 companies used in this study. These results are shown in Table 4.9. Note that for the interpolated series, we used the weekly PCHIP interpolation method of order 1, as it ranked higher across the error measures for both EPS and FCF, ensuring consistency in our comparisons.

There are two key observations to be made from Table 4.9. First, even before interpolation, the two series exhibit different magnitudes of change. The baseline FCF series is more than

three times as volatile as the baseline EPS series. This could be attributed to the fact that EPS is of greater interest to shareholders, as managers are motivated to maintain positive EPS values, and a sudden decrease often results in poor market reactions [8]. However, the volatility in FCF is often associated with increased business investments, which is typical for a growing company requiring significant investments [5].

Second, both the EPS and FCF series see a significant reduction in the magnitudes of changes when interpolated, which indicates that interpolation has indeed smoothed the original data. In both cases, the magnitude of change from one quarter to the next drops more than twofold. For FCF, this smoothing simplifies the regression task by reducing the complexity of the data, making it easier for the estimator to identify trends. Conversely, the EPS data, which is already smoother than FCF, becomes 'too smooth', causing the OLS estimator to undermine volatility in the series.

The next question concerns why the weekly PCHIP interpolation method was helpful, while other methods ranked lower in the FCF data sets. This may be due to the characteristics of the PCHIP method, which tends to produce fewer extreme values across all 50 data sets. Unlike other interpolation methods, which can introduce local oscillations (i.e., extreme spikes or drops above or below the original data), PCHIP is better at preserving the original shape of the data, especially in contexts of rapidly changing values [147]. In contrast, other methods produce monotonic series that may distort OLS predictions for FCF data.

Overall, in the FCF data sets, interpolation makes the data smooth enough for the OLS estimator to accurately predict quarter-ahead FCF values. The weekly PCHIP interpolation method of order 1 shows better performance for these data sets, likely due to its ability to handle dynamic, rapidly changing data. In the EPS data sets, interpolation smooths the data too much, making it harder for the OLS estimator to capture volatility.

Next, we will discuss the effects of scaling and transformation on the FCF and EPS data sets.

Transformation effects on data

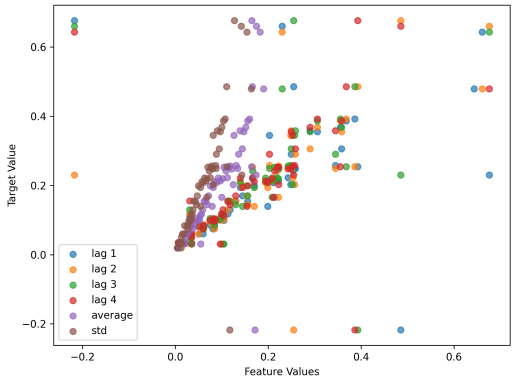
Previously, we observed that the EPS series responded positively to the Quantile Transformation (QT), while the FCF series exhibited no response to either scaling or transformation methods. In this section, we examine how QT modifies these data sets by applying it to a subset of randomly selected companies from the sample of 50 companies we experimented on, namely: Comcast Corporation (CMCSA), Alphabet Inc. (GOOG), Patterson-UTI Energy Inc. (PTEN), Kimberly-Clark Corporation (KMB), and The Sherwin-Williams Company (SHW). More detailed information on these companies can be found in Table A-1 in the Appendix, Section A.

We present figures for GOOG and PTEN here for illustrative purposes, with the corresponding plots for other companies available in the Appendix, Section A.

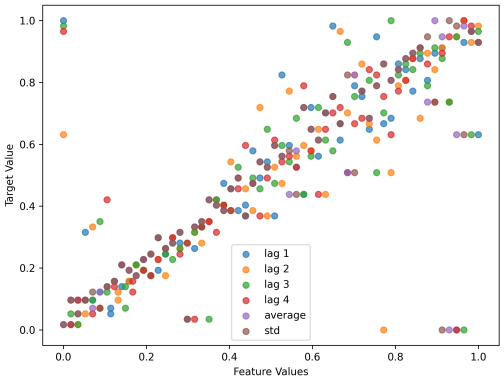
As previously noted, the Quantile Transformation is designed to map feature and target variables to a uniform distribution. This transformation moves every transformed feature closer to the transformed target, with all values now falling within the 0-1 range, following the same, uniform distribution. This makes extreme EPS and FCF values more influential on predictions. To illustrate, Figure 4.5 presents a scatterplot with the target variable (EPS) on the y-axis and all its features on the x-axis for both GOOG and PTEN. Similar plots for the other companies are shown in Figure A-1 in the Appendix, Section A. Note that the number of features in each data set differs, as our features matrix represents past lags of the target series. The optimal number of past lags, which minimizes the Akaike Information Criterion (AIC), was determined for each dataset.

From Figure 4.5, it is evident that the original data in plots (a) and (b) tends to cluster

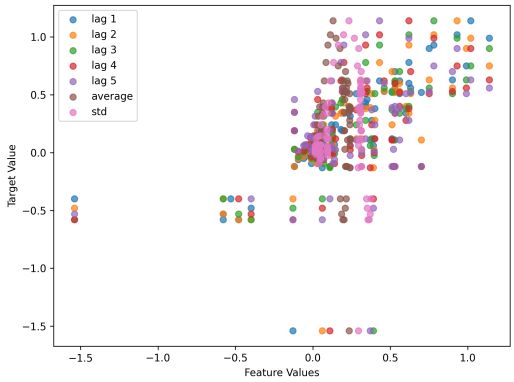
Figure 4.5: *EPS: scatter plot of the target series (y-axis) against its features (x-axis), transformed (right) and original (left)*



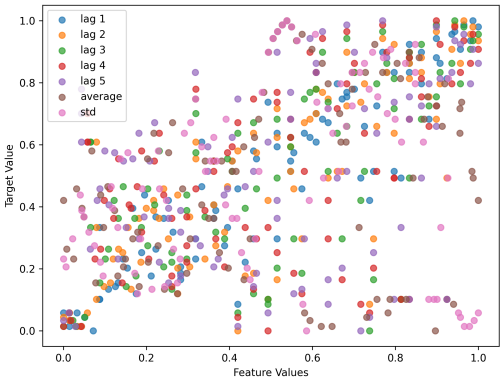
(a) GOOG: original data scatter plot



(c) GOOG: transformed data scatter plot



(b) PTEN: original data scatter plot



(d) PTEN: transformed data scatter plot

around certain points: most values are concentrated in a certain place of graph. Some data points are visibly distant from this cluster, again supporting our earlier observation of the more monotonic nature of the EPS series.

After the transformation, many of the outliers are drawn closer to the main cluster, which exposes the estimator to these outliers during regression. Specifically, the higher the EPS values on the y-axis, the higher the corresponding feature values on the x-axis. This suggests that the Quantile Transformation emphasizes the linear relationship between the EPS series and its lags, which complements the OLS estimation process.

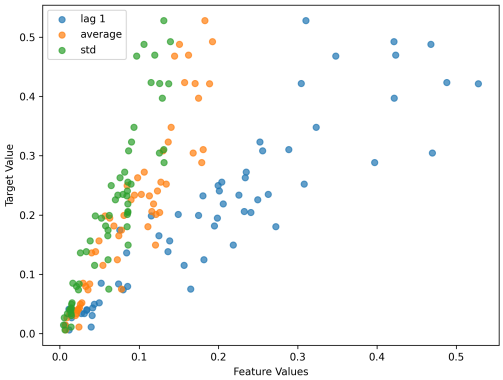
Next, we turn to the FCF data sets. Figure 4.6 shows the scatter plots for GOOG and PTEN, with additional plots available in the Appendix (Figure A-2 in Section A).

When transformed, the data points for FCF appear nearly random in some cases, making it difficult to establish a clear relationship between the target (FCF) and the features. This is particularly evident for the PTEN dataset, as seen in subfigures (b) and (d) of Figure 4.6, and in the KMB and SHW data sets in Figure A-2 in Appendix, Section A.

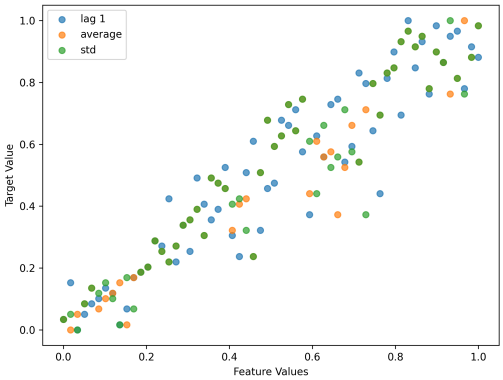
This randomness in the transformed FCF data reinforces that the relationship between the target and features is not strictly linear, which likely explains why the Quantile Transformation did not improve prediction accuracy for FCF. In some cases, the prediction accuracy improved slightly, but not enough to make QT a better-performing technique compared to the baseline.

In conclusion, the Quantile Transformation, while effective in minimizing regression error for EPS data sets, worsened inference for the FCF data sets. The transformation enhanced the linearity of the EPS series, potentially improving OLS accuracy. However, since FCF data does not exhibit a linear relationship, QT did not provide similar benefits. Therefore, while QT is not a necessary procedure in our forecasting pipeline, it can be a useful tool for datasets like EPS, where linearity is more blatant. Next section concludes this chapter.

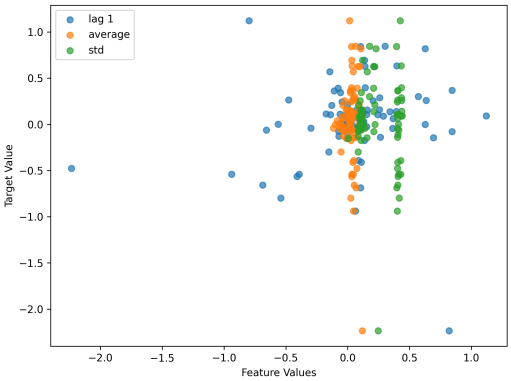
Figure 4.6: *FCF: scatter plot of the target series (y-axis) against its features (x-axis), transformed (right) and original (left)*



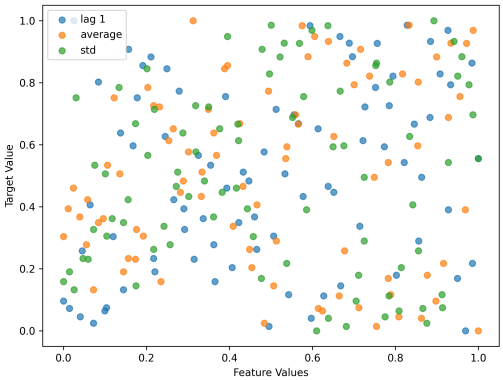
(a) GOOG: original data scatter plot



(c) GOOG: transformed data scatter plot



(b) PTEN: original data scatter plot



(d) PTEN: transformed data scatter plot

4.5 Chapter Conclusions

In this chapter, we outlined the characteristics of our data sets and examined the impact of interpolation methods and scaling/transformation techniques on these sets. Specifically, we investigated how these techniques influence the accuracy of predictions made by the Ordinary Least Squares (OLS) regression estimator. All experiments were conducted on a representative set of 50 data sets.

Generally, our data sets are non-stationary, meaning that their mean and variance change over time. Additionally, most of our data sets exhibit a trend and are not normally distributed. This did not pose a significant problem, as normalization did not provide any clear benefits. Since the data is already in the same units, scaling was unnecessary. However, empirically, we found that a specific type of transformer applied to the EPS data sets minimized the error.

We found that the PCHIP interpolation technique of the first order, which assumes the data arrives on a weekly basis rather than quarterly, was effective at minimizing error measures for the FCF data sets but not for the EPS ones. Generally, interpolation methods create a smoothing effect on outliers in model outcomes, as seen with the FCF data sets. In contrast, for the EPS data sets, where the OLS regression was more successful at distinguishing outliers from general data points, all interpolation techniques increased the error between the model's inferred values and the true data points.

The second part of experiments stresses the benefits transformation/scaling techniques bring to the estimation process. In particular, we found that the Quantile Transformation allowed the OLS estimator to more accurately explore the relationships between the EPS target values and their lags, by making the linearity pattern in these data sets more obvious to the estimator. In contrast, applying the transformation to the FCF data sets distorted the

relationship between the target and features so much, that visibly these seems random.

The contribution of this chapter lies in the resulting pipelines for both the EPS and FCF data sets. For the remainder of this study, we will use the expanding window forecasting algorithm, augmented with interpolation for the FCF data sets and with Quantile Transformation for the EPS data sets. The next chapter will examine the performance of machine learning and statistical estimators also increasing the number of EPS and FCF data sets to 100.

Chapter 5

Single Estimators

5.1 Introduction

In this section, we conduct a series of experiments to examine how Machine Learning (ML) and Statistical Estimators (SE) perform on our sparse Earnings Per Share (EPS) and Free Cash Flow (FCF) data sets. In the previous Chapter 4, we outlined technical details of both the EPS and FCF data sets and selected appropriate data pipelines for each series. Specifically, we found that the EPS data sets responded well to Quantile Transformation, which forces the data to follow a uniform distribution within the 0-1 range. This transformation technique emphasizes the linearity in the EPS data, as evidenced by lower regression error across all proposed metrics (MAE, MAPE, sMAPE).

On the other hand, the weekly PCHIP of order 1 interpolation, when applied to the FCF data sets, achieved the highest average rank according to the Friedman Average Ranking test across all of our proposed error measures. Interpolation allows us to assume that the data arrives at weekly, rather than quarterly intervals, effectively smoothing outliers and enabling better generalization by the OLS regression model.

In this chapter, we expand the range of both Machine Learning (ML) and Statistical Estimators (SE) and increase the number of data sets from the initial 50 to 100. The decision to expand the data set is driven by our intention to test the generalization capabilities of our approach across a wider range of stocks from the financial markets, all of which meet similar filtering criteria. The additional 50 data sets included in this chapter were selected using the same characteristics outlined in Chapter 2: companies from the manufacturing industry, with a market capitalization greater than US $1 billion, and publicly traded on the financial markets since or prior to 2008.

While classical statistical estimation (SE) methods are commonly applied in regression problems with limited data observations [148], our primary objective is to investigate whether ML estimators can achieve the same level of accuracy, or even outperform, SE methods. Furthermore, we explore the properties of the ML and SE techniques that may influence the estimation process, especially with respect to the transformed EPS and interpolated FCF series.

Throughout this study, we distinguish between ML and SE methods by highlighting the fact that ML models apply regularization to the parameters of an estimator (whether coefficients or hyperparameters), while SE methods focus solely on minimizing a cost function. Additionally, we examine how the limited number of observations in our data sets may affect the performance of both statistical and machine learning models.

In the following sections, we first outline the methodology used in our experiments in Section 5.2, followed by an overview of the data and parameter tuning for models with tunable hyperparameters in Section 5.3. The results are presented in Section 5.4, and conclusions are drawn in Section 5.5.

5.2 Methodology

Throughout this thesis, we address a regression problem where the goal is to estimate the next quarter's EPS/FCF value using a series of past lags, along with mean and standard deviation. As discussed in previous chapters, our target variables are sparse, non-stationary series with a high number of outliers, and in most cases, they follow a non-Gaussian distribution. This chapter investigates how these characteristics affect the estimation procedure of Machine Learning (ML) and Statistical Estimators (SE) in regression models.

We assume a linear relationship between the target variable and its past lags, with no other data included in the feature set.

This assumption allows us to explore how various data patterns, such as sparsity, outliers, and non-stationarity — affect model performance while maintaining the simplicity of the estimators. Introducing non-linearity could risk obscuring the impact of these data characteristics. For instance, if we allowed for complex polynomial functions or other non-linear transformations, we might not be able to establish if worse performance results from model complexity or from the inherent properties of the data, such as its sparse size.

Despite assuming linearity, we do introduce data augmentation for the FCF and transformation for the EPS data sets, as described in the previous chapter. These adjustments aim to address the sparsity and outlier issues, frequent in our selection of data sets, while improving the overall model performance. These adjustments are summarized in the next section.

5.2.1 Data preprocessing

For both the EPS and FCF series, the input data consists of past lags, as well as the mean and standard deviation of the target variable (next quarter's EPS/FCF values). The number of

past lags is determined by the Akaike Information Criterion (AIC), with the optimal number yielding the minimum AIC forming the feature set.

Statistical tests on a subset of 50 data sets indicate that both series exhibit a tendency to trend over time. Therefore, we include the mean of the target variable to provide our estimators with insight into the direction of the series.

Given the non-stationary nature of most EPS and FCF series—meaning the data distribution may change over time—we augment the feature set with the standard deviation of the target variable. This feature helps capture changes in volatility, complementing the mean by providing information on fluctuations in the data over the observed period.

In Chapter 4, we also designed two pipelines: one utilizing Quantile Transformation for the EPS series, and the other employing PCHIP interpolation of 1st order, assuming weekly data frequency, for the FCF series. These methods were empirically shown to improve the OLS regression model, which served as a proxy due to its straightforward optimization and interpretability.

Quantile Transformation is a non-parametric feature transformation technique that maps the original data values to their respective quantiles [25]. This transformation method minimizes the impact of marginal outliers, ensuring that extreme values do not disproportionately affect the model's output. For example, a very high EPS value followed by a very low can create a large gap between a feature and its target. To address this, we configured the transformer to map values to a uniform distribution within the 0-1 range for these experiments.

PCHIP interpolation applied to the FCF data sets generates a monotonic series approximation based on the existing data points, as described in Chapter 2. The advantage of this method against other candidate interpolation techniques is its ability to preserve the shape and monotonicity of the series, avoiding the oscillations that can arise from interpolation methods

that prioritize smoothness over monotonicity [149]. In highly volatile data, PCHIP may miss local trends by producing overly generalized approximations between data points. However, empirical experiments on a subset of 50 data sets, conducted in Chapter 4, show that this interpolation method was the most effective compared to other candidates for the FCF series.

In the next section, we explain the experimental setup used in this chapter.

5.3 Experimental Setup

In this chapter, apart from increasing the sample size, we use a broader range of both statistical and machine learning regression models, described at length in Section 2. Our objective is to determine how both types of estimators handle data sparsity and to identify the properties of sparse series that influence estimator performance.

As mentioned previously, in this set of experiments we employ a selection of ML estimators that have tunable hyperparameters. The next section introduces the hyperparameters tuning process.

5.3.1 Parameter Tuning and Results Interpretation

For hyperparameter tuning, we split the data sets into 60% training, 20% validation, and 20% test subsets. We perform grid search over candidate hyperparameter combinations, selecting the set that minimizes the Root Mean Squared Error (RMSE) on the validation set. For each data set, we select the individual hyper-parameters that yield the lowest RMSE on the validation subset, allowing for tailored optimization per data set. RMSE is chosen because it is sensitive to larger errors and provides a clear metric for comparing models with different error distributions, making it appropriate for parameter tuning in this context. Proposed

hyper-parameters with relevant range of values for every ML algorithm are given in Table B-3 of the Appendix Section B.

Prior to grid searching, we employ data preprocessing to our data sets. In particular, for all EPS series, we fit the Quantile Transformer to respective stock's training data, mapping the original values to their respective quantiles, and then transform the validation set without refitting the transformer.

For the FCF data sets, subject to PCHIP interpolation, we do not interpolate the validation data to avoid over-smoothing, as it could distort the model's ability to capture dynamic trends. Instead, we first interpolate the training set, tune the hyper-parameters on the validation set, and then predict on the test set.

We carefully set the grid search space — all possible combinations of hyper-parameters, relevant to a ML estimator, to mitigate overfitting due to the sparsity of the series. For example, we limit the maximum number of trees in the Random Forest Regressor to 8 and restrict the number of neighbors to 14 for the K-Nearest Neighbors Regressor. Regularization techniques such as l1 and l2 penalties in LASSO, ARD, and BR, or tree depth in Decision Tree (DT), Random Forest (RF), and the number of nodes and layers in Multi-Layer Perceptron (MLP), are critical for controlling the fitness of the model to the data. These penalties help prevent the model from perfectly fitting the training data, which could lead to high errors when predicting on the test set.

Inference on the test set follows the procedure outlined in Chapter 4, Algorithms 2 and 3 for EPS and FCF series, respectively. The results are interpreted using the Friedman test, which ranks models based on prediction performance, followed by the Hommel's correction for statistical significance.

In the next section, we present the results of our experiments.

5.4 Results

In this section, we present the results of our experiments conducted on a set of 100 randomly selected companies that meet the criteria outlined in Chapter 2. The results are organized into several parts. First, for both EPS and FCF series, we present summary statistics and the results of the Friedman Test in Subsections 5.4.1 and 5.4.2, respectively. These statistical analyses provide an overview of the performance of the models.

Following this, Section 5.4.3 provides a detailed discussion of the obtained results, split into EPS and FCF series. This section explores the insights derived from the experiments and the significance of the findings. The results are then summarized in the concluding Section 5.5, which offers final remarks.

Note that all results in this section are based on predictions made on the test subset of the data sets. Before computing any metrics, we de-transform both the predicted and true data and exclude the artificially generated data. This ensures that the metrics reflect the performance of the model on real-world data, rather than on augmented inputs.

We begin with the description of the results for the EPS series in the next section.

5.4.1 Earnings Per Share Results

The summary statistics of the results for EPS forecasting are shown in Table 5.1. Notice that the MAPE error measure has exploded for some models. As discussed in Chapter 2, when the denominator of the MAPE formula is close to zero, it can cause the MAPE score to become extremely large. This phenomenon is observed in the EPS data presented here, where some values become unreasonably high.

From Table 5.1, it is observed that for the MAE error measure, the BR estimator has the

Table 5.1: *EPS: ML and SE forecasting results. Minimum value in each respective category is highlighted in* ***boldface***

Regressor	Error	Mean	Std. Dev.	Minimum	Median	Maximum
	MAE	0.7459	0.61091	0.02823	**0.51758**	**2.95877**
ARD	MAPE	1.15E+12	7.84E+12	0.08152	**0.66204**	7.23E+13
	sMAPE	32.5529	19.0478	4.07467	28.8716	88.7325
	MAE	1.95097	2.04048	0.04861	1.28827	12.4908
ARIMA	MAPE	7.09E+12	6.45E+13	0.13618	1.61308	6.41E+14
	sMAPE	46.9334	19.3127	7.3306	44.447	89.0417
	MAE	**0.74435**	**0.60179**	0.02678	0.54659	2.98745
BR	MAPE	1.17E+12	7.86E+12	0.07831	0.67304	7.23E+13
	sMAPE	32.3072	18.5712	3.91688	28.4101	78.508
	MAE	0.82644	0.79137	0.0502	0.59395	6.13264
DT	MAPE	1.28E+12	1.10E+13	0.11187	0.75045	1.08E+14
	sMAPE	35.6157	19.1268	6.0034	31.8366	90.2946
	MAE	0.80339	0.91137	0.028	0.53952	7.01702
HR	MAPE	1.12E+12	7.50E+12	0.08276	0.80245	6.83E+13
	sMAPE	**30.5534**	18.623	4.09854	27.6609	**74.5993**
	MAE	0.79923	0.71735	0.04121	0.5508	4.84414
KNN	MAPE	**8.95E+11**	**7.05E+12**	0.11316	0.71213	**6.89E+13**
	sMAPE	34.2269	19.028	6.1342	30.4293	89.165
	MAE	0.98384	1.14125	0.13366	0.68317	10.3683
LASSO	MAPE	1.21E+12	8.68E+12	0.15728	0.70282	8.00E+13
	sMAPE	41.0962	17.0538	8.64868	38.2159	80.1939
	MAE	1.43847	2.02078	0.22302	0.89896	13.1717
MLP	MAPE	1.50E+12	1.08E+13	0.18988	0.84936	9.31E+13
	sMAPE	51.578	**15.5537**	11.0019	51.2033	86.8683
	MAE	0.75165	0.77237	**0.02582**	0.51874	6.21485
OLS	MAPE	1.14E+12	8.10E+12	**0.07612**	0.69636	7.69E+13
	sMAPE	31.0005	19.3254	**3.77864**	27.981	84.7159
	MAE	0.81437	0.6894	0.05299	0.6015	4.29442
RF	MAPE	2.19E+12	**1.49E+13**	0.11973	0.67336	1.11E+14
	sMAPE	35.1476	18.9726	6.4123	31.5862	90.2034
	MAE	0.88352	1.15328	0.02781	0.60869	8.0369
RLM	MAPE	1.34E+12	9.78E+12	0.08293	0.81141	9.35E+13
	sMAPE	31.2529	18.7271	4.07287	**27.6074**	80.0174
	MAE	0.78747	0.739	0.03596	0.53413	5.0594
SES	MAPE	1.06E+12	8.21E+12	0.08836	0.70188	8.00E+13
	sMAPE	32.949	19.8265	4.5712	28.62	90.3022
	MAE	0.92742	1.04882	0.0331	0.6779	8.01863
WLS	MAPE	1.23E+12	9.90E+12	0.09488	0.91982	9.70E+13
	sMAPE	32.6783	18.0939	4.8153	28.7732	80.4459

minimum average error of 0.74435. For other error measures, the KNN estimator shows the lowest MAPE error of 1.21E+12. Additionally, the HR estimator demonstrates the lowest average sMAPE of 30.5534. Although some classical statistical estimators perform similarly across the sample of 100 data sets, ARIMA and WLS perform poorly compared to the ML algorithms. The MLP algorithm, while ranking among the worst, demonstrates the lowest standard deviation for sMAPE errors, suggesting that its regression error is concentrated around the average error for most data sets. These observations are further supported by the median values across the error measures. The ARD estimator results in the lowest median MAE and MAPE. The BR estimator, along with OLS, HR, LASSO, RLM, and RF, are among the better-performing estimators across regression error metrics. Consider also the difference between median errors in training and test sets, provided in Table 5.2. We propose to study median values to avoid the influence of extremely high or low errors.

Median train and test set error values in Table 5.2 indicate that, overall, our proposed estimators tend to overfit the training data — they are biased toward training observations. Notably, the KNN estimator achieves near-zero MAE, MAPE, and sMAPE on the training set, but this performance does not transfer to the test set.

The ARIMA estimator, on the other hand, exhibits high variance in the MAPE error measurement: it shows relatively low test set error despite having a significantly higher training error. Conversely, its MAE performance shows the overfit, with training score nearly double the test, and almost a 'fair' fit with sMAPE error measurement.

Across other estimators, we observe varying degrees of bias. Importantly models that demonstrate stronger overall performance on the out-of-sample test subset, show a smaller degree of bias, i.e. ARD could be said to fit well, with relatively low difference between median train and test MAPE and sMAPE scores.

Table 5.2: *EPS: Median of Train (marked 'Train') and Test (marked 'Test') Set Errors Across a Sample of 100 Data Sets. Values in* ***boldface*** *indicate the lowest value in the respective column.*

MAE			**MAPE**			**sMAPE**		
Regressor	**Train**	**Test**	**Regressor**	**Train**	**Test**	**Regressor**	**Train**	**Test**
ARD	0.1741	**0.5175**	ARD	0.7033	**0.6620**	ARD	22.8248	28.8716
ARIMA	0.6345	1.2882	ARIMA	3.5172	1.6130	ARIMA	44.7970	44.4469
BR	0.1760	0.5465	BR	0.7086	0.6730	BR	22.1495	28.4101
DT	0.1650	0.5939	DT	0.6829	0.7504	DT	21.6449	31.8365
HR	0.1582	0.5395	HR	0.7031	0.8024	HR	19.9665	27.6609
KNN	$\mathbf{2E-16}$	0.5508	KNN	$\mathbf{3E-16}$	0.7121	KNN	$\mathbf{2E-14}$	30.4292
LASSO	0.2301	0.6831	LASSO	0.7312	0.7028	LASSO	28.8028	38.2158
MLP	0.3020	0.8989	MLP	1.3128	0.8493	MLP	37.4301	51.2032
OLS	0.1739	0.5187	OLS	0.6832	0.6963	OLS	21.9309	27.9810
RF	0.1607	0.6014	RF	0.6889	0.6733	RF	21.4788	31.5862
RLM	0.1640	0.6086	RLM	0.7156	0.8114	RLM	20.6188	**27.6074**
SES	0.4391	0.5341	SES	3.4812	0.7018	SES	47.6743	28.6199
WLS	0.1770	0.6778	WLS	0.7186	0.9198	WLS	23.1317	28.7731

Table 5.3: *EPS: Friedman Test of forecasting errors. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Regressor	**Rank**	**p-val**	**Regressor**	**Rank**	**p-val**	**Regressor**	**Rank**	**p-val**
OLS	4.86	control	BR	5.07	control	HR	4.08	control
HR	4.90	9.42E-01	ARD	5.41	5.35E-01	OLS	4.69	2.74E-01
BR	4.95	9.42E-01	SES	5.44	5.35E-01	RLM	4.76	2.74E-01
ARD	5.03	9.42E-01	OLS	5.82	5.31E-01	BR	5.17	1.49E-01
SES	5.31	9.42E-01	KNN	6.10	2.51E-01	SES	5.59	**2.62E-02**
RLM	5.94	2.54E-01	RF	6.34	1.07E-01	ARD	5.79	**1.02E-02**
KNN	6.28	6.38E-02	DT	6.43	6.88E-02	WLS	5.90	**6.13E-03**
RF	7.08	**4.50E-04**	HR	6.59	**4.34E-02**	KNN	6.71	**1.46E-05**
DT	7.11	**3.57E-04**	LASSO	6.77	**1.74E-02**	RF	7.55	**3.10E-09**
WLS	7.25	**1.49E-04**	RLM	7.50	**1.03E-04**	DT	7.96	**2.19E-11**
LASSO	9.25	**2.21E-14**	WLS	8.22	**1.24E-07**	LASSO	10.03	**6.67E-26**
MLP	11.03	**9.70E-28**	MLP	9.21	**8.06E-13**	ARIMA	11.30	**7.22E-38**
ARIMA	11.95	**1.72E-36**	ARIMA	12.09	**8.86E-36**	MLP	11.49	**7.84E-40**

To rank our selection of regression estimators by their respective performance, we conduct the Friedman average ranking test. A statistically significant difference between the model with the lowest rank (the 'control' algorithm) and the other models is determined using the Hommel's post-hoc pairwise comparison between the control and rest estimators. The null hypothesis states that there is no difference between two groups. A p-value less than 0.05 indicates statistical significance at the $\alpha = 5\%$ level, meaning that the algorithm outperforms other candidates (reject the null). The Friedman results are shown in Table 5.3.

Results in Table 5.3 indicate that for the MAE error, OLS serves as the control algorithm with an average rank of 4.8687, significantly outperforming ARIMA, DT, LASSO, MLP, RF, and

WLS.

For MAPE errors, BR is the control algorithm, with an average rank of 5.0707 across all 100 data sets, significantly outperforming HR, ARIMA, LASSO, MLP, RLM, and WLS.

Finally, in terms of sMAPE, HR acts as the control estimator for this group, with an average rank of 4.0808, significantly outperforming SES, ARIMA, DT, KNN, LASSO, MLP, RF, and WLS.

So far, we have observed that simpler optimization functions tend to perform better across the error measures: linear estimators (both statistical and ML), show better performance than more complex ones. However, the difference between statistical and linear ML models is not substantial, as some statistical models are outperformed by ML algorithms. This trend is evident in all three error measures: in the MAPE group, the BR estimator significantly outperforms both the WLS and ARIMA estimators. Similarly, HR, as the control estimator for sMAPE, significantly outperforms both ARIMA and WLS.

Although statistical estimators like OLS serve as the control for MAE, they do not significantly outperform more complex ML models like ARD, BR, HR, and KNN. Therefore, in sparse datasets, certain ML algorithms demonstrate performance comparable to, or even better than, statistical models. In the next section, we examine how similar regression model types perform on FCF data sets.

5.4.2 Free Cash Flows Per Share Results

As in the previous subsection, we first present the summary statistics for the performance of our candidate model types, outlined in Table 5.4.

According to the results in Table 5.4, in the MAE error category, the HR estimator achieves the lowest average score of 0.9166, with RLM and OLS showing comparable results. LASSO demonstrates the lowest average MAPE of 1.465855, followed by KNN and RF. The RLM

Table 5.4: *FCF: ML and SE forecasting results. Minimum value in each respective category is highlighted in* ***boldface***

Regressor	Error	Mean	Std. Dev.	Minimum	Median	Maximum
	MAE	0.94792	1.09135	0.09922	0.64635	9.48691
ARD	MAPE	1.58169	2.47646	0.1478	0.99207	21.8384
	sMAPE	36.3373	18.7468	7.40593	35.1177	83.3887
	MAE	1.62076	1.813	0.09866	1.067	14.1239
ARIMA	MAPE	3.48169	4.81037	0.22893	2.09918	33.2296
	sMAPE	47.9918	19.7963	11.0375	50.2808	89.3059
	MAE	0.94475	1.05385	0.10174	0.65806	9.00807
BR	MAPE	1.58989	2.43898	0.14709	1.00485	21.4453
	sMAPE	36.2016	18.5686	7.39127	34.5688	79.3623
	MAE	1.19463	1.59065	0.0918	0.89374	14.0693
DT	MAPE	2.00571	2.78388	0.16909	1.14824	19.9108
	sMAPE	40.8114	18.7168	8.39649	41.2244	81.339
	MAE	**0.9166**	**0.9751**	0.10006	**0.64483**	8.03927
HR	MAPE	1.58611	2.5036	0.1454	0.96463	21.9906
	sMAPE	**35.6955**	18.0524	7.23168	**34.0282**	77.2455
	MAE	0.97111	1.00077	0.10391	0.65088	8.02428
KNN	MAPE	1.49012	2.22453	0.16313	0.93762	19.1861
	sMAPE	37.6241	17.9849	7.75897	36.6972	**73.5482**
	MAE	0.96352	0.98387	0.19708	0.69566	8.21571
LASSO	MAPE	**1.46585**	2.54057	0.16362	**0.90572**	23.8392
	sMAPE	41.028	19.9654	8.53684	38.4035	98.2851
	MAE	1.2081	1.73105	0.15709	0.80848	16.4313
MLP	MAPE	1.575	2.65591	0.26093	1.05438	23.6316
	sMAPE	44.6809	**17.6801**	15.7884	41.7842	89.5712
	MAE	0.94332	1.03875	0.10167	0.65126	8.79351
OLS	MAPE	1.61222	2.44855	0.14447	1.02872	20.718
	sMAPE	36.5929	18.7699	7.23404	34.8286	84.2797
	MAE	1.01936	0.99624	0.0989	0.70349	**7.34269**
RF	MAPE	1.53956	**2.04565**	0.16807	1.05558	**15.9174**
	sMAPE	38.9161	18.4643	8.69915	39.2003	82.2
	MAE	0.91958	0.98416	0.10146	0.64755	8.18703
RLM	MAPE	1.6044	2.53489	**0.14405**	0.97436	21.8535
	sMAPE	35.8191	18.0169	**7.15913**	34.6885	77.0579
	MAE	1.49971	1.8155	**0.08793**	0.97871	14.9667
SES	MAPE	3.07685	4.24365	0.1909	1.97855	31.2205
	sMAPE	46.295	20.0886	9.13817	47.3676	85.007
	MAE	1.39648	1.57233	0.10256	0.93168	12.9202
WLS	MAPE	2.39587	2.84965	0.14887	1.5404	20.9617
	sMAPE	43.0903	19.7382	7.27023	45.6705	79.96

estimator, which previously ranked highly based on average MAE and MAPE, shows the lowest average sMAPE of 35.69552. Across all three error measures, ARIMA and SES demonstrate consistently worse average performance.

Regarding the median of our error measures, the HR estimator stands out with the lowest median MAE and sMAPE. The lowest median MAPE is found for the LASSO estimator. Additionally, ARD, RLM, KNN, and HR exhibit lower median error measures, whereas, confirming earlier observations, ARIMA, SES, and MLP remain among the worst-performing estimators.

In addition to the descriptive statistics, we also check the median train and test error scores across our sample of 100 data sets, given in Table 5.5.

Table 5.5: *FCF: Median of Train (marked 'Train') and Test Set (marked 'Test') Errors Across a Sample of 100 Data Sets. Values in* ***boldface*** *indicate the lowest value in the respective column.*

MAE			**MAPE**			**sMAPE**		
Regressor	**Train**	**Test**	**Regressor**	**Train**	**Test**	**Regressor**	**Train**	**Test**
ARD	0.2948	0.6463	ARD	1.3562	0.9920	ARD	34.4859	35.1176
ARIMA	0.6362	1.0670	ARIMA	1.6663	2.0991	ARIMA	69.2057	50.2807
BR	0.2955	0.6580	BR	1.3576	1.0048	BR	34.2572	34.5688
DT	0.2711	0.8937	DT	1.4205	1.1482	DT	35.0150	41.2244
HR	0.2899	**0.6448**	HR	1.3263	0.9646	HR	33.1740	**34.0281**
KNN	**0**	0.6508	KNN	**0**	0.9376	KNN	**0**	36.6971
LASSO	0.3121	0.6956	LASSO	1.4328	**0.9057**	LASSO	39.9324	38.4034
MLP	0.3293	0.8084	MLP	1.8183	1.0543	MLP	38.0742	41.7841
OLS	0.2961	0.6512	OLS	1.3728	1.0287	OLS	34.8306	34.8286
RF	0.2045	0.7034	RF	1.0843	1.0555	RF	27.4640	39.2002
RLM	0.2921	0.6475	RLM	1.3316	0.9743	RLM	34.1464	34.6885
SES	0.8168	0.9787	SES	4.4565	1.9785	SES	56.2883	47.3676
WLS	0.4003	0.9316	WLS	1.6890	1.5404	WLS	41.2696	45.6704

From Table 5.5, we observe that many estimators exhibit overfitting behavior across the

data sets. Similar to the EPS series, the KNN estimator yields zero error on the training set but performs poorly on the test set, indicating significant overfitting.

Interestingly, the ARD estimator displays signs of underfitting, particularly evident in the MAPE error score. In contrast, the better-performing estimators — such as ARD, BR, HR, LASSO, and OLS — generally exhibit more balanced performance, with training and test set errors closely aligned. This is especially visible from their sMAPE scores.

Meanwhile, other models, including ARIMA and MLP, show a relatively high degree of bias, as indicated by the discrepancy between their training and test errors.

In line with the EPS data sets, we conduct the Friedman test to rank our candidate model types based on the resulting error measures. The Friedman average ranking test is conducted in two steps. In the first step we perform ranking,where each data set is assigned an estimator that shows the lowest error (MAE, MAPE or sMAPE) on this data, with a higher rank indicating lower error. The average rank, referred to as 'Rank' in Table 5.6, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control method (marked as 'control' in Table 4.2) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that all methods (i.e., the OLS regression errors, measured as MAE, MAPE and sMAPE, respectively) are drawn from the same distribution.

Table 5.6 summarizes the results of this statistical test, where statistically significant differences at the 5% level between the top-performing ('control') model and the worst-performing model types are highlighted in boldface.

As suggested by results in Table 5.6, with MAE as the error measure, RLM serves as the control model, achieving an average rank of 4.06. It is statistically significant compared to a

Table 5.6: *FCF: Friedman Test of forecasting errors. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			**MAPE**			**sMAPE**		
Regressor	**Rank**	**P-Val**	**Regressor**	**Rank**	**P-Val**	**Regressor**	**Rank**	**P-Val**
RLM	4.06	control	LASSO	4.94	control	RLM	4.3	control
HR	4.09	9.57E-01	HR	5.05	8.42E-01	HR	4.43	8.13E-01
ARD	4.61	6.36E-01	KNN	5.14	8.42E-01	BR	4.92	5.21E-01
BR	4.74	4.77E-01	RLM	5.22	8.42E-01	ARD	4.96	4.62E-01
OLS	4.84	4.70E-01	BR	5.75	5.65E-01	OLS	5.26	3.25E-01
KNN	6.25	**3.50E-04**	ARD	5.82	4.40E-01	KNN	6.33	**1.14E-03**
LASSO	6.25	**3.50E-04**	OLS	5.9	3.53E-01	LASSO	7.04	**3.92E-06**
RF	7.19	**9.26E-08**	RF	6.25	1.22E-01	RF	7.16	**1.45E-06**
DT	8.87	**1.98E-17**	MLP	6.72	**9.84E-03**	DT	8.23	**7.71E-12**
MLP	8.95	**6.09E-18**	DT	8.18	**3.63E-08**	WLS	8.82	**2.04E-15**
SES	9.92	**1.94E-25**	WLS	9.54	**6.70E-16**	SES	9.4	**2.04E-19**
WLS	10.02	**3.00E-26**	SES	10.78	**3.15E-25**	MLP	9.81	**1.60E-22**
ARIMA	11.21	**1.85E-37**	ARIMA	11.71	**1.20E-33**	ARIMA	10.34	**6.63E-27**

set of estimators, including KNN, LASSO, RF, DT, WLS, SES, MLP, and ARIMA. In the MAPE error category, LASSO ranks highest, with an average rank of 4.94, and is significant against MLP, DT, WLS, SES, and ARIMA. Lastly, in the sMAPE category, RLM ranks highest with an average score of 4.3 and is significant compared to KNN, LASSO, RF, DT, WLS, SES, MLP, and ARIMA.

Similar to the EPS data sets, we observe that more complex estimators, such as MLP, ARIMA, and tree-based models, tend to perform worse than simpler estimators like ARD, BR, and HR. Interestingly, some ML estimators in both the EPS and FCF data sets perform as well as, or even outperform, statistical models. In the next section, we explore the reasons behind this phenomenon.

5.4.3 Results Discussion and Analysis

For a more thorough discussion, we split this section into two parts: the first describes the two data sets—one exhibiting high error scores across estimators and the other with low error scores. This distinction helps highlight properties that may have influenced the estimation process, either positively or negatively. The following subsection details the behavior of estimators on these data sets.

In the main body of this chapter, we present figures for two companies that, in both the FCF and EPS cases, show the highest and lowest regression errors across all proposed metrics. Similar figures for eight other companies, randomly chosen from a sample of 98 data sets (excluding the ones with the highest and lowest errors), are provided in the Appendix, Section B of this thesis. Hewlett-Packard Inc. (HP) is a data set that typically exhibits higher error scores in both the EPS and FCF cases, while Fastenal Co. (FAST) demonstrates better predictability for the quarter-ahead EPS/FCF value. Results for other 8 companies are given in

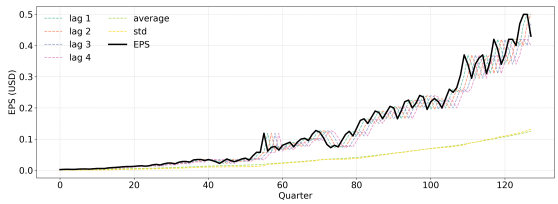
the Appendix, Section B. Notice that the results of fitting various estimators on these data sets fall between the two extreme cases, which we use as representative examples throughout this chapter.

Our analysis begins with an overview of observations about the data sets the next Subsection.

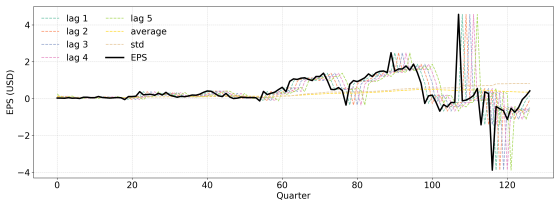
Data Properties

We begin by analyzing the behavior of the target variables, EPS and FCF, and their relationships with the features: past lags, mean, and standard deviation. Figure 5.1 show the full data sets (training, validation and test subsets) for the FAST and HP data sets, EPS and FCF series, respectively. These figures display the target variable alongside its features in the train, validation and test subsets, providing a visual representation of the dataset structure.

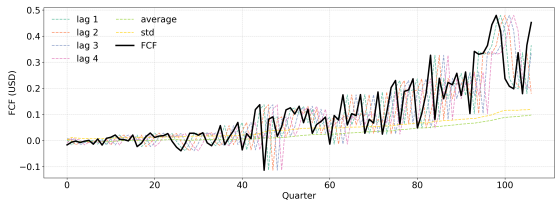
Figure 5.1: *Representative FAST and HP Data Sets: EPS and FCF series*



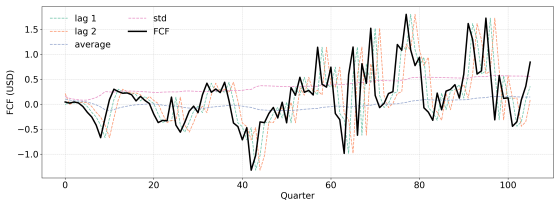
(a) EPS: FAST



(b) EPS: HP



(c) FCF: FAST



(d) FCF: HP

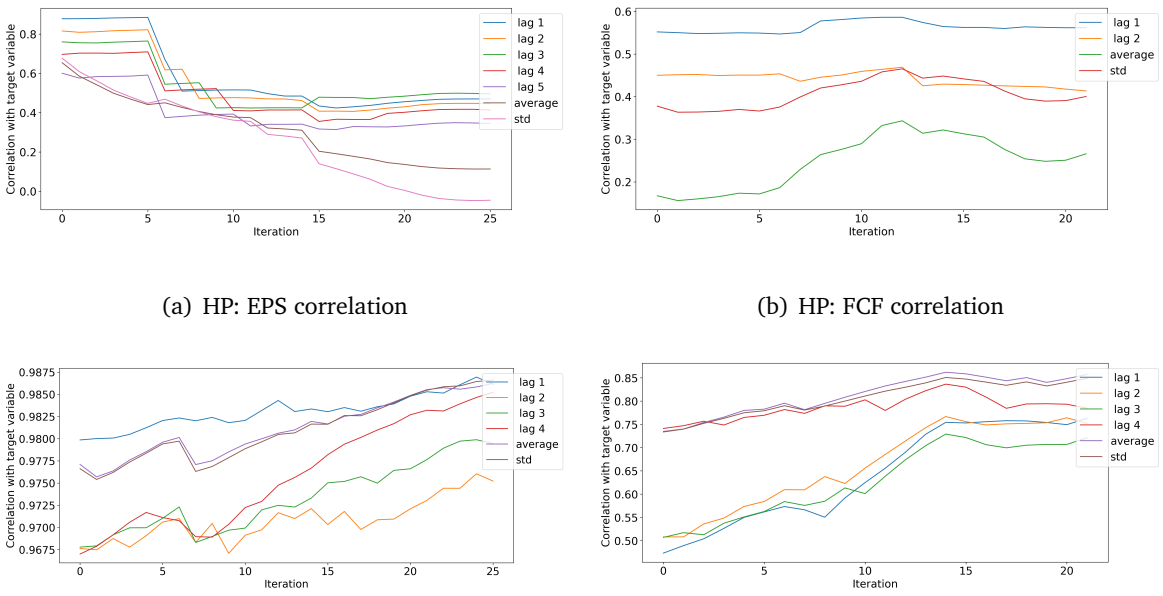
Notice that in both FCF and EPS data sets, the target variable and its features show dynamic

behaviour, with sudden up-/down-ward movements. Meanwhile, the mean and standard deviation features appear smoother, as they filter out short-term fluctuations and align with the overall trend of the target variable.

As mentioned in the previous chapter, the FCF datasets tend to exhibit higher volatility than the EPS datasets. However, the EPS plots display large spikes too, particularly towards the end of the HP dataset, followed by smaller fluctuations. These spikes impact the relationships within the data: the higher the variance of a feature, the stronger its correlation with the target variable. Consequently, these fluctuations inflate the correlation coefficients.

Periods of high volatility persist across the lags in the dataset. For example, in datasets with large errors (indicating poor model fit), the correlations are unstable, as shown in Figure 5.2, which illustrates the correlation of features with the target variable across iterations in the test set.

Figure 5.2: *EPS and FCF correlations of the target variable with its features through test-set iterations: comparison between poorly and well-fitted data sets*



(a) HP: EPS correlation

(b) HP: FCF correlation

(c) FAST: EPS correlation

(d) FAST: FCF correlation

As shown in Figure 5.2, the EPS correlations are unstable and tend to decrease with each iteration in the test set. This pattern is consistent across data sets with high error rates. The FCF data set, however, shows more stability: correlations for lags 1 and 2 are roughly 0.6 and 0.45, respectively. The 'average' feature increases its influence over time, though the correlations remain relatively low, suggesting a moderate dependence of the target variable on its features. In contrast, datasets that perform better, such as the FAST dataset, exhibit high and stable correlations.

In conclusion, in datasets with better performance, the target variable shows a high and stable correlation with its features, which improves as the training data expands. Both types of datasets exhibit volatility, but high-performing datasets show a trend, while lower-performing datasets do not. The volatility in the features tends to cluster, and extreme fluctuations in the target variable are propagated to the lag features. Meanwhile, the moments (mean and standard deviation) are relatively smoother.

The following section will analyze the performance of the estimators and explore how these data properties influence the estimation results.

Impact on Estimators

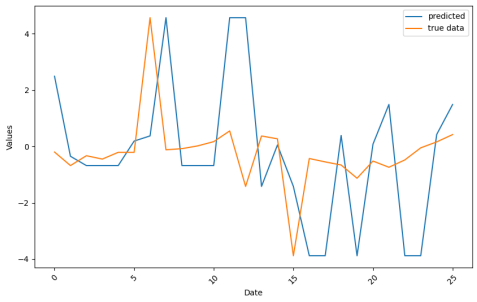
In this section, we examine the performance of various estimators. Both our EPS and FCF series exhibit similar patterns: when within-sample correlations are high, the error scores tend to be lower, and vice versa. However, we have observed that machine learning models often perform similarly to statistical estimators.

For instance, consider the performance of the ARIMA model, one of the estimators that generally yields higher error scores. Its performance on the less stable HP dataset is shown in Figure 5.3.

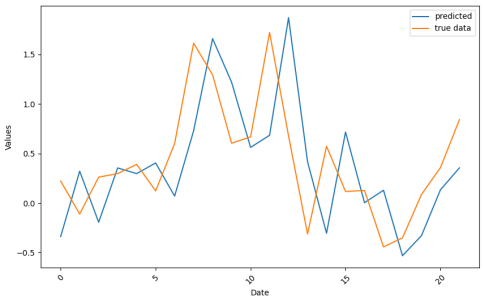
This estimator captures variability in the dataset but tends to make more volatile predictions during less volatile periods, as shown by the test set predictions for the HP dataset in Figure 5.3. This suggests that the coefficients were properly optimized, although at times the predictions are lagged by one period.

In its best-case dataset, where correlations between the target and feature variables are stable and increasing, such as in the FAST dataset, ARIMA successfully captures the trend in the series but misses small increments, as shown in Figure 5.3. This conclusion is further validated by other datasets, as depicted in Figures B-5 for EPS predictions and B-6 for FCF series predictions.

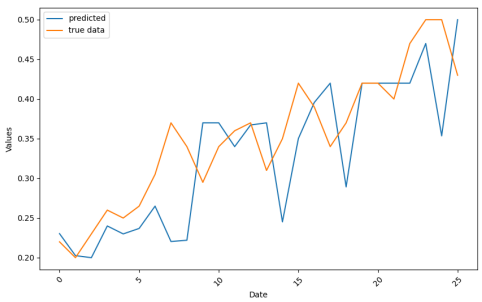
Figure 5.3: *ARIMA prediction results: EPS and FCF forecasts on HP and FAST test sets*



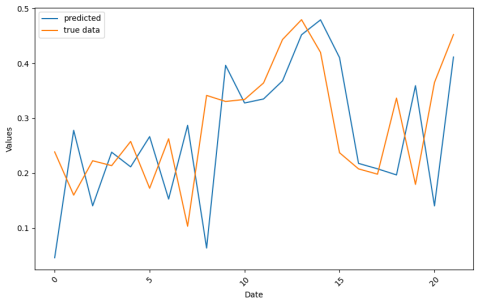
(a) HP: EPS test set predictions



(b) HP: FCF test set predictions



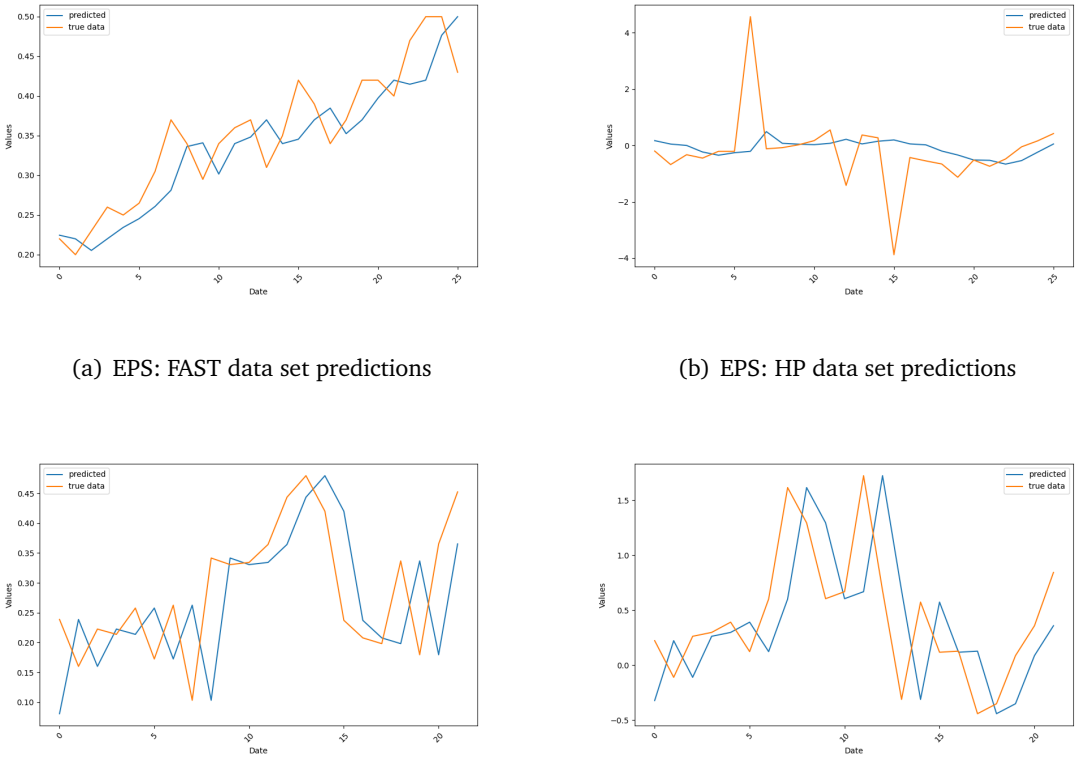
(c) FAST: EPS test set predictions



(d) FAST: FCF test set predictions

The SES model, another statistical estimator, yields different results between FCF and EPS predictions. For EPS predictions, it infers a straight line through the data, capturing the trend rather than the variance. While this minimizes regression error in datasets like FAST, it overlooks large deviations in datasets like HP. Generally, SES predicts the FCF data sets, visibly better than the EPS ones. Predictions of the SES model for EPS and FCF datasets, against the actual data, are provided in Figure 5.4.

Figure 5.4: *SES prediction results for EPS and FCF on FAST and HP data sets*



(a) EPS: FAST data set predictions

(b) EPS: HP data set predictions

(c) FCF: FAST data set predictions

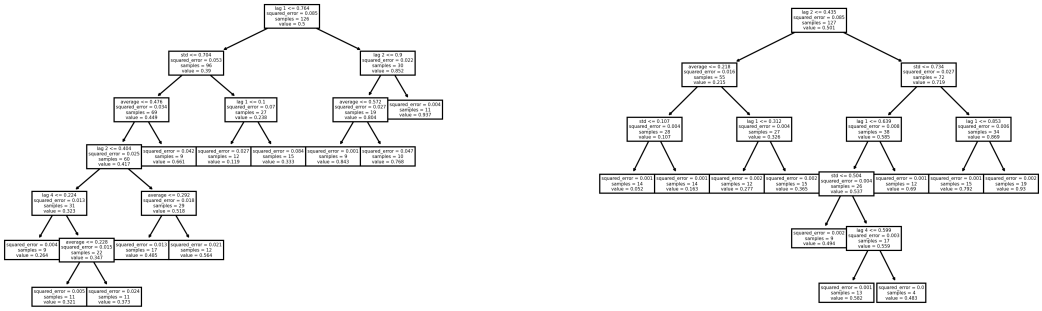
(d) FCF: HP data set predictions

It is possible that the observed effect is due to interpolation, which has a smoothing effect on the series. Additionally, in the FCF series, HP dataset, the predicted values closely resemble the previous step in the ground-truth data. Since all of the features used to fit this model

are past lags, it is apparent that the SES model is simply copying the first lag of the series, ignoring the rest of the information. This effect is not useful for the modeling task: rather than predicting the future value, we get an exact approximation of one of its lags.

Another notable effect occurs in tree-based estimators, specifically Decision Trees (DT) and Random Forests (RF). The fitted tree structures in the FCF and EPS cases differ. Therefore, consider the best-fit tree for both selected EPS datasets in Figure 5.5.

Figure 5.5: *EPS: Decision Tree Structures for HP and FAST datasets*



(a) HP dataset decision tree

(b) FAST dataset decision tree

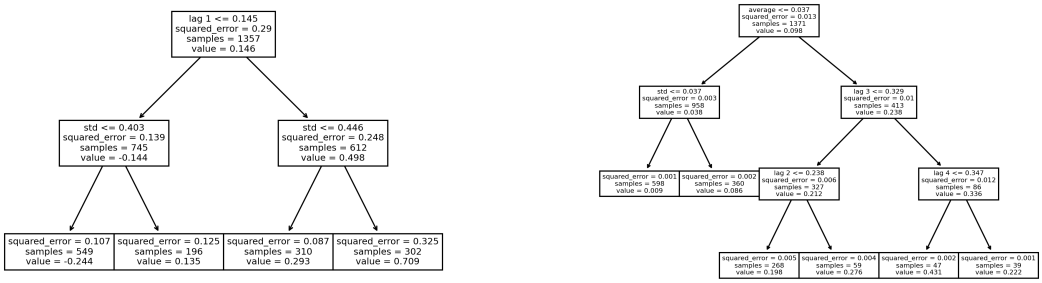
In the HP tree (which corresponds to worse-performing predictions depicted as (a) in Figure 5.5), there is a leftward skew. The 'lag 1' feature drives the inference, meaning that if it fails to meet a certain threshold, the estimation process can be misled. This is particularly true given the dynamic correlations observed in this dataset.

In contrast, the better-performing FAST tree in Figure 5.5 has a more balanced structure, with more estimated nodes explored during the training stage. Given that it is one of the better-performing datasets, this structure appears to be more suitable for future inference.

More balanced structures are also observed in the FCF datasets. Here, both the HP and

FAST datasets exhibit an equal number of nodes in each leaf, as shown in Figure 5.6.

Figure 5.6: *FCF: Decision Tree Structures for HP and FAST datasets*



(a) HP dataset decision tree

(b) FAST dataset decision tree

In the case of the FCF series, the HP dataset indeed builds a more balanced, granular tree. However, this estimator fails to capture the relationships between the features and the target, as indicated by the high squared error (0.087) at one of the lower nodes, compared to the much smaller errors in the FAST dataset.

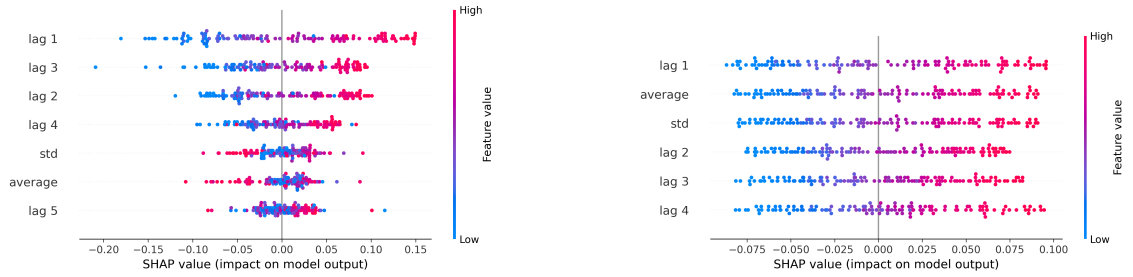
In the FAST dataset, shown in subplot (b) of Figure 5.6, the DT estimator performs better at inference, minimizing errors at every node. This can be attributed to the high correlation between the features and the target, which makes it easier for the DT estimator to make accurate inferences.

The key observation from examination of the tree structures for FCF and EPS data sets suggest that the highly imbalanced trees (ones with nodes on either side being deeper than nodes on the other), are a sign of overfitting, and therefore of the poor generalization of a DT estimator to the unseen data. Our argument is due to the fact that in the majority of cases, data sets with lower error measures typically show a more balanced tree structure, whereas

trees displaying a high bias are typically deeper.

Despite using different optimization approaches, non-parametric methods face similar challenges. An examination of Shapley values force plots, which show how each feature observation influences the estimator's outcome [150], reveals that for the KNN forecasting model, single observations have the most impact. Figure 5.7 illustrates the contribution of each feature in estimating the EPS target variable.

Figure 5.7: *EPS: KNN Shapley values force plots*

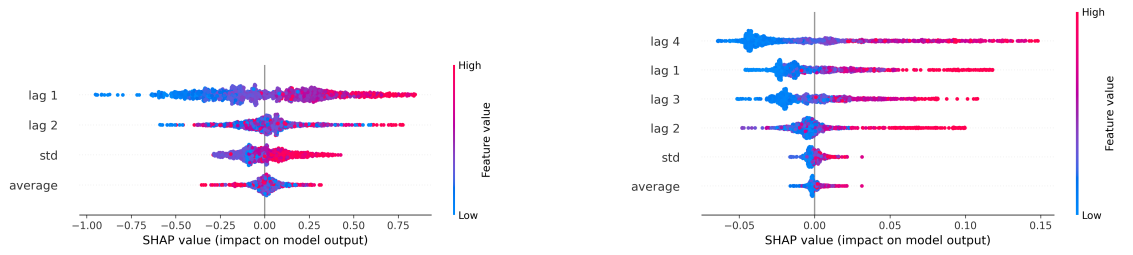


(a) HP Shapley values

(b) FAST Shapley values

The FCF data sets, subject to interpolation, display similar Shapley value patterns, as shown in Figure 5.8.

Figure 5.8: *FCF: KNN Shapley values force plots*



(a) HP Shapley values

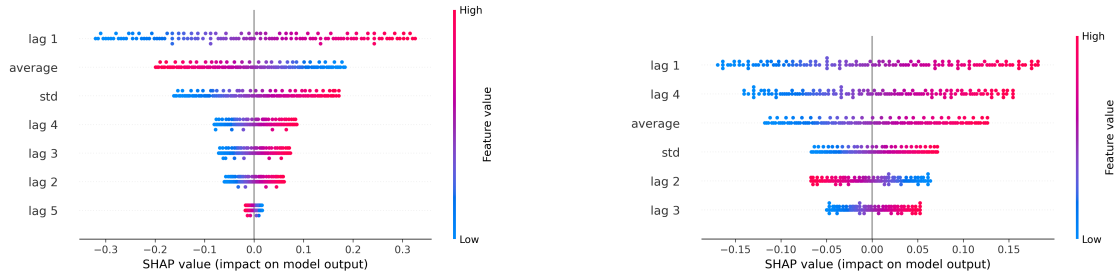
(b) FAST Shapley values

In the HP data set (a set with higher error measures), a few data observations significantly impacted the performance of the KNN estimator: some data records in the 'lag 1' vector have a

Shapley value close to -1.0. In contrast, in the FAST data set (a better-performing dataset with lower error measures), the values are more evenly distributed across the scale. Interestingly, the most influential feature, 'lag 4,' contains many data points with a maximum impact of 0.15. Therefore, all features contributed to the final outcome. While outliers are present in the figure, their contribution is likely neglected by the estimator, given the high impact of other features. It is also noteworthy that, in better-performing series, all features had some impact on the outcome, whereas in worse-performing ones, typically only one feature dominated the estimation.

We observed a group of better-performing regression model types in our selection. Analyzing their Shapley values confirms our earlier conclusion about the equal contribution of features in better-performing estimators. In particular, consider the Shapley values for the HR estimator — one with a lower ranking across all error measures —– depicted in Figure 5.9.

Figure 5.9: *EPS: HR Shapley values force plots*



(a) HP Shapley values

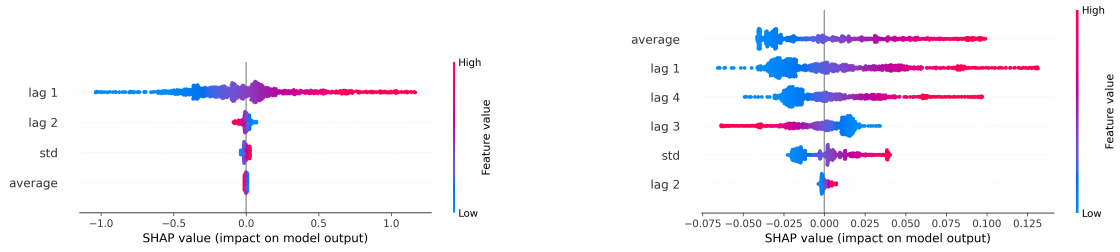
(b) FAST Shapley values

In Figure 5.9, there is rarely a single data observation that affects the outcome the most. Additionally, the range of feature impact on the estimation differs: in the HP data set, the most influential feature, 'lag 1', has Shapley values in the range of +/- 0.30, while the second most influential feature has values in the +/- 0.20 range. In contrast, for the FAST data set, the highest impact is made by the 'lag 1' feature with Shapley values in the +/- 0.15 range, while

the second most influential feature, 'lag 4', has an impact in the +/- 0.10 range.

This phenomenon is consistent in the interpolated FCF data sets, as shown in Figure 5.10. The HP data set, which exhibits higher error, displays a pattern previously observed in the EPS series. The key difference lies in the magnitude of the impact: 'lag 1' has an impact in the range of +/- 1.0, while the remaining features have almost no influence. In contrast, the FAST data set shows the opposite pattern.

Figure 5.10: *FCF: HR Shapley values force plots*



(a) HP Shapley values

(b) FAST Shapley values

A similar effect is observed with the other top-performing estimators, including OLS, RLM, LASSO, ARD, and BR, and other data sets given in the Appendix, Section B, Figures B-13, B-15 for the EPS predictions and Figures B-14, B-13 for the FCF predictions. These better performing models tend to assign the highest weight to a single feature in data sets with higher errors, while distributing the impact more evenly across features in data sets with better performance. The next section will summarize the results and findings of this chapter.

5.5 Chapter Conclusions

In this chapter, we conducted a series of experiments using a selection of machine learning (ML) and statistical modeling (SM) estimators on a sample of 100 datasets, covering both EPS and FCF time series. Our results suggest that, in sparse series, ML estimators can perform

comparably to traditional statistical models typically used for such problems.

Specifically, the LASSO, BR, ARD, and HR estimators outperformed the ARIMA, WLS, and SES models. Non-parametric models such as KNN, Decision Trees (DT), and Random Forests (RF) tended to overfit, performing worse than the top models but still achieving lower MAE, MAPE, and sMAPE scores than ARIMA and SES. The MLP estimator, likely due to its complex structure and estimation process, exhibited the poorest performance across both EPS and FCF series.

Robust models, those utilizing Huber loss (Robust) as the objective function, tended to show better performance. We attribute the poor performance of certain datasets to several factors.

First, recall that we used an expanding window approach. As we iterated through the test subsets of each dataset, the number of training data points increased. In datasets that showed lower error across our proposed metrics, we observed a stable high positive or negative correlation, which either remained constant or increased as the training data expanded.

Second, in certain datasets, the features with the highest variance generally had a greater impact on the regression outcome. The highest variance was typically observed in features that contained outliers from the target series. Due to the sparsity of our data, outliers were difficult to distinguish from regular data points, leading to biased estimations. In contrast, in datasets where estimators performed better, the Shapley values indicated that the impact was more evenly distributed across the feature set. This allowed each feature to contribute to the estimation, with outliers exerting less influence on the overall prediction.

Third, we recognize that different estimators capture different aspects of the relationship between the target variable and its features. This is evident from the fact that estimators sometimes place varying weights on the same set of features.

We propose that combining information from both ML and SM estimators could enhance the solution to our estimation problem. Specifically, since each estimator places different weights on various features, averaging their performance may yield better results. Furthermore, the 'exchange of information' can also be achieved by using the estimated coefficients from different models, as these will differ across estimators. This approach is explored next.

Chapter 6

Transfer Learning and Averaging

6.1 Introduction

In the previous Chapter 5, we examined the performance of ML and SE estimators on the sparse EPS and FCF datasets. Building on top of the results from Chapter 4, we applied two different pre-processing pipelines: Quantile Transformation to the EPS series and PHCIP interpolation to the FCF series, respectively. After the preprocessing applied, we fed the data to the ML and SE regression estimators in order to measure the performance of the two types of the quantitative modeling techniques, with past lags, mean and standard deviation serving as the input features.

Using the Friedman ranking test procedure applied to the set of error measures, namely Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and symmetric Mean Absolute Percentage Error (sMAPE), we identified a group of estimators that consistently outperform other candidate estimators. This group includes both ML and SM regression models, specifically: ARD, BR, OLS, LASSO, RLM, and HR. Additionally, in datasets with lower error measures, these estimators tend to be impacted by a larger number of features:

empirically, it is shown that in the less performing regression model-types, a single input feature is often used by such estimators to make a prediction. The existence of a group of consistently better performing estimators, rather than a single top-performing model-type, suggests that we can choose a regressor suited to a given dataset. However, due to the relatively small size of the typical datasets in our selection, even these top-performing estimators tend to overfit to training data, meaning that a model's performance on the training or validation subset may not be used as indication of better out-of-sample performance of such estimator, as evident from examination of median errors across our sample of 100 data sets in train and test subsets. Further, it is hard to determine if statistical or machine learning estimators perform better — across our sample of data sets ML and SE model-types tend to show similar performance, on average.

Building on top of this argument, in this chapter we propose and test methodology that allows to combine predictions of Statistical and Machine Learning estimators by means of averaging and transfer learning. In particular, we let Bayesian Model Averaging determine which model(s) performs better in the transfer learning setting, and weigh those that show better performance more when obtaining the final prediction. Combining Statistical and ML estimators in such a manner helps to overcome their respective limitations (in a given case).

For each dataset, the prediction is done in two steps. The first step involves obtaining a measure of quality for each model, expressed as 'evidence' of high performance in the TL domain. Without loss of generality, we assume that, if we have 100 datasets (i.e. companies), we may use 99 of these as the TL domain, and use BMA to make a prediction for the remaining one dataset (i.e. the 'target' company). In order to obtain this measure of quality for each model, we train the respective model normally on each of the 99 datasets, resulting in 99 unique feature-coefficient vectors for a given model. Given this set of vectors, we can then use

Kernel Density Estimation (KDE), to estimate the probability density function (pdf) within the space of all possible coefficient configurations, for this particular set of vectors, but weighted via their respective performance in each case. The resulting empirically-derived pdf over this space, captures 'evidence' of high performance in the TL domain, which can be evaluated for any arbitrary vector of coefficients. Therefore, once the target company has been trained on a particular model, the resulting parameterisation can be evaluated as above, to obtain a measure of quality for that parameterisation, as informed by the TL domain.

The second step then involves using the BWA framework: for each estimator, we use the trained model to make a prediction for the target company, but this prediction is then weighted according to the height on the PDF curve at the point corresponding to the particular coefficient configuration used by the estimator. In other words, our approach weighs each prediction according to the degree of 'evidence of high performance' corresponding to the particular model coefficients, as obtained from the transfer setting. Therefore, the overall prediction for the target company emerges as a weighted average of individual model predictions, where higher weights indicate better anticipated estimator performance, given knowledge obtained from external data.

In other words, our approach weighs each prediction according to the degree of 'evidence of high performance' corresponding to the particular model coefficients, as obtained from the transfer setting. Therefore, predictions for each data point represent a weighted average, where higher weights indicate better estimator performance with respect to external data.

This chapter is structured as follows: Section 6.2 outlines methodology used for Transfer Learning Averaging framework. Immediately after, Section 6.3 shows the results of our experiments. Conclusions are the final part of this Chapter, outlined in Section 6.5.

6.2 Methodology

In Section 6.1, we stated that a prediction for an unseen data point is represented as the weighted average over a number of predictions — one per estimator — obtained from a pool of estimators involving both machine learning (ML) and statistical models (SE). The weight assigned to each estimator is determined by a probability density score, which reflects the degree of evidence regarding how well the estimator performs over the domain of possible coefficient configurations, as determined from other datasets. This score is inferred using Kernel Density Estimation (KDE), as explained in Section 6.2.1. For this inference step, we assume a linear dependency of the target variable, EPS, on its past lags, mean, and standard deviation moments. This assumption is supported by the simplicity and interpretability of linear models.

In other words, our regression model can be expressed akin to Equation 6.1:

$$\hat{y_t} = \theta_1 \times y_{t-1} + ... + \theta_5 \times y_{t-5} + \theta_6 \times \mu + \theta_7 \times \sigma \tag{6.1}$$

where the estimate $\hat{y_t}$ is linearly dependent on its five past lags, mean (μ), and standard deviation (σ). For all datasets, the number of lags is fixed for two reasons. First, five lags capture relatively recent information. Second, our methodology requires that every estimator have a consistent number of input features for each training dataset, such that all companies can be expressed as equivalent vectors whose components reflect the respective coefficients. Because a collection of coefficients serves as the input matrix to the Kernel Density Estimation when building the pdf, we cannot have one company with 2 lags and another with 4 — the input matrix must have the same number of elements in every row. Second, as mentioned in Chapter 4, we use the Akaike Information Criterion to pick the appropriate number of lags per data set. Empirically, we observed that the total majority of companies in our sample of 100

data sets, show smallest AIC at 5 lags. Those with fewer lags show no differences in our key error measures.

As established by the results in Chapter 4, we apply two separate data preprocessing pipelines for the respective series. Specifically, for both forming the parameterization space and subsequent estimation, we apply Quantile Transformation to the EPS series and PCHIP interpolation to the PCHIP series. The next section introduces the methodology we pursue in this chapter.

6.2.1 Transfer Learning Averaging Methodology

Transfer Learning allows compensating for individual estimators' lack of data observations by using information available from other, similar domains, providing general and thus 'transferrable' information about the problem. The main advantage of our approach is that knowledge is shared between estimators within the same data domain. We employ Bayesian Averaging of Statistical and Machine Learning regression models to combine their predictions. As will be explained later in this section, this method allows us to dynamically assign a weight to each estimator's prediction, with the weights mathematically constrained between 0 and 1. Dynamic assignment is due to the fact that weights are directly tied to each estimator's performance on the other 99 data sets. For comparison, the simple average would assign the same weight to every estimator, regardless of its performance. As a result, the prediction in our methodology is expressed as Equation 6.2.1.

$$\bar{P_n} = \frac{\sum_{k=1}^{K} D_n^{(k)} P_n^{(k)}}{\sum_{k=1}^{K} D_n^{(k)}} \tag{6.2}$$

where $\bar{P}_n$ denotes a prediction for data set n made with algorithm k, k represents an algorithm from the set of K algorithms, and D_k is the density score for algorithm k obtained using Kernel Density Estimation, serving as a weighting that reflects the 'evidence' for that model: the extent to which the model is known to perform well more generally (i.e. when considering all datasets). In other words, the numerator is the sum of predictions from candidate algorithms, pre-multiplied by the respective algorithms' density score. The denominator is the sum of density scores across all algorithms for the given data set. The density score here refers to the probability of algorithm k's parameterization within the parameterization space, obtained from other datasets. Since the density score for a specific model type is divided by the sum of densities for all model types used, it is ensured that the density score for each individual model can be treated as a weight: the result of this division will be scaled between 0 and 1.

The parameterization space consists of the coefficients of algorithm k with respect to other datasets, excluding the one for which we make a forecast, to avoid data leakage. For the set of 100 companies used, the parameterization space is a matrix $S^{m\times(N-1)}$, with m denoting the number of coefficients, and N being the total number of datasets, excluding the one for which we make forecasts. m coincides with the number of features used (7, as per Equation 6.1), and N is equal to 99, as we run experiments for 100 companies.

As explained, the parameterization space is used as in the input for the Kernel Density estimation, mathematically expressed as:

$$\tau(\boldsymbol{\theta}_n; h) = \frac{1}{h\sqrt{2\pi}} \exp\left\{-\frac{1}{2} \cdot \frac{\boldsymbol{\theta}_n^\top \boldsymbol{\theta}_n}{h^2}\right\} \tag{6.3}$$

Here, $\tau(x; h)$ represents the kernel function, $\boldsymbol{\theta}_n^{(k)}$ denotes the optimal coefficient vector for dataset n (given algorithm k), and h is the bandwidth parameter, which controls the size of the kernel's 'local neighbourhood', and thus the smoothness of the resulting density function. In

our experiments, the Gaussian kernel is used with $h = 1$. We acknowledge that for the choice of the bandwidth h more complex and dynamic methods exist. However, our decision to set $h = 1$ as fixed and symmetric is due to our intention to keep the model averaging process simple and digestible for comparison. Further, we empirically observed that standard deviation of the coefficients of both EPS and FCF regressions in the Transfer set is approximately 1, making $h = 1$ a reasonable choice.

For an unseen set of regression parameters, the computation of the density score is given as:

$$D_n^{(k)} = \frac{\sum_{i=1}^{N-1} w_i^{(k)} \cdot \tau\big(\boldsymbol{\theta}_i^{(k)} - \boldsymbol{\theta}_n^{(k)}\big)}{\sum_{i=1}^{N-1} w_i^{(k)}} \tag{6.4}$$

where $D_n^{(k)}$ is the density score, $\boldsymbol{\theta}_n^{(k)}$ is the point in the parameterisation domain where this density is being evaluated, $\boldsymbol{\theta}_i^{(k)}$ is the resulting parameterisation vector for company i after having being trained with algorithm k, and finally the term $w_i^{(k)}$ denotes the weight given to company i, serving as a measure of how well algorithm k performs on dataset i, and is obtained by applying the following logistic function to the MSE score:

$$w_i^k = 1 - \frac{1}{1 + \exp\{-\epsilon_i^{(k)}\}} \tag{6.5}$$

where $\epsilon_i^{(k)}$ is the Mean Squared Regression Error (MSE), calculated over the training subset of respective data set. This is because our data sets are limited in number of observations. In particular, the training data is subset our estimators used for fitting: in case of overfitting, the MSE would result in extremely low values not representative of the actual fitness. Additionally, computing it over the validation set would result, again, in an extremely low MSE value for

those estimators that we fine-tune (the ML branch). Hence, using the train and validation sets represents the only viable trade-off for computing the MSE for our purposes.

In other words, we estimate the prior probability distribution of each model's parameter space by fitting and adding Gaussian kernels at the specific parameter vectors of each of the other 99 companies, weighted by their 'performance'; $D_n^{(k)}$ then represents the height of the resulting probability distribution at position $\boldsymbol{\theta}_n^{(k)}$. Note that, without the $w_i^{(k)}$ weights, the density estimation would have only given evidence of 'how common' $\theta_n^{(k)}$ is, based on information coming from the other companies. If the parameterization is seen as some sort of 'profile' or 'personality' of a company (in the context of algorithm k), then the density at $\theta_n^{(k)}$ represents how well that personality is represented in the Transfer set. The addition of w terms converts this to making the density at each point a measure of the degree of evidence that the parameterization is performant, or how well that company is represented, judging preferentially on the basis of performant companies rather than over all companies. Due to the sparsity of our datasets, the MSE is computed on both the training and validation sets combined.

Apart from the newly proposed transfer learning approach, our pipelines outlined in Chapter 4 stay the same. We use the expanding window forecasting approach to forecast in the test set. In EPS series, we use the Quantile Transformation to both the target and features' variables. In FCF series, we use PCHIP interpolation that allows us to assume the new data arrival on the weekly, rather than quarterly basis. These tools were also applied to the parametrization space used for KDE estimation: EPS coefficients are estimated on the transformed data and FCF parameters on the interpolated series.

In the next section, we report results of our experiments.

6.3 Results

In this section, we present the results of applying transfer learning with a Bayesian averaging component to the algorithm selection discussed in the previous subsection, including ARD, BR, HR, LASSO, OLS, RLM, and WLS. Effectively, these estimators represent OLS variants; the trained coefficients are directly comparable to the coefficients of a standard OLS model, although with varying cost functions and parameter regulation methods. In practice, models such as ARIMA or MLP could also have been included in our experiments; their coefficient space would have been dimensionally different from that of the aforementioned seven estimators, ruling out the direct comparison of the KDE spaces. Hence, we avoid using them for transfer learning experiments.

In order to draw the comparison of how our methodology stands against the use of a single regression model type, we used the top-performers — the highest ranked estimators as established with the Friedman Average ranking statistical test in Chapter 5, as the benchmark models for this set of experiments, summarized for convenience in Table 6.1.

Table 6.1: *Benchmark results summary*

Series Type	**EPS**			**FCF**		
Error Measure	**MAE**	**MAPE**	**sMAPE**	**MAE**	**MAPE**	**sMAPE**
Control Estimator	**OLS**	**BR**	**HR**	**RLM**	**LASSO**	**RLM**
Mean	0.75165	1.17E+12	30.55336	0.91958	1.46585	35.81909
Standard Deviation	0.77237	7.86E+12	18.62303	0.98416	2.54057	18.01686
Min	0.02581	0.07831	4.09854	0.10146	0.16362	7.15913
Median	0.51874	0.67303	27.66093	0.64755	0.90572	34.6885
Max	6.21485	7.23E+13	74.59926	8.18703	23.8392	77.05792

As per Table 6.1, our benchmarks are OLS, BR and HR for EPS, and RLM, LASSO and RLM again, for FCF series, with respect to error measures: MAE, MAPE, sMAPE.

In addition to these single estimators, we employ the simple ('NAIVE') averaging of the aforementioned estimators for the comparative purposes, i.e. whether our methodology beats the simple averaging, where each estimator has even contribution to the prediction outcome.

By design of our methodology the number of estimators to be used for averaging can be treated as a hyper-parameter. Therefore, apart from combining all seven estimators, we also explore combinations of these estimators in pairs, triplets, and other groupings. We first present results of the EPS series, followed by those of FCF experiments.

6.3.1 EPS results

The top-performing combinations, listed in Table 6.2, are identified by ranking them using the Friedman test across all possible combinations for each respective error measure. For simplicity, the averaging results are denoted as 'BWA(n)', where 'n' represents the number of estimators combined (from 2 to 6).

Table 6.2: *EPS: Top-ranked model-types combinations. BWA(n) denotes the number n of models used for averaging*

COMBINATION	**MAE**	**MAPE**	**sMAPE**
Baseline	OLS	BR	HR
BWA(2)	HR-ARD	ARD-BR	HR-BR
BWA(3)	HR-ARD-BR	HR-ARD-BR	RLM-HR-BR
BWA(4)	RLM-HR-ARD-BR	OLS-LASSO-HR-ARD	RLM-HR-ARD-BR
BWA(5)	OLS-RLM-HR-ARD-BR	OLS-RLM-LASSO-HR-ARD	OLS-RLM-HR-ARD-BR
BWA(6)	OLS-RLM-HR-ARD-BR-WLS	OLS-RLM-LASSO-HR-ARD-BR	OLS-RLM-HR-ARD-BR-WLS

Table 6.3 outlines the summary of error distributions for the transfer learning experiment.

Notice that, 'BWA(7)' denotes the result of averaging of all seven model-types and 'NAIVE' represents the simple average methodology.

These results suggest that the predictions made with our methodology bring benefits in terms of increasing the accuracy of forecasts as compared to using the individual estimator (referred to as 'Baseline') and simple averaging of the same 7 regressors (referred to as 'NAIVE'). Notably, the standard deviation decreases when multiple estimators are used for prediction. Next, Table 6.4 presents the final Friedman test comparison of the combinations from Table 6.2.

The Friedman average ranking test is conducted in two steps. In the first step, regression methods are ranked for each data set based on their performance (i.e., MAE, MAPE, or sMAPE), with a higher rank indicating lower error. The average rank, referred to as 'Rank' in Table 6.4, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control method (marked as 'control' in Table 6.4) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that all methods (i.e., the OLS regression errors, measured as MAE, MAPE and sMAPE, respectively) are drawn from the same distribution.

Results depicted in Table 6.4 suggest important implications. For MAE, MAPE and sMAPE, there exists a combination of fewer than 7 model-types that outperforms the single benchmark estimator at a 5% significance level. Moreover, the result of combining all 7 model-types is also statistically outperformed by a combination of fewer estimators at a 5% significance level. In particular, the combination of RLM-HR-ARD-BR is the control for the MAE measure. Similarly, OLS-RLM-LASSO-HR-ARD is the top-ranked (control) combination for MAPE measures. Finally, RLM-HR-BR is the control combination for sMAPE error measurements. Notably, the 'naive'

Table 6.3: *EPS: Transfer Learning regression error summary statistics over 100 companies. Minimum value in each category is highlighted in* ***boldface***

Method	Error Measure	Mean	Std. Dev.	Min	Median	Max
	MAE	0.75165	0.77237	0.02581	0.51874	6.21485
Baseline	MAPE	1.167E+12	7.858E+12	7.831E-02	6.730E-01	7.230E+13
	sMAPE	30.5533	18.6230	4.09854	27.6609	74.5992
	MAE	0.75242	0.72370	0.03355	0.51076	4.30845
NAIVE	MAPE	1.195E+12	8.480E+12	8.142E-02	7.227E-01	8.020E+13
	sMAPE	30.7137	18.1118	4.35476	27.5513	79.6623
	MAE	0.63659	0.55620	0.02003	**0.44960**	2.90813
BWA(2)	MAPE	7.768E+11	6.323E+12	**5.644E-02**	6.284E-01	6.220E+13
	sMAPE	27.31376	17.83229	2.849486	**24.25039**	68.74795
	MAE	0.63984	0.55809	0.02035	0.45729	2.90617
BWA(3)	MAPE	7.995E+11	**6.189E+12**	5.916E-02	6.388E-01	**6.030E+13**
	sMAPE	**27.13541**	**17.67014**	2.82522	24.76198	**67.55550**
	MAE	**0.636033**	0.55564	**0.020394**	0.45008	**2.90167**
BWA(4)	MAPE	7.712E+11	6.941E+12	7.653E-02	**6.156E-01**	6.880E+13
	sMAPE	27.37324	17.87506	2.883657	24.47756	68.93462
	MAE	0.636826	**0.55171**	0.02049	0.45524	2.91703
BWA(5)	MAPE	7.865E+11	7.081E+12	6.622E-02	6.266E-01	7.02E+13
	sMAPE	27.41672	17.90846	2.891734	24.45839	70.90860
	MAE	0.643041	0.552278	0.020137	0.463891	2.933588
BWA(6)	MAPE	**7.693E+11**	6.911E+12	6.270E-02	6.310E-01	6.850E+13
	sMAPE	27.57921	17.77825	**2.81045**	24.43242	71.22289
	MAE	0.649708	0.560237	0.020057	0.460913	2.936132
BWA(7)	MAPE	8.462E+11	7.293E+12	5.678E-02	6.401E-01	7.220E+13
	sMAPE	28.13213	18.12167	2.856022	24.69830	74.78230

Table 6.4: *EPS: Friedman Test results. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**p-val**	**Method**	**Rank**	**p-val**	**Method**	**Rank**	**p-val**
BWA(4)	3.51	control	BWA(5)	3.695	control	BWA(3)	3.15	control
BWA(2)	3.63	7.29E-01	BWA(6)	3.75	8.62E-01	BWA(2)	3.43	4.19E-01
BWA(5)	3.86	6.25E-01	BWA(4)	3.83	8.62E-01	BWA(4)	3.75	2.12E-01
BWA(3)	3.99	4.68E-01	BWA(2)	3.98	8.62E-01	BWA(5)	4.12	**1.53E-02**
BWA(6)	4.21	1.86E-01	BWA(3)	4.16	7.18E-01	BWA(6)	4.17	**1.02E-02**
BWA(7)	4.82	**7.79E-04**	BWA(7)	4.55	6.79E-02	BWA(7)	5.25	**6.71E-09**
NAIVE	4.95	**1.94E-04**	NAIVE	5.42	**3.82E-06**	NAIVE	5.28	**4.03E-09**
Baseline	7.03	**2.06E-23**	Baseline	6.61	**2.75E-16**	Baseline	6.89	**2.50E-26**

averaging technique performs worse compared to the control methodology. This is likely because each estimator is assigned an equal weight, resulting in a small contribution from each (since the averaging assigns the same weight to all estimators). This highlights the advantages of our proposed methodology, which allows for dynamic weighting based on the performance of each estimator on other, similar datasets.

We next consider the implications of our methodology to the FCF data sets.

6.3.2 FCF results

We followed the same procedure as described in Section 6.3.1 to identify the top-ranked combinations in each category. These results are summarized in Table 6.5.

Table 6.6 depicts summary statistics of results of our proposed methodology, the naive approach and baseline results.

Table 6.5: *FCF: Top-ranked model-types combinations. BWA(n) denotes the number n of models used for averaging*

COMBINATION	MAE	MAPE	sMAPE
Baseline	RLM	LASSO	RLM
BWA(2)	LASSO-HR	LASSO-HR	HR-ARD
BWA(3)	RLM-LASSO-HR	LASSO-HR-ARD	RLM-HR-ARD
BWA(4)	RLM-LASSO-HR-ARD	RLM-LASSO-HR-ARD	RLM-LASSO-HR-ARD
BWA(5)	OLS-RLM-LASSO-HR-ARD	OLS-RLM-LASSO-HR-ARD	OLS-RLM-HR-ARD-BR
BWA(6)	OLS-RLM-LASSO-HR-ARD-BR	OLS-RLM-LASSO-HR-ARD-BR	OLS-RLM-LASSO-HR-ARD-BR

Results for the FCF differ from those of the EPS series. First, the minimum average MAPE and sMAPE are achieved by the baseline model. Additionally, the lowest standard deviation is attained by the combination of 6 model-types (BWA(6)). The lowest median MAPE is also produced by the baseline model. At the same time, the lowest median MAE and sMAPE results are achieved by BWA(4) and BWA(3), respectively. Now, consider how these results translate into the Friedman ranking test, presented in Table 6.7.

The Friedman average ranking test is conducted in two steps. In the first step, regression methods are ranked for each data set based on their performance (i.e., MAE, MAPE, or sMAPE), with a higher rank indicating lower error. The average rank, referred to as 'Rank' in Table 6.7, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control method (marked as 'control' in Table 6.7) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that all methods (i.e., the OLS regression errors, measured as MAE, MAPE and sMAPE,

Table 6.6: *FCF: Transfer Learning error-score summary statistics over 100 companies. Minimum value in each category is highlighted in* ***boldface***

Method	Error measure	Mean	Std. Dev.	Min	Median	Max
	MAE	0.919575	0.98416	0.101457	0.647547	8.18703
Baseline	MAPE	**1.465855**	2.54057	0.163621	**0.905719**	23.83922
	sMAPE	**35.81909**	**18.01686**	**7.159129**	34.6885	**77.05792**
	MAE	0.92265	0.98386	0.10299	0.64831	8.14012
NAIVE	MAPE	1.57604	2.43811	0.15067	0.95085	21.8007
	sMAPE	36.05695	17.94680	7.712541	34.07787	73.93169
	MAE	**0.913353**	**0.97373**	0.113626	0.642767	**8.097124**
BWA(2)	MAPE	1.518504	2.501799	0.149569	0.959443	22.57839
	sMAPE	35.92945	18.37849	7.278942	34.46989	82.62183
	MAE	0.914061	0.978805	0.104367	0.641402	8.150634
BWA(3)	MAPE	1.539032	2.489742	0.148962	0.954749	22.25185
	sMAPE	35.92804	18.19211	7.270376	**34.43649**	82.38804
	MAE	0.922915	1.031578	0.100349	**0.636046**	8.804111
BWA(4)	MAPE	1.556182	2.503543	0.14757	0.952017	22.15925
	sMAPE	36.08357	18.24624	7.262758	34.67848	81.35213
	MAE	0.92679	1.035178	0.0985	0.641428	8.837274
BWA(5)	MAPE	1.566158	2.490003	**0.147023**	0.96355	21.85575
	sMAPE	36.27452	18.41871	7.288516	34.69977	80.97185
	MAE	0.931226	1.046976	**0.09773**	0.645918	8.983521
BWA(6)	MAPE	1.571833	**2.480273**	0.147078	0.986959	21.80394
	sMAPE	38.62848	20.02297	7.32137	36.5014	96.88514
	MAE	1.080065	1.311022	0.112664	0.713509	11.21646
BWA(7)	MAPE	1.827305	2.601091	0.148318	1.089375	**21.68352**
	sMAPE	38.64761	20.08558	7.440908	36.4191	96.82387

respectively) are drawn from the same distribution.

Table 6.7: *FCF: Friedman Test results. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

MAE			MAPE			sMAPE		
Method	**Rank**	**P-Val**	**Method**	**Rank**	**P-Val**	**Method**	**Rank**	**P-Val**
BWA(4)	3.80	control	BWA(2)	3.69	control	Baseline	3.80	control
BWA(3)	4.04	4.88E-01	BWA(4)	3.99	3.86E-01	BWA(3)	3.94	6.86E-01
BWA(2)	4.22	4.51E-01	BWA(3)	4.00	3.86E-01	BWA(2)	3.99	6.86E-01
BWA(5)	4.29	3.38E-01	Baseline	4.11	3.86E-01	BWA(4)	4.28	4.98E-01
Baseline	4.34	3.38E-01	BWA(5)	4.58	**4.08E-02**	NAIVE	4.44	2.59E-01
BWA(6)	4.65	7.07E-02	NAIVE	4.66	**2.55E-02**	BWA(5)	4.51	1.62E-01
NAIVE	4.66	6.52E-02	BWA(6)	5.05	**5.18E-04**	BWA(7)	5.50	**5.54E-06**
BWA(7)	6.00	**1.50E-09**	BWA(7)	5.92	**8.50E-10**	BWA(6)	5.54	**3.23E-06**

From Table 6.7 we observe that the results of applying our methodology to the FCF data sets is drastically different from those of the EPS. In particular, we observe that BWA(4) is the 'control' methodology in MAE error measurement group, significant only against the BWA(7) — the transfer learning methodology that combines all 7 estimators, with Baseline and naive averaging approaches ranked 4.34 and 4.66, respectively. Similarly, the transfer learning BWA(2) methodology is the 'control' forecasting approach in MAPE group, significant against three other methodologies, namely: naive, BWA(6) and BWA(7). Finally, the sMAPE error measurement group is the only one where the Baseline approach — a single estimator, is the 'control' methodology, significant against the BWA(7) and BWA(6) transfer learning regressions.

From both the EPS and FCF results, we observe that our methodology is helpful at minimiz-

ing the regression error, in the majority cases. Specifically, we saw that the in the EPS case, the transfer learning approach is better than the baseline and the naive averaging approaches at a statistical significance. At the same time, we saw that in the FCF case, the transfer learning is the 'control' approach in two our of three regression error measures. We acknowledge the fact that, based on the results from this section, it could be said that the improvement over the baseline results is "tiny", given the complexity of our proposed approach. However, it is worth emphasizing the fact that the EPS and FCF predictions will be used as inputs to the valuation models. As explained in Chapter 2, these valuation methods represent equations where the estimated next quarter FCF/EPS value is taken in relation to some other variables. Therefore, as will be discussed in depth in the next Chapter 7, even a seemingly insignificant improvement in the regression error measurement may lead to improved monetary gains from one financial quarter onto the next. In turn, that leads to larger gains, when profits are measured over longer periods of time. Before going deeper into applied analysis of our proposed model, we would like to conduct a further analysis of our transfer learning results in the next section.

6.4 Results Discussion

For the interpretation of results in this chapter, we use the two data sets, namely Caterpillar Inc. (CAT) and eBay Inc. (EBAY) as the representative company for demonstration purposes in this part, with other 8 examples randomly picked from the pool of 100 data sets present in the Appendix, Section C of this thesis.

Examination of Tables 6.4 and 6.7 reveals a notable finding: the weighted with the aid of transfer learning average of statistical and ML estimators generally improves performance on sparse datasets. Importantly, the transfer learning component of our methodology assigns

greater weight to estimators that perform better across most datasets. However, we observed that combining all 7 proposed models performed worse and was statistically outperformed by certain subsets, even to a degree of statistical significance as observed in the FCF cases.

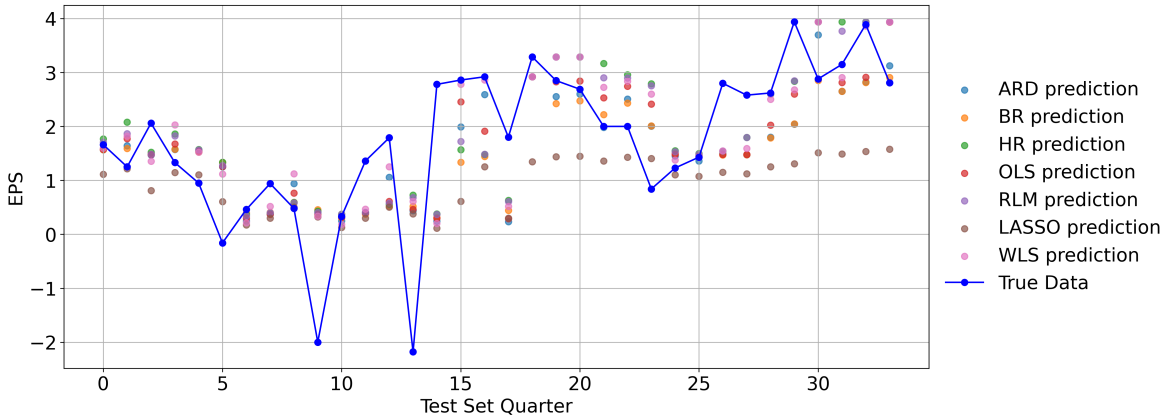
In all three cases, the naive averaging technique is also worse (statistically significantly worse in the EPS case) than the control transfer learning-based averaging method. This phenomenon can be attributed to the fact that in our framework, each estimator is assigned a weight in the density approximation to the parameter space, although some weights may be marginal. In our methodology, the number of models is considered as a hyperparameter — a higher number of estimators leads to a more complex prediction being made. What we observe is effectively more complex models overfitting, yet there is an optimal complexity for our Bayesian Model Averaging approach; e.g., in the optimal model bias-variance balance sense.

As an example of this, Figures 6.1 and 6.2 show Caterpillar Inc., EPS and FCF datasets, respectively, with the same graphs for other data sets given the Appendix Section C, Figure C-3 for EPS and Figure C-4.

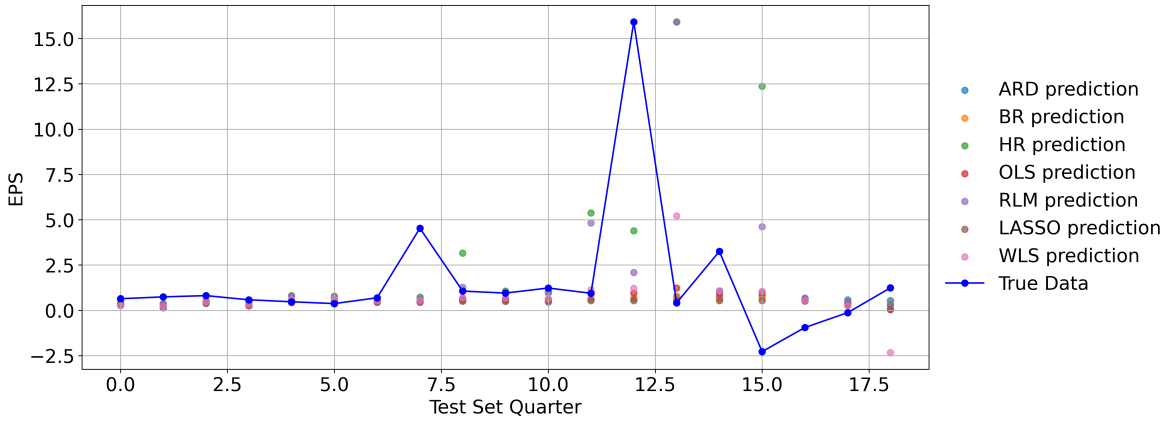
We can see that in some areas of the the EPS timeline (e.g., up to quarter 14 on the 'EPS-CAT' graph), predictions from various single estimators are fairly consistent — they are 'grouped' or 'clustered' with each other, whereas from quarter 15 onwards the opposite effect emerges: predictions made by estimators are more spread around the true value, visibly distant from each other. We observe the same behaviour on the 'CAT-EBAY' graph: the majority of predictions done by various estimators are 'grouped' together, visibly not far from a genuine data point, with few outlier cases around the outlier at quarter 11.

Similarly, estimators yield such behaviour for the FCF series, with WLS often making extremely high or low predictions regarding the next quarter's value: as indicated by the pink

Figure 6.1: *EPS: Single estimators; test set predictions. Dots denote individual estimator predictions, solid line denotes actual data.*

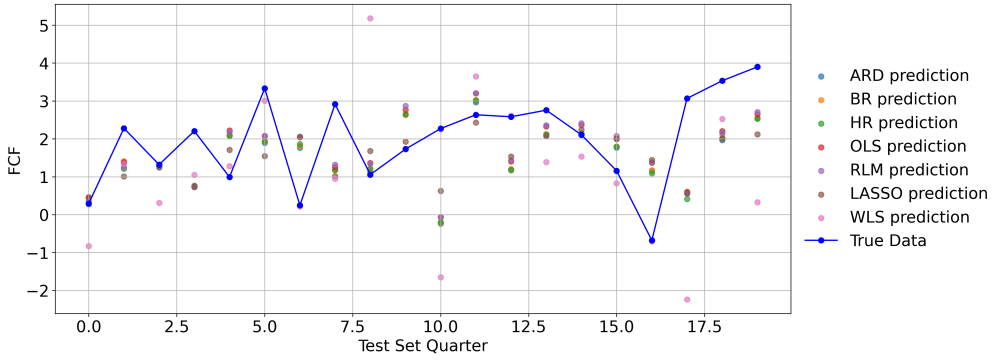


(a) EPS-CAT: Single Estimators Predictions

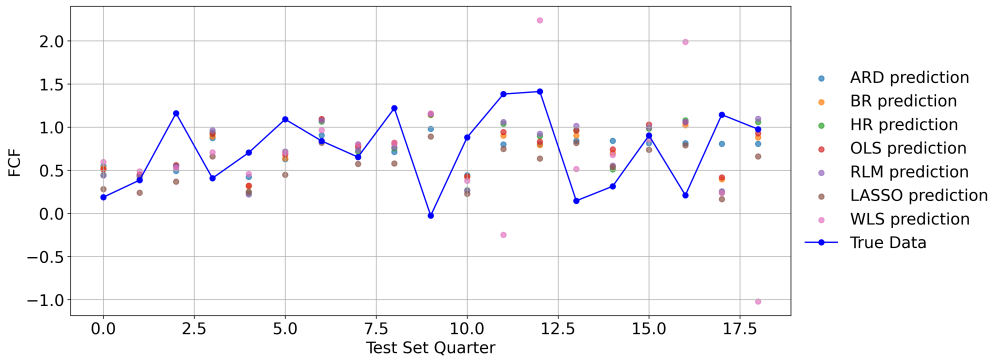


(b) EPS-EBAY: Single Estimators Predictions

Figure 6.2: *FCF: Single estimators; test set predictions. Dots denote individual estimator predictions, solid line denotes actual data.*



(a) FCF-CAT: Single Estimators Predictions

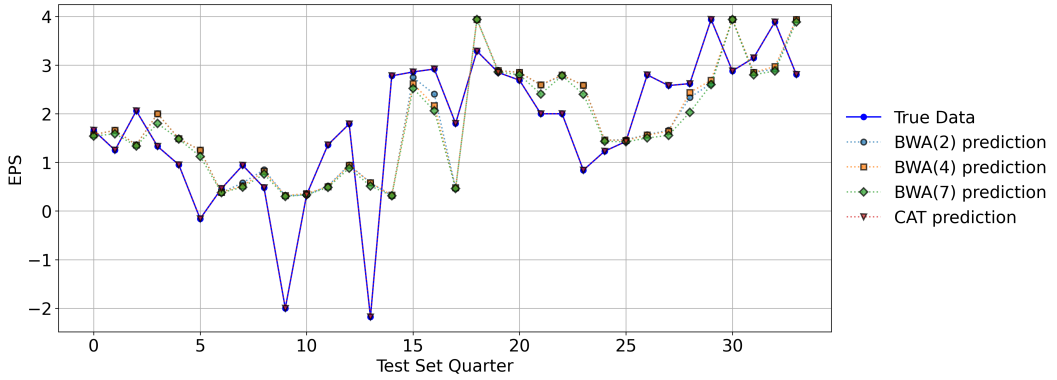


(b) FCF-EBAY: Single Estimators Predictions

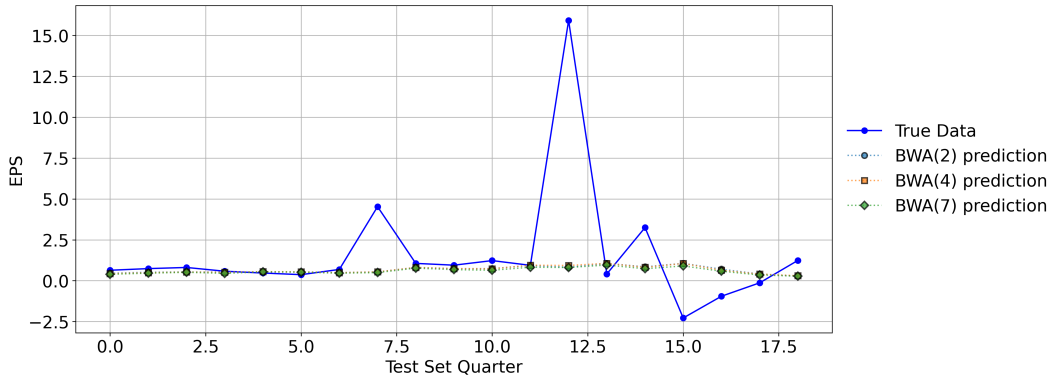
dots, at first, 8th and 17th quarters from the CAT predictions, evident from the 'FCF-CAT' figure. The 'FCF-EBAY' figure supports this observation. Further, *visibly* all estimators tend to over-/under-shoot the actual series, possibly due to the volatility of past lags as discussed in Chapters 4 and 5 for the dynamic FCF values. Otherwise, the majority of estimators tend to predict approximately similar values differing in decimal places.

The Friedman average ranking results from Tables 6.4 and 6.7 would suggest that excluding algorithms with worse rank from the averaging process might lead to improved performance. Figures 6.3 and 6.4 therefore shows forecasts for 2, 4, and 7 averaged model-types for representative EPS and FCF data sets, respectively.

Figure 6.3: *EPS Transfer Learning results; test set predictions. Dotted lines denote averaged predictions of 2, 4, and 7 models. Solid line denotes actual data.*

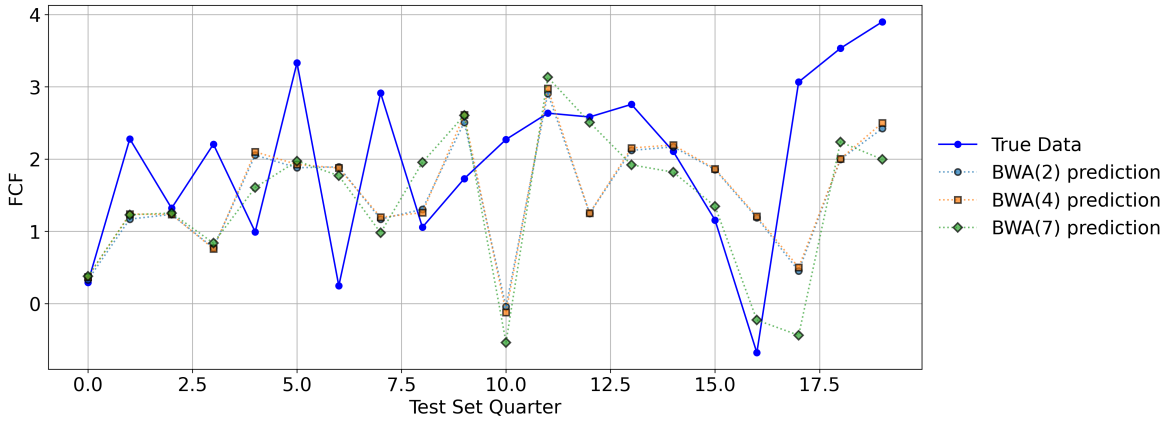


(a) EPS-CAT: BWA(n) Predictions

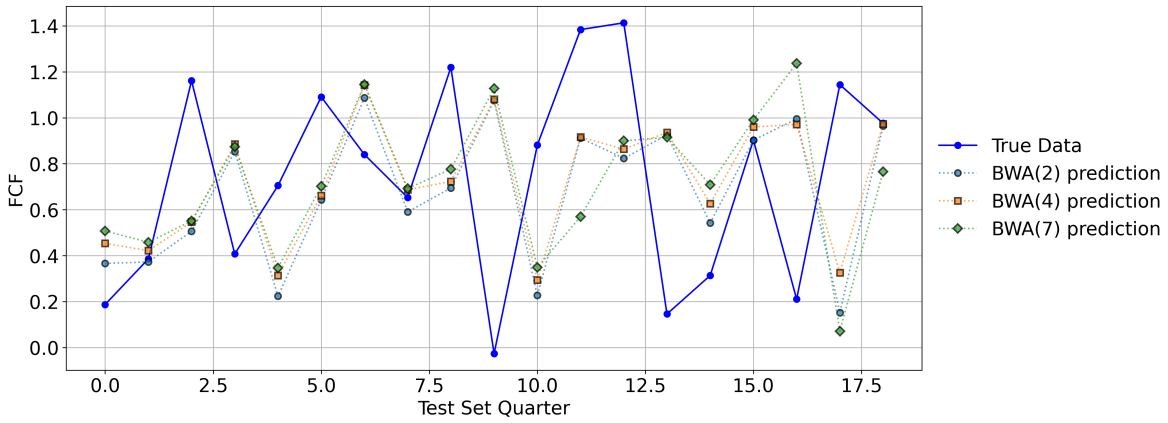


(b) EPS-EBAY: BWA(n) Predictions

Figure 6.4: *FCF Transfer Learning results; test set predictions. Dotted lines denote averaged predictions of 2, 4, and 7 models. Solid line denotes actual data*



(a) FCF-CAT: BWA(n) Predictions



(b) FCF-EBAY: BWA(n) Predictions

We note that in the EPS and FCF series, despite the fact that the average ranking of the BWA(4) is generally higher than that of BWA(7), which signals a better performance of the former, in this particular data set there does not appear to be too much *visible* difference between averaged estimators.

Rarely, there is a difference in the FCF predictions: BWA(4) occasionally makes visibly higher or lower prediction of the target series. This is unsurprising to an extent, since we would expect the 'better' estimators to contribute the lion's share to the averaging process.

Extreme weight values would be required for the 'worse' estimators to significantly influence the end-result (and we do not observe such extreme values in this particular example). We also note that the averaged algorithms are not only more consistent with each other, but also with respect to the underlying ground truth compared to the range of individual predictions from Figures 6.1 and 6.2. This observation can be extrapolated to other series: observe similar patterns in Figures C-1 for the EPS and Figures C-2 for the FCF data sets, respectively, in the Appendix Section C.

Another finding aligns with the theory: averaging results of estimators with diverse optimization and penalty functions yields consistently better results [23]. Specifically, superior performance tends to arise from models employing varied cost functions, as outlined in Chapter 2, as well as different coefficient penalty characteristics (L1 and L2 regularization). Furthermore, both Statistical Models (SE) and Machine Learning (ML) estimators capture distinct information from the series. Therefore, the weighted average of all estimators, both SE and ML where weight is assigned given how well an estimator fits similar data sets, consistently yields higher accuracy compared to using a single estimator. In next section we summarize our findings for this chapter.

6.5 Conclusion

This chapter introduces a transfer learning technique for averaging predictions over multiple models, by leveraging external information regarding the model's performance when evaluated on similar datasets.

Our approach assigns weights to estimators by considering how 'likely' this estimator's parameter vector is, with reference to the empirical probability distribution of all possible

parameterizations resulting from the application of the same estimator onto other, external datasets. As this empirical distribution is obtained via a Kernel Density Estimation process, modifying the contribution of each dataset's kernel according to its performance further modifies the underlying distribution. This adjustment expresses not only how 'likely' but also how 'performant' the estimator is at any particular location in the parameterization space. These weights can then be used as model priors in a Bayesian averaging process, serving as weights that dictate the contribution of each estimator onto the final prediction.

We found that combining up to 5 models generally improves forecasting accuracy in most cases of EPS and FCF data sets, compared to either including all the proposed estimators into the averaging process or using estimators individually, especially in sparse datasets. This method effectively integrates insights from both ML and Statistical estimators.

This chapter introduced the final estimation methodology to the EPS and FCF series. In next chapter, we apply predictions of those respective series to the Forward Price/Earnings and Discounted Cash Flows valuation models, explained in greater details in Chapter 2, in order to establish the economic benefits our models brings to the fundamental investment procedure.

Chapter 7

Out-Of-Sample Testing

7.1 Introduction

In this chapter, we demonstrate that predicting the Earnings Per Share (EPS) and Free Cash Flows (FCF) with our proposed transfer learning approach yields practical applications when the next quarter estimates are used as inputs to the Price-Earnings (PE) and Discounted Cash Flows (DCF) models, respectively. Valuation models are used by fundamental investors to derive the intrinsic value of a business, which, in contrast to the market price, states the price worth paying for a stock, given its future prospects. For a more rigorous treatment of these and other valuation methods refer to Sections 2.2.1 and 2.2.2 of Chapter 2 of this thesis.

The primary objective of this chapter is to determine whether the proposed forecasting methodology enhances the PE and FCF fundamental investment approaches. Specifically, we assess the performance of a stock portfolio in which intrinsic value fo stocks exceeds the market price at a given point in time. As discussed in Chapter 2, intrinsic value represents the reasonable price one should pay for a stock based on its future prospects [2].

The intrinsic value is calculated using valuation models. For this study, we employed two

such models. The first is the PE model, which incorporates the predicted EPS value for the upcoming quarter. The second is the DCF model, which relies on the predicted FCF for the same period.

In this thesis, our primary focus has been to identify the regression methodology that minimizes error when making quarter-ahead predictions for EPS and FCF target values, respectively. Hence, in Chapter 6, we introduced the transfer learning method, which combines the predictive power of the top-performing group of ML and SE estimators identified in Chapter 5. Additionally, Chapter 4 discusses the properties of these series, outlining the pipeline that interpolates the FCF values and transforms the EPS values.

This chapter presents the results of backtesting the valuation techniques augmented with our methodology. Backtesting allows us to assess how the portfolio performed over a specified period of time. A portfolio of stocks refers to a group of financial securities purchased according to specific guidelines. In our case, these guidelines are outlined in Chapter 2: we add a stock to the portfolio if its intrinsic value exceeds the market price at time t, and sell the stock if it has previously been purchased and its intrinsic value is now lower than the market price at time t.

We benchmark our portfolios against the Standard and Poor's 500 broad market index, S&P 500 for short. The market index is also a portfolio of stocks: the S&P 500 is the collection of more than 500 largest (by market capitalization) companies, traded on the U.S. stock exchanges. The market capitalization is the total worth of a company, given its current market price and the number of shares in circulation, physically issued by a company, at a point in time. The S&P500 index was chosen because it includes companies from various sectors, representing a diversified portfolio of stocks, which we consider as a potential alternative to the portfolios created in this study. More specific guidelines regarding the backtesting

methodology are outlined in Section 7.2. The experimental setup is described in Section 7.3, while the results and their analysis are presented in Section 7.4. Conclusions for this section are provided in Section 7.5.

7.2 Methodology

The need for more accurate forecasts arises because overstatement or understatement of EPS/FCF values can lead to financial losses. Capturing it with minimal error results in financial gains or the avoidance of financial catastrophe.

To measure the benefits our proposed methodology brings, we conduct back-testing, which allows us to evaluate how a portfolio would have performed if we had bought a selection of undervalued stocks in a past period. 'Undervalued' in this context means that its intrinsic value is higher than its market price.

In this section, we first outline the valuation rules followed by our program. These rules are designed to mimic the decision-making process undertaken by a fundamental investor when determining the intrinsic value of a stock. The rules are detailed in Section 7.2.1. The benefits an investor gains from the proposed regression approach are measured using the standard portfolio performance ratios described in Section 7.2.2.

7.2.1 Simulation procedure

Generally, our simulation program is designed to mimic a decision-making process akin to that of a fundamental investor. Every day, thousands of financial professionals study financial reports produced by companies to determine the intrinsic value of a stock. Their aim is to assess the prospects of a particular stock for the next financial period.

Our program works in a similar way. At a high level, the program scans through its investment universe of 100 stocks and computes the DCF/PE intrinsic values for each. Then, if an intrinsic value is higher than the price of a stock at any point during the quarter, the stock will be added to the portfolio; otherwise, it will be ignored. The stock is then held until the next quarter, during which the program evaluates it again. If its intrinsic value is less than the market price, the stock will be sold; otherwise, it will continue to be held.

A stock cannot be bought multiple times, as this would violate our principle of equal contribution to the portfolio and force us to optimize the portfolio weights. We avoid this, as the portfolio's performance could then be attributed to the optimization process rather than the stock selection process, which is the primary objective of this study. We are more inclined toward observing the effects of portfolio formation rather than maximizing portfolio returns. In other words, since our study primarily focuses on the impact of forecasting accuracy on portfolio performance, in terms of how this affects the portfolio's composition over time. In other words, since the forecasting process decides what stocks enter or leave a portfolio at different points in time, this means that different forecasting algorithms will form very different portfolios over a given period of time, whose financial performance can be evaluated directly.

In light of this, the stock can exist in several states: it can either be in the portfolio or out in the market. If a stock is in the market, it can only be bought or ignored. If a stock is in the portfolio, it can only be held or sold. We also restrict short-selling, an investment strategy whereby investor profits from the depreciation of an asset's price. This restriction is due to the fact that short selling is largely available to institutional rather than individual investors. Additionally, this investment vehicle has limited profit potential: a stock's price can drop by as much as 100%. The risk then becomes infinite, as a stock can appreciate indefinitely if

it is bought by other market participants [8]. All together, this is outside the scope of value investment reasoning [2], which we aim to mimic through our program.

In addition, we do not consider bid/ask quotes on stocks and other liquidity nuances. The bid price is the price at which a stock can be sold, while the ask price is the one at which it can be purchased [5] by an investor. Therefore, in our investment simulation, every stock has only one price: the closing price of the day. The bid/ask quotes are updated every second during a trading day and usually change by decimal places. Hence, their inclusion in the simulation is not necessary, as the buy/sell procedure is executed on a quarterly basis, with stocks held for at least 80 days (the typical number of trading days in a financial quarter). Ignoring the bid/ask spread does not lead to major consequences, as the spread itself is typically in decimals and would only cause a marginal effect on quarterly price appreciation or depreciation. Furthermore, any stock held in the portfolio can be sold; there are no liquidity constraints.

Finally, we ignore any capital gains tax imposed on profits. Capital gains tax is subject to different jurisdictions and varies depending on the size of the stake sold as well as the profit or loss made. Including this would overly complicate our simulation, distorting the final results and preventing us from conducting a deeper analysis of the performance of our methodology.

In the next section, we introduce the three portfolio performance metrics used to determine the success of our portfolio-building efforts.

7.2.2 Performance metrics

The three measures, which we explain below, operate on daily returns. Daily returns are the percent change in stock price from one day to the next, as illustrated in Equation 7.1.

$$R_p = \frac{p_d - p_{d-1}}{p_{d-1}} \tag{7.1}$$

where p_d denotes the total portfolio value on day d, with p_{d-1} denoting the portfolio value on the previous day, and R_p is the output of the equation. This represents the percent change in the portfolio value — how much the value of our portfolio increased or decreased on a daily basis.

The first portfolio performance metric we use is the Sharpe ratio, which measures the risk-premium stock return per unit of risk [12], as expressed in Equation 7.2.

$$\text{Sharpe Ratio} = \frac{E[R_p - R_f]}{\sigma_p} \tag{7.2}$$

where $E[]$ is the expectation (mathematical mean) operator, R_p represents the daily returns of our portfolio, R_f is the risk-free rate, and σ_p is the standard deviation of portfolio returns on a daily basis. Here, the numerator is the risk premium — the reward an investor gains from holding a stock (a risk-bearing asset) over three year US Treasury bonds (the risk-free asset).

The standard deviation embedded as the denominator in the Sharpe Ratio measures daily risk. However, it reveals nothing about systematic risk. Systematic risk refers to the potential for losing funds due to events that affect the entire market, such as an economic recession or a stock market collapse [151]. To account for systematic risk, financial professionals use the Treynor Ratio, outlined in Equation 7.3.

$$\text{Treynor Ratio} = \frac{E[R_p - R_f]}{\beta_p} \tag{7.3}$$

The numerator in Equation 7.3 is the same as in the Sharpe Ratio in Equation 7.2, while the denominator is the portfolio beta, denoted as β_p. The portfolio beta is found by regressing

portfolio returns against market returns. In other words, β_p is the slope coefficient of the linear regression, where the X-input data is the market daily percent change and the y-output is the daily portfolio returns.

The Sharpe Ratio and Treynor Ratio measure the reward gained with respect to daily fluctuations of funds on the market and the risk of losing funds due to a financial collapse. However, it is also important to understand how much the portfolio gains relative to the minimum historically observed drop in portfolio value. This is what the third ratio — the Sortino Ratio — measures [152], as presented in Equation 7.4.

$$\text{Sortino Ratio} = \frac{E[R_p - R_f]}{DD_p} \tag{7.4}$$

where, similarly to the previous two measures, the numerator is the expectation of stock returns net of the risk-free rate. The denominator, DD_p, is the downside risk of portfolio p. The downside risk in the Sortino Ratio is the standard deviation of daily returns below the expected (average daily returns). The computation of downside risk is outlined in Equation 7.5.

$$DD_p = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \min(0, (R_{p,i} - E[R_p])^2)} \tag{7.5}$$

The Sortino ratio is computed on a daily basis, where the index $i = 1, \ldots, N$ represents the days, with N denoting the total number of days in a sample. In particular, we make use of the min function, which selects the minimum between zero and the squared difference between the return of portfolio p on day i and the expected (arithmetic mean) portfolio daily returns $E[R_p]$.

For illustrative purposes, we also disclose the expected rate of return and the standard

deviation. However, because daily fluctuations of the prices are typically in decimal places, we multiply both numbers by 252 — the number of trading days per year. The standard deviation is multiplied by $\sqrt{252}$, as the standard deviation function represents the square root of variance.

For all these metrics, except for the Annualized Standard Deviation, a higher value indicates better performance of the corresponding portfolio. A higher Annualized Standard Deviation, on the other hand, implies an increased risk for the respective portfolio, which is undesirable.

Our experiments are performed on 100 stocks and compared to the market index and the ground true data, which is described in length in next section.

7.3 Experimental Setup

7.3.1 Data

Similarly to the previous sections, we used a sample of stocks from U.S. publicly traded companies with a market capitalization of over US $1 billion, specifically from the manufacturing industries. In earlier chapters, the same test set was used for all experiments. However, for clarity in this section, we utilized data from Quarter 1, 2022, to Quarter 2, 2024. This subset of EPS and FCF series was not used in previous chapters, making the tests in this section entirely out-of-sample experiment. Tables D-1 and D-2 in the Appendix, Section D, contain more information regarding timelines used for regression and backtesting, for each data set in our EPS and FCF series selection, respectively. The main purpose of using the updated timelines is to avoid the bias in our study: we do not know which methodology yields most accurate results on the added subsets of data.

For the evaluation of financial results, we used pricing data, which enables us to compute

various portfolio performance metrics on a daily basis. This characteristic of the pricing data also allows us to calculate PE ratios daily. To ensure objectivity, we allow our PE-based model to randomly select a PE ratio for the computation of intrinsic value at each quarter.

In practice, it is common to perform portfolio optimization: buy more or less of some shares, to minimize the standard deviation or other measure of risk. In this study, we perform no portfolio optimization, as it is outside of the scope of the present study: we seek to establish the benefits our EPS and FCF forecasting methodology brings to the fundamental investors, not to find the most profitable trading strategy or least volatile portfolio formation technique. Therefore, each stock in the portfolio is assigned an equal weight. In other words, every stock purchased by the program, following the rules outlined above, has an equal contribution to portfolio performance.

7.3.2 Benchmarks

Up to this chapter, our study stressed the performance of various ML and SE regression estimators in the sparse time-series data. In Chapter 5 we used a set of single estimators to make forecasts, with Chapter 6 building on top of that narrative, by introducing a transfer learning methodology for combining the predictive power of two branches of regression estimators used in this study. Thus, in addition to comparing one of our portfolios to the market index, we also compare the performance of various portfolios against each other. We consider a collection of stocks selected using the transfer learning methodology introduced in Chapter 6, along with those studied in Chapter 5. Specifically, we include ARIMA, a model type that was consistently ranked lower than the top-performing estimators at a statistically significant level. We also incorporate predictions from the 'second-best' estimator in each error metric to assess whether our transfer learning methodology provides an advantage. Specifically,

we include: LASSO, HR, OLS, ARIMA, and MLP. Further, in tandem with the previous Chapter 6 we use the naive averaging as one of the benchmarks in this chapter. We refer to the naive averaging approach as to 'NAIVE' when presenting results. In our work, the naive averaging is the simple arithmetic average across predictions made with the same collection of estimators used as input models to our proposed transfer learning approach.

Apart from the aforementioned regression models, for certain tests we include actual data, referred to as 'ACT', to demonstrate the highest possible result we can achieve, assuming perfect foresight into the future. For practical reasons we exclude this benchmarks from comparison procedures, such as Friedman average ranking: this portfolio exactly 'knows' what the EPS/FCF value will be next quarter.

Finally, the market benchmark — the S&P 500 market index, referred to as 'MKT' serves as the primary comparison parameter. The MKT represents a single alternative to the assets in our portfolios, as it is a diversified set of more than 500 stocks (503 as of 2025, the title 'S&P500' is kept for historical reasons and convenience).

In the next section, we reveal the results of conducted experiments.

7.4 Results

This section is divided into several parts. First, we present the financial ratios and an overview of the backtesting results in Subsection 7.4.1. Second, we conduct a deeper analysis of our portfolios and the relationship between the regression error and the portfolio performance in Subsection 7.4.2.

7.4.1 Financial Performance

In this section, we first present the results of both DCF and PE valuation method portfolios, followed by a Friedman test analysis to rank the portfolios. Since the objective is to maximize reward per unit of risk, we rank the portfolios from highest to lowest ratio. We begin with the DCF portfolios.

We used several conventions to display values in tables. '(TL)' denotes portfolios where valuation for stocks is done with predictions based on the transfer learning methodology. Therefore, '(TL)PE:MAE' refers to the portfolio, where EPS predictions used for Price-Earnings valuation model are done with combination of estimators that act as control in the MAE category, as per Friedman Average ranking test. In a similar way, the '(TL)DCF:MAPE' denotes the transfer learning based portfolio where the Discounted Cash Flows model is used to perform valuations. In accordance with the previous Chapter 6, for EPS and FCF regressions, we found the following transfer learning combinations to rank lower in each respective error measures, summarized in Table 7.1.

Table 7.1: *Highest Ranked Estimator Combinations*

Error Measure	**EPS**	**FCF**
MAE	RLM-HR-ARD-BR	RLM-LASSO-HR-ARD
MAPE	OLS-RLM-LASSO-HR-ARD	LASSO-HR
sMAPE	RLM-HR-BR	RLM

Finally, the '(TL)DCF:ALL' and '(TL)PE:ALL' refers to transfer learning portfolios where predictions are made by averaging the outputs of all forecasting model types, as outlined in Chapter 6.

We begin the presentation of our results with the DCF portfolios, which are outlined in the next section.

DCF portfolios

Table 7.2 summarizes the ratios for the DCF portfolios and the benchmarks.

Table 7.2: *DCF: Summary ratios. Best values are highlighted in* ***boldface***

Portfolio	**Annualized Average Return**	**Annualized Std. Dev.**	**Sharpe Ratio**	**Treynor Ratio**	**Sortino Ratio**
MKT	0.05032	**0.18781**	0.004435	0	0.006244
DCF:ACT	0.122762	0.204856	0.026342	0.000414	0.039032
(TL)DCF:ALL	0.063717	0.20659	0.008117	0.000126	0.011615
DCF:ARIMA	0.047192	0.202967	0.003133	0.000048	0.004512
DCF:HR	0.063474	0.188323	0.008823	0.000137	0.012498
DCF:LASSO	0.055429	0.216144	0.005342	0.000086	0.007695
(TL)DCF:MAE	0.05707	0.200433	0.006277	0.000098	0.009072
(TL)DCF:MAPE	**0.088435**	0.205076	**0.015769**	**0.000248**	**0.023035**
DCF:RLM	0.066265	0.201074	0.009138	0.000143	0.013246
(TL)DCF:sMAPE	0.066265	0.201074	0.009138	0.000143	0.013246
NAIVE	0.048405	0.201611	0.003533	0.000055	0.005073

Table 7.2 suggests the following implications. The highest Annualized Average Rate of Return ('Annualized Average Return' in Table 7.2), is observed in the portfolio made with transfer learning predictions produced by the highest-ranked estimators, according to the MAPE error metric. This implies that, on an annual basis, the portfolio added an 8.8% gain every year, compared to the market benchmark, which added approximately 5%. For comparison, the actual data portfolio achieved a 12% annual gain.

The lowest annualized standard deviation is observed in the market portfolio. In fact, this is the only measure where the market outperforms every portfolio in our selection. The highest

risk measure is seen in the LASSO-based portfolio, while the transfer learning-based portfolios all exhibit approximately 20% annualized deviations.

The highest Sharpe, Treynor, and Sortino ratios are all achieved by the transfer learning, MAPE-based portfolios. This indicates that this portfolio, in particular, provides the highest return per unit of daily deviation (Sharpe ratio), the best compensation for potential negative systematic events (Treynor ratio), and the highest reward for returns below expectations (Sortino ratio).

Next, we conduct a Friedman test ranking analysis to assess how all our portfolios are ranked based on their respective daily portfolio returns. The results of the Friedman test are presented in Table 7.3.

The Friedman average ranking test is conducted in two steps. In the first step, every portfolio is ranked based on its average daily portfolio return, with a higher rank indicating higher return. The average rank, referred to as 'Rank' in Table 7.3, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control method (marked as 'control' in Table 7.3) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that each portfolios' returns are drawn from the same distribution.

For comparative purposes, we exclude the portfolio based on Actual data ('ACT'), as its inclusion might distort the results and analysis.

The Friedman test results conducted on DCF-based daily portfolio returns, presented in Table 7.2, suggest that the transfer-learning, MAPE-based model-type combination serves as the control for the group. It is statistically significant against the market (MKT), naive averaging (naive), HR, MAE, RLM, and sMAPE portfolios.

Table 7.3: *DCF: Friedman Test results. The Hommel's post-hoc correction p-value (the 'p-val' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

Portfolio	**Rank**	**p-value**
(TL)DCF:MAPE	5.031702899	control
(TL)DCF:ALL	5.037137681	9.76E-01
DCF:LASSO	5.083333333	9.76E-01
ARIMA	5.221014493	8.97E-01
NAÏVE	5.551630435	**1.73E-02**
(TL)DCF:sMAPE	5.601449275	**8.85E-03**
RLM	5.601449275	**8.85E-03**
(TL)DCF:MAE	5.727355072	**9.45E-04**
MKT	6.00	**8.62E-07**
HR	6.144927536	**9.05E-09**

We acknowledge that the combination of all 7 estimators, denoted as '(TL)DCF:ALL' in Table 7.3 is the second best according to average ranking, not significant against the 'control'. Due to its design in the transfer learning methodology, these two are bounded to be close: possibly, predictions of estimators used in the '(TL)DCF:MAPE' were assigned higher weights in the '(TL)DCF:ALL', therefore leading to almost the same result. The minor difference in the results of the two is due to other, less accurate estimators also used in the '(TL)DCF:ALL' methodology.

Next, we consider results of portfolios based on the PE valuation model.

PE portfolios

Table 7.4 summarizes the key financial ratios for PE-based portfolios.

Results presented in Table 7.4 reveal several important conclusions. In terms of annualized return performance, the transfer-learning-based combination of all estimators portfolio, labeled

Table 7.4: *PE: Summary ratios. Best values are highlighted in* ***boldface***

Portfolio	Annualized Average Return	Annualized Std. Dev.	Sharpe Ratio	Treynor Ratio	Sortino Ratio
MKT	0.05032	0.18781	0.004435	0	0.006244
PE:ACT	0.158914	0.190842	0.04021	0.00052	0.058373
(TL)PE:ALL	**0.152753**	0.200654	**0.036309**	**0.000483**	**0.053073**
PE:ARD	0.093637	0.193688	0.018392	0.000255	0.026594
PE:ARIMA	0.085697	0.195623	0.015653	0.000204	0.022522
PE:HR	0.104118	**0.185365**	0.022779	0.000302	0.03294
(TL)PE:MAE	0.108326	0.196682	0.022813	0.000295	0.033069
(TL)PE:MAPE	0.103521	0.196846	0.021257	0.000276	0.030784
PE:OLS	0.101419	0.194783	0.020805	0.00028	0.029885
(TL)PE:sMAPE	0.116576	0.197611	0.025336	0.000329	0.036808
PE:NAIVE	0.110223	0.190443	0.024191	0.000330	0.035043

as '(TL)PE:ALL', yields the highest results, gaining approximately 15.27% per annum, on average, over the backtesting period. Notice how close returns of this portfolio are to that of the actual data, denoted as 'PE:ACT', which sees a 15.89% annualized return. In comparison, our benchmark market, labeled as MKT in Table 7.4, gains 5.03% annually. Results presented in Table 7.4 reveal several important conclusions. In terms of annualized return performance, the transfer-learning-based combination of all estimators portfolio, labeled as '(TL)PE:ALL', yields the highest results, gaining approximately 15.27% per annum, on average, over the backtesting period. Notice how close this return is to the actual data portfolio, denoted as PE:ACT, which sees a 15.89% annualized return. In comparison, our benchmark market, labeled as MKT in Table 7.4, gains 5.03% annually.

The lowest annualized standard deviation is produced by the HR-based portfolio, which is also smaller than that of the benchmark market portfolio, although only by decimal places.

The transfer-learning-based portfolios are riskier than the benchmark, as risk is defined by the standard deviation. The highest annualized deviation is produced by the '(TL)PE:ALL' portfolio. Interestingly, this suggests an important implication that aligns with financial theory: the higher the risk, the higher the potential gain [8]. It is possible that the higher standard deviation is due to the absence of portfolio optimization, which falls outside the scope of this work.

Once again, the transfer-learning-based portfolio, '(TL)PE:ALL', displays the highest Sharpe, Treynor, and Sortino ratios. Similar to the DCF portfolios, this indicates that this particular portfolio offers the highest premium per unit of risk, as reflected by the Sharpe ratio. Furthermore, this portfolio provides the highest reward per unit of systematic risk, although, as with all portfolios, that reward is close to zero. Finally, regarding the Sortino ratio, this means that the '(TL)PE:ALL' portfolio delivers the best yield even when portfolio returns fall below the expected level.

In addition to the portfolio performance metrics outlined above, we conduct the Friedman test to assess the statistical significance between the returns of the PE portfolios. The results of the Friedman test are presented in Table 7.5.

The Friedman average ranking test is conducted in two steps. In the first step, each portfolio is ranked based on its daily performance (i.e., daily return), with a higher rank indicating a higher return. The average rank, referred to as 'Rank' in Table 7.5, for each method is then computed across all data sets.

In the second step, the method with the highest average rank—referred to as the control portfolio (marked as 'control' in Table 7.5) — is statistically compared to the others. We use the Hommel's post-hoc correction to evaluate significance at the 5% level. The null hypothesis is that returns of each portfolio are drawn from the same distribution.

Table 7.5: *PE: Friedman Test results. The Hommel's post-hoc correction p-value (the 'p-value' column) in boldface indicate statistical significance at 5% significance against the 'control' method.*

Portfolio	Rank	p-value
(TL)PE:ALL	4.99	(control)
(TL)PE:sMAPE	5.32	7.36E-02
PE:ARIMA	5.36	7.36E-02
(TL)PE:MAPE	5.37	7.36E-02
(TL)PE:MAE	5.40	6.24E-02
PE:ARD	5.45	5.20E-02
PE:OLS	5.54	**1.89E-02**
PE:NAIVE	5.63	**3.65E-03**
MKT	5.84	**2.89E-05**
PE:HR	6.07	**3.59E-08**

The Friedman test procedure suggests the '(TL)PE:ALL' portfolio as the control, significantly outperforming the market benchmark and individual estimators such as HR, OLS, and ARD. No significant difference was found among the transfer-learning-based portfolios. Additionally, the ARIMA-based portfolio does not significantly underperform the control method.

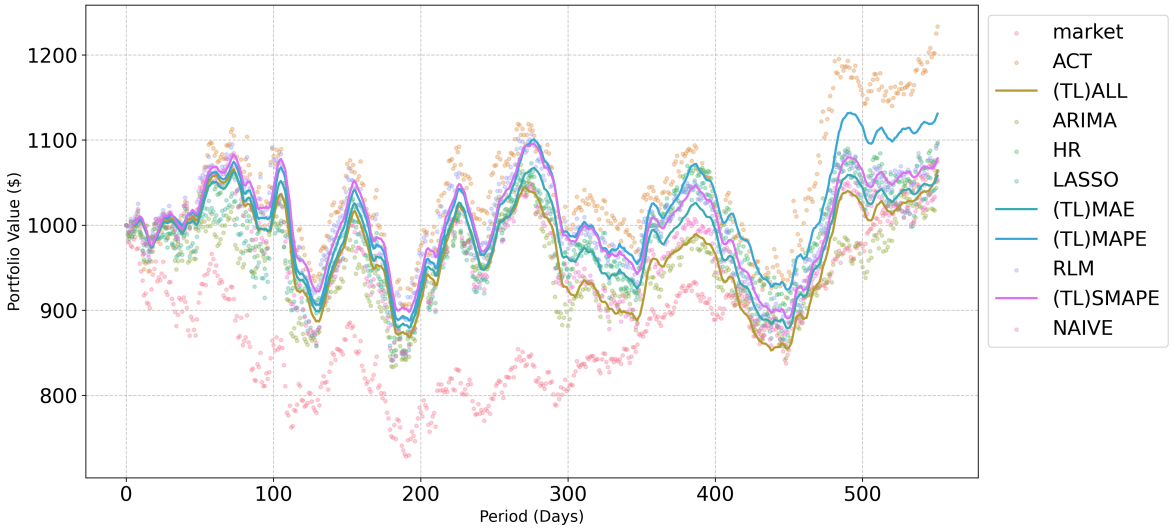
In the next section, we conduct a more in-depth analysis of the different DCF and PE portfolios, providing additional insights into their respective performances.

7.4.2 Results Analysis

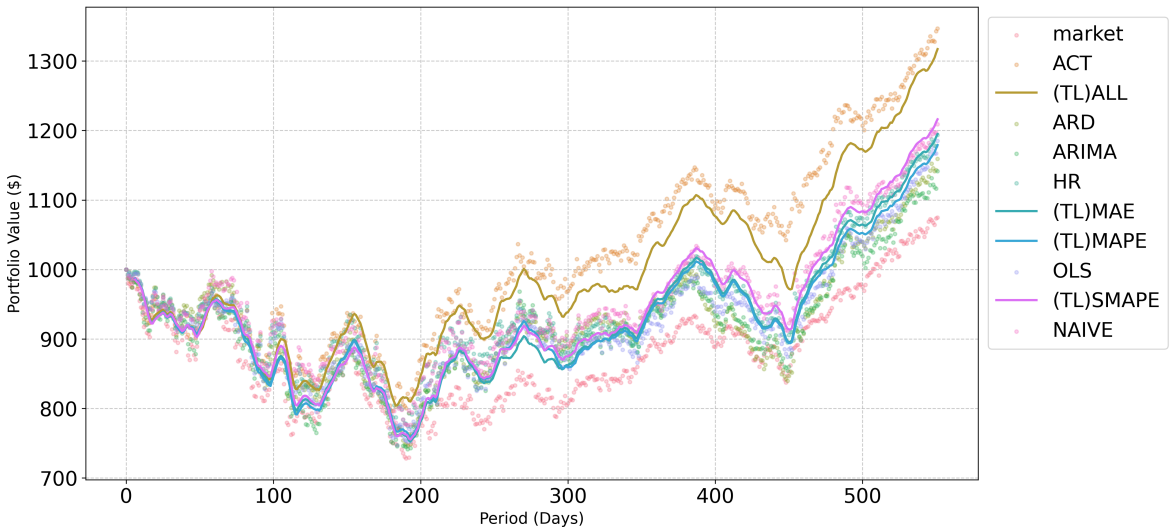
The back-testing results suggest that each of our transfer learning portfolios generates better daily returns than the market.

First, we consider how our portfolios performed during the back-testing period. Specifically, we allocated an initial investment of imaginary $ 1000 for the simulation. Figure 7.1 presents two graphs for the DCF and PE portfolios, respectively.

Figure 7.1: *The movement of $1000 invested through the simulation period*



(a) DCF portfolios



(b) PE portfolios

Our top-ranked portfolios, (TL)DCF:MAPE and (TL)PE:ALL, achieved profits of $150 and $290, respectively. Notably, PE portfolios generally outperformed DCF portfolios, indicating higher daily returns for the former. This could be attributed to two factors. First, the EPS, used for PE intrinsic value estimation, exhibited generally lower errors, as discussed in earlier chapters of this thesis. Second, EPS signals often suggest higher potential profits for investors [11], which may drive stock prices upward. According to the PE valuation formula outlined in Section 2, Equation 2.3, an increase in earnings compared to the previous quarter is one way to push the intrinsic value higher. Therefore, under the rules of our simulation, a stock with higher earnings than the current quarter will be included in the portfolio.

Apart from that, notice the tight movements of the portfolios together. Even when our portfolios outperform the market, we still observe the dynamics of price fluctuations. This information is reflected in the high beta coefficients of every portfolio we consider, as presented in Table 7.6. Beta coefficient is the value obtained from regressing portfolio returns against those of the market (simple OLS regression), where a higher coefficient is indicative of closer co-movement of the overall market and a portfolio.

Table 7.6 provides important insights into the systematic risk of our portfolios. A higher β value indicates that a stock or portfolio moves more in line with the market, with a value of 1 signifying perfect co-movement with the market.

This suggests that, although our PE portfolios yield higher returns, they also result in higher systematic risk compared to the DCF portfolios. In other words, investing based on PE-estimated intrinsic value exposes the investor to higher level of systematic risk than when using the DCF valuation method. This can be attributed to the fact that FCF, the foundation of the DCF formula as outlined in Chapter 2, Equation 2.8, reflects the company's net earnings after accounting for necessary expenses and financing operations [5]. These financing decisions

Table 7.6: *DCF and PE beta values per portfolio*

DCF		**PE**	
Portfolio	β_p	**Portfolio**	β_p
(TL)DCF:ALL	0.838256	(TL)PE:ALL	0.949428
DCF:ARIMA	0.82752	PE:ARIMA	0.94121
DCF:HR	0.763429	PE:HR	0.878958
(TL)DCF:MAE	0.805159	(TL)PE:MAE	0.956608
(TL)DCF:MAPE	0.818507	(TL)PE:MAPE	0.953621
(TL)DCF:sMAPE	0.806423	(TL)PE:sMAPE	0.958004
NAIVE	0.808683	NAIVE	0.878196
DCF:RLM	0.806423	PE:OLS	0.910183
DCF:LASSO	0.839429	PE:ARD	0.87775

are typically based on the expectation of generating higher future cash flows, often extending beyond a single quarter [5].

In other words, the primary reason for the outlined behavior is that a company may sacrifice short-term EPS growth in favor of increasing investments in its business, with the expectation that these investments will generate future returns. This would be reflected in future increases in DCF, while potentially leading to a decrease in near-term EPS.

The second reason for the higher beta of PE-based portfolios is that during economic downturns, companies typically report a decrease in quarterly EPS, and vice versa. As a result, an increase in a company's EPS tends to drive a corresponding rise in its stock price. For the majority of stocks, this increase in stock prices leads to a higher market index value, as described by [8]. It is possible that this relationship results in a higher covariance—and consequently a higher beta value—between the portfolios and the market.

Along with a high beta, our portfolios are also highly correlated with each other, as indicated in Figure 7.2. This figure presents the correlation matrix of all our portfolios,

including the market.

From Figure 7.2, DCF portfolios are moderately correlated with the market (at approximately the 0.75 level), while PE portfolios exhibit very high correlations, typically greater than 0.95. Additionally, DCF and PE portfolios show a moderate correlation with each other, around 0.85, but not exceeding this threshold. This suggests that investors may benefit from pursuing both strategies, as it provides some diversification, potentially minimizing losses during systematic downturns.

Notice that our transfer learning methodology yields higher returns than both the market and most single estimators. We also observed that PE-based portfolios generate higher returns than DCF-based portfolios, which can be attributed to the fact that EPS estimation had lower errors than FCF estimation. This suggests that there may be a relationship between regression errors and portfolio performance.

To establish relationships between regression errors and portfolio returns, we conducted correlation tests between the average and median errors for each estimation method, annualized daily returns, Sharpe ratio, Treynor ratio, and Sortino ratio of our resulting portfolios. The correlations are summarized in Tables 7.7 and 7.8, for DCF-FCF and PE-EPS series, respectively. Note that for regression error computation, we only included the regression errors generated by the Transfer Learning methodology.

In both cases, the correlations indicate a moderately negative relationship between regression error and the performance ratios. This suggests that a lower error corresponds to higher daily returns on the portfolio. It is important to note that the regression error measures the distance between model-inferred and target values. Specifically, both of these values could be negative. Therefore, a better score indicates a smaller distance between either positive or negative values.

Figure 7.2: *Correlations of DCF, PE and market portfolios*

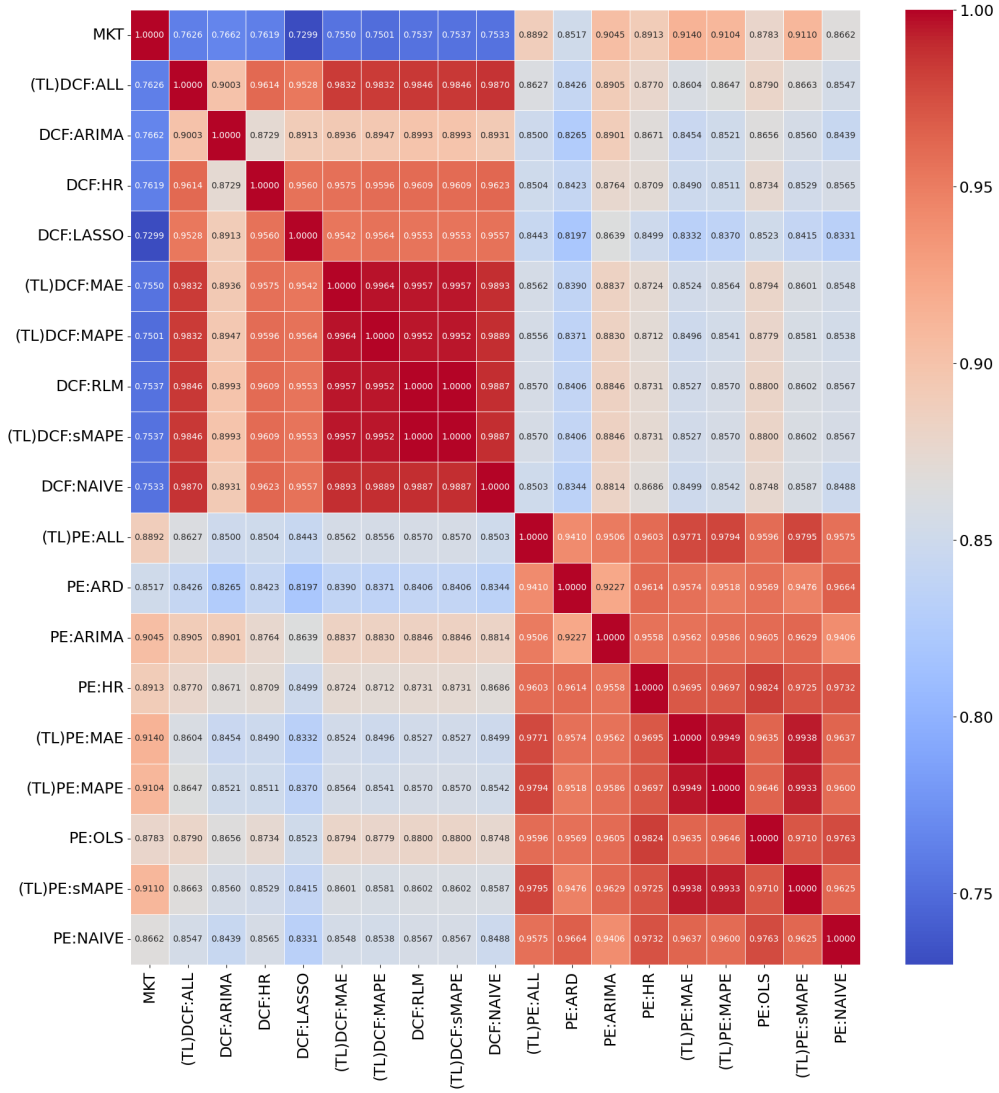


Table 7.7: *Correlations between DCF generated returns and FCF errors*

Correlation between	Annualized Average Return	Sharpe Ratio	Treynor Ratio	Sortino Ratio
Average MAE	-0.57984	-0.5908	-0.59592	-0.58497
Median MAE	-0.58978	-0.60182	-0.60543	-0.59535
Average MAPE	-0.55023	-0.55351	-0.56003	-0.54798
Median MAPE	-0.54007	-0.5436	-0.54988	-0.53743
Average sMAPE	-0.65564	-0.68208	-0.6805	-0.67489
Median sMAPE	-0.62791	-0.64631	-0.64707	-0.63837

Table 7.8: *Correlations between PE generated returns and EPS errors*

Correlation between	Annualized Average Return	Sharpe Ratio	Treynor Ratio	Sortino Ratio
Average MAE	-0.50499	-0.52237	-0.52937	-0.51996
Median MAE	-0.49626	-0.51586	-0.52438	-0.51327
Average MAPE	-0.48284	-0.50522	-0.51446	-0.50246
Median MAPE	-0.48565	-0.49867	-0.50840	-0.49647
Average sMAPE	-0.56051	-0.57666	-0.56897	-0.57444
Median sMAPE	-0.53701	-0.55569	-0.55682	-0.55329

In contrast, for the valuation models to work correctly, the resulting intrinsic value must be positive and higher than the current stock price. Therefore, if the predicted EPS or DCF value is negative, the resulting intrinsic value will also be negative, and it will fall outside the scope of our portfolio selection methodology. This excludes the possibility of perfectly correlated errors and portfolio measures, as short selling is restricted. Therefore, it is possible that the current correlations represent the highest achievable in our methodology.

In next section, we summarize results of this chapter with concluding remarks.

7.5 Conclusions

In this chapter, we conducted a back-testing simulation, forming portfolios using the DCF and PE valuation models. For both models, we used quarter-ahead predictions of FCF and EPS, mined through the transfer learning methodology outlined in Chapter 6.

The simulation is programmed to buy a stock if its intrinsic value exceeds the observed market price and to sell it otherwise. Portfolio revisions occur quarterly, in line with the publication of quarterly financial results.

Our results suggest that both DCF and PE portfolios outperform the market index, with statistical significance at the 5% level. The transfer learning portfolios also yield the highest Sharpe, Treynor, and Sortino ratios, which are commonly used to measure the quality of an investment strategy. However, performance comes at a cost: our highest-ranking portfolio also exhibits the highest standard deviation. This may be due to the fact that no portfolio optimization was performed, as we aimed to evaluate the performance of our strategy without the influence of additional optimization techniques.

In absolute terms, our PE-based portfolios outperform the DCF-based portfolios. This

is primarily due to the fact that the DCF-based method requires a long-term perspective. Additionally, the EPS series, which is used in the PE valuation model, has direct implications on stock returns, as highlighted in financial literature.

Importantly, our results show a moderate correlation between portfolio metrics and regression errors. Since companies with negative FCF/EPS series are excluded from the simulation, we speculate that this correlation reflects an inverse relationship between portfolio returns and regression errors.

In the next chapter, we summarize the results and key findings of this study.

Chapter 8

Conclusions

This work introduces an innovation to the most common stock valuation methodologies. Specifically, we propose a methodology for predicting the EPS and FCF series, which are used by the Forward PE and DCF valuation models. Given that these series are sparse, limited in the number of observations, and exhibit high excess kurtosis, we outline a methodology to model them. Additionally, we investigate the performance of machine learning (ML) and statistical estimation (SE) methods for forecasting such series. At its core, this study addresses a supervised regression problem, where the target variables (EPS, FCF) are regressed against a set of features (past lags, mean, and standard deviation). In this chapter, we summarize the key information and contributions of this work, beginning with the data pipelines in Section 8.1.

8.1 Interpolation and Transformations

In Chapter 4, we conducted a series of experiments using the OLS estimator as a proxy for the forecasting model on a set of 50 companies. We found that both the EPS and FCF datasets are

non-stationary, non-normally distributed, and exhibit a tendency toward either a positive or negative trend. Additionally, this chapter introduces two data preprocessing pipelines.

The first data pipeline utilizes PCHIP interpolation, which was found to be the most suitable for the FCF series. Assuming the data arrives on a weekly rather than a quarterly basis, the interpolation smooths out the series, making it easier for the OLS regression to model the relationship between the FCF variable and its features. In contrast, for the EPS series, all interpolation methods led to further overfitting of the model, by making the interpolated series 'too smooth' and leading the estimator to biased predictions.

The second data pipeline, which is more appropriate for the EPS series than for FCF, uses Quantile Transformation. This technique not only scales the data to the 0-1 range but also forces the data to follow a uniform distribution.

From the results of our experiments, we observe that the transformer highlights the linear relation of the EPS with its lagged features, therefore simplifying the estimation process for the OLS estimator. In contrast, in a more dynamic FCF data sets, the Quantile Transformer bring no benefits.

These data pipelines are then used for further experiments, summarized next.

8.2 Machine Learning and Statistical Estimators Experiments

In Chapter 5, we expand our selection of datasets from 50 to 100 companies, while also introducing the additional estimators outlined in Chapter 2. The diverse selection of cost functions and regularization properties of the estimator coefficients enables us to investigate the impact of sparse series on inference.

First, we found that for both the FCF and EPS series, the SE and ML estimators perform

interchangeably, with no statistically significant difference between the various parametric models. Furthermore, depending on the error metric used, the Friedman test ranking highlights either the ML or SE estimator.

Second, there is a set of estimators, predominantly ARIMA, MLP, KNN, SES, DT, and RF, that do not perform well on sparse series. This is because MLP, DT, and RF subsample the training data, leading to optimization on even smaller subsets, which makes the estimators more prone to overfitting. On the other hand, ARIMA, KNN, and SES tend to make forecasts that overlook the local variability of the data, resulting in poor generalization to the test data.

Third, we observed that in datasets with low errors, the target variable and features generally exhibit high positive or negative correlations. Additionally, in these datasets, the correlations tend to be stable or intensify as we progress through the test set. In contrast, for datasets with higher errors, the correlations tend to approach zero or fluctuate rapidly, making it difficult for estimators to accurately infer relationships between the target and features.

Since there is no clear distinction in performance between ML and SE, we can combine their strengths through transfer learning to overcome their mutual limitations. This approach is the central focus of Chapter 6, which is summarized in the next section.

8.3 Transfer Learning Bayesian Averaging of estimators

Transfer learning is a method designed to address the primary challenge of our series: the limited number of observations. Additionally, we found that parametric estimators tend to perform better than non-parametric ones.

Therefore, we propose a two-step approach for forecasting a data point in a dataset. First, we transfer the knowledge of an estimator's performance across other datasets. Second, based

on this prior step, we estimate a weight for each estimator, determined by its realized error on the other datasets.

Our methodology offers several benefits. First, learning and transferring knowledge occurs within the same data domain. Second, different estimators provide unique insights into the data. These two advantages result in a significant improvement over the benchmark methodology. On the other hand, a major limitation of our methodology is its computational complexity. However, since the data arrives quarterly, the accuracy of predictions outweighs concerns about runtime performance.

This methodology is then used to create stock portfolios, which are backtested against the market benchmark in Chapter 7, summarized in the next section.

8.4 Out-of-sample testing summary

In the final chapter of this study, we conduct out-of-sample testing. Specifically, for each stock in our selection, we perform Forward PE and DCF valuations. If the stock is undervalued, it is bought; otherwise, it is sold. The collection of undervalued stocks purchased forms a portfolio, which is then tracked against the broad market benchmark. Our results indicate that the proposed methodology yields the top-performing portfolio.

We recognize that the transfer learning-based portfolio carries higher risk, which aligns with financial literature: the higher the risk, the higher the potential reward. Additionally, portfolio optimization could help mitigate this risk, though it is outside the scope of our study. Beyond outperforming market benchmarks, our methodology also surpasses the performance of non-transfer learning-based estimators, further validating the effectiveness of our approach.

Furthermore, the analysis of regression errors and portfolio metrics reveals a moderately

negative relationship between the two: the higher the regression error, the lower the portfolio performance. This supports our earlier hypothesis regarding the importance of accurately estimating the EPS and FCF values. Notably, we believe the correlations are only moderately negative due to the fact that while FCF and EPS values can be negative, asset prices cannot. Specifically, a negative FCF/EPS value results in a negative 'intrinsic value.' In such cases, our program either ignores the stock or sells it if it is already in the portfolio. Therefore, it is possible that the moderate correlation observed is the highest achievable in this type of measurement.

Next section outlines possible future directions of this study.

8.5 Further Research

In Chapter 3, we observed the growing popularity of Generative Adversarial Networks (GANs) applied to the generation of synthetic time series data. These techniques could be integrated into our proposed methodology to simulate potential scenarios of how the target and features might behave, given their distributions. This would enable the training of estimators based on hypothetical relationships, potentially further minimizing regression errors. Additionally, GANs are known to be effective at isolating the noise from the data, therefore, studying the merits of explicit denoising of fundamental financial time-series data prior to fitting regression estimators is another topic for further research.

Our proposed methodology limits the number of possible input features. Therefore, caching and creating a parameterization space of varying sizes could potentially reduce the regression error further. In this context, it would be valuable to identify a methodology to generate possibly more complicated combinations of greater number of statistical/ML estimators. Such

approach could help in further selecting more advanced data augmentation/transformation techniques for each dataset individually, as well as in choosing the optimal combination of estimation algorithms.

Throughout our study, we employed Kernel Density Estimation (KDE) to estimate the density score of parameterization. However, more advanced techniques, such as Gaussian Mixture Models and others, could also be applied for density estimation. Since we relied on regression coefficients for knowledge transfer, we were limited in the number of estimators we could utilize. Coefficients indicate which features were most useful for regressing the target against the features. There are other indicators that could be leveraged, allowing for a broader set of estimators to be considered.

Finally, in terms of valuation, we focused solely on the EPS and FCF series. First, it would be possible to combine the estimated intrinsic values obtained from the PE and DCF models, thereby estimating the variability of such values. Second, additional valuation models could be explored. In this case, it would be necessary to estimate other financial series, such as revenues, total assets, and others.

Appendix

Appendix A: Chapter 4

In this section of the appendix, we present metadata for the 50 data sets used in Chapter 4, including the full legal name of each entity, along with its economic industry and sector. This information is summarized in Table A-1.

For both the EPS and FCF series of each company, we also report the number of available observations before and after applying monthly or weekly interpolation. It is important to note that the interpolation method itself does not determine the number of interpolated data points; rather, this depends on the original sample size. Accordingly, we use first-order PCHIP interpolation to generate the interpolated data sets. The results are summarized in Tables A-2 and A-3 for the EPS and FCF series, respectively.

In addition, we present the p-values from the statistical tests conducted in Chapter 4. These are summarized in Tables A-4 and A-5 for the EPS and FCF series, respectively.

In the main part of Chapter 4, we presented scatterplots for two of five companies, randomly selected for demonstration. Here, we include the corresponding scatterplots for the remaining three companies: Comcast Corporation (CMCSA), Kimberly-Clark Corporation (KMB), and The Sherwin-Williams Company (SHW). These plots are provided in Figures A-1 and A-2 for

the EPS and FCF series, respectively.

Table A-1: *Metadata for 50 companies used in Chapter 4*

Ticker	**Full Business Name**	**Industry**	**Sector**
AMAT	Applied Materials Inc.	Semiconductor Equipment & Materials	Technology
AMZN	Amazon.com Inc.	Internet Retail	Consumer Cyclical
ANSS	ANSYS Inc.	Software - Application	Technology
APD	Air Products and Chemicals Inc.	Specialty Chemicals	Basic Materials
APH	Amphenol Corporation	Electronic Components	Technology
ATI	ATI Inc.	Metal Fabrication	Industrials
BBY	Best Buy Co. Inc.	Specialty Retail	Consumer Cyclical
CAH	Cardinal Health Inc.	Medical Distribution	Healthcare
CAT	Caterpillar Inc.	Farm & Heavy Construction Machinery	Industrials
CF	CF Industries Holdings Inc.	Agricultural Inputs	Basic Materials
CHRW	C.H. Robinson Worldwide Inc.	Integrated Freight & Logistics	Industrials
CLX	The Clorox Company	Household & Personal Products	Consumer Defensive
CMCSA	Comcast Corporation	Telecom Services	Communication Services
COO	The Cooper Companies Inc.	Medical Instruments & Supplies	Healthcare
COST	Costco Wholesale Corporation	Discount Stores	Consumer Defensive
CTXS	Citrix Systems Inc.	Software - Application	Technology
CVS	CVS Health Corporation Inc.	Healthcare	Healthcare Plans

Table A-1 continued

Ticker	Full Business Name	Industry	Sector
DE	Deere & Company	Farm & Heavy Construction Machinery	Energy
DHI	D.R. Horton Inc.	Residential Construction	Consumer Cyclical
DISH	DISH Network Corporation	Telecom Services	Communication Services
DRI	Darden Restaurants Inc.	Restaurants	Consumer Cyclical
ECL	Ecolab Inc.	Specialty Chemicals	Basic Materials
EMR	Emerson Electric Co.	Specialty Industrial Machinery	Industrials
GOOG	Alphabet Inc.	Internet Content & Information	Communication Services
GWW	W.W. Grainger Inc.	Industrial Distribution	Industrials
HD	The Home Depot Inc.	Home Improvement Retail	Consumer Cyclical
INTU	Intuit Inc.	Software - Application	Technology
JNJ	Johnson & Johnson	Drug Manufacturers - General	Healthcare
KMB	Kimberly-Clark Corporation	Household & Personal Products	Consumer Defensive
KSS	Kohl's Corporation	Department Stores	Consumer Cyclical
MAS	Masco Corporation	Building Products & Equipment	Industrials
MLM	Martin Marietta Materials Inc.	Building Materials	Basic Materials
MRK	Merck & Co. Inc.	Drug Manufacturers - General	Healthcare
NKE	NIKE Inc.	Footwear & Accessories	Consumer Cyclical

Table A-1 continued

Ticker	Full Business Name	Industry	Sector
NRG	NRG Energy Inc.	Utilities - Independent Power Producers	Utilities
NVDA	NVIDIA Corporation	Semiconductors	Technology
OI	O-I Glass Inc.	Packaging & Containers	Consumer Cyclical
PM	Philip Morris International Inc.	Tobacco	Consumer Defensive
PTEN	Patterson-UTI Energy Inc.	Oil & Gas Drilling	Energy
SHW	The Sherwin-Williams Company	Specialty Chemicals	Basic Materials
SSD	Simpson Manufacturing Co. Inc.	Lumber & Wood Production	Basic Materials
TEX	Terex Corporation	Farm & Heavy Construction Machinery	Industrials
TGNA	TEGNA Inc.	Broadcasting	Communication Services
TTC	The Toro Company	Tools & Accessories	Industrials
URBN	Urban Outfitters Inc.	Apparel Retail	Consumer Cyclical
VZ	Verizon Communications Inc.	Telecom Services	Communication Services
WLY	John Wiley & Sons Inc.	Publishing	Communication Services
WM	Waste Management Inc.	Waste Management	Industrials
WMT	Walmart Inc.	Discount Stores	Consumer Defensive
XRAY	DENTSPLY SIRONA Inc.	Medical Instruments & Supplies	Healthcare

Table A-1 continued

Ticker	Full Business Name	Industry	Sector

Table A-2: *Earnings Per Share number of observations original, after applying monthly/weekly interpolation, used in Chapter 4*

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
AMAT	101	26	302	1306
AMZN	79	20	235	1018
ANSS	86	22	256	1110
APD	101	26	301	1306
APH	98	25	292	1266
ATI	80	21	238	1031
BBY	101	26	300	1301
CAH	85	22	253	1097
CAT	133	34	397	1723
CF	52	13	154	667
CHRW	71	18	211	914
CLX	101	26	301	1305
CMCSA	101	26	301	1306
COO	101	26	301	1305
COST	88	23	262	1137

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
CTXS	88	23	262	1137
CVS	101	26	301	1306
DE	101	26	301	1305
DHI	97	25	289	1253
DISH	88	23	262	1136
DRI	82	21	244	1058
ECL	101	26	301	1306
EMR	101	26	301	1306
GOOG	58	15	172	744
GWW	101	26	301	1306
HD	101	26	302	1306
INTU	92	23	274	1188
JNJ	133	34	397	1722
KMB	102	26	304	1319
KSS	93	24	277	1201
MAS	101	26	301	1306
MLM	88	23	262	1136
MRK	101	26	301	1306
NKE	101	26	301	1306
NRG	56	15	166	718
NVDA	70	18	208	901
OI	98	25	292	1266

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
PM	48	13	142	614
PTEN	92	23	274	1188
SHW	101	26	301	1306
SSD	92	23	274	1188
TEX	101	26	301	1306
TGNA	101	26	301	1306
TTC	101	26	303	1318
URBN	88	23	262	1136
VZ	101	26	301	1306
WLY	101	26	301	1305
WM	79	20	235	1018
WMT	101	26	301	1305
XRAY	92	23	274	1188

Table A-3: *Free Cash Flows: number of observations original, after applying monthly/weekly interpolation, used in Chapter 4*

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
AMAT	86	22	256	1110
AMZN	78	20	232	1005
ANSS	82	21	244	1057

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
APD	73	19	217	940
APH	85	22	253	1097
ATI	79	20	235	1018
BBY	84	21	249	1079
CAH	86	22	256	1110
CAT	78	20	232	1005
CF	51	13	151	654
CHRW	68	17	202	874
CLX	83	21	247	1070
CMCSA	72	18	214	927
COO	88	23	262	1135
COST	82	21	245	1061
CTXS	82	21	245	1061
CVS	84	22	250	1084
DE	76	19	226	979
DHI	88	22	262	1136
DISH	85	22	253	1097
DRI	83	21	247	1071
ECL	85	22	253	1097
EMR	86	22	256	1110
GOOG	60	15	178	770
GWW	84	22	250	1083

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
HD	84	22	250	1084
INTU	82	21	244	1058
JNJ	85	22	254	1097
KMB	84	22	250	1083
KSS	84	22	249	1083
MAS	85	22	253	1097
MLM	81	21	241	1044
MRK	85	22	253	1097
NKE	83	21	247	1071
NRG	75	19	223	966
NVDA	72	19	214	927
OI	84	22	250	1083
PM	44	12	130	562
PTEN	84	22	250	1083
SHW	85	22	253	1097
SSD	85	22	253	1097
TEX	84	22	250	1083
TGNA	85	22	253	1097
TTC	81	21	241	1045
URBN	84	22	250	1084
VZ	63	16	187	810
WLY	68	18	202	875

Data set	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
WM	79	20	235	1018
WMT	84	22	250	1084
XRAY	84	22	250	1083

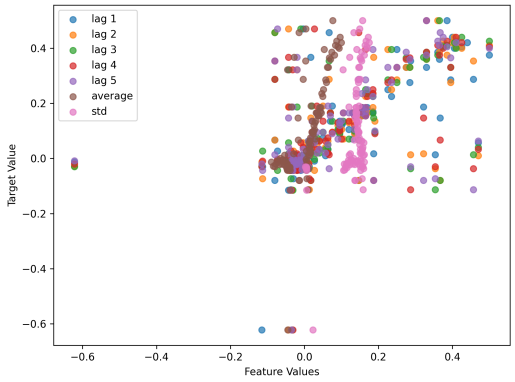
Table A-4: *P-Values from statistical tests conducted on Earnings Per Share data sets in Chapter 4. 'SW' denotes Shapiro-Wilk test, 'DF' denotes Augmented Dickey-Fuller test, 'MK' denotes Mann-Kendall test*

Data Set	SW	DF	MK	Data Set	SW	DF	MK
SHW	1.56e-12	1.00000	0.04436	INTU	4.38e-17	1.00000	0.00035
CAT	3.04e-12	0.91490	1.96e-06	VZ	1.57e-10	1.62e-06	1.44e-11
ANSS	3.40e-11	0.99830	0.00664	KSS	1.35e-10	0.43721	5.11e-15
WLY	3.24e-13	0.83823	1.30e-08	WM	7.14e-08	0.75680	0.00940
ECL	2.07e-18	4.12e-15	2.46e-05	AMAT	2.24e-14	0.99314	0.01532
DRI	1.70e-09	0.00241	0.00027	CMCSA	3.59e-13	0.74913	0.00718
DHI	1.02e-11	0.96384	0.01503	TEX	1.43e-06	0.00159	1.12e-07
CHRW	4.08e-06	0.96417	0.01182	GOOG	1.11e-08	0.98128	3.43e-05
CAH	3.96e-20	1.48e-19	6.30e-07	NVDA	2.53e-18	1.00000	3.29e-11
TGNA	4.12e-23	0.00001	0.58983	CLX	7.57e-07	0.08434	2.16e-07
NRG	4.31e-12	5.64e-12	0.36021	XRAY	5.10e-20	0.00170	1.75e-05
BBY	1.89e-12	0.37466	0.03736	MLM	2.37e-15	1.00000	1.91e-11
PM	0.03469	0.20896	6.14e-06	HD	2.98e-13	0.99357	7.64e-09
JNJ	1.53e-19	0.99583	0.00011	COO	2.48e-23	6.08e-07	5.11e-05
APD	3.67e-14	0.93602	0.00068	ATI	2.50e-14	0.00101	0.14407
WMT	0.00017	0.89754	2.58e-05	APH	8.44e-11	1.00000	4.18e-05
TTC	2.33e-11	0.96794	0.00857	CTXS	1.76e-10	0.04954	2.34e-08
DE	2.46e-13	0.98739	8.71e-06	MAS	2.31e-12	0.29982	4.72e-09
DISH	9.56e-07	0.00002	2.00e-14	COST	4.54e-10	1.00000	0.00089
MRK	1.37e-09	0.38663	8.29e-07	NKE	1.00e-08	0.99678	5.18e-18
PTEN	1.60e-07	0.04013	0.09911	KMB	2.43e-05	0.69403	2.26e-05
SSD	2.66e-13	0.98289	4.30e-05	EMR	3.25e-23	0.99770	0.00045
OI	1.18e-15	1.67e-21	0.21646	CF	1.39e-06	0.08327	0.04054
CVS	1.92e-08	0.65294	0.01936	AMZN	8.04e-15	0.01316	0.00772
URBN	1.12e-09	0.96117	8.35e-11	GWW	1.62e-13	1.00000	6.47e-13

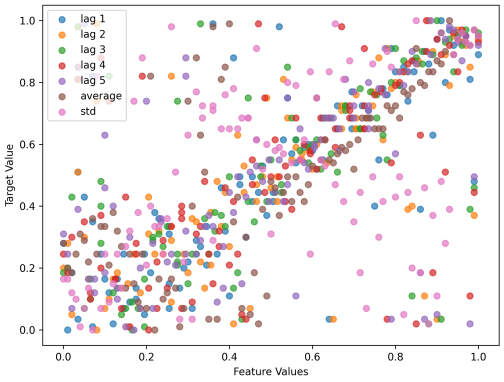
Table A-5: *P-Values from statistical tests conducted on Free Cash Flows data sets in Chapter 4. 'SW' denotes Shapiro-Wilk test, 'DF' denotes Augmented Dickey-Fuller test, 'MK' denotes Mann-Kendall test*

Data Set	**SW**	**DF**	**MK**	**Data Set**	**SW**	**DF**	**MK**
SHW	1.77e-7	8.93011e-1	1.51e-11	INTU	2.19e-14	1e+0	2.65616e-2
CAT	7.97e-6	9.94514e-1	4.77e-12	VZ	5.97565e-1	2.90586e-1	7.32e-6
ANSS	7.09e-7	9.98576e-1	3.5987e-5	KSS	4.54e-5	5.96799e-1	2.58652e-4
WLY	1.3573e-3	3.88973e-1	4.08475e-1	WM	1.11119e-3	4.99064e-1	1.0289e-4
ECL	3.75e-6	9.32055e-1	4.44e-16	AMAT	1.67e-12	9.98809e-1	9.98794e-3
DRI	3.08601e-3	9.96751e-1	3.21e-9	CMCSA	4.58996e-4	9.86569e-1	2.38957e-2
DHI	5.96e-6	1.0164e-2	7.54061e-2	TEX	2.34029e-1	3.81338e-1	5.59444e-2
CHRW	9.11e-11	7.24158e-4	1.5e-7	GOOG	1.68e-8	8.77153e-1	8.23929e-3
CAH	1.13e-7	8.19263e-1	1.12e-5	NVDA	6.01e-18	1e+0	5.00213e-2
TGNA	1.19e-11	5.65678e-1	1.76404e-4	CLX	1.76e-5	3.91335e-1	6.57e-9
NRG	1.56e-5	1.58528e-4	4.7099e-3	XRAY	3.34e-10	4.38181e-1	8.12e-10
BBY	1.27e-9	2.04417e-1	3.56509e-3	MLM	5.01e-20	5.23624e-2	1.42167e-4
PM	7.72094e-1	4.25999e-2	3.54584e-2	HD	1.12e-9	9.99023e-1	4.0125e-3
JNJ	8.58663e-4	6.63747e-1	3.63594e-2	COO	2.91e-5	4.38457e-1	6.44e-15
APD	6.24e-7	8.47954e-1	1.65989e-1	ATI	2.7341e-3	4.98505e-3	6.17885e-2
WMT	2.31e-5	7.13884e-1	1.11e-6	APH	1.53e-9	1e+0	1.342e-3
TTC	6.0629e-2	3.31741e-1	3.18e-5	CTXS	1.54e-6	7.0409e-1	2.1542e-2
DE	9.02699e-3	8.26439e-1	5.40319e-2	MAS	3.6135e-1	4.6569e-1	1.83816e-4
DISH	6.4952e-2	4.55484e-1	1.18e-10	COST	9.39e-9	9.97676e-1	2.23e-10
MRK	3.24689e-2	2.49522e-2	1.59085e-4	NKE	9.27e-11	9.96765e-1	1.55e-15
PTEN	1.04e-7	1.47e-13	2.31179e-1	KMB	1.55274e-2	8.58769e-1	9.15e-12
SSD	2.65e-10	9.986e-1	5.07e-6	EMR	1.37085e-4	3.46466e-1	1.02e-9
OI	1.06348e-3	5.49858e-3	2.24051e-2	CF	2.0764e-2	2.80352e-1	3.58293e-4
CVS	2.32e-8	5.29294e-1	1.2934e-3	AMZN	6.14e-10	5.40593e-1	1.98532e-3
URBN	4.14e-9	6.69123e-1	2.43859e-4	GWW	4.26e-8	1e+0	1.9289e-4

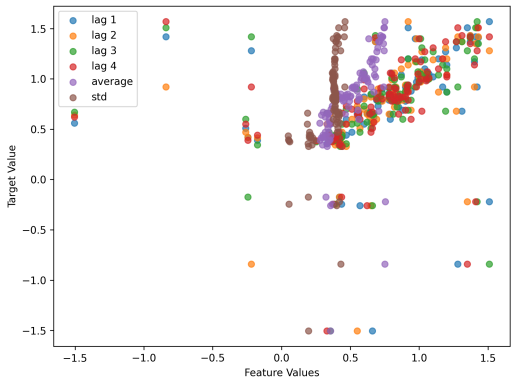
Figure A-1: *EPS: scatter plot of the target series (y-axis) against its features (x-axis), transformed (right) and orignal (left), for three companies randomly selected from the pool of 50 data sets*



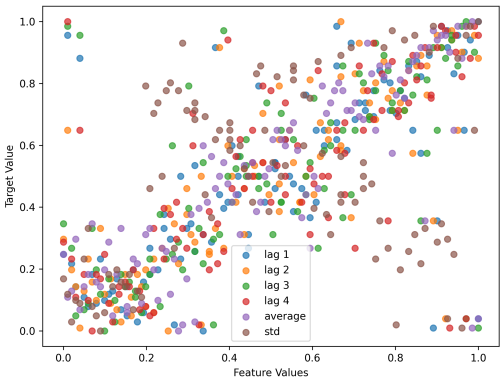
(a) CMCSA: original data scatter plot



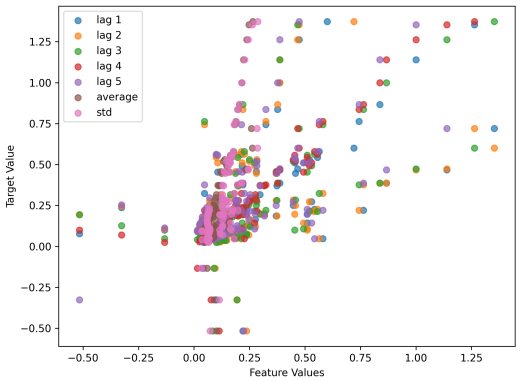
(d) CMCSA: transformed data scatter plot



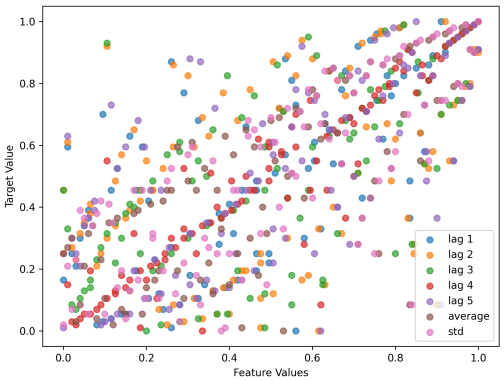
(b) KMB: FCF original data scatter plot



(e) KMB: transformed data scatter plot

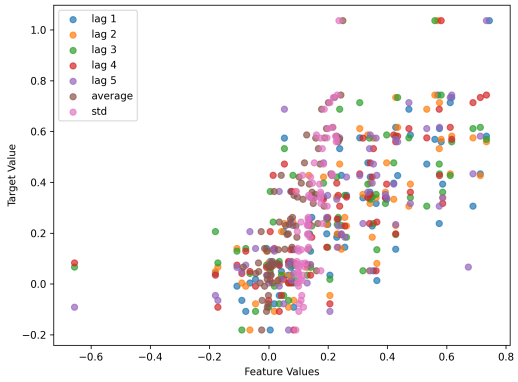


(c) SHW: FCF original data scatter plot

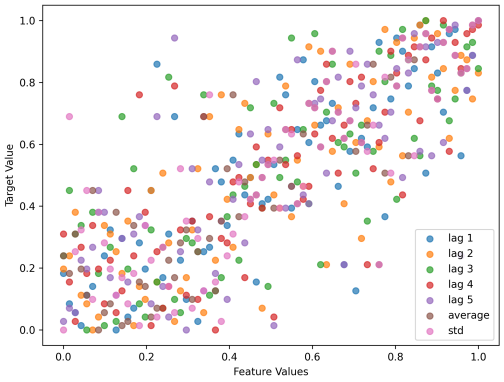


(f) SHW: transformed data scatter plot

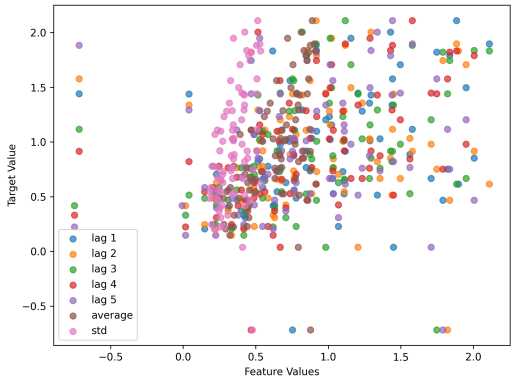
Figure A-2: *FCF: scatter plot of the target series (y-axis) against its features (x-axis), transformed (right) and orignal (left), for three companies randomly selected from the pool of 50 data sets*



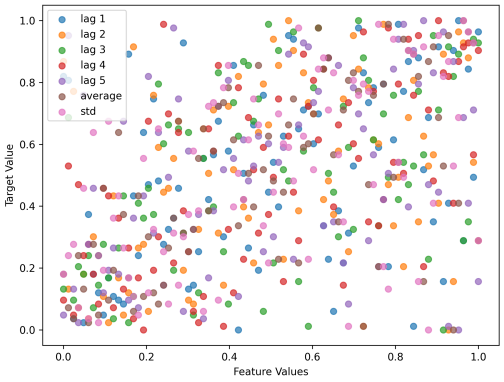
(a) CMCSA: original data scatter plot



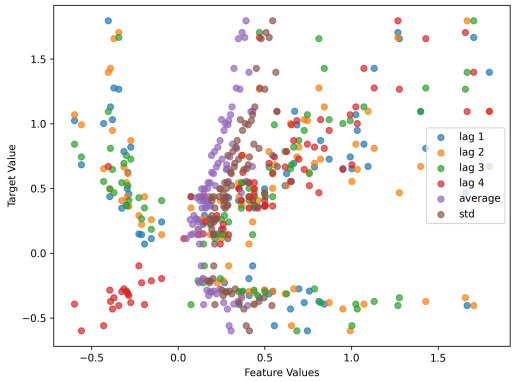
(d) CMCSA: transformed data scatter plot



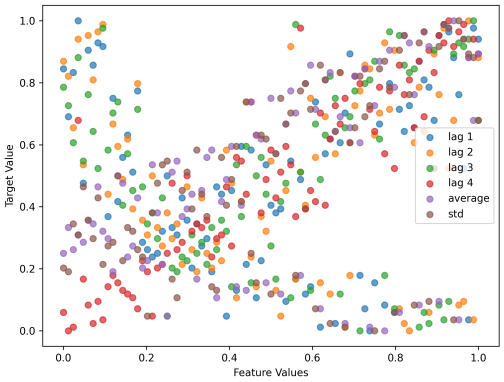
(b) KMB: FCF original data scatter plot



(e) KMB: transformed data scatter plot



(c) SHW: FCF original data scatter plot



(f) SHW: transformed data scatter plot

Appendix B: Chapter 5

In Chapter 5, we conduct a set of experiments using machine learning and statistical estimators, applied to an expanded sample of 100 data sets. Accordingly, this section of the thesis begins by presenting metadata for all 100 companies included in the experiments. This information is summarized in Table B-1.

As established in Chapter 4, interpolation improves forecasting accuracy, particularly for FCF data sets. Table B-2 reports the number of available data points across the 100 FCF data sets, both before and after interpolation.

Chapter 5 is the first chapter we use ML estimators in. Because these estimators have tunable hyperparameters, we perform parameter tuning using the grid-search on the hold-out validation subset of each of 100 sample data sets. Therefore, in this part of the thesis we provide a table of proposed hyperparmeter values for each estimator in Table B-3.

We then introduce the results of applying machine learning and statistical regression models. To assess the performance of the proposed estimators, we present a range of visualizations. As in the previous chapter, we highlight a selected subset of companies, while additional plots are included in this appendix. Specifically, this appendix provides plots for both EPS and FCF series of The Sherwin-Williams Company (SHW), Caterpillar Inc. (CAT), ANSYS Inc. (ANSS), John Wiley & Sons Inc. (WLY), Ecolab Inc. (ECL), Darden Restaurants Inc. (DRI), D. R. Horton Inc. (DHI), and C. H. Robinson Worldwide Inc. (CHRW).

Figures B-1 and B-2 show the training, validation, and test subsets of the target series, along with their lags, for EPS and FCF data respectively. Figures B-3 and B-4 illustrate the autocorrelation between the target variable and its past values, allowing us to explore temporal dependencies. Since we use an expanding window approach, as described in Section 4.2.2 of

Chapter 4, correlations are measured at each iteration over the test set.

Next, we present predictions generated by various estimators. Figures B-5 and B-6 show the forecasts made using the ARIMA model, while Figures B-7 and B-8 display results from the Simple Exponential Smoothing (SES) model.

Among the machine learning estimators employed is the Decision Tree (DT). To illustrate how DTs generate predictions, we include visualizations of their learned structures in Figures B-9 and B-11 for EPS and FCF series respectively (Figures B-10 and B-12 are continuation of main graphs).

Finally, to better understand feature influence in the estimation process, we apply Shapley value analysis. In this section of the Appendix, we present Shapley values force plots for KNN and HR estimators across the remaining eight companies. In particular, Figures B-13 and B-14 show the results for the KNN estimator, while Figures B-15 and B-16 present those for the HR estimator.

Table B-1: *Metadata for 100 companies used in Chapter 5 we conducted regression experiments on*

Ticker	**Full Business Name**	**Industry**	**Sector**
ALB	Albemarle Corporation	Specialty Chemicals	Basic Materials
AMAT	Applied Materials Inc.	Semiconductor Equipment & Materials	Technology
AMD	Advanced Micro Devices Inc.	Semiconductors	Technology
AMZN	Amazon.com Inc.	Internet Retail	Consumer Cyclical
ANSS	ANSYS Inc.	Software - Application	Technology
APD	Air Products and Chemicals Inc.	Specialty Chemicals	Basic Materials

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
APH	Amphenol Corporation	Electronic Components	Technology
ATI	ATI Inc.	Metal Fabrication	Industrials
AZO	AutoZone Inc.	Specialty Retail	Consumer Cyclical
BA	The Boeing Company	Aerospace & Defense	Industrials
BALL	Ball Corporation	Packaging & Containers	Consumer Cyclical
BBWI	Bath & Body Works Inc.	Specialty Retail	Consumer Cyclical
BBY	Best Buy Co. Inc.	Specialty Retail	Consumer Cyclical
BWA	BorgWarner Inc.	Auto Parts	Consumer Cyclical
CAH	Cardinal Health Inc.	Medical Distribution	Healthcare
CAT	Caterpillar Inc.	Farm & Heavy Construction Machinery	Industrials
CE	Celanese Corporation	Chemicals	Basic Materials
CF	CF Industries Holdings Inc.	Agricultural Inputs	Basic Materials
CHRW	C.H. Robinson Worldwide Inc.	Integrated Freight & Logistics	Industrials
CL	Colgate-Palmolive Company	Household & Personal Products	Consumer Defensive
CLX	The Clorox Company	Household & Personal Products	Consumer Defensive
CMCSA	Comcast Corporation	Telecom Services	Communication Services
COO	The Cooper Companies Inc.	Medical Instruments & Supplies	Healthcare
COST	Costco Wholesale Corporation	Discount Stores	Consumer Defensive
CSCO	Cisco Systems Inc.	Communication Equipment	Technology

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
CTXS	Citrix Systems Inc.	Software - Application	Technology
CVS	CVS Health Corporation Inc.	Healthcare	Healthcare Plans
CVX	Chevron Corporation	Oil & Gas Integrated	Energy
DE	Deere & Company	Farm & Heavy Construction Machinery	Energy
DHI	D.R. Horton Inc.	Residential Construction	Consumer Cyclical
DISH	DISH Network Corporation	Telecom Services	Communication Services
DRI	Darden Restaurants Inc.	Restaurants	Consumer Cyclical
DVN	Devon Energy Corporation	Oil & Gas E&P	Energy
EA	Electronic Arts Inc.	Electronic Gaming & Multimedia	Communication Services
EBAY	eBay Inc.	Internet Retail	Consumer Cyclical
ECL	Ecolab Inc.	Specialty Chemicals	Basic Materials
EIX	Edison International	Utilities - Regulated Electric	Utilities
EMN	Eastman Chemical Company	Specialty Chemicals	Basic Materials
EMR	Emerson Electric Co.	Specialty Industrial Machinery	Industrials
FAST	Fastenal Company	Industrial Distribution	Industrials
FL	Foot Locker Inc.	Apparel Retail	Consumer Cyclical
GLW	Corning Incorporated	Electronic Components	Technology
GME	GameStop Corp.	Specialty Retail	Consumer Cyclical

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
GOOG	Alphabet Inc.	Internet Content & Information	Communication Services
GPC	Genuine Parts Company	Auto Parts	Consumer Cyclical
GVA	Granite Construction Incorporated	Engineering & Construction	Industrials
GWW	W.W. Grainger Inc.	Industrial Distribution	Industrials
HAL	Halliburton Company	Oil & Gas Equipment & Services	Energy
HD	The Home Depot Inc.	Home Improvement Retail	Consumer Cyclical
HP	Helmerich & Payne Inc.	Oil & Gas Drilling	Energy
IBM	International Business Machines Corporation	Information Technology Services	Technology
IFF	International Flavors & Fragrances Inc.	Specialty Chemicals	Basic Materials
INTC	Intel Corporation	Semiconductors	Technology
INTU	Intuit Inc.	Software - Application	Technology
JNJ	Johnson & Johnson	Drug Manufacturers - General	Healthcare
KMB	Kimberly-Clark Corporation	Household & Personal Products	Consumer Defensive
KO	The Coca-Cola Company	Beverages - Non-Alcoholic	Consumer Defensive
KSS	Kohl's Corporation	Department Stores	Consumer Cyclical
M	Macy's Inc.	Department Stores	Consumer Cyclical
MAS	Masco Corporation	Building Products & Equipment	Industrials

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
MDLZ	Mondelez International Inc.	Confectioners	Consumer Defensive
MLM	Martin Marietta Materials Inc.	Building Materials	Basic Materials
MMM	3M Company	Conglomerates	Industrials
MO	Altria Group Inc.	Tobacco	Consumer Defensive
MRK	Merck & Co. Inc.	Drug Manufacturers - General	Healthcare
MSFT	Microsoft Corporation	Software - Infrastructure	Technology
NFLX	Netflix Inc.	Entertainment	Communication Services
NKE	NIKE Inc.	Footwear & Accessories	Consumer Cyclical
NOV	NOV Inc.	Oil & Gas Equipment & Services	Energy
NRG	NRG Energy Inc.	Utilities - Independent Power Producers	Utilities
NVDA	NVIDIA Corporation	Semiconductors	Technology
OI	O-I Glass Inc.	Packaging & Containers	Consumer Cyclical
OLN	Olin Corporation	Chemicals	Basic Materials
PARA	Paramount Global	Entertainment	Communication Services
PG	The Procter & Gamble Company	Household & Personal Products	Consumer Defensive
PM	Philip Morris International Inc.	Tobacco	Consumer Defensive
POOL	Pool Corporation	Industrial Distribution	Industrials
PTEN	Patterson-UTI Energy Inc.	Oil & Gas Drilling	Energy

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
QRTEA	Qurate Retail Inc.	Internet Retail	Consumer Cyclical
RSG	Republic Services Inc.	Waste Management	Industrials
SCHL	Scholastic Corporation	Publishing	Communication Services
SHW	The Sherwin-Williams Company	Specialty Chemicals	Basic Materials
SLGN	Silgan Holdings Inc.	Packaging & Containers	Consumer Cyclical
SSD	Simpson Manufacturing Co. Inc.	Lumber & Wood Production	Basic Materials
STLD	Steel Dynamics Inc.	Steel	Basic Materials
TEX	Terex Corporation	Farm & Heavy Construction Machinery	Industrials
TGNA	TEGNA Inc.	Broadcasting	Communication Services
TTC	The Toro Company	Tools & Accessories	Industrials
URBN	Urban Outfitters Inc.	Apparel Retail	Consumer Cyclical
UVV	Universal Corporation	Tobacco	Consumer Defensive
VMC	Vulcan Materials Company	Building Materials	Basic Materials
VRSN	VeriSign Inc.	Software - Infrastructure	Technology
VZ	Verizon Communications Inc.	Telecom Services	Communication Services
WBD	Warner Bros. Discovery Inc.	Entertainment	Communication Services

Table B-1 continued

Ticker	Full Business Name	Industry	Sector
WLY	John Wiley & Sons Inc.	Publishing	Communication Services
WM	Waste Management Inc.	Waste Management	Industrials
WMT	Walmart Inc.	Discount Stores	Consumer Defensive
X	United States Steel Corporation	Steel	Basic Materials
XRAY	DENTSPLY SIRONA Inc.	Medical Instruments & Supplies	Healthcare

Table B-2: *Free Cash Flows: number of observations for each of 100 data sets used for regression experiments*

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
ALB	84	22	250	1083
AMAT	86	22	256	1110
AMD	84	22	251	1084
AMZN	78	20	232	1005
ANSS	82	21	244	1057
APD	73	19	217	940
APH	85	22	253	1097
ATI	79	20	235	1018
AZO	66	17	196	852
BA	78	20	232	1005

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
BALL	85	22	253	1097
BBWI	84	21	249	1083
BBY	84	21	249	1079
BWA	84	22	250	1083
CAH	86	22	256	1110
CAT	78	20	232	1005
CE	52	14	154	666
CF	51	13	151	654
CHRW	68	17	202	874
CL	85	22	253	1097
CLX	83	21	247	1070
CMCSA	72	18	214	927
COO	88	23	262	1135
COST	82	21	245	1061
CSCO	87	22	259	1123
CTXS	82	21	245	1061
CVS	84	22	250	1084
CVX	84	22	250	1083
DE	76	19	226	979
DHI	88	22	262	1136
DISH	85	22	253	1097
DOV	84	22	250	1083

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
DRI	83	21	247	1071
DVN	84	22	250	1083
EA	84	21	250	1083
EBAY	74	19	220	953
ECL	85	22	253	1097
EIX	84	22	250	1083
EMN	84	22	250	1083
EMR	86	22	256	1110
FAST	85	22	253	1097
FL	71	18	212	914
GLW	85	22	253	1097
GME	65	17	193	836
GOOG	60	15	178	770
GPC	85	22	253	1097
GVA	84	22	250	1083
GWW	84	22	250	1083
HAL	86	22	256	1110
HD	84	22	250	1084
HP	84	22	250	1083
IBM	88	23	262	1136
IFF	68	18	202	875
INTC	85	22	254	1097

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
INTU	82	21	244	1058
JNJ	85	22	254	1097
KMB	84	22	250	1083
KO	81	21	241	1044
KSS	84	22	249	1083
M	84	21	249	1083
MAS	85	22	253	1097
MDLZ	66	17	196	849
MLM	81	21	241	1044
MMM	84	22	250	1083
MO	81	21	241	1044
MRK	85	22	253	1097
MSFT	86	22	256	1110
NFLX	62	16	184	797
NKE	83	21	247	1071
NOV	79	20	235	1018
NRG	75	19	223	966
NVDA	72	19	214	927
OI	84	22	250	1083
OLN	84	22	250	1083
PARA	49	13	145	628
PG	86	22	256	1110

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
PM	44	12	130	562
POOL	81	21	241	1044
PTEN	84	22	250	1083
QRTEA	46	12	136	588
RSG	76	19	226	979
SCHL	83	21	247	1071
SHW	85	22	253	1097
SLGN	79	20	235	1018
SSD	85	22	253	1097
STLD	78	20	232	1005
TEX	84	22	250	1083
TGNA	85	22	253	1097
TTC	81	21	241	1045
URBN	84	22	250	1084
UVV	55	14	163	705
VMC	85	22	253	1097
VRSN	75	19	223	966
VZ	63	16	187	810
WBD	52	13	154	667
WLY	68	18	202	875
WM	79	20	235	1018
WMT	84	22	250	1084

Ticker	Train set size	Test set size	Monthly Interpolated	Weekly Interpolated
X	84	22	250	1083
XRAY	84	22	250	1083

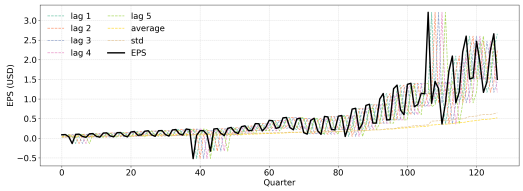
Table B-3: *Hyperparameters for ML Estimators*

Estimator	Values Range
Automatic Relevance Determination	
threshold_lambda	50, 100, 500, 1000, 1500
lambda_1	0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9
Bayesian Ridge	
alpha_1	0, 2, 4, 6, 10
alpha_2	0,1
lambda_1	0,1,2,3
lambda_2	0,1,2,3
alpha_init	1,2,3
lambda_init	1,2
Decision Tree	
min_samples_split	2,4,6
max_depth	4,6,8,10,12
min_samples_leaf	3,5,7,9,11

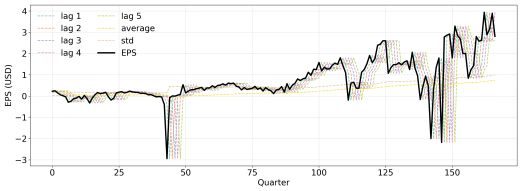
Estimator	Values Range
max_features	1.0, ‘log2’, ‘sqrt’
max_leaf_nodes	4,6,8,10,12
Huber Regressor	
alpha	0.05, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5
fit_intercept	true, false
K-Nearest Neighbours	
algorithm	‘kd_tree’, ‘ball_tree’, ‘brute’
p	2,4,6,8,10
n_neighbors	5, 7, 9, 12, 14
weights	‘distance’
Lasso	
alpha	0.05, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5
selection	‘cyclic’, ‘random’
fit_intercept	true, false
positive	true, false
Multi-Layer Perceptron	
hidden_layer_sizes	(4, 1), (6, 1), (8, 1), (16, 1), (2, 4, 1), (4, 6, 1), (6, 8, 1)
batch_sizes	4,8,16
Random Forest	
n_estimators	2,4,6,8
max_features	‘log2’, ‘sqrt’, 1.0

Estimator	**Values Range**
max_depth	2,4,6
min_samples_split	8, 10, 12, 14
min_samples_leaf	4,6,8,10
bootstrap	true, false

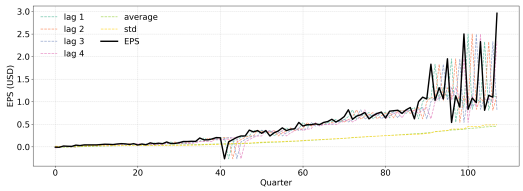
Figure B-1: *EPS: Train, validation and test subset plots of the target variable (Earnings Per Share) with its features for the 8 representative companies*



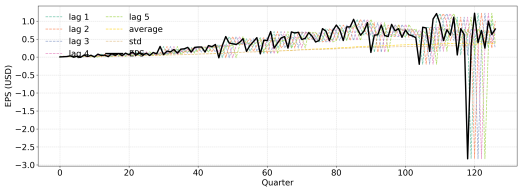
(a) EPS: SHW data set



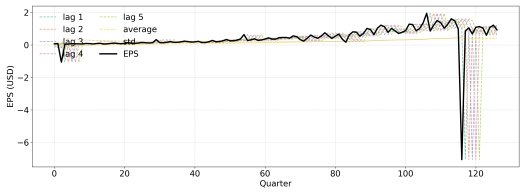
(b) EPS: CAT data set



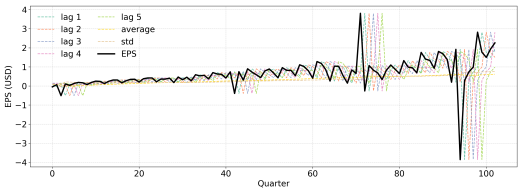
(c) EPS: ANSS data set



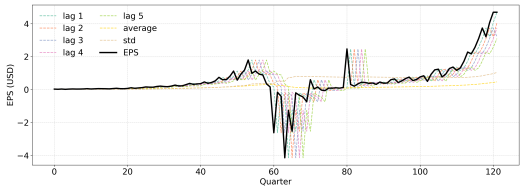
(d) EPS: WLY data set



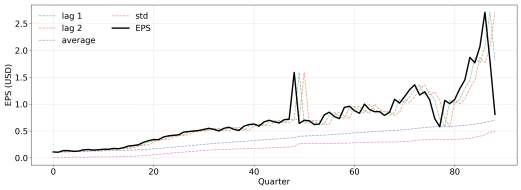
(e) EPS: ECL data set



(f) EPS: DRI data set

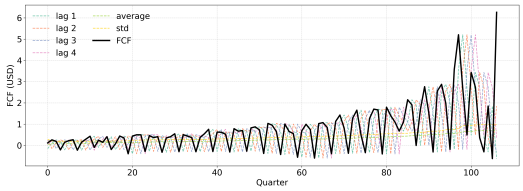


(g) EPS: DHI data set

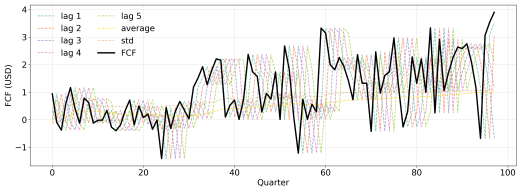


(h) EPS: CHRW data set

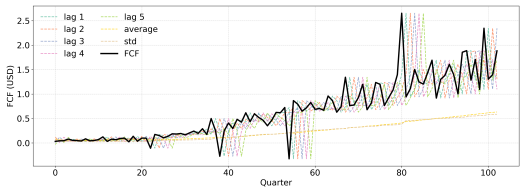
Figure B-2: *FCF: Train, validation and test subset plots of the target variable (Free Cash Flows) with its features for 8 representative companies*



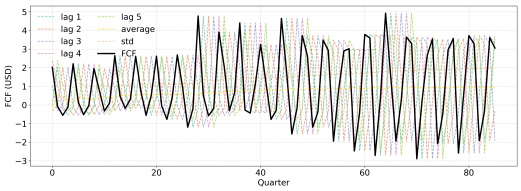
(a) FCF: SHW data set



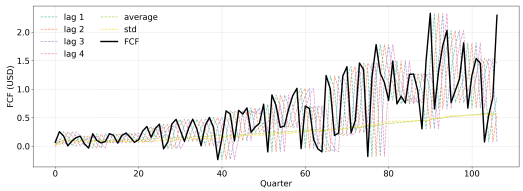
(b) FCF: CAT data set



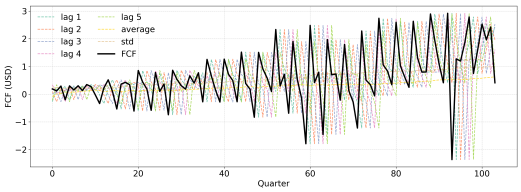
(c) FCF: ANSS data set



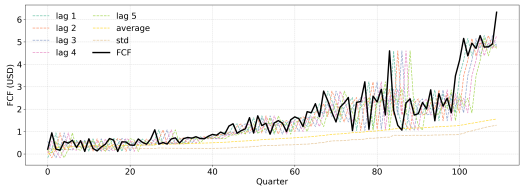
(d) FCF: WLY data set



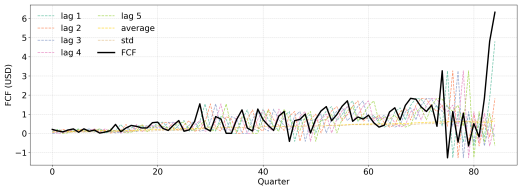
(e) FCF: ECL data set



(f) FCF: DRI data set

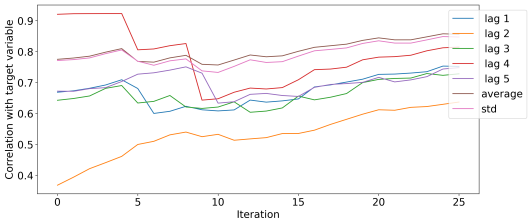


(g) FCF: DHI data set

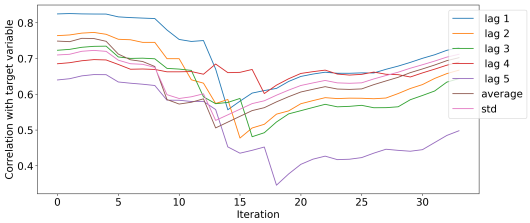


(h) FCF: CHRW data set

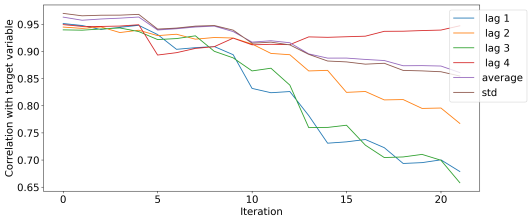
Figure B-3: EPS data sets: correlations of target variable with its features, through iterations in the test set for 8 representative companies



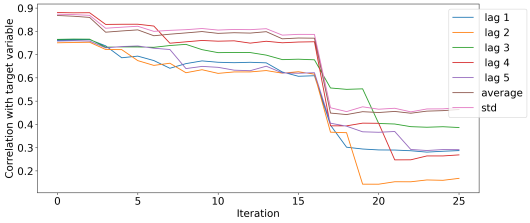
(a) EPS: SHW data set



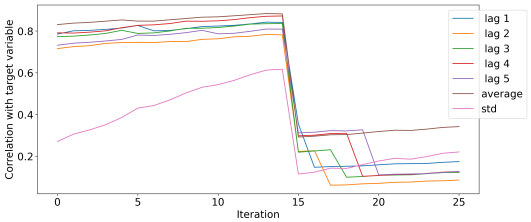
(b) EPS: CAT data set



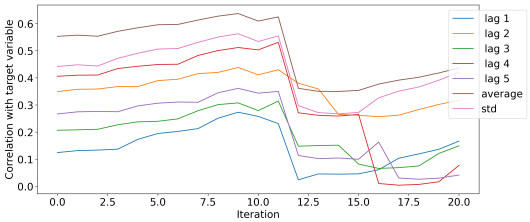
(c) EPS: ANSS data set



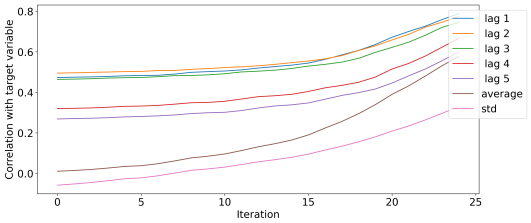
(d) EPS: WLY data set



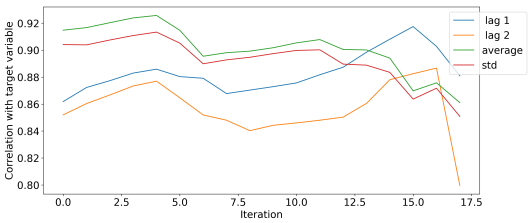
(e) EPS: ECL data set



(f) EPS: DRI data set

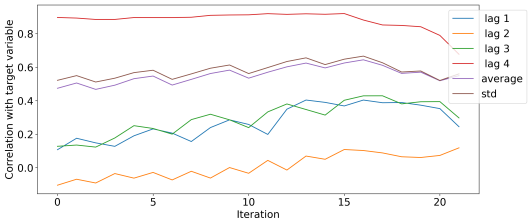


(g) EPS: DHI data set

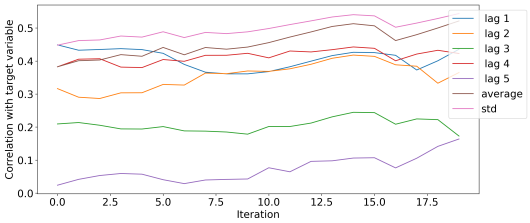


(h) EPS: CHRW data set

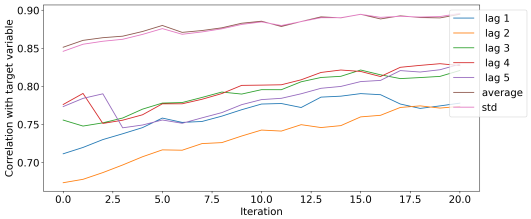
Figure B-4: *FCF data sets: correlations of target variable with its features, through iterations in the test set for 8 representative companies*



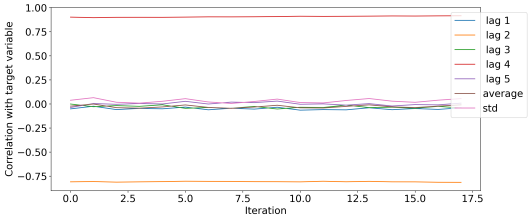
(a) FCF: autocorrelations SHW data set



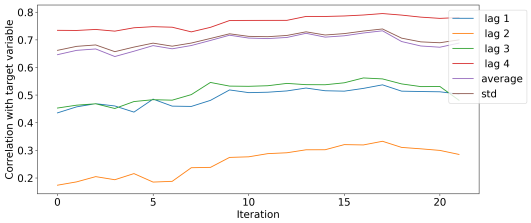
(b) FCF: CAT data set



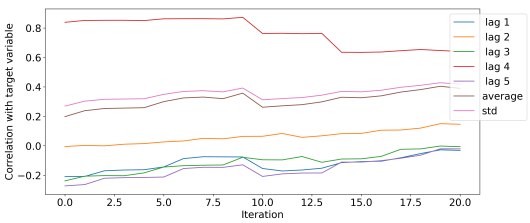
(c) FCF: ANSS data set



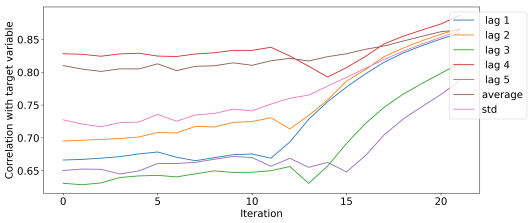
(d) FCF: WLY data set



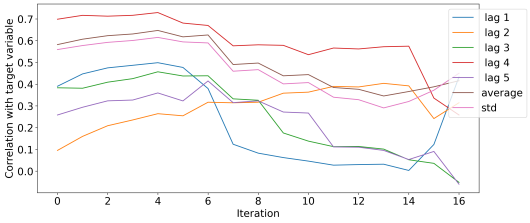
(e) FCF: ECL data set



(f) FCF: DRI data set

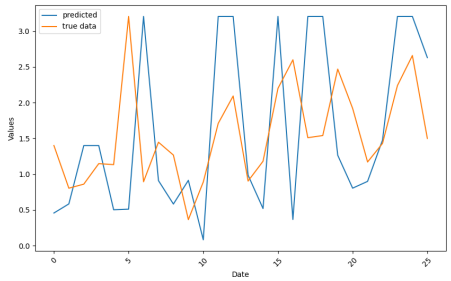


(g) FCF: DHI data set

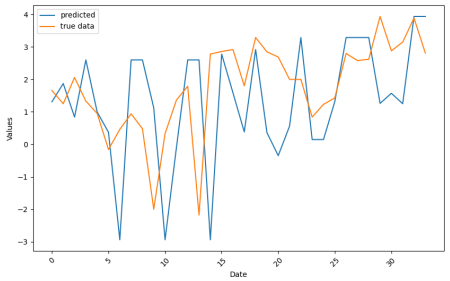


(h) FCF: CHRW data set

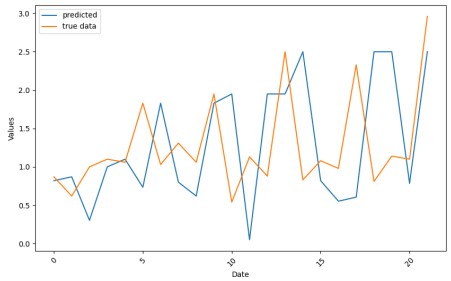
Figure B-5: *ARIMA: EPS predictions for 8 representative companies*



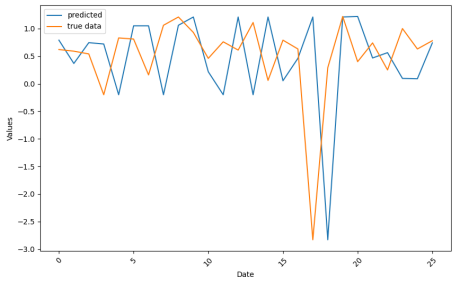
(a) EPS: SHW predictions



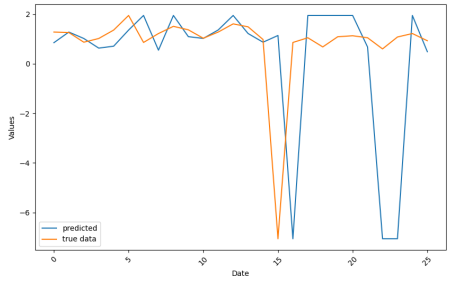
(b) EPS: CAT predictions



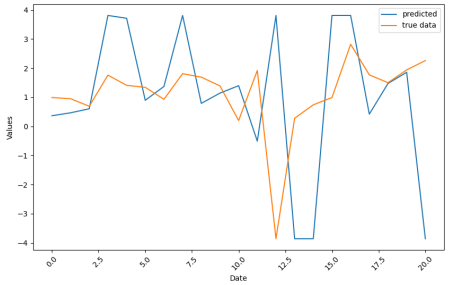
(c) EPS: ANSS predictions



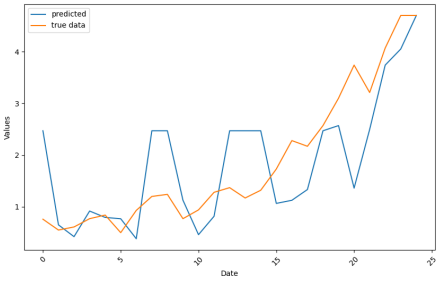
(d) EPS: WLY predictions



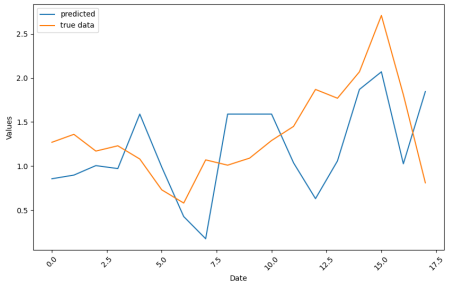
(e) EPS: ECL predictions



(f) EPS: DRI predictions

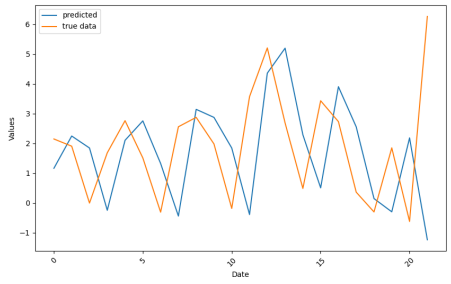


(g) EPS: DHI predictions

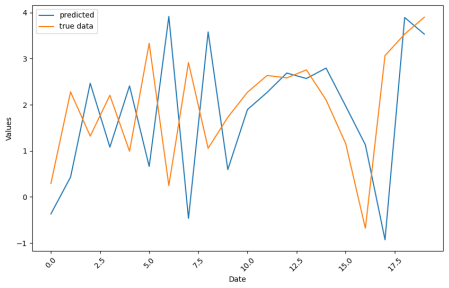


(h) EPS: CHRW predictions

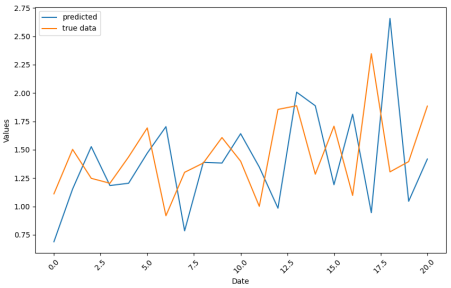
Figure B-6: *ARIMA: FCF predictions for 8 representative companies*



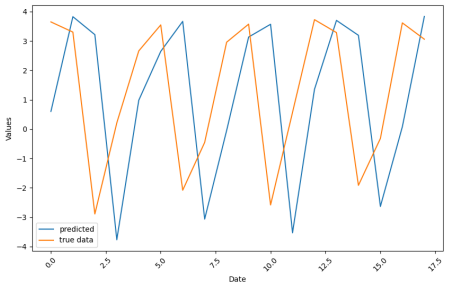
(a) FCF: SHW predictions



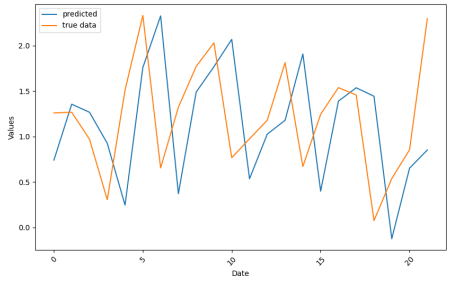
(b) FCF: CAT predictions



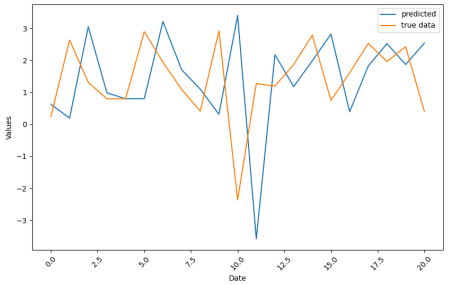
(c) FCF: ANSS predictions



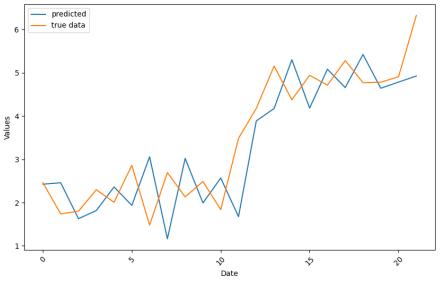
(d) FCF: WLY predictions



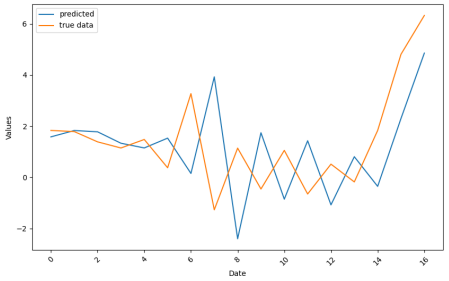
(e) FCF: ECL predictions



(f) FCF: DRI predictions

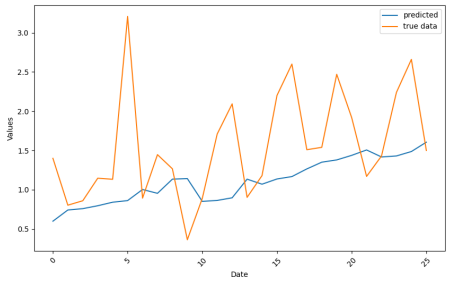


(g) FCF: DHI predictions

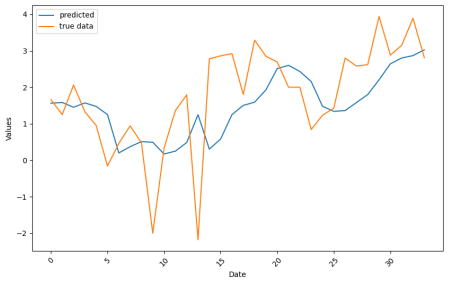


(h) FCF: CHRW predictions

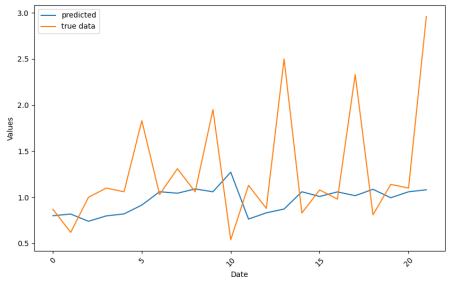
Figure B-7: *SES: EPS predictions for 8 representative companies*



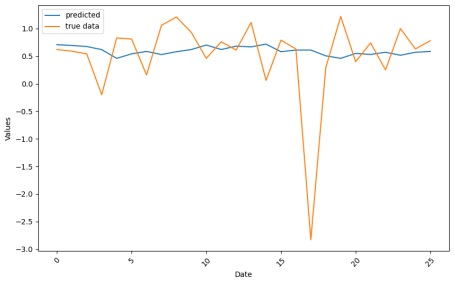
(a) EPS: SHW predictions



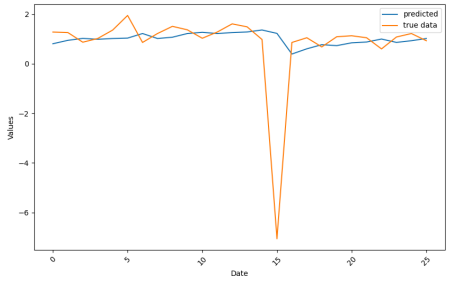
(b) EPS: CAT predictions



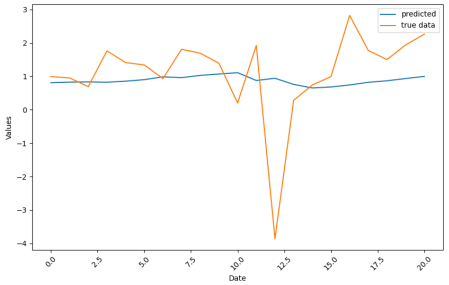
(c) EPS: ANSS predictions



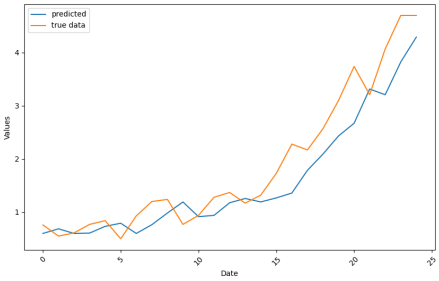
(d) EPS: WLY predictions



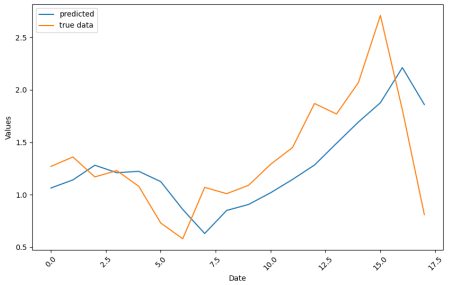
(e) EPS: ECL predictions



(f) EPS: DRI predictions

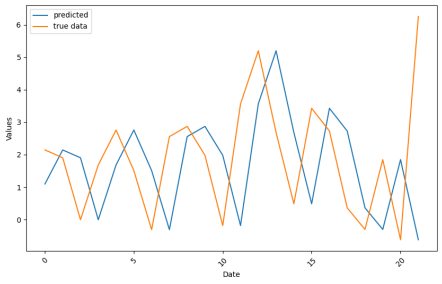


(g) EPS: DHI predictions

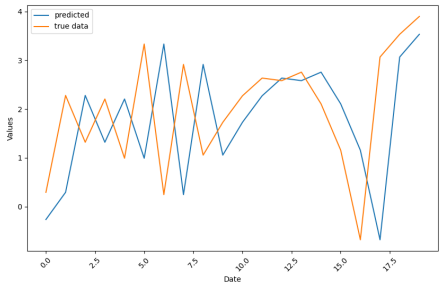


(h) EPS: CHRW predictions

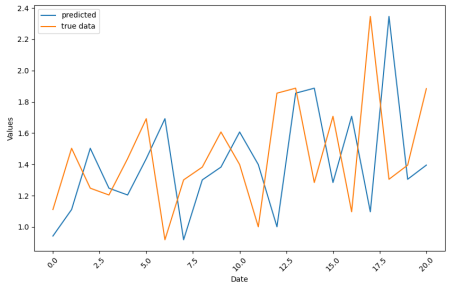
Figure B-8: *SES: FCF predictions for 8 representative companies*



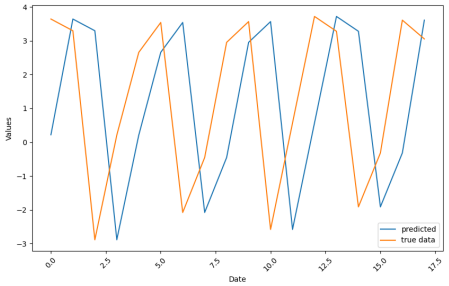
(a) FCF: SHW predictions



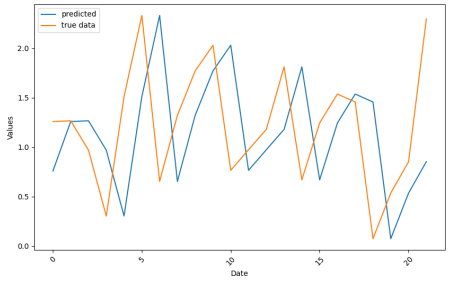
(b) FCF: CAT predictions



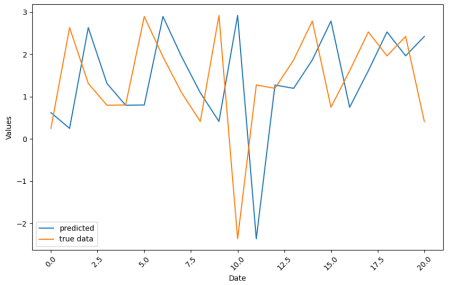
(c) FCF: ANSS predictions



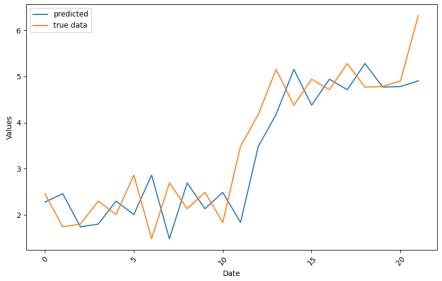
(d) FCF: WLY predictions



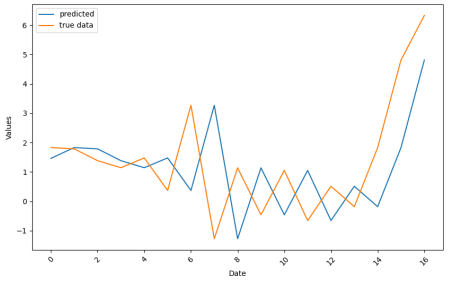
(e) FCF: ECL predictions



(f) FCF: DRI predictions

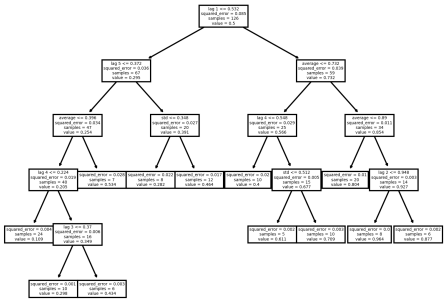


(g) FCF: DHI predictions

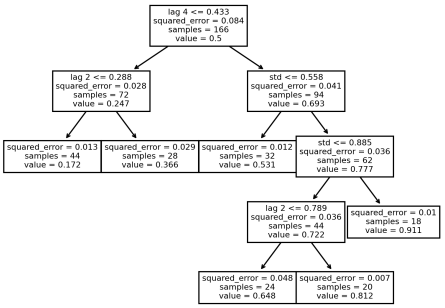


(h) FCF: CHRW predictions

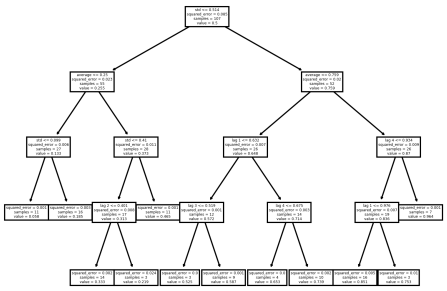
Figure B-9: *Decision Tree: EPS best-fit tree structures for 8 representative companies*



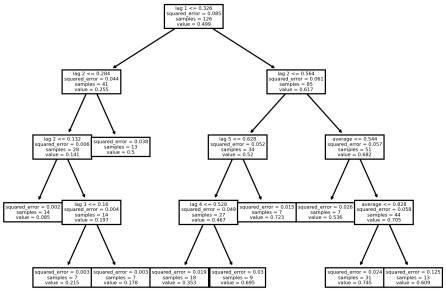
(a) EPS: SHW tree



(b) EPS: CAT tree

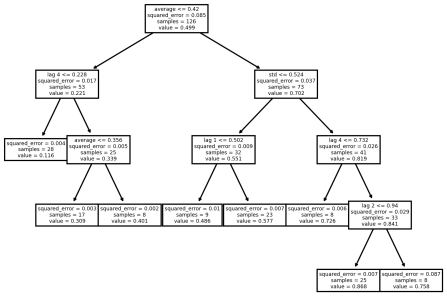


(c) EPS: ANSS tree

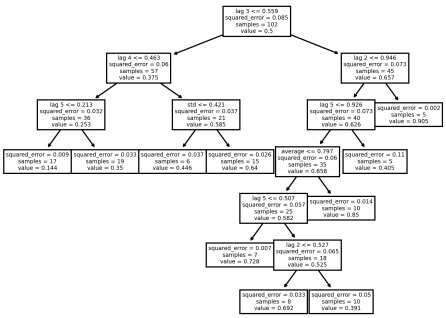


(d) EPS: WLY tree

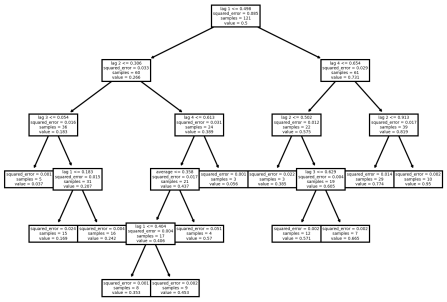
Figure B-10: *Decision Tree: EPS best-fit tree structures 8 representative companies, continued*



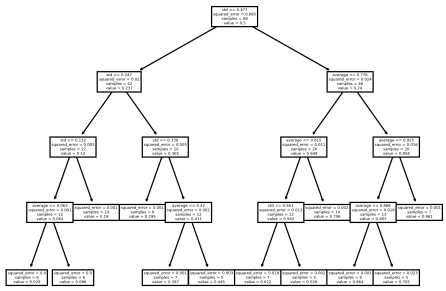
(a) EPS: ECL tree



(b) EPS: DRI tree

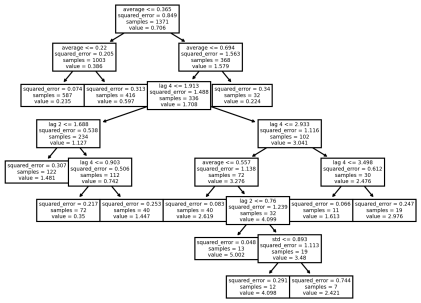


(c) EPS: DHI tree

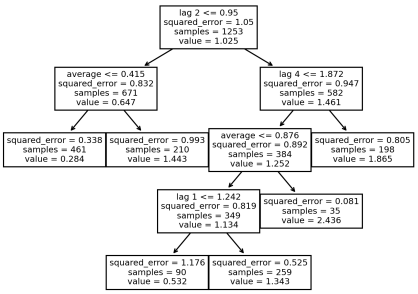


(d) EPS: CHRW tree

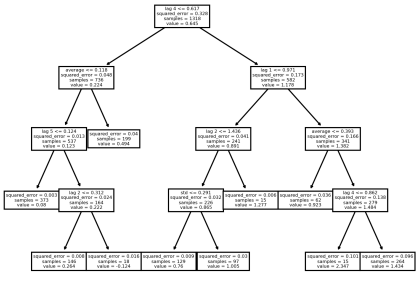
Figure B-11: *Decision Tree: FCF best-fit tree structures for 8 representative companies*



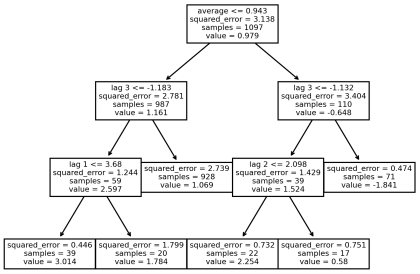
(a) FCF: SHW tree



(b) FCF: CAT tree

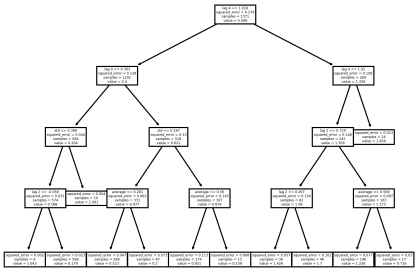


(c) FCF: ANSS tree

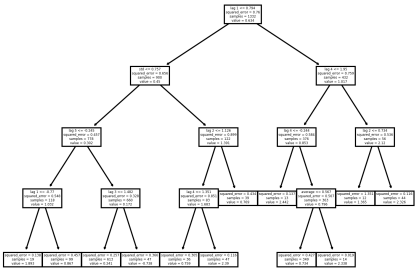


(d) FCF: WLY tree

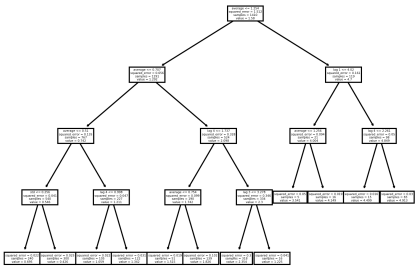
Figure B-12: *Decision Tree: FCF best-fit tree structures for 8 representative companies, continued*



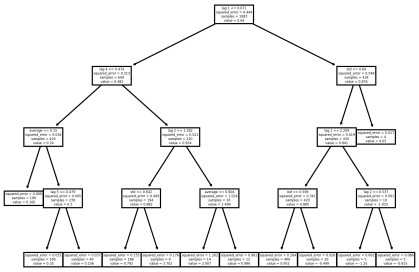
(a) FCF: ECL tree



(b) FCF: DRI tree

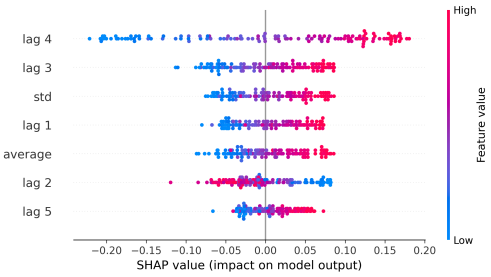


(c) FCF: DHI tree

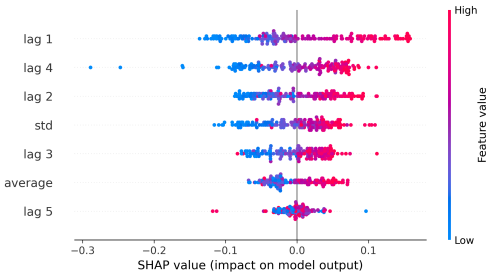


(d) FCF: CHRW tree

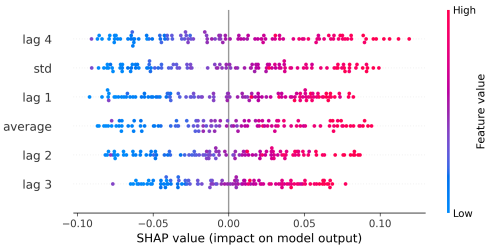
Figure B-13: *Shapley Values: KNN-EPS predictions for 8 representative companies*



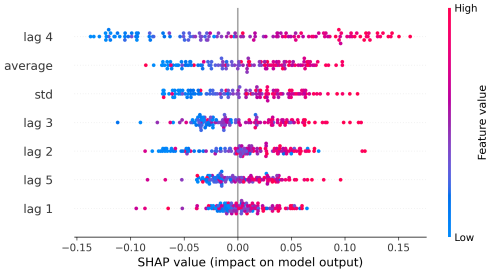
(a) EPS: SHW shapley values



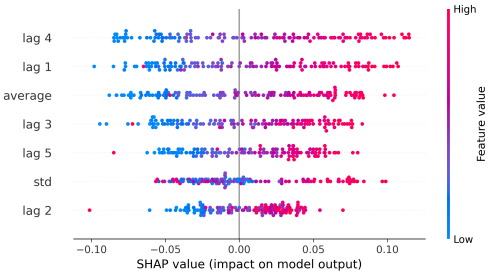
(b) EPS: CAT shapley values



(c) EPS: ANSS shapley values

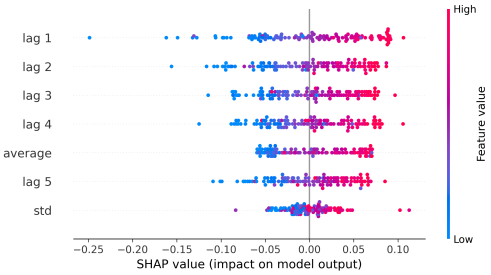


(d) EPS: WLY shapley values

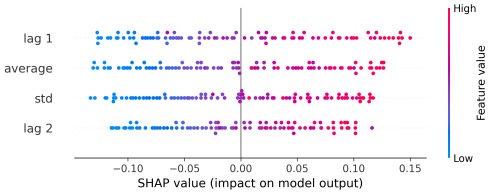


(e) EPS: ECL shapley values

(f) EPS: DRI shapley values

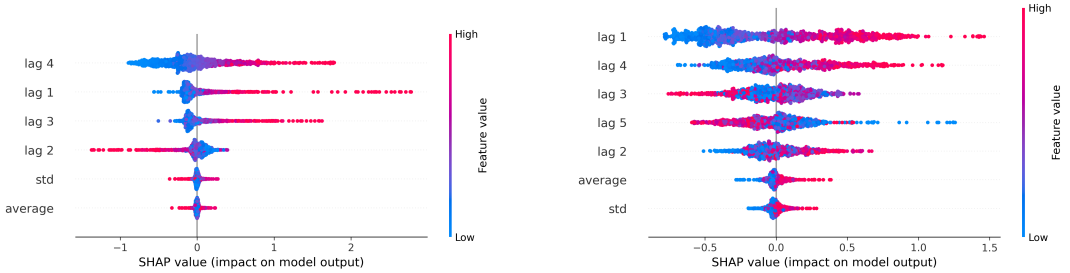


(g) EPS: DHI shapley values



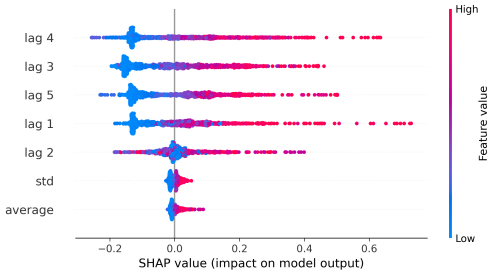
(h) EPS: CHRW shapley values

Figure B-14: *Shapley Values: KNN-FCF predictions for 8 representative companies*

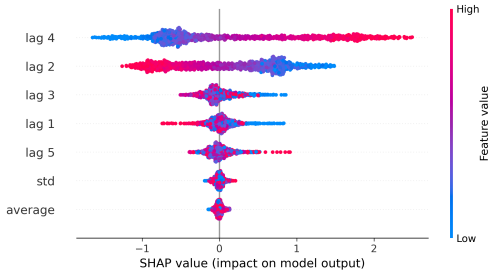


(a) FCF: SHW shapley values

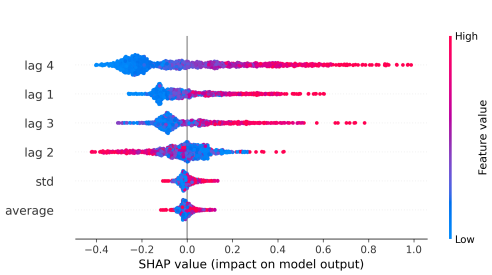
(b) FCF: CAT shapley values



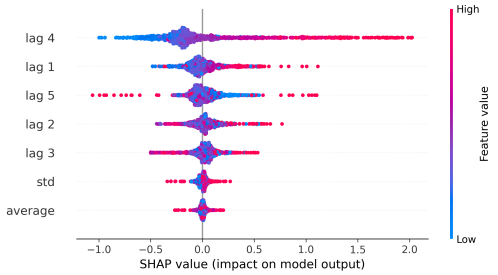
(c) FCF: ANSS shapley values



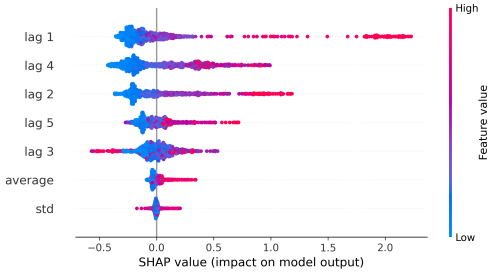
(d) FCF: WLY shapley values



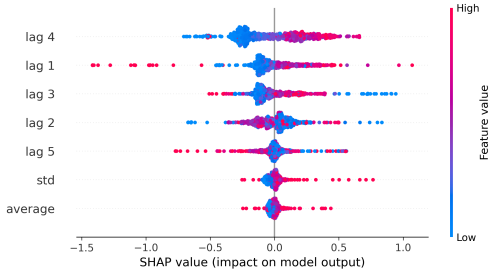
(e) FCF: ECL shapley values



(f) FCF: DRI shapley values

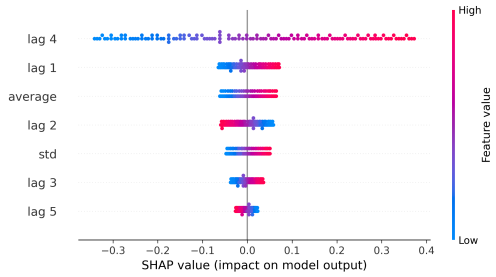


(g) FCF: DHI shapley values

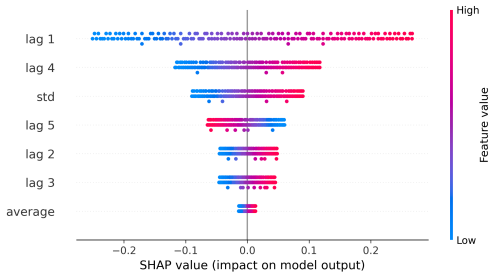


(h) FCF: CHRW shapley values

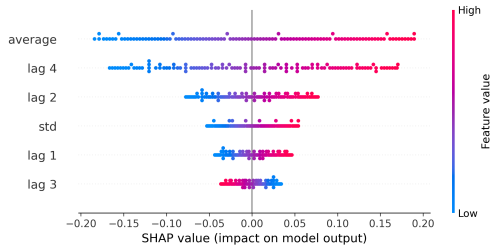
Figure B-15: *Shapley Values: HR-EPS predictions for 8 representative companies*



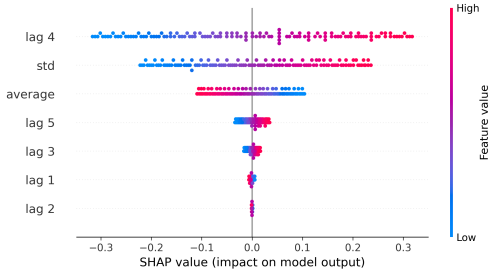
(a) EPS: SHW shapley values



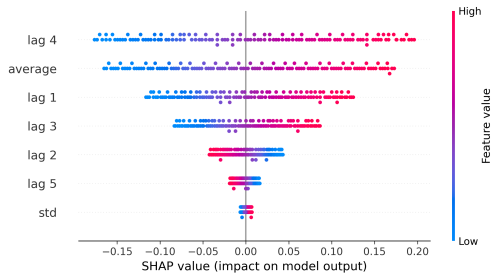
(b) EPS: CAT shapley values



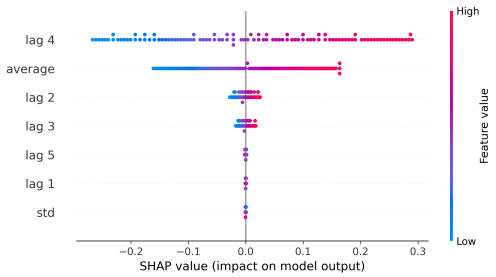
(c) EPS: ANSS shapley values



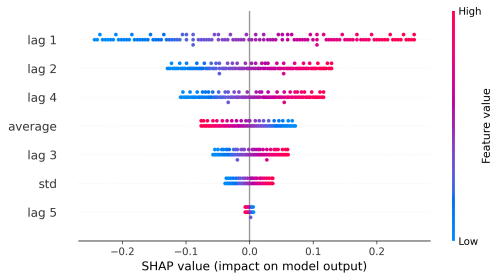
(d) EPS: WLY shapley values



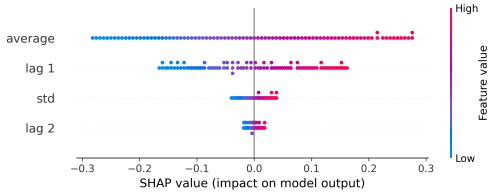
(e) EPS: ECL shapley values



(f) EPS: DRI shapley values

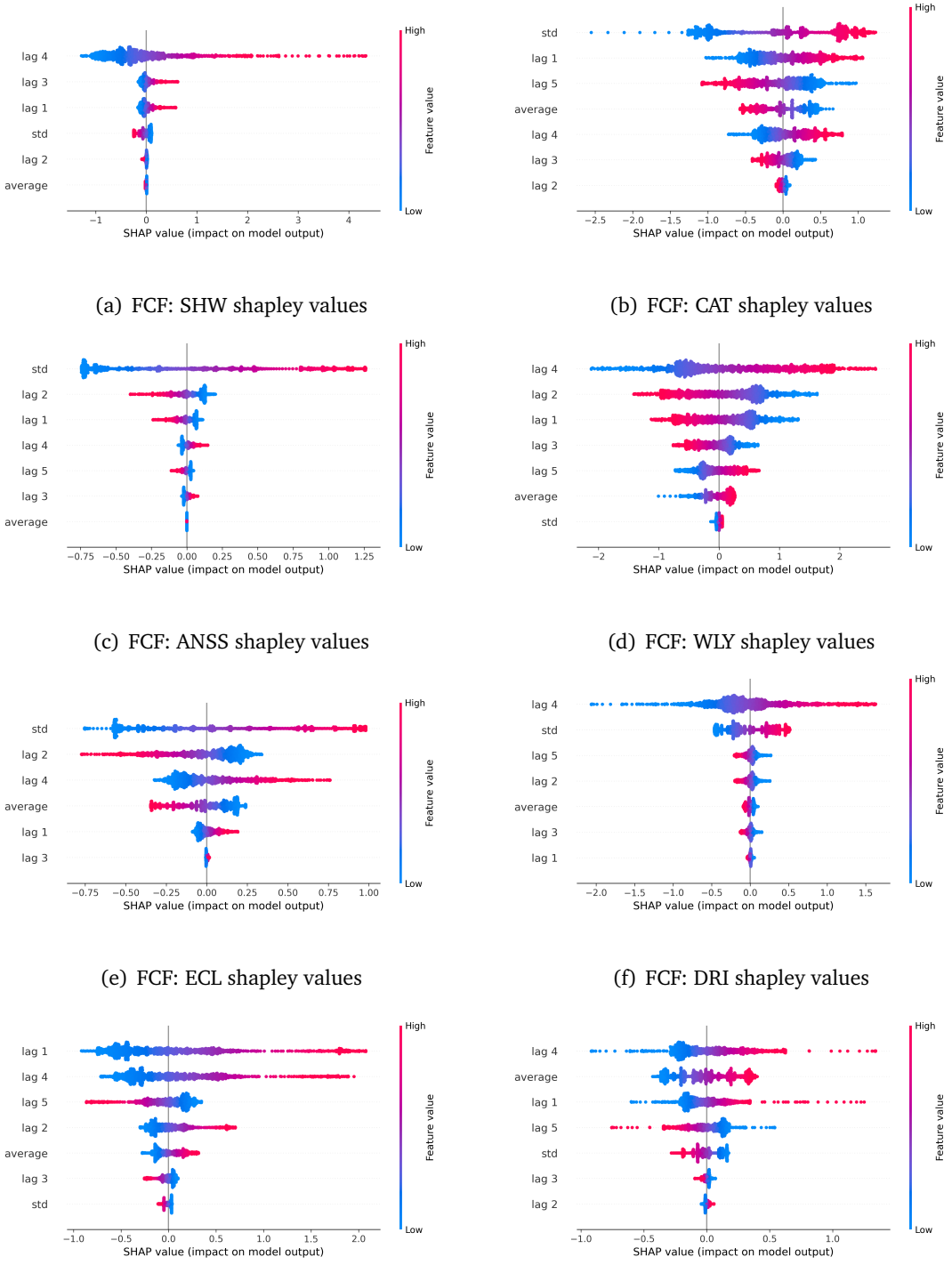


(g) EPS: DHI shapley values



(h) EPS: CHRW shapley values

Figure B-16: *Shapley Values: HR-FCF predictions for 8 representative companies*



(a) FCF: SHW shapley values

(b) FCF: CAT shapley values

(c) FCF: ANSS shapley values

(d) FCF: WLY shapley values

(e) FCF: ECL shapley values

(f) FCF: DRI shapley values

(g) FCF: DHI shapley values

(h) FCF: CHRW shapley values

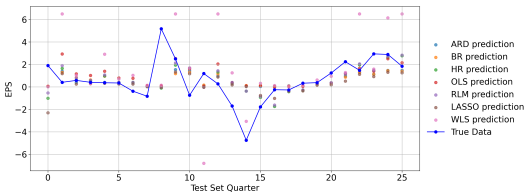
Appendix C: Chapter 6

In Chapter 6, we introduce a transfer learning methodology that enables us to treat a prediction for a data point as a weighted average of outputs from statistical and machine learning estimators. We then analyze the performance of this approach by comparing it to predictions made using individual estimators.

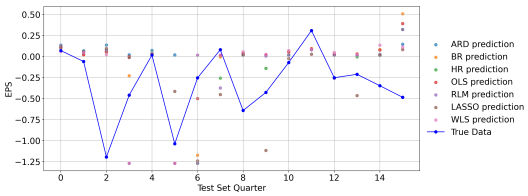
For the analysis, we present results from two data sets in detail, with results from eight additional data sets included in this chapter. These additional data sets correspond to the following companies: Devon Energy Corporation (DVN), GameStop Corp. (GME), NRG Energy Inc. (NRG), O-I Glass Inc. (OI), The Toro Company (TTC), AutoZone Inc. (AZO), Johnson & Johnson (JNJ), and Steel Dynamics Inc. (STLD).

Figures C-1 and C-2 show the predictions for the EPS and FCF series, respectively, made using a collection of single estimators. Subsequently, Figures C-3 and C-4 display the corresponding predictions generated using our transfer learning methodology.

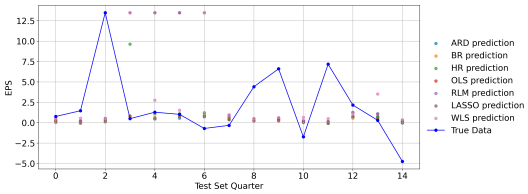
Figure C-1: *EPS: Single Estimators Predictions For 8 Random Data Sets*



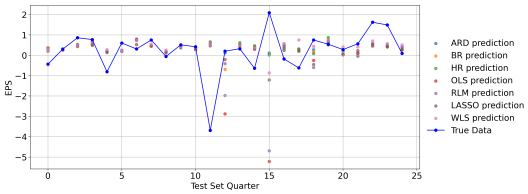
(a) DVN data set predictions



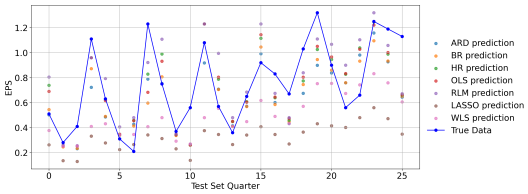
(b) GME data set predictions



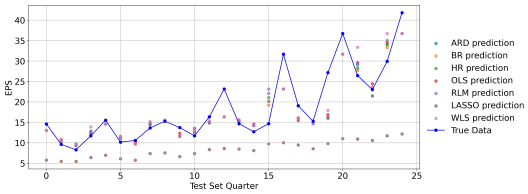
(c) NRG data set predictions



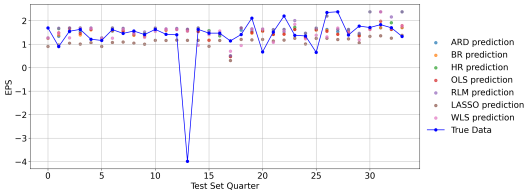
(d) OI data set predictions



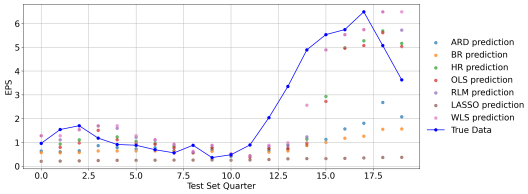
(e) TTC data set predictions



(f) AZO data set predictions

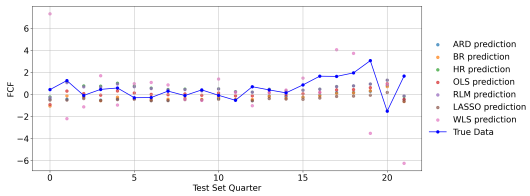


(g) JNJ data set predictions

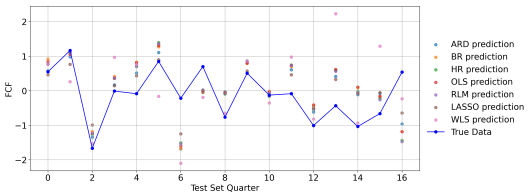


(h) STLD data set predictions

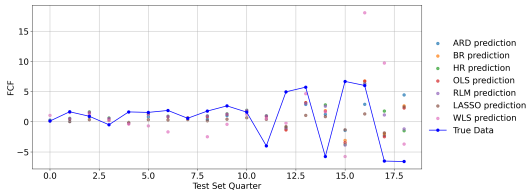
Figure C-2: *FCF: Single Estimators Predictions For 8 Random Data Sets*



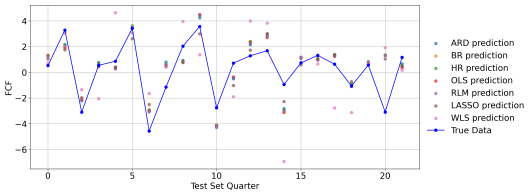
(a) DVN data set predictions



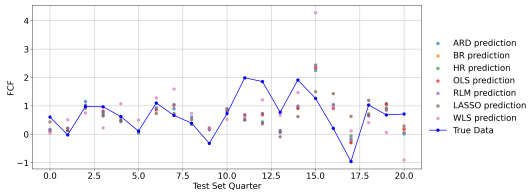
(b) GME data set predictions



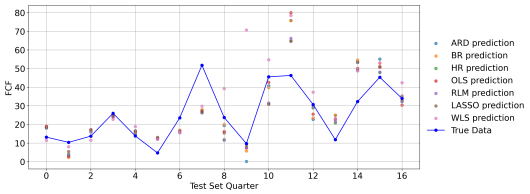
(c) NRG data set predictions



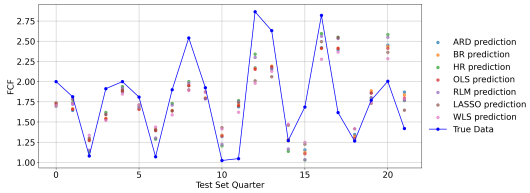
(d) OI data set predictions



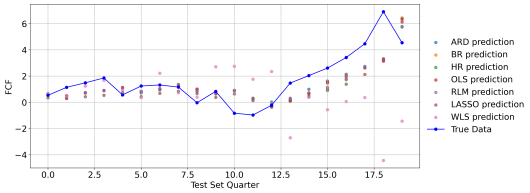
(e) TTC data set predictions



(f) AZO data set predictions

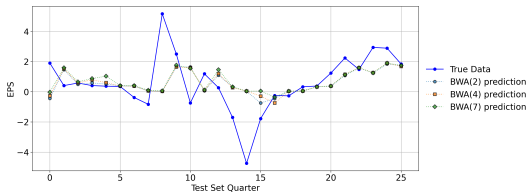


(g) JNJ data set predictions

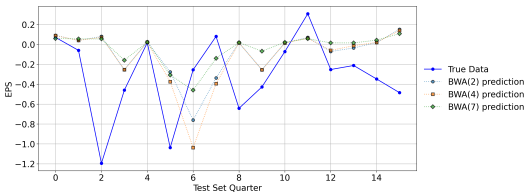


(h) STLD data set predictions

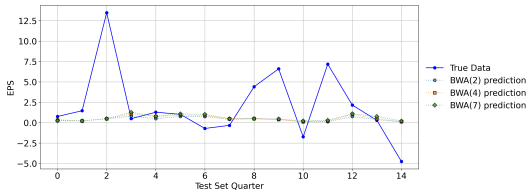
Figure C-3: *EPS: BWA(n) Predictions For 8 Random Data Sets*



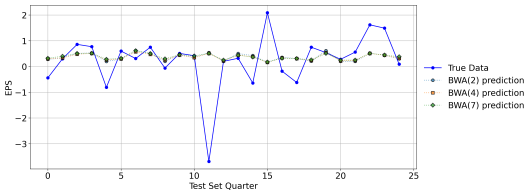
(a) DVN data set predictions



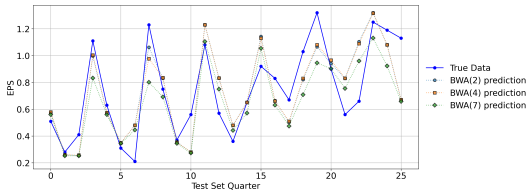
(b) GME data set predictions



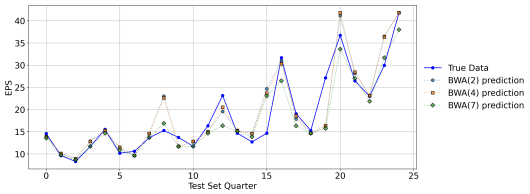
(c) NRG data set predictions



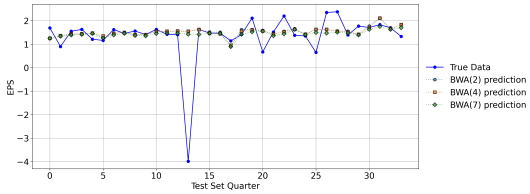
(d) OI data set predictions



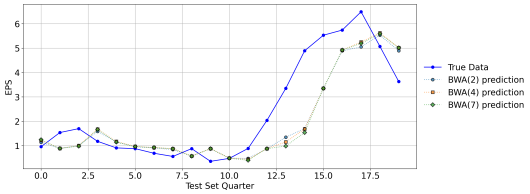
(e) TTC data set predictions



(f) AZO data set predictions

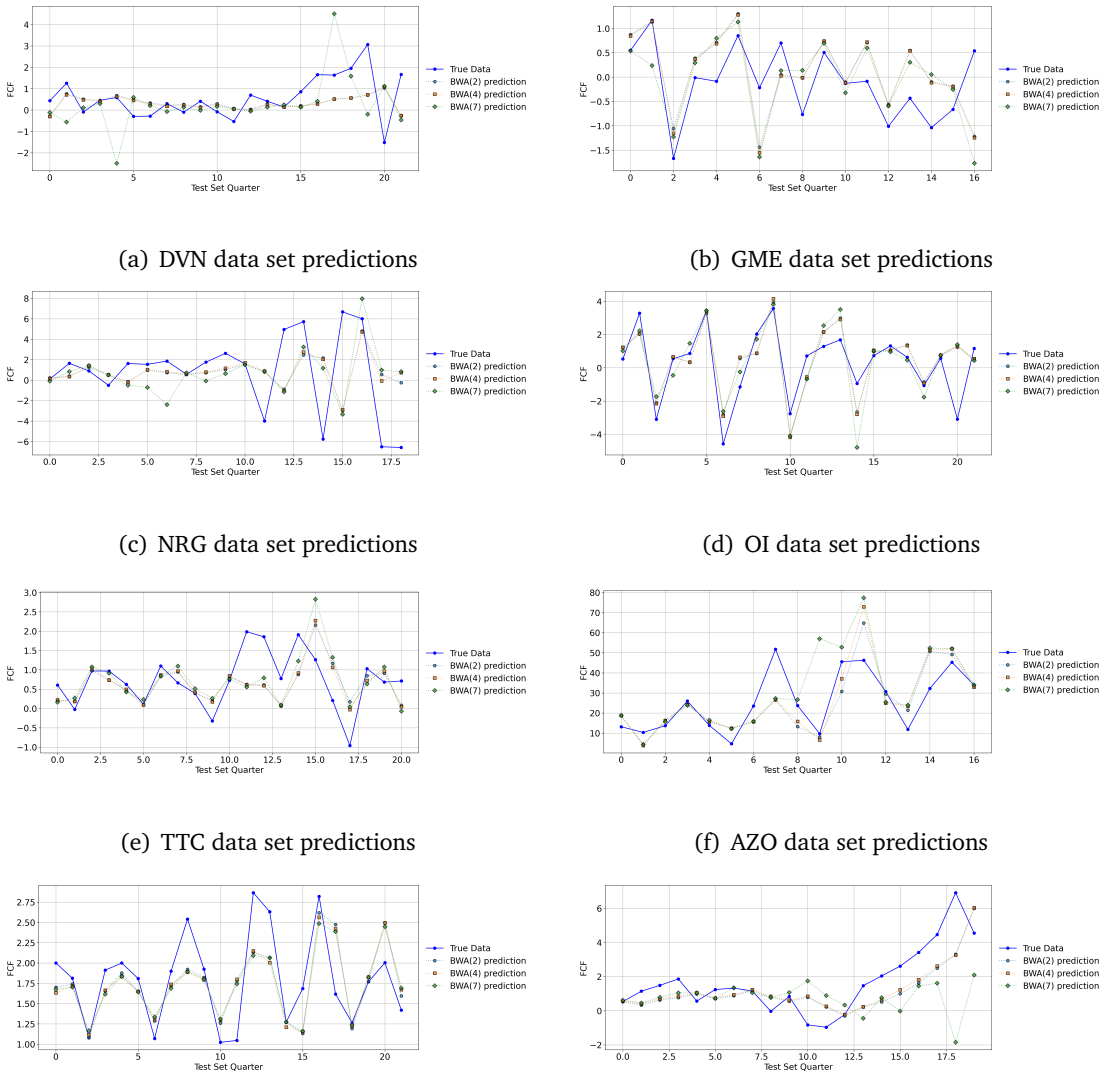


(g) JNJ data set predictions



(h) STLD data set predictions

Figure C-4: *FCF: BWA(n) Predictions For 8 Random Data Sets*



(a) DVN data set predictions

(b) GME data set predictions

(c) NRG data set predictions

(d) OI data set predictions

(e) TTC data set predictions

(f) AZO data set predictions

(g) JNJ data set predictions

(h) STLD data set predictions

Appendix D: Chapter 7

In Chapter 7, we demonstrate practical applications of our forecasting approach through out-of-sample backtesting. To reduce bias in our analysis, we extend the timeline for each data set. Each data set is divided into four consecutive periods: training, validation, testing, and

simulation. Tables D-1 and D-2 summarize the time periods corresponding to each phase for the EPS and FCF data sets, respectively.

Table D-1: *EPS: Timelines of data sets used for training, validation, regression, and simulation testing*

Data set	**Training period start**	**Validation period start**	**Regression testing period start**	**Simulation period start**	**Simulation period end**
ALB	1994-06-30	2015-12-31	2017-06-30	2022-12-31	2024-06-30
AMAT	1991-04-28	2015-01-25	2016-07-31	2022-10-30	2024-07-28
AMD	1991-06-30	2015-03-28	2016-09-24	2022-12-31	2024-06-29
AMZN	1998-06-30	2016-12-31	2018-03-31	2022-12-31	2024-06-30
ANSS	1996-06-30	2016-06-30	2017-09-30	2022-12-31	2024-06-30
APD	1991-03-31	2014-12-31	2016-06-30	2022-09-30	2024-06-30
APH	1992-06-30	2015-06-30	2016-12-31	2022-12-31	2024-06-30
ATI	1997-12-31	2016-09-30	2017-12-31	2022-12-31	2024-06-30
AZO	1991-08-31	2014-11-22	2016-08-27	2022-08-27	2024-05-27
BA	1981-06-30	2012-09-30	2014-09-30	2022-12-31	2024-06-30
BALL	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-27
BBWI	1990-08-04	2014-05-03	2015-10-31	2022-01-29	2024-02-03
BBY	1990-09-01	2014-05-03	2015-10-31	2022-01-29	2024-01-27
BWA	1995-09-30	2016-03-31	2017-09-30	2022-12-31	2024-06-30
CAH	1995-12-31	2015-12-31	2017-03-31	2022-06-30	2024-06-30
CAT	1981-06-30	2012-09-30	2014-09-30	2022-12-31	2024-06-30

Table D-1 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
CE	2006-06-30	2018-12-31	2019-09-30	2022-12-31	2024-06-30
CF	2006-12-31	2018-12-31	2019-12-31	2022-12-31	2024-06-30
CHRW	2001-09-30	2017-09-30	2018-09-30	2022-12-31	2024-06-28
CL	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
CLX	1990-12-31	2014-09-30	2016-03-31	2022-06-30	2024-06-30
CMCSA	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
COO	1991-04-30	2015-01-31	2016-07-31	2022-10-31	2024-07-31
COST	1995-02-12	2015-11-22	2017-02-12	2022-08-28	2024-05-12
CSCO	1991-01-27	2014-10-25	2016-04-30	2022-07-30	2024-07-27
CTXS	1997-03-31	2016-03-31	2017-06-30	2022-06-30	2023-09-30
CVS	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
CVX	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30
DE	1991-04-30	2015-01-31	2016-07-31	2022-10-30	2024-07-28
DHI	1992-06-30	2015-03-31	2016-09-30	2022-09-30	2024-06-30
DISH	1995-06-30	2016-03-31	2017-06-30	2022-12-31	2024-06-30
DOV	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
DRI	1996-11-24	2016-02-28	2017-05-28	2022-05-29	2024-05-27
DVN	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30
EA	1991-03-31	2014-06-30	2016-03-31	2022-03-31	2024-03-31

Table D-1 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
EBAY	1999-12-31	2017-03-31	2018-06-30	2022-12-31	2024-06-29
ECL	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
EIX	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
EMN	1994-06-30	2015-12-31	2017-06-30	2022-12-31	2024-06-30
EMR	1991-03-31	2014-12-31	2016-06-30	2022-09-30	2024-06-30
FAST	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
FL	1990-07-28	2014-05-03	2015-10-31	2022-01-29	2024-08-03
GLW	1991-06-16	2015-03-31	2016-09-30	2022-12-31	2024-06-30
GME	2002-08-03	2017-04-29	2018-05-05	2022-01-29	2024-05-04
GOOG	2005-03-31	2018-09-30	2019-06-30	2022-12-31	2024-06-30
GPC	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
GVA	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30
GWW	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
HAL	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
HD	1990-07-29	2014-05-04	2015-11-01	2022-01-30	2024-07-28
HP	1991-03-31	2014-12-31	2016-06-30	2022-09-30	2024-06-30
IBM	1981-06-30	2012-09-30	2014-09-30	2022-12-31	2024-06-30
IFF	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
INTC	1987-06-27	2014-03-29	2015-12-26	2022-12-31	2024-06-29

Table D-1 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
INTU	1994-01-31	2015-07-31	2017-01-31	2022-07-31	2024-07-31
JNJ	1981-06-30	2012-09-30	2014-09-28	2023-01-01	2024-06-30
KMB	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
KO	1981-06-30	2012-09-28	2014-09-26	2022-12-31	2024-06-28
KSS	1993-01-31	2014-11-01	2016-04-30	2022-01-29	2024-08-03
M	1993-07-31	2015-01-31	2016-07-30	2022-01-29	2024-08-03
MAS	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
MDLZ	2001-06-30	2017-09-30	2018-09-30	2022-12-31	2024-06-30
MLM	1995-06-30	2016-03-31	2017-06-30	2022-12-31	2024-06-30
MMM	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30
MO	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
MRK	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
MSFT	1987-12-31	2013-12-31	2015-09-30	2022-06-30	2024-06-30
NFLX	2003-09-30	2018-03-31	2019-03-31	2022-12-31	2024-06-30
NKE	1990-11-30	2014-08-31	2016-02-29	2022-05-31	2024-05-31
NOV	1998-06-30	2016-12-31	2018-03-31	2022-12-31	2024-06-30
NRG	2005-06-30	2018-09-30	2019-06-30	2022-12-31	2024-06-30
NVDA	2000-04-30	2016-10-30	2017-10-29	2022-01-30	2024-07-28
OI	1992-06-30	2015-06-30	2016-12-31	2022-12-31	2024-06-30

Table D-1 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
OLN	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30
PARA	2007-03-31	2019-03-31	2019-12-31	2022-12-31	2024-06-30
PG	1990-12-31	2014-09-30	2016-03-31	2022-06-30	2024-06-30
PM	2008-06-30	2019-06-30	2020-03-31	2022-12-31	2024-06-30
POOL	1997-06-30	2016-09-30	2017-12-31	2022-12-31	2024-06-30
PTEN	1994-06-30	2015-12-31	2017-06-30	2022-12-31	2024-06-30
QRTEA	2007-09-30	2019-03-31	2019-12-31	2022-12-31	2024-06-30
RSG	1999-06-30	2017-03-31	2018-06-30	2022-12-31	2024-06-30
SCHL	1992-05-30	2014-11-30	2016-05-31	2022-05-31	2024-05-31
SHW	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
SLGN	1997-06-30	2016-09-30	2017-12-31	2022-12-31	2024-06-30
SSD	1994-06-30	2015-12-31	2017-06-30	2022-12-31	2024-06-30
STLD	1998-03-31	2016-12-31	2018-03-31	2022-12-31	2024-06-30
TEX	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
TGNA	1991-06-30	2015-03-29	2016-09-30	2022-12-31	2024-06-30
TTC	1991-02-01	2015-01-30	2016-07-29	2022-10-31	2024-05-03
URBN	1994-07-31	2015-04-30	2016-07-31	2022-01-31	2024-07-31
UVV	1990-06-30	2014-06-30	2015-12-31	2022-03-31	2024-06-30
VMC	1991-03-31	2015-03-31	2016-09-30	2022-12-31	2024-06-30

Table D-1 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
VRSN	1999-06-30	2017-03-31	2018-06-30	2022-12-31	2024-06-30
VZ	1991-06-30	2015-03-31	2016-09-30	2022-12-31	2024-06-30
WBD	2006-09-30	2018-12-31	2019-09-30	2022-12-31	2024-06-30
WLY	1990-10-31	2014-07-31	2016-01-31	2022-04-30	2024-04-30
WM	1998-06-30	2016-12-31	2018-03-31	2022-12-31	2024-06-30
WMT	1990-07-31	2014-04-30	2015-10-31	2022-04-29	2024-29-07
X	1992-06-30	2015-06-30	2016-12-31	2022-12-31	2024-06-30
XRAY	1994-06-30	2015-12-31	2017-06-30	2022-12-31	2024-06-30

Table D-2: *FCF: Timelines of data sets used for training, validation, regression, and simulation testing*

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
ALB	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
AMAT	1996-07-28	2012-04-29	2017-10-29	2023-01-29	2024-07-28
AMD	1996-09-29	2012-06-30	2017-09-30	2022-12-31	2024-06-29
AMZN	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
ANSS	1997-06-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30

Table D-2 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
APD	2000-06-30	2013-12-31	2018-06-30	2022-12-31	2024-06-30
APH	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
ATI	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
AZO	2002-05-04	2014-08-30	2018-11-17	2022-11-19	2024-05-04
BA	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
BALL	1996-09-29	2012-07-01	2017-09-30	2022-12-31	2023-12-31
BBWI	1996-11-02	2012-07-28	2017-10-28	2022-10-29	2024-08-03
BBY	1996-11-30	2012-08-04	2017-10-28	2022-10-29	2024-08-03
BWA	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
CAH	1996-03-31	2012-03-31	2017-09-30	2022-12-31	2024-06-30
CAT	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
CE	2006-09-30	2016-06-30	2019-09-30	2022-12-31	2024-06-30
CF	2007-03-31	2016-09-30	2019-12-31	2022-12-31	2024-06-30
CHRW	2001-12-31	2014-09-30	2018-12-31	2022-12-31	2024-06-28
CL	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
CLX	1997-03-31	2012-09-30	2017-12-31	2022-12-31	2024-06-30
CMCSA	2000-09-30	2014-03-31	2018-09-30	2022-12-31	2024-06-30
COO	1995-07-31	2012-01-31	2017-07-31	2023-01-31	2024-07-31
COST	1997-05-11	2012-09-02	2017-11-26	2022-11-20	2024-05-12

Table D-2 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
CSCO	1996-04-28	2012-04-28	2017-10-28	2023-01-28	2024-07-27
CTXS	1997-03-31	2012-03-31	2017-06-30	2022-06-30	2023-09-30
CVS	1996-09-28	2012-06-30	2017-09-30	2022-12-31	2024-06-30
CVX	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
DE	1999-07-31	2013-10-31	2018-07-29	2023-01-29	2024-07-28
DHI	1995-06-30	2011-12-31	2017-06-30	2022-09-30	2024-06-30
DISH	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
DOV	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
DRI	1997-02-23	2012-08-26	2017-11-26	2022-11-27	2024-05-26
DVN	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
EA	1996-12-31	2012-09-30	2017-12-31	2022-12-31	2024-06-30
EBAY	2000-03-31	2013-09-30	2018-06-30	2022-12-31	2024-06-30
ECL	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
EIX	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
EMN	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
EMR	1996-06-30	2012-03-31	2017-09-30	2022-12-31	2024-06-30
FAST	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
FL	2000-10-30	2014-02-01	2018-08-04	2022-10-29	2024-08-03
GLW	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30

Table D-2 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
GME	2002-11-02	2014-11-01	2018-11-03	2022-10-29	2024-05-04
GOOG	2005-06-30	2015-12-31	2019-06-30	2022-12-31	2024-06-30
GPC	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
GVA	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
GWW	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
HAL	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
HD	1996-10-27	2012-07-29	2017-10-29	2023-01-29	2024-07-28
HP	1997-06-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30
IBM	1995-09-30	2012-03-31	2017-09-30	2022-12-31	2024-06-30
IFF	2001-09-30	2014-06-30	2018-09-30	2022-12-31	2024-06-30
INTC	1996-09-28	2012-06-30	2017-09-30	2022-12-31	2024-06-29
INTU	1997-07-31	2012-10-31	2018-01-31	2023-01-31	2024-07-31
JNJ	1996-09-29	2012-07-01	2017-10-01	2023-01-01	2024-06-30
KMB	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
KO	1997-09-30	2012-09-28	2017-12-31	2022-12-31	2024-06-28
KSS	1996-11-02	2012-07-28	2017-10-28	2023-01-28	2024-08-03
M	1996-11-02	2012-07-28	2017-10-28	2022-10-29	2024-08-03
MAS	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
MDLZ	2002-09-30	2014-09-30	2018-12-31	2022-12-31	2024-06-30

Table D-2 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
MLM	1997-09-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30
MMM	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
MO	1997-09-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30
MRK	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
MSFT	1996-03-31	2012-03-31	2017-09-30	2022-12-31	2024-06-30
NFLX	2003-09-30	2015-03-31	2019-03-31	2022-12-31	2024-06-30
NKE	1997-02-28	2012-08-31	2017-11-30	2022-11-30	2024-05-31
NOV	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
NRG	1999-09-30	2013-09-30	2018-06-30	2022-12-31	2024-06-30
NVDA	2000-10-29	2014-04-27	2018-10-28	2023-01-29	2024-07-28
OI	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
OLN	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
PARA	2007-09-30	2016-09-30	2019-12-31	2022-12-31	2024-06-30
PG	1996-03-31	2012-03-31	2017-09-30	2022-12-31	2024-06-30
PM	2009-06-30	2017-09-30	2020-06-30	2022-12-31	2024-06-30
POOL	1997-09-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30
PTEN	1997-09-30	2012-09-30	2017-12-31	2022-12-31	2024-06-30
QRTEA	2008-06-30	2016-12-31	2019-12-31	2022-09-30	2024-06-30
RSG	1999-09-30	2013-09-30	2018-06-30	2022-12-31	2024-06-30

Table D-2 continued

Data set	Training period start	Validation period start	Regression testing period start	Simulation period start	Simulation period end
SCHL	1997-02-28	2012-08-31	2017-11-30	2022-11-30	2024-05-31
SHW	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
SLGN	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
SSD	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
STLD	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
TEX	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
TGNA	1996-09-30	2012-06-24	2017-09-30	2022-12-31	2024-06-30
TTC	1997-08-01	2012-08-03	2017-10-31	2022-10-31	2024-05-03
URBN	1996-10-31	2012-07-31	2017-10-31	2023-01-31	2024-07-31
UVV	2005-12-31	2016-03-31	2019-09-30	2022-12-31	2024-06-30
VMC	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
VRSN	1999-09-30	2013-09-30	2018-06-30	2022-12-31	2024-06-30
VZ	2003-09-30	2015-03-31	2019-03-31	2022-12-31	2024-06-30
WBD	2007-03-31	2016-09-30	2019-12-31	2022-12-31	2024-06-30
WLY	2001-01-31	2013-10-31	2018-01-31	2022-04-30	2024-04-30
WM	1998-09-30	2013-03-31	2018-03-31	2022-12-31	2024-06-30
WMT	1996-10-31	2012-07-31	2017-10-31	2023-01-31	2024-07-31
X	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30
XRAY	1996-09-30	2012-06-30	2017-09-30	2022-12-31	2024-06-30

Bibliography

[1] Mahinda Mailagaha Kumbure, Christoph Lohrmann, Pasi Luukka, and Jari Porras. Machine learning techniques and data for stock market forecasting: a literature review. *Expert Systems with Applications*, page 116659, 2022.

[2] Benjamin Graham, David Le Fevre Dodd, Sidney Cottle, et al. *Security analysis*. McGraw-Hill New York, 2008.

[3] Germania Vayas-Ortega, Cristina Soguero-Ruiz, José-Luis Rojo-Álvarez, and Francisco-Javier Gimeno-Blanes. On the differential analysis of enterprise valuation methods as a guideline for unlisted companies assessment (i): Empowering discounted cash flow valuation. *Applied Sciences*, 10(17), 2020.

[4] Tina Hviid Rydberg. Realistic statistical modelling of financial data. *International Statistical Review*, 68(3):233–258, 2000.

[5] Tim Koller, Marc Goedhart, David Wessels, et al. *Valuation: measuring and managing the value of companies*. John Wiley & Sons, 2020.

[6] Kees Camfferman and Stephen A Zeff. *Financial reporting and global capital markets: A history of the international accounting standards committee, 1973-2000*. Oxford University Press, USA, 2007.

[7] Barry J Epstein, Ralph Nach, and Steven M Bragg. *Wiley GAAP 2010: Interpretation and application of generally accepted accounting principles*. John Wiley & Sons, 2009.

[8] Aswath Damodaran. *Investment valuation: Tools and techniques for determining the value of any asset*. John Wiley & Sons, 2012.

[9] Glen Arnold. *The Financial Times Guide to Investing: The definitive companion to investment and the financial markets*. Pearson UK, 2012.

[10] Wendy McKenzie. *FT guide to using and interpreting company accounts*. Pearson Education, 2009.

[11] Stewart L Brown. Earnings changes, stock prices, and market efficiency. *The Journal of Finance*, 33(1):17–28, 1978.

[12] William F Sharpe. The capital asset pricing model: a "multi-beta" interpretation. In *Financial Dec Making Under Uncertainty*, pages 127–135. Elsevier, 1977.

[13] Efthimios G Demirakos, Norman C Strong, and Martin Walker. What valuation models do analysts use? *Accounting horizons*, 18(4):221–240, 2004.

[14] John Hoggett, John Medlin, Keryn Chalmers, Claire Beattie, Andreas Hellmann, and Jodie Maxfield. *Financial accounting*. John Wiley & Sons, 2024.

[15] Robert Parrino, Thomas W Bates, Stuart L Gillan, and David S Kidwell. *Fundamentals of corporate finance*. John Wiley & Sons, 2025.

[16] Zhu Fangshu. Review of us gaap and ifrs convergence: Revenue recognition aspects. *Research Journal of Management Sciences ISSN*, 2319:1171, 2015.

[17] Li Zhang, Han Zhang, and SuMin Hao. An equity fund recommendation system by combing transfer learning and the utility function of the prospect theory. *The Journal of Finance and Data Science*, 4(4):223–233, 2018.

[18] Guglielmo d'Amico and Riccardo De Blasis. A review of the dividend discount model: From deterministic to stochastic models. *Statistical Topics and Stochastic Models for Dependent Data with Applications*, pages 47–67, 2020.

[19] Michael Brennan. A note on dividend irrelevance and the gordon valuation model. *The Journal of Finance*, 26(5):1115–1121, 1971.

[20] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Springer, 2006.

[21] SC Nayak, Bijan B Misra, and Himansu Sekhar Behera. Impact of data normalization on stock index forecasting. *International Journal of Computer Information Systems and Industrial Management Applications*, 6:13–13, 2014.

[22] Francois Chollet. *Deep learning with Python*. Simon and Schuster, 2021.

[23] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[24] Felipe Tomazelli Lima and Vinicius MA Souza. A large comparison of normalization methods on time series. *Big Data Research*, 34:100407, 2023.

[25] Bo Peng, Robert K Yu, Kevin L DeHoff, and Christopher I Amos. Normalizing a large number of quantitative traits using empirical normal quantile transformation. In *BMC proceedings*, volume 1, pages 1–5. Springer, 2007.

[26] Alessandra Lunardi. *Interpolation theory*, volume 16. Springer, 2018.

[27] Thierry Blu, Philippe Thévenaz, and Michael Unser. Linear interpolation revitalized. *IEEE Transactions on Image Processing*, 13(5):710–719, 2004.

[28] Sky McKinley and Megan Levine. Cubic spline interpolation. *College of the Redwoods*, 45(1):1049–1060, 1998.

[29] FN Fritsch and RE Carlson. Piecewise cubic interpolation methods. Technical report, California Univ., Livermore (USA). Lawrence Livermore Lab., 1978.

[30] Fumio Hayashi. *Econometrics*. Princeton University Press, 2011.

[31] David Ruppert and Matthew P Wand. Multivariate locally weighted least squares regression. *The annals of statistics*, pages 1346–1370, 1994.

[32] Chun Yu and Weixin Yao. Robust linear regression: A review and comparison. *Communications in Statistics-Simulation and Computation*, 46(8):6261–6282, 2017.

[33] Gregory P Meyer. An alternative probabilistic interpretation of the huber loss. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 5261–5269, 2021.

[34] Art B Owen. A robust hybrid of lasso and ridge regression. *Contemporary Mathematics*, 443(7):59–72, 2007.

[35] Prapanna Mondal, Labani Shit, and Saptarsi Goswami. Study of effectiveness of time series modeling (arima) in forecasting stock prices. *International Journal of Computer Science, Engineering and Applications*, 4(2):13, 2014.

[36] Susana Lima, A Manuela Gonçalves, and Marco Costa. Time series forecasting using holt-winters exponential smoothing: An application to economic data. In *AIP conference proceedings*, volume 2186. AIP Publishing, 2019.

[37] Everette S Gardner Jr. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1):1–28, 1985.

[38] Marc Peter Deisenroth, A Aldo Faisal, and Cheng Soon Ong. *Mathematics for machine learning*. Cambridge University Press, 2020.

[39] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[40] Jerry Eriksson. *Optimization and regularization of nonlinear least squares problems*. Verlag nicht ermittelbar Jerusalem, 1996.

[41] David JC MacKay et al. Bayesian nonlinear modeling for the prediction competition. *ASHRAE transactions*, 100(2):1053–1062, 1994.

[42] Michael E Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun), 2001.

[43] David W Aha. *Lazy learning*. Springer Science & Business Media, 2013.

[44] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.

[45] Ryan Tibshirani and Larry Wasserman. Nonparametric regression. *Statistical Machine Learning, Spring*, 2013.

[46] Steven S Skiena. *The algorithm design manual*, volume 2. Springer, 1998.

[47] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[48] Trevor Hastie. The elements of statistical learning: data mining, inference, and prediction, 2009.

[49] Leo Breiman. Random forests. *Machine learning*, 45(1), 2001.

[50] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[51] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010.

[52] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

[53] Salvador Garcia and Francisco Herrera. An extension on" statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of machine learning research*, 9(12), 2008.

[54] Anthony Brabazon, Michael Kampouridis, and Michael O'Neill. Applications of genetic programming to finance and economics: past, present, future. *Genetic Programming and Evolvable Machines*, 21(1):33–53, 2020.

[55] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

[56] Souhaib Ben Taieb, James W Taylor, and Rob J Hyndman. Hierarchical probabilistic forecasting of electricity demand with smart meter data. *Journal of the American Statistical Association*, 116(533):27–43, 2021.

[57] Joos-Hendrik Böse, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Dustin Lange, David Salinas, Sebastian Schelter, Matthias Seeger, and Yuyang Wang. Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, 10(12):1694–1705, 2017.

[58] Gourav Kumar, Sanjeev Jain, and Uday Pratap Singh. Stock market forecasting using computational intelligence: A survey. *Archives of computational methods in engineering*, 28(3):1069–1101, 2021.

[59] Gaurang Sonkavde, Deepak Sudhakar Dharrao, Anupkumar M Bongale, Sarika T Deokate, Deepak Doreswamy, and Subraya Krishna Bhat. Forecasting stock market prices using machine learning and deep learning models: A systematic review, performance analysis and discussion of implications. *International Journal of Financial Studies*, 11(3):94, 2023.

[60] Gael M Martin, David T Frazier, Worapree Maneesoonthorn, Rubén Loaiza-Maya, Florian Huber, Gary Koop, John Maheu, Didier Nibbering, and Anastasios Panagiotelis. Bayesian forecasting in economics and finance: A modern review. *International Journal of Forecasting*, 40(2):811–839, 2024.

[61] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[62] Shakir Khan and Hela Alghulaiakh. Arima model for accurate time series stocks forecasting. *International Journal of Advanced Computer Science and Applications*, 11(7), 2020.

[63] Jiaqi Qin, Zheng Tao, Shansong Huang, and Gaurav Gupta. Stock price forecast based on arima model and bp neural network model. In *2021 IEEE 2nd international conference on big data, artificial intelligence and internet of things engineering (ICBAIE)*, pages 426–430. IEEE, 2021.

[64] Andrew C Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[65] Shuhaidah Abdul Shukor, Suliadi Firdaus Sufahani, Kamil Khalid, Mohd Helmy Abd Wahab, Syed Zulkarnain Syed Idrus, Asmala Ahmad, and Tamil Selvan Subramaniam. Forecasting stock market price of gold, silver, crude oil and platinum by using double exponential smoothing, holt's linear trend and random walk. In *Journal of Physics: Conference Series*, volume 1874, page 012087. IOP Publishing, 2021.

[66] Sun-Yong Choi. Industry volatility and economic uncertainty due to the covid-19 pandemic: Evidence from wavelet coherence analysis. *Finance research letters*, 37:101783, 2020.

[67] Nils Bertschinger and Iurii Mozzhorin. Bayesian estimation and likelihood-based comparison of agent-based volatility models. *Journal of Economic Interaction and Coordination*, 16(1):173–210, 2021.

[68] Paul Owusu Takyi and Isaac Bentum-Ennin. The impact of COVID-19 on stock market performance in Africa: A Bayesian structural time series approach. *Journal of Economics and Business*, 115:105968, 2021.

[69] S Kevin. *Commodity and financial derivatives*. PHI Learning Pvt. Ltd., 2024.

[70] Krzysztof Drachal. Forecasting the crude oil spot price with Bayesian symbolic regression. *Energies*, 16(1):4, 2022.

[71] Kerry Back. *A course in derivative securities: Introduction to theory and computation*. Springer, 2005.

[72] Andrew Carverhill and Dan Luo. A Bayesian analysis of time-varying jump risk in S&P 500 returns and options. *Journal of Financial Markets*, 64:100786, 2023.

[73] Rui Gao, Yaqiong Li, and Yanfei Bai. Numerical pricing of exchange option with stock liquidity under Bayesian statistical method. *Communications in Statistics-Theory and Methods*, 51(10):3312–3333, 2022.

[74] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654, 1973.

[75] Mohit Beniwal, Archana Singh, and Nand Kumar. Forecasting long-term stock prices of global indices: A forward-validating genetic algorithm optimization approach for support vector regression. *Applied Soft Computing*, 145:110566, 2023.

[76] Rebecca Abraham, Mahmoud El Samad, Amer M Bakhach, Hani El-Chaarani, Ahmad Sardouk, Sam El Nemar, and Dalia Jaber. Forecasting a stock trend using genetic

algorithm and random forest. *Journal of Risk and Financial Management*, 15(5):188, 2022.

[77] Mohammad Arashi and Mohammad Mahdi Rounaghi. Analysis of market efficiency and fractal feature of NASDAQ stock exchange: Time series modeling and forecasting of stock index using ARMA-GARCH model. *Future Business Journal*, 8(1):14, 2022.

[78] Chenyao Ma and Sheng Yan. Deep learning in the chinese stock market: the role of technical indicators. *Finance Research Letters*, 49:103025, 2022.

[79] Jingyi Liang and Guozhu Jia. China futures price forecasting based on online search and information transfer. *Data Science and Management*, 5(4):187–198, 2022.

[80] Kaijian He, Qian Yang, Lei Ji, Jingcheng Pan, and Yingchao Zou. Financial time series forecasting with the deep learning ensemble model. *Mathematics*, 11(4):1054, 2023.

[81] Matthias X Hanauer, Marina Kononova, and Marc Steffen Rapp. Boosting agnostic fundamental analysis: Using machine learning to identify mispricing in european stock markets. *Finance Research Letters*, 48:102856, 2022.

[82] Alexandre Rubesam. Machine learning portfolios with equal risk contributions: Evidence from the brazilian market. *Emerging Markets Review*, 51:100891, 2022.

[83] Dhruhi Sheth and Manan Shah. Predicting stock market using machine learning: best and accurate way to know future stock prices. *International Journal of System Assurance Engineering and Management*, 14(1):1–18, 2023.

[84] Helmut Wasserbacher and Martin Spindler. Machine learning for financial forecasting, planning and analysis: recent developments and pitfalls. *Digital Finance*, 4(1):63–88, 2022.

[85] Daniel Hoang and Kevin Wiegratz. Machine learning methods in finance: Recent applications and prospects. *European Financial Management*, 29(5):1657–1701, 2023.

[86] Turan G Bali, Heiner Beckmeyer, Mathis Moerke, and Florian Weigert. Option return predictability with machine learning and big data. *The Review of Financial Studies*, 36(9):3548–3602, 2023.

[87] Chaojie Wang, Yuanyuan Chen, Shuqi Zhang, and Qiuhui Zhang. Stock market index prediction using deep transformer model. *Expert Systems with Applications*, 208:118128, 2022.

[88] Ian WR Martin and Stefan Nagel. Market efficiency in the age of big data. *Journal of financial economics*, 145(1):154–177, 2022.

[89] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

[90] Zhen Zeng, Rachneet Kaur, Suchetha Siddagangappa, Saba Rahimi, Tucker Balch, and Manuela Veloso. Financial time series forecasting using cnn and transformer. *arXiv preprint arXiv:2304.04912*, 2023.

[91] Caosen Xu, Jingyuan Li, Bing Feng, and Baoli Lu. A financial time-series prediction model based on multiplex attention and linear transformer structure. *Applied Sciences*, 13(8):5175, 2023.

[92] Paul Bilokon and Yitao Qiu. Transformers versus lstms for electronic trading. *arXiv preprint arXiv:2309.11400*, 2023.

[93] Saqib Aziz, Michael Dowling, Helmi Hammami, and Anke Piepenbrink. Machine learning in finance: A topic modeling approach. *European Financial Management*, 28(3):744–770, 2022.

[94] Shengting Wu, Yuling Liu, Ziran Zou, and Tien-Hsiung Weng. S_i_lstm: stock price prediction based on multiple data sources and sentiment analysis. *Connection Science*, 34(1):44–62, 2022.

[95] T Swathi, N Kasiviswanath, and A Ananda Rao. An optimal deep learning-based lstm for stock price prediction using twitter sentiment analysis. *Applied Intelligence*, 52(12):13675–13688, 2022.

[96] Payal Soni, Yogya Tewari, and Deepa Krishnan. Machine learning approaches in stock price prediction: a systematic review. In *Journal of Physics: Conference Series*, volume 2161, page 012065. IOP Publishing, 2022.

[97] Jacob Boudoukh, Ronen Israel, and Matthew Richardson. Biases in long-horizon predictive regressions. *Journal of financial economics*, 145(3):937–969, 2022.

[98] Stefano Cassella, Benjamin Golez, Huseyin Gulen, and Peter Kelly. Horizon bias and the term structure of equity returns. *The Review of Financial Studies*, 36(3):1253–1288, 2023.

[99] Xiaojing Fan, Chunliang Tao, and Jianyu Zhao. Advanced stock price prediction with xlstm-based models: Improving long-term forecasting. *Preprints*, (2024082109), 2024.

[100] Markus Vogl, Peter Gordon Rötzel, and Stefan Homes. Forecasting performance of wavelet neural networks and other neural network topologies: A comparative study based on financial market data sets. *Machine Learning with Applications*, 8:100302, 2022.

[101] Alamir Labib Awad, Saleh Mesbah Elkaffas, and Mohammed Waleed Fakhr. Stock market prediction using deep reinforcement learning. *Applied System Innovation*, 6(6):106, 2023.

[102] Santosh Kumar Sahu, Anil Mokhade, and Neeraj Dhanraj Bokde. An overview of machine learning, deep learning, and reinforcement learning-based techniques in quantitative finance: recent progress and challenges. *Applied Sciences*, 13(3):1956, 2023.

[103] Razib Hayat Khan, Jonayet Miah, Md Minhazur Rahman, Md Maruf Hasan, and Muntasir Mamun. A study of forecasting stocks price by using deep reinforcement learning. In *2023 IEEE World AI IoT Congress (AIIoT)*, pages 0250–0255. IEEE, 2023.

[104] Yuanyuan Yu, Yu Lin, Xianping Hou, and Xi Zhang. Novel optimization approach for realized volatility forecast of stock price index based on deep reinforcement learning model. *Expert Systems with Applications*, 233:120880, 2023.

[105] Yuling Huang, Chujin Zhou, Kai Cui, and Xiaoping Lu. A multi-agent reinforcement learning framework for optimizing financial trading strategies based on timesnet. *Expert Systems with Applications*, 237:121502, 2024.

[106] Taylan Kabbani and Ekrem Duman. Deep reinforcement learning approach for trading automation in the stock market. *IEEE Access*, 10:93564–93574, 2022.

[107] Shuo Sun, Rundong Wang, and Bo An. Reinforcement learning for quantitative trading. *ACM Transactions on Intelligent Systems and Technology*, 14(3):1–29, 2023.

[108] Yasmeen Ansari, Sadaf Yasmin, Sheneela Naz, Hira Zaffar, Zeeshan Ali, Jihoon Moon, and Seungmin Rho. A deep reinforcement learning-based decision support system for automated stock market trading. *IEEE Access*, 10:127469–127501, 2022.

[109] Nan Li. An iteration algorithm for american options pricing based on reinforcement learning. *Symmetry*, 14(7):1324, 2022.

[110] Blanka Horvath, Josef Teichmann, and Žan Žurič. Deep hedging under rough volatility. *Risks*, 9(7):138, 2021.

[111] Loris Cannelli, Giuseppe Nuti, Marzio Sala, and Oleg Szehr. Hedging using reinforcement learning: Contextual k-armed bandit versus q-learning. *The Journal of Finance and Data Science*, 9:100101, 2023.

[112] David Valle-Cruz, Vanessa Fernandez-Cortez, Asdrúbal López-Chau, and Rodrigo Sandoval-Almazán. Does twitter affect stock market decisions? financial sentiment analysis during pandemics: A comparative study of the h1n1 and the covid-19 periods. *Cognitive computation*, 14(1):372–387, 2022.

[113] Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. Pixiu: A large language model, instruction data and evaluation benchmark for finance. *arXiv preprint arXiv:2306.05443*, 2023.

[114] John Affleck-Graves, Larry R Davis, and Richard R Mendenhall. Forecasts of earnings per share: Possible sources of analyst superiority and bias. *Contemporary Accounting Research*, 6(2):501–517, 1990.

[115] Patricia M Fairfield, Sundaresh Ramnath, and Teri Lombardi Yohn. Do industry-level analyses improve forecasts of financial performance? *Journal of Accounting Research*, 47(1), 2009.

[116] L Brooks and Dale Buckmaster. Price-change accounting models and disaggregated monetary gains and losses. *Quarterly Review of Economics and Business (Spring 1979)*, pages 115–130, 1979.

[117] Hossein Etemadi, Ahmad Ahmadpour, and Seyed Mohammad Moshashaei. Earnings per share forecast using extracted rules from trained neural network by genetic algorithm. *Computational Economics*, 46(1), 2015.

[118] Marko Kureljusic and Lucas Reisch. Revenue forecasting for european capital market-oriented firms: A comparative prediction study between financial analysts and machine learning models. *Corporate Ownership & Control*, 19(2), 2022.

[119] Lin Zhu, Mingzhu Yan, and Luyi Bai. Prediction of enterprise free cash flow based on a backpropagation neural network model of the improved genetic algorithm. *Information*, 13(4), 2022.

[120] James M Patell. Corporate forecasts of earnings per share and stock price behavior: Empirical test. *Journal of accounting research*, pages 246–276, 1976.

[121] Susan M Machuga, Ray J Pfeiffer, and Kiran Verma. Economic value added, future accounting earnings, and financial analysts' earnings per share forecasts. *Review of Quantitative Finance and Accounting*, 18:59–73, 2002.

[122] Lawrence D Brown and Michael S Rozeff. Univariate time-series models of quarterly accounting earnings per share: A proposed model. *Journal of Accounting Research*, pages 179–189, 1979.

[123] Peter D Easton, Martin M Kapons, Steven J Monahan, Harm H Schütt, and Eric H Weisbrod. Forecasting earnings using k-nearest neighbors. *The Accounting Review*, 99(3):115–140, 2024.

[124] Qing Cao and Mark E Parry. Neural network earnings per share forecasting models: A comparison of backward propagation and the genetic algorithm. *Decision Support Systems*, 47(1):32–41, 2009.

[125] Jan Alexander Fischer, Philipp Pohl, and Dietmar Ratz. A machine learning approach to univariate time series forecasting of quarterly earnings. *Review of Quantitative Finance and Accounting*, 55:1163–1179, 2020.

[126] Swati Jadhav, Hongmei He, and Karl Jenkins. Prediction of earnings per share for industry. In *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*, volume 1, pages 425–432. IEEE, 2015.

[127] Xi Chen, Yang Ha Cho, Yiwei Dou, and Baruch Lev. Predicting future earnings changes using machine learning and detailed financial data. *Journal of Accounting Research*, 60(2), 2022.

[128] Yi Jiang and Stewart Jones. Corporate distress prediction in china: A machine learning approach. *Accounting & Finance*, 58(4), 2018.

[129] Vic Anand, Robert Brunner, Kelechi Ikegwu, and Theodore Sougiannis. Predicting profitability using machine learning. *Available at SSRN 3466478*, 2019.

[130] Qi-Qiao He, Patrick Cheong-Iao Pang, and Yain-Whar Si. Transfer learning for financial time series forecasting. In *PRICAI 2019: Trends in Artificial Intelligence: 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26–30, 2019, Proceedings, Part II 16*, pages 24–36. Springer, 2019.

[131] Haoyang Cao, Haotian Gu, Xin Guo, and Mathieu Rosenbaum. Risk of transfer learning and its applications in finance. *Available at SSRN 4624427*, 2023.

[132] Nikolay Laptev, Jiafan Yu, and Ram Rajagopal. Reconstruction and regression loss for time-series transfer learning. In *Proceedings of the Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) and the 4th Workshop on the Mining and LEarning from Time Series (MiLeTS), London, UK*, volume 20, pages 1–8, 2018.

[133] Eirini Christinaki, Tasos Papastylianou, Sara Carletto, Sergio Gonzalez-Martinez, Luca Ostacoli, Manuel Ottaviano, Riccardo Poli, and Luca Citi. Well-being forecasting using a parametric transfer-learning method based on the fisher divergence and hamiltonian monte carlo. *EAI Endorsed Transactions on Bioengineering and Bioinformatics*, 1(1), 2020.

[134] Zhenyuan Zhang, Pengfei Zhao, Peng Wang, and Wei-Jen Lee. Transfer learning featured short-term combining forecasting model for residential loads with small sample sets. *IEEE Transactions on Industry Applications*, 58(4):4279–4288, 2022.

[135] Cheolhwan Oh, Seungmin Han, and Jongpil Jeong. Time-series data augmentation based on interpolation. *Procedia Computer Science*, 175:64–71, 2020.

[136] Terry T Um, Franz MJ Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulić. Data augmentation of wearable sensor data for

parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM international conference on multimodal interaction*, pages 216–220, 2017.

[137] Xingxing Wang, Peilin Ye, Yelin Deng, Yinnan Yuan, Yu Zhu, and Hongjun Ni. Influence of different data interpolation methods for sparse data on the construction accuracy of electric bus driving cycle. *Electronics*, 12(6):1377, 2023.

[138] Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 6989–6997, 2023.

[139] Kaleb E Smith and Anthony O Smith. Conditional gan for timeseries generation. *arXiv preprint arXiv:2006.16477*, 2020.

[140] Shiyu Liu, Rohan Ghosh, and Mehul Motani. Towards better long-range time series forecasting using generative adversarial networks. *arXiv preprint arXiv:2110.08770*, 2021.

[141] Christopher F Baum and Jesús Otero. Unit-root tests for explosive behavior. *The Stata Journal*, 21(4):999–1020, 2021.

[142] C Caruso and F Quarta. Interpolation methods comparison. *Computers & Mathematics with Applications*, 35(12):109–126, 1998.

[143] Grzegorz Słowiński. Influence of data dimension reduction, feature scaling and activation function on machine learning performance. *Proceedings http://ceur-ws. org ISSN*, 1613:0073, 2021.

[144] Khaled H Hamed and A Ramachandra Rao. A modified mann-kendall trend test for autocorrelated data. *Journal of hydrology*, 204(1-4):182–196, 1998.

[145] Marco Peixeiro. *Time series forecasting in python*. Simon and Schuster, 2022.

[146] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

[147] Ramdas Gore, Bharti Gawali, and Deepak Pachpatte. Weather parameter analysis using interpolation methods. In *Artificial Intelligence and Applications*, volume 1, pages 260–272, 2023.

[148] Hema Sekhar Reddy Rajula, Giuseppe Verlato, Mirko Manchia, Nadia Antonucci, and Vassilios Fanos. Comparison of conventional statistical methods with machine learning in medicine: diagnosis, drug development, and treatment. *Medicina*, 56(9):455, 2020.

[149] Frederick N Fritsch and Judy Butland. A method for constructing local monotone piecewise cubic interpolants. *SIAM journal on scientific and statistical computing*, 5(2):300–304, 1984.

[150] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41:647–665, 2014.

[151] Charles W Hodges, Walton RL Taylor, and James A Yoder. Beta, the treynor ratio, and long-run investment horizons. *Applied Financial Economics*, 13(7):503–508, 2003.

[152] Frank A Sortino and Lee N Price. Performance measurement in a downside risk framework. *the Journal of Investing*, 3(3):59–64, 1994.