University of Essex
Department of Mathematical Sciences

ON THE PREDICTABILITY OF U.S. STOCK MARKET USING MACHINE LEARNING AND DEEP LEARNING TECHNIQUES

A Thesis submitted for the award of

Doctor of Philosophy

by

Jonathan Iworiso

Supervisor: Dr. Spyridon Vrontos

March 25, 2020

Table of Contents

List of Tables

List of Figures

Abstract

Conventional market theories are considered to be inconsistent approach in modern financial analysis. This thesis focuses mainly on the application of sophisticated machine learning and deep learning techniques in stock market statistical predictability and economic significance over the benchmark conventional efficient market hypothesis and econometric models. Five chapters and three publishable papers were proposed altogether, and each chapter is developed to solve specific identifiable problem(s).

Chapter one gives the general introduction of the thesis. It presents the statement of the research problems identified in the relevant literature, the objective of the study and the significance of the study. Chapter two applies a plethora of machine learning techniques to forecast the direction of the U.S. stock market. The notable sophisticated techniques such as regularization, discriminant analysis, classification trees, Bayesian and neural networks were employed. The empirical findings revealed that the discriminant analysis classifiers, classification trees, Bayesian classifiers and penalized binary probit models demonstrate significant outperformance over the binary probit models both statistically and economically, proving significant alternatives to portfolio managers. Chapter three focuses mainly on the application of regression training (RT) techniques to forecast the U.S. equity premium. The RT models demonstrate significant evidence of equity premium predictability both statistically and economically relative to the benchmark historical average, delivering significant utility gains. Chapter four investigates the statistical predictive power and economic significance of financial stock market data by deep learning techniques. Chapter five give the summary,

conclusion and present area(s) of further research.

The techniques are proven to be robust both statistically and economically when forecasting the equity premium out-of-sample using recursive window method. Overall, the deep learning techniques produced the best result in this thesis. They seek to provide meaningful economic information on mean-variance portfolio investment for investors who are timing the market to earn future gains at minimal risk.

Dedication

This doctoral thesis is dedicated to my lovely wife **Mrs. Joy Uzoamaka Iworiso**, a virtuous woman with honour and integrity. She stood firmly with me from the beginning to the end of this terminal degree programme regardless of the distance as young couples. Our love will remain fresh, waxing stronger and stronger, from grace to greater grace till Jesus comes.

Acknowledgements

My profound gratitude goes to God Almighty for his love, faithfulness, mercy, protection and provision throughout this PhD programme. May his name be glorified forever.

I wish to express my heartfelt gratitude to the Federal Government of Nigeria for counting me worthy among millions of Nigerians to fund my PhD degree programme in the United Kingdom.

Suffice it to say that a research has never been successful without a laid-down direction, may I express my sincere appreciation to my amiable and distinguished supervisor **Dr. Spyridon Vrontos** for painstakingly supervising my PhD research. His constructive criticisms and suggestions contributed immensely to boost my academic status as doctoral scholar to face the task ahead.

Finally, I am grateful to all persons who have contributed in one way or the other to the success of this achievement.

List of Publications

The first paper (chapter 2 of this thesis) is published in a reputable international journal. The second and third papers (chapters 3 and 4 of this thesis) have been submitted for publication, and are currently on peer review.

1. Iworiso, J. & Vrontos, S. (2019). On the Directional Predictability of Equity Premium Using Machine Learning Techniques. Journal of Forecasting, 1-21. https://doi.org/10.1002/for.2632

2. Iworiso, J. & Vrontos, S. (2019). Forecasting the U.S. Equity Premium with Regression Training Techniques. Journal of Empirical Finance. Unpublished (peer review in progress).

3. Iworiso, J. & Vrontos, S. (2019). On the Predictability of Equity Premium Using Deep Learning Techniques. International Journal of Forecasting. Unpublished (peer review in progress).

Chapter 1

Introduction

A notable quest in modern financial literature is the search for more suitable models that can explicitly model and forecast financial time series data with utmost precision to guarantee investors future expectation at extremely low volatility. In this case, the researchers are expected to present models that can significantly outperformed the conventional financial modelling and forecasting approach. A number of statistical and econometric models have been applied by various scholars over the years. Although the empirical findings in most situations are proven to outperformed the conventional approaches but are generally considered to be weak as compared to a rule of thumb (i.e., at least 50%) especially in sign forecasting and market dynamics (Leitch and Tanner, 1991; Christoffersen and Diebold, 2006; Anatolyev and Gospodinov, 2010; Pönkä, 2016), and hence, the need for further research in this field of study. The empirical findings in Nyberg (2011, 2013) demonstrates the effectiveness of static and dynamic statistical models in financial time series analysis but the resulting predictive power of the models seems to be weak and required further statistical and economic performance evaluation measures to affirm the stability in finance. The classification and level estimation techniques in Leung et al. (2000) greatly provides evidence of useful predictability over the conventional financial methodologies known as the benchmark buy and hold (B&H) trading strategy, with the discriminant classifier emerging as the superior model in this perspective. The quest for a superior model to forecast equity risk premium out-of-sample comparative to the benchmark historical average with convincing statistical and economic performance evaluation measures to a mean-variance portfolio investor, is another crucial debatable research issue in modern finance (Goyal and Welch, 2003; Campbell and Thompson, 2005, 2007; Rapach et al., 2007; Neely et al., 2014; Baetje and Menkhoff, 2016; Bai, 2010; Li and Tsiakas, 2017). Although the statistical and econometric models used in the afore-mentioned studies demonstrate feasibility and evidence of both statistically and economically significant predictability but further investigation is required to authenticate the fate of investors timing the market to maximize profit at minimal risk.

The earnest expectation of the stock market outcome to mean-variance investors above the treasury bill rate led to the quest for a meaningful estimate of excess stock return or equity premium (Campbell, 2008). The equity premium is the difference between the expected return on the market portfolio (SP500) and the risk free interest rate. Thus, investors can expect this return from holding the market portfolio in excess of the return on the 3-month Treasury bills. It is considered to be the most crucial concept in finance, owing to portfolio allocation decisions and cost of capital estimates. It is worth noting that the backbone of investment strategies depends on the ability to predict future returns but the predictability itself does not necessarily guarantee the investor's profit from the trading strategy based on the resulting forecasts (Campbell and Thompson, 2005; Bai, 2010). An academic debating question in modern review of financial studies is that: can any other model accurately forecast the equity premium better than the forecasts from the historical average? Goyal and Welch (2007) have argued that no other variable beats a simple forecast based on the historical average, and concluded that the in-sample correlations conceal a systematic failure of the financial and economic variables out-of-sample. Contrary to this empirical analysis, the results in Rapach et al. (2007) reveals that despite the failure of the individual model forecasts to outperform the historical mean forecasts, the combination of the individual model forecasts yield statistically and economically significant gains, relative to the historical mean, consistently over time. Although some of the indicators used as predictors appeared to be good statistically significant predictors of the equity premium in-sample at some specific horizons but are relatively poor in the out-of-sample forecasting ability. Fama and French (2002) verified that the estimates from economic fundamentals especially in the dividend growth model, appeared to produce lower standard errors resulting to corresponding better precision than the estimates from the benchmark historical average model. The combination approach in Neely et al. (2014) confirmed that both technical indicators and macroeconomic variables displayed statistically and economically significant evidence of in-sample and out-of-sample forecasting ability, with the technical indicators seemingly outperforming the macroeconomic variables. The empirical analysis suggests that the combination of both technical indicators and macroeconomic variables will significantly improve the equity risk premium forecasts rather than using a single set of the predictor variables alone. However, the findings require further investigation for either corroboration or refutation. Following this argument, Baetje and Menkhoff (2016) demonstrated that the predictive abilities of both indicators seem to possess similar quality when assessed by their respective long term forecast errors. Unlike the economic indicators that loses predictive ability on a long run, the technical indicators maintain or increase stability over time, and hence, the technical indicators consistently outperformed the economic indicators over time. The application of forecasts combination in Rapach et al. (2010) confirmed that combination of forecasts yields statistically and economically significant out-of-sample gains consistently on a long run, compared to the benchmark historical average. Therefore, the forecasts combination approach in Rapach et al. (2010) appeared to maintain a long run statistical and economic stability.

However, evaluating the mean squared errors (MSEs) and Sharpe ratios (SRs) alone, do not provide adequate evidence to justify the superiority of a specific model over other competitive models. Some existing papers focused mainly on evaluating the model prediction errors, expected returns on portfolio investment and the corresponding Sharpe ratio. In addition to these parametric measures, Nyberg (2011) included the Diebold-Mariano (DM) statistical tests and the Pesaran-Timmermann (PT) directional predictability tests only in the in-sample case, but do not investigate the DM among the models in the out-of-sample case. The benchmark used in the study was the expected return on a portfolio investment held on a risk-free interest rate, based on the buy and hold trading strategy. Goyal and Welch (2003), Campbell and Thompson (2005) were mainly concerned with examining the predictive ability of individual predictor variables relative to the benchmark historical average, using the out-of-sample statistical goodness of fit tests and the SRs, whereas Campbell and Thompson (2007); Campbell (2008), Goyal and Welch (2007) and Rapach et al. (2010) added an important concept known as the *utility gain*, which serves as an additional benchmark for comparing the economic performance of a model to a portfolio investor with a portfolio held on the risk-free Treasury bill. However, some important concepts such as cumulative returns among others were not included in their studies. It is imperative to explore adequate statistical predictive and economic significance tests especially in the out-of-sample forecasting models, to determine superiority among the resulting models used in the study.

In modern research, the use of machine learning and deep learning techniques is drawing rapid attention in financial time series analysis. Machine learning is the study of sophisticated algorithms and mathematical models existing between input features and target output or to learn and recognize patterns in order to improve the performance on a specific task typically by computer systems. In this case, the algorithm builds a model from sample training data and use the resulting model to make predictions without being explicitly programmed to perform the specific task. The feasibility of some machine learning techniques in finance (Kumar and Thenmozhi, 2006; Chen, 2011; Huang and Wu, 2008; Ince and Trafalis, 2007;

Pahwa et al., 2017; Patel et al., 2015) with desirable predictive performance had led to the introduction of more sophisticated learning technique known as *deep learning* which have the ability to extract features from a large raw dataset without relying on prior knowledge of predictors. Day and Lee (2016) described *deep learning* as deep neural network, which is a more sophisticated aspect of machine learning. It is a form of machine learning technique that involved the use of data to train a model or recognize pattern(s) or label instances in order to make predictions from new data in a more sophisticated manner (Heaton et al., 2017). Machine learning and deep learning techniques are shown in empirical literature to be useful techniques to learn data, recognize patterns such as speech recognition, and to classify instances such as digital image classification. They appeared to be useful modern research techniques for analysis in computer science, biology, medicine, linguistics, physics, statistics, economics and finance.

It is worth noting that the machine learning algorithms/models introduced in financial and econometric literature do not provide adequate statistical predictive and economic significant measuring tools for comparing their performances with the conventional efficient market hypothesis and econometric models. Suffice it to say that the justification of a predictive model in terms of superiority over the conventional models in finance depends on both statistical and economic performance measures. The findings in Kumar and Thenmozhi (2006), Chen (2011) and Chen and Hao (2017) are promising, but do not provide adequate statistical and economic measures to demonstrate the superiority of machine learning techniques over the benchmark approaches in finance. Yoshihara et al. (2014) shows that recurrent deep neural networks appeared to be more effective approach over support vector machines (SVM) and deep belief networks when predicting the trend of stock market, especially when the process is focused on specific period after a known significant event in financial domain. The analysis provides a controversial superiority of recurrent neural networks over the deep belief network in this direction. The empirical findings in (Zhao et al., 2017) shows the superiority of deep learning ensemble with stacked denoising autoencoders for modelling and forecasting crude oil prices over the bootstrap aggregation and other machine learning techniques used in the literature. The empirical results in Hu et al. (2018), Feuerriegel and Fehrer (2016), Heaton et al. (2016) also confirmed that the application of deep learning techniques in financial analysis seek to outperformed both the standard methods in finance and the conventional machine learning techniques. Armano et al. (2005) introduced a hybrid genetic-neural architecture to forecast stock indexes with consideration of realistic trading commission, and appeared to be promising in the selected application task. The empirical findings also demonstrate evidence of superior outperformance over the benchmark buy and hold strategy for a large test sample size. Contrary to these analyses, is the empirical results in Krauss et al. (2017) in which a random forest outperformed some notable deep learning techniques. It was shown that the random forest outperformed both deep neural networks (DNN) and gradient boosted trees in the investigation of statistical arbitrage on SP500, and concluded that a further investigation by hyper-parameter optimization for the deep neural networks is required as an area of future research work.

The statement of the problem lies on the provision of superior technique that can significantly outperform the existing conventional econometric models and to fill the identifiable research gaps in the existing literature. The objective of this study is to explore the sophisticated machine learning and deep learning techniques to model financial stock market data in order to make predictions and evaluate their performances with robustness, and to demonstrate superior outperformance of these proposed methodologies in the thesis over the benchmark approaches used in the existing literature. The study aimed to introduce additional statistical and economic performance evaluation measures that can authenticate the long-run effectiveness and consistency of the proposed models in relation to optimistic investment approach for yielding future gains. Therefore the outcome of this study shall provide significant statistical and economic information to stock market investors and portfolio managers on the need for optimal portfolio assessment with a view to maximize profit at minimal risk when timing the market. It shall also fill the identifiable research gaps, enrich empirical literature and present area of further research to future researchers on the subject matter.

Chapter 2

Directional Predictability of U.S. Stock Market Using Machine Learning Techniques

2.1 Introduction

Stock market participants aim at maximising returns on portfolio investments at minimal risk. Consequently, forecasting stock market returns has received considerable attention in recent years. The majority of papers have focused on the forecast accuracy of competing models and examined if there is evidence of predictability, which can lead to economic gains. However, devising successful trading strategies is contingent on the directional accuracy of the underlying models. The literature on directional predictability is sparse, and the empirical findings offer limited support. For example, the findings in (Chevapatrakul, 2013; Christoffersen and Diebold, 2006; Nyberg and Pönkä, 2016) provide weak evidence of directional stock market predictability. Although the predictive power of the models employed so far are shown to be weak in statistical terms, they seem to provide economic value. Thus, the emphatic challenge lies in the development of a suitable directional predictive model involving the relevant financial and economic variables.

The application of some benchmark econometric models used in previous findings are shown to be weak in terms of predictive performance. The introduction of recursive and alternative rolling windows out-of-sample estimation and forecasting techniques used by Nyberg (2008, 2011), Pönkä (2016) provide statistically significant evidence of the directional predictability of stock market returns, but the predictive power of the models are shown to be relatively weak, and hence, there is a need to introduce sophisticated machine learning techniques, as proposed in this study to improve the predictive task of the models.

This chapter focuses on the application of sophisticated machine learning techniques on binary probit and classification models to forecast the direction of the U.S. excess stock market returns. The machine learning techniques employed include classification and regression trees (CART), such as Bagging, Boosting and Discriminant Analysis classifiers, Bayesian classifiers, Neural Networks and regularization techniques, such as Ridge, Least Absolute Shrinkage and Selection Operator (LASSO), and Elastic Net. To compare our findings with the previous literature, we also include four variants of the benchmark binary probit models, namely, the static, stepwise static, dynamic and stepwise dynamic models. The application of CART forecasting models aims to explore all covariates as ensembles to learn the data, train the classification model, recognize patterns, classify instances and to forecast future binary outcomes. With respect to penalised binary probit models, it is important to note that the presence of shrinkage penalty vector norms could result to a bias in coefficient estimates, reduction in the forecast errors and improvement in predictive performance via the so-called bias-variance trade off. Thus, the proposal of CART and penalized predictive models in this chapter aims at yielding superior statistical predictive performance and economic significance compared to the benchmark econometric models typically employed in the literature to date.

2.2 Literature Review

A notable quest in modern financial econometric literature is the application of suitable techniques to predict the sign of stock market returns. A review of relevant empirical literature has revealed that the use of econometric models for the directional predictability of excess stock returns are known to produce weak predictive power, poor statistical goodness of fit and low predictive accuracies, among others; see (Pesaran and Timmermann, 1995; Nyberg, 2011; Leung et al., 2000; Chevapatrakul, 2013; Leitch and Tanner, 1991; Pönkä, 2016), even though the empirical results seems to provide economic significance.

The previous findings on directional predictability by Anatolyev and Gospodinov (2010), and Hong and Chung (2003) have employed a logistic regression model to predict the sign of U.S. stock market returns using relevant financial variables as the key predictors, and their results provide evidence of predictability, but the overall predictive power is relatively weak, compared to a rule of thumb (i.e., at least 50%). In an attempt to determine market timing and asset allocation decisions between stocks and risk-free assets, some researchers considered the role of conditional mean and volatility while predicting the sign of asset returns. Christoffersen and Diebold (2006) have opined that the direction of asset returns is predictable, as volatility dependence produces sign dependence, so long as expected returns are nonzero. This notion seems to be true, as other existing papers have also provided significant statistical evidence of the sign predictability of the U.S. stock market returns and economic recession status by application of static, dynamic, autodynamic and error correction models, both in-sample and out-of-sample (Nyberg, 2011; Kauppi and Saikkonen, 2008; Nyberg and Pönkä, 2016; Nyberg, 2013).

The static and dynamic probit models proposed by Nyberg (2011) to predict the direction of monthly U.S. excess stock returns recursively appears to have outperformed the autoregressive moving average with exogenous inputs models (ARMAX), vector autoregressive-generalized autoregressive conditional heteroskedasticity (VAR-GARCH) models, etc. used by previous researchers. The idea was based on the approach used by Kauppi and Saikkonen (2008), Estrella and Mishkin (1998) to obtain U.S. economic recession forecasts using variables such as the U.S. term spread and lagged stock returns, among others.

However, according to the Nyberg (2011) paper, the Estrella's statistical goodness of fit values in the various probit models are very low in all cases. The positive values of the Sharpe ratios signified that investors are likely to have positive returns on portfolio investments. The percentage of correct matches as a statistical performance evaluation measure in the existing papers are relatively low, hence the need to employ more advanced sophisticated models that can yield a better degree of accuracy with the smallest prediction error.

The underlying challenges associated with the use of financial and economic variables to predict stock market returns has prompted researchers to introduce sophisticated statistical or machine learning algorithms to improve the predictive task and the overall performance of the resulting models under consideration. It is noticeable from the empirical literature that machine learning techniques, which include Random Forest, Linear Discriminant Analysis (LDA), k-Nearest Neighbour, Tree-based Classification, Recursive Partitioning, Bagging and Boosting, Logistic Regression, Support Vector Machine (SVM), Ridge Regression, Least Absolute Shrinkage and Selection Operator (LASSO), Least Angle Regression and Elastic Nets, are useful for the analysis of financial econometric time series (Roy et al., 2015; Sermpinis et al., 2017; Li and Chen, 2014; Inoue and Kilian, 2008; Zhou et al., 2015; Hsu et al., 2008; Park and Sakaori, 2013; Chen, 2016; Stock and Watson, 2012; Lin and McClean, 2001; Kim and Swanson, 2014; Hajek et al., 2014; Shen et al., 2014; Pahwa et al., 2017; Swanson and White, 1997). Khaidem et al. (2016) used the Random Forest method to predict the direction of stock market prices. The algorithm appears to be robust in predicting the future direction of the stock market movement, thus minimizing the risk of investment in the stock market with good predictive accuracy.

The ridge regression introduced by Hoerl and Kennard (1970), and the least absolute shrinkage and selection operator (LASSO) introduced by Tibshirani (1996) are found to be useful statistical or machine learning techniques for econometric models. The ridge regression reduces multicollinearity and minimizes the model prediction error but does not perform feature selection; the LASSO shrinks the model coefficients towards zero and performs feature selection and model interpretability. The aim is to introduce bias in the model coefficient estimates and minimize the prediction error.

The empirical analysis in Inoue and Kilian (2008) revealed that bagging has large reductions in prediction mean square errors (PMSEs) in inflation forecasting. Kim and Swanson (2014) suggest that the model averaging does not dominate other well designed prediction model specification methods, and that the use of hybrid combination factor and shrinkage methods produced the best predictions as compared to principal components, bagging, boosting, least angle regression, among others. On the other hand, the empirical results from Zhou et al. (2015) showed no statistically significant difference between the best classification performance of the models with yearly feature selection guided by data mining techniques and the one involving domain knowledge; hence, their predictive accuracies seems to be the same.

The use of the LASSO linear regression model for stock market forecasting in Roy et al. (2015) using monthly data revealed that the LASSO method yield sparse solutions and performs extremely well when the number of features is less than the number of observations, and that the LASSO linear regression model outperforms the ridge linear regression model. Modelling the market implied ratings using LASSO variable selection techniques in Sermpinis et al. (2017) and forecasting macroeconomic time series using LASSO-based approaches and their forecast combinations with dynamic factor models in Li and Chen (2014) all reflect statistical evidence of the superior predictive power of LASSO.

The outperformance of the aforementioned statistical learning algorithmic techniques over the benchmark econometric and statistical modelling techniques has prompted modern researchers to proceed into a more advanced concept, i.e., the deep learning techniques based on artificial intelligence, which encompasses support vector machines (SVM) and neural networks (NNET). However, the contrasting arguments of various scholars on the predictive performance by SVM and NNET as compared to the previous literature has placed this notion pending for further statistical investigation. The application of artificial neural networks (ANN) in forecasting financial markets and stock prices in Shahpazov et al. (2014) demonstrated the outperformance of the NNET over previous techniques used in the existing literature. Again, the findings in de Oliveira et al. (2013) also revealed that the application of artificial neural networks yielded the minimum mean square prediction error (MSE) and correct direction rates. Controversially, the analytical results by Moreno and Olmeda (2007) show that the ANN do not provide evidence of superior performance over the conventional linear models. The findings in Ding et al. (2013), applying the concept for daily data and market sentiment, shows the outperformance of SVM over NNET and logistic regression. The SVM seems to be the most accurate machine learning model for predicting stock market movement, but the statistical tests do not provide significant statistical evidence of better performance over NNET and logistic regression. Patel et al. (2015) confirmed the outperformance of Random Forest over ANN, SVM and the genetic algorithm (GA) for input data with continuous values. Ballings et al. (2015) also presented random forest as the top machine learning algorithm over others and recommended the inclusion of ensembles in algorithmic sets when predicting the direction of stock market prices. The findings in Zheng (2006) demonstrated the superiority of boosting and bagging of NNET over SVM and logistic regression when forecasting the daily directional movements of stocks.

It is obvious, based on the reviewed existing empirical literature, that machine learning techniques played an enormous role in financial econometric time series. Thus, the application of the proposed sophisticated machine learning recursive out-of-sample forecasting models for the directional predictability of the U.S. stock market returns in this paper aimed to yield significant results and outperform the benchmark econometric models and aimed to enrich the empirical literature for further relevant scholarly research work.

2.3 Research Methodology

This section gives a detailed theoretical approach to excess stock return modelling as a binary time series, the static and dynamic binary probit forecasting models; the application of machine learning techniques which include the Ridge, LASSO and Elastic Net probit models; the classification and regression trees (CART); followed by the forecasting/predictive model performance evaluation for easy comparison.

2.3.1 Equity Premium Direction Modelling as a Binary Time Series

Let R_t be the monthly U.S. excess stock market return over the risk-free interest rate denoted by rf_t, and let I_t^s denote the binary-valued dependent variable. The sign of the monthly equity premium is modelled as the return sign binary indicator, as follows:

$$I_t^s = \begin{cases} 1, \text{ if } R_t > 0 \text{ i.e., positive excess stock market return} \\ 0, \text{ if } R_t \leq 0 \text{ i.e., negative or zero excess stock market return.} \end{cases} \tag{2.1}$$

R_t is calculated as follows:

$$R_t = \ln\left(\frac{P_t}{P_{t-1}}\right) - rf_{t-1} \tag{2.2}$$

where P_t is the price of the stock index at period t and rf_{t-1} is the risk-free interest rate at period $t-1$ (Pesaran, 2015; Leung et al., 2000). The distribution of the return sign binary indicator I_t^s conditional on $\Re_{t-1}$ follows Bernoulli with probability p_t, as follows:

$$I_t^s|\Re_{t-1} \sim Bernoulli(p_t),$$

where $\Re_{t-1}$ is the information set of the covariates.

2.3.2 The Static and Dynamic Binary Probit Models

Christoffersen and Diebold (2006) showed that if R_t is distributed as follows:

$$R_t|\Re_{t-1} \sim N(\mu, \sigma^2_{t|t-1})$$

and displays no conditional mean dependence and conditional variance dependence, then there exists a link between the volatility dynamics and the sign dynamics. The conditional probability of a positive excess stock market return $Prob_{t-1}(R_t > 0)$ is as follows:

$$Prob_{t-1}(R_t > 0) = 1 - \Gamma\Big(\frac{-\mu}{\sigma_{t|t-1}}\Big) = \Gamma\Big(\frac{\mu}{\sigma_{t|t-1}}\Big)$$

where $\Gamma(.)$ is the $N(0,1)$ cumulative distribution function, and the forecast horizon used is equal to 1. The sign of equity premium is predictable if the conditional probability of positive equity premium $Prob_{t-1}(R_t > 0) > 0.5$ for a threshold of 0.5, varies with the information set $\Re_{t-1}$, which invariably implies a direction of change in the forecastability or predictability of the equity premium (Chevapatrakul, 2013).

Given the conditional expectation E_{t-1}, conditional on the information set $\Re_{t-1}$. The conditional probability of a positive equity premium sign employing the binary probit indicator $I_t^s = 1$ is as follows:

$$p_t = E_{t-1}(I_t^s) = Prob_{t-1}(I_t^s = 1) = Prob_{t-1}(R_t > 0) = \Gamma(\Psi_t)$$

where $\Gamma(.)$ is a standard normal cumulative distribution function.
The static and dynamic binary probit models can be obtained from this direction, using the fact that the autocorrelation between any two successive numerical values of the equity premium is statistically negligible.
Thus, the static binary probit forecasting model is defined as follows:

$$\Psi_{t+1}(\beta) = \alpha + \mathbf{Z}_t'\beta \tag{2.3}$$

where α is the model intercept;
$\mathbf{Z}_t$ is k-dimensional covariate vector of predictors of equity premium;
β is $k \times 1$ vector of unknown coefficients (Nyberg, 2011; Nyberg and Pönkä, 2016).

Using the uncorrelated assumption $Cor.(I_{t+1}^s, I_t^s) = 0$, the historical value of the equity premium sign indicator I_t^s is included in the static binary probit forecasting model, which results to the dynamic binary probit forecasting model. Thus, the dynamic binary probit forecasting model is

$$\Psi_{t+1}(\beta) = \alpha + \sum_{i=1}^{p} \eta_i I_t^s + \mathbf{Z}_t'\beta \tag{2.4}$$

where α is the model intercept;
η is an unknown coefficient of the lagged equity premium sign indicator;
$\mathbf{Z}_t$ is k-dimensional covariate vector of the predictors of equity premium;
β is $k \times 1$ vector of unknown coefficients;
p is the lag order of the equity premium sign indicator (Kauppi and Saikkonen, 2008).

Thus, the benchmark forecasts from the static binary probit model are based on the link function:

$$Prob(I^s_{t+1} = 1|\Re_t) = \Gamma(\alpha + \mathbf{Z}'_t\beta) \tag{2.5}$$

and the benchmark forecasts from the dynamic binary probit model are based on the link function:

$$Prob(I^s_{t+1} = 1|\Re_t) = \Gamma(\alpha + \sum_{i=1}^{p} \eta_i I^s_t + \mathbf{Z}'_t\beta) \tag{2.6}$$

where $p \geq 1$ is the lag order (Kauppi and Saikkonen, 2008).

2.3.2.1 Stepwise Variable Selection using Akaike Information Criterion

The stepwise variable selection is a step-by-step selection technique which seeks to screen the predictive variables of a specific model by an automatic iterative procedure. It involves a screening process in that in each step, a predictor variable is considered for inclusion or elimination from the set of predictor variables based on the significant status determined by an information criterion. In this study, the bidirectional (forward-and-backward) stepwise approach with the Akaike information criterion (StepAIC) was used for further investigation of the static and dynamic binary probit models.

2.3.2.2 Likelihood Estimation of Binary Probit Model Parameters

The parameters of the binary probit models defined in (2.3) and (2.4) can be estimated by maximum likelihood method. Given the function:

$$Prob(I^s_{t+1} = 1|\Re_t) = \Gamma(\Psi_{t+1}(\beta)) \tag{2.7}$$

The likelihood function of β is defined as follows:

$$L(\beta) = \prod_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) \prod_{(I^s_{t+1}=0)} \Big(1 - \Gamma\big(\Psi_{t+1}(\beta)\big)\Big) \tag{2.8}$$

The log-likelihood function is defined as follows:

$$lnL(\beta) = \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1 - \Gamma\big(\Psi_{t+1}(\beta)\big)\Big) \tag{2.9}$$

Thus, the maximum likelihood estimator (MLE) of β is obtained as follows:

$$\hat{\beta}_{ML} = \arg\max_{\beta} \left\{ \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1 - \Gamma\big(\Psi_{t+1}(\beta)\big)\Big) \right\}$$

where $\Gamma(.)$ is the standard normal cumulative distribution function (Estrella and Mishkin, 1998; Pesaran, 2015; Kedem and Fokianos, 2005)

2.3.3 The Penalized Binary Probit Models for Stock Return Predictability

This section examined the penalized likelihood binary probit models employing the relevant Ridge, LASSO and Elastic Net structures. The inclusion of a penalty vector norm in the log-likelihood function of the ordinary binary probit model results in the penalized binary probit model. It is worth noting that in the penalized likelihood binary probit models, the coefficient estimates are shrunk towards zero. The regularized biased coefficients are known to have significantly reduced variances, that could result in smaller forecasting errors.

2.3.3.1 The Ridge Binary Probit Model

The ridge binary probit model aims to reduce multicollinearity and minimize the prediction error of the model and is based on the ridge regression introduced by Hoerl and Kennard (1970). Given the log-likelihood function of the ordinary binary probit model (2.9), the ridge log-likelihood probit function introduces a penalty on the ℓ_2-norm of β:

$$lnL(\beta^{\lambda}) = \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1-\Gamma\big(\Psi_{t+1}(\beta)\big)\Big) - \lambda \sum_{j=1}^{k} \beta_j^2 = lnL(\beta) - \lambda \parallel \beta \parallel_2^2 \tag{2.11}$$

where $lnL(\beta)$ is the unrestricted log-likelihood function of the probit model; $\parallel \beta \parallel_2^2 = \sqrt{\sum_{j=1}^{k} \beta_j^2}$ is the ℓ_2-vector norm of β; $\lambda > 0$ is the ridge tuning parameter which controls the amount of regularization of the norm of β.
Thus, the maximum likelihood estimator of the ridge binary probit model is given by the following:

$$\hat{\beta}^{\lambda}_{RMLE} = \arg\max_{\beta} \left\{ \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1-\Gamma\big(\Psi_{t+1}(\beta)\big)\Big) - \lambda \sum_{j=1}^{k} \beta_j^2 \right\}$$

where $\hat{\beta}^{\lambda}$ is the maximizer of the ordinary probit model.

2.3.3.2 The LASSO Binary Probit Model

The Least Absolute Shrinkage and Selection Operator (LASSO) introduced by Tibshirani (1996) as a shrinkage and selection technique for linear regression models is extended to binary probit models. The proposed LASSO binary probit model aims to shrink the probit model coefficients toward zero, yielding bias parameter estimates, resulting in the model interpretability and identification of the covariates most strongly associated with the equity premium direction.
To obtain the LASSO coefficient $\hat{\beta}^{\lambda}_{LMLE}$, the maximization of the log-likelihood

function of the ordinary binary probit model (2.9) will include a shrinkage penalty on ℓ_1-norm of β. The introduction of the constraint into the probit model is expressed by incorporating a shrinkage penalty to the log-likelihood of the model. Thus, the constraint maximization for the log-likelihood becomes:

$$lnL(\beta^{\lambda}) = \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1-\Gamma\big(\Psi_{t+1}(\beta)\big)\Big) - \lambda \sum_{j=1}^{k} |\beta_j| = lnL(\beta) - \lambda \parallel \beta \parallel_1 \tag{2.12}$$

where $lnL(\beta)$ is the unrestricted log-likelihood function of the probit model; $\parallel \beta \parallel_1 = \sqrt{\sum_{j=1}^{k} |\beta_j|}$ is the ℓ_1-vector norm of β; $\lambda > 0$ is the LASSO tuning parameter, which controls the amount of shrinkage (regularization) of the norm of β.
The vector $\hat{\beta}^{\lambda}_{LMLE}$, that maximizes $lnL(\beta^{\lambda})$ is the LASSO binary probit estimator of β, hence, the LASSO binary probit model coefficient estimates are obtained by

$$\hat{\beta}^{\lambda}_{LMLE} = \arg\max_{\beta} \left\{ \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1 - \Gamma\big(\Psi_{t+1}(\beta)\big)\Big) - \lambda \sum_{j=1}^{k} |\beta_j| \right\}$$

where $\hat{\beta}^{\lambda}$ is the maximizer of the ordinary probit model.

2.3.3.3 The Elastic Net Binary Probit Model

The elastic net (EN) is a regularized technique that linearly combines the ℓ_1 and ℓ_2 penalties of the LASSO and Ridge. The elastic net probit coefficient estimates $\hat{\beta}^{\lambda,\alpha}_{EMLE}$ are obtained by maximizing the log-likelihood function, which penalized the size of the model coefficients based on both the ℓ_1-vector norm and ℓ_2-vector norm of β, defined by the following:

$$lnL(\beta^{\lambda,\alpha}) = \sum_{(I^s_{t+1}=1)} \Gamma\Big(\Psi_{t+1}(\beta)\Big) + \sum_{(I^s_{t+1}=0)} \Big(1-\Gamma\big(\Psi_{t+1}(\beta)\big)\Big) - \lambda\Big((1-\alpha)\sum_{j=1}^{k} \frac{\beta_j^2}{2} + \alpha \sum_{j=1}^{k} |\beta_j|\Big) \tag{2.13}$$

Thus, the parameter estimates of the elastic net binary probit model will be given by the following:

$\hat{\beta}^{\lambda,\alpha}_{EMLE}=$

$$\arg\max_{\beta}\left\{\sum_{(I^s_{t+1}=1)}\Gamma\Big(\Psi_{t+1}(\beta)\Big)+\sum_{(I^s_{t+1}=0)}\Big(1-\Gamma\big(\Psi_{t+h}(\beta)\big)\Big)-\lambda\Big((1-\alpha)\sum_{j=1}^{k}\frac{\beta_j^2}{2}+\alpha\sum_{j=1}^{k}|\beta_j|\Big)\right\}$$

where λ and α are the elastic net tuning parameters (Zou and Hastie, 2005). In this study, the penalty factor $\alpha = 0.5$ was employed, which results to an elastic net probit model.

To choose the tuning parameters λ, λ_1, λ_2 in Ridge, LASSO and Elastic Net, we need a validation set in which the predictive value of the specific penalized binary probit model could be compared for various values of the tuning parameter, and the optimal tuning parameter should be chosen such that the error rate is minimal. In this study, the best tuning parameter employing cross-validation was chosen for each model.

2.3.4 Classification and Regression Trees for Stock Return Predictability

The concept of classification and regression trees (CART) was first introduced by Breiman et al. (1984), which involves the use of decision tree learning procedures to build a model that can predict the value of a target variable based on several input variables. There are many classification algorithms, including decision trees, rule-based learners, exemplar learners, discriminant functions, neural networks and Bayesian networks, that are considered to be useful in modern forecasting. There are also ways of combining them into ensemble classifiers, such as bagging, boosting, and random forest. The consistent CART models in this study are as follows:

2.3.4.1 Bagging

Bagging or bootstrap aggregating was introduced in 1994 by Breiman (1996) to improve classification by combining classifications of randomly generated training datasets, to reduce the biases and variances in a tree-based analysis. Bagging implies fitting a model, including all potential points on the original training set. It appears to effectively remove the instability of a decision rule by averaging across resamples and to avoid overfitting (Zheng, 2006).

Let $\mathbf{S} = \{(Z_1, y_1), (Z_2, y_2), ..., (Z_t, y_t), ..., (Z_T, y_T)\}$ denote the training sample, where T is the number of observations in the training sample, $\mathbf{Z}_t$ is a vector of k covariates, and $y_t \in \{-1, 1\}$ indicates a positive or negative return for each t. The classification into one of the two groups is defined as follows:

$$\hat{\Psi}(\mathbf{Z}) = sign\Big(\hat{\delta}(\mathbf{Z}_t) - \tau_B\Big), \hat{\Psi}(\mathbf{Z}) \in \{-1, 1\}$$

where τ_B is the threshold (cut-off value); $\hat{\delta}(\mathbf{Z}_t)$ is the base classifier that learned the covariates in the training sample; $\hat{\delta}(\mathbf{Z}_t) > \tau_B$ implies a positive return classification, while $\hat{\delta}(\mathbf{Z}_t) < \tau_B$ implies a negative return classification (Lemmens and Croux, 2006).

The decision tree classification score is given by the following:

$$\hat{\delta}(\mathbf{Z}) = 2\hat{\rho}(\mathbf{Z}) - 1$$

where $\hat{\rho}(\mathbf{Z})$ is the predicted probability of a positive return estimated by the tree. For each bootstrap sample $S_b^*, b = 1, 2, ..., B$, a classifier can be estimated assigning B score functions $\hat{\delta}_1^*(\mathbf{Z}), \hat{\delta}_2^*(\mathbf{Z}), ..., \hat{\delta}_b^*(\mathbf{Z}), ..., \hat{\delta}_B^*(\mathbf{Z})$.

These functions are afterwards aggregated into a score, as follows:

$$\hat{\delta}_{bag}(\mathbf{Z}) = \frac{1}{B}\sum_{b=1}^{B} \hat{\delta}_b(\mathbf{Z})$$

Thus, the final classification is obtained as follows:

$$\hat{\Psi}_{bag}(\mathbf{Z}) = sign\Big(\hat{\delta}_{bag}(\mathbf{Z}) - \tau_B\Big), \hat{\Psi}(\mathbf{Z}) \in \{-1, 1\} \tag{2.14}$$

2.3.4.2 Random Forest

A random forest (RF) classifier, see Breiman (2001), is a specific type of bootstrap aggregating based on a random subset of the input features (Ballings et al., 2015; Kumar and Thenmozhi, 2006; Creamer, 2009). A random forest classifier consists of an ensemble classification algorithm that involves the use of trees as base classifiers. It consists of a combination of classifiers in which each classifier contributes an individual vote for assigning the most frequent class to the input vector $\mathbf{Z}$, defined by the following:

$$\delta_{RF}^{\hat{B}} = majority\ vote \left\{\hat{\delta}_b(\mathbf{Z})\right\}_{b=1}^{B} \tag{2.15}$$

where $\hat{\delta}_b(\mathbf{Z})$ is the class prediction of the b^{th} random forest tree; $\mathbf{Z}$ is the input vector; $b = 1, 2, ..., B$.

The Gini index approach suggested by Breiman et al. (1984) is a suitable measure for selecting the best splits which determines the impurity of a given element with respect to the classes, and hence, it is employed for selecting the best split at each node.

Given a training dataset $\mathbf{S}$, involving a set of covariates and categorical target outcome, the Gini index can be computed as follows:

$$I(\tau) = \sum_{i}\sum_{j\neq i}\frac{h(\delta_i, \mathbf{S})}{|\mathbf{S})|}\frac{h(\delta_j, \mathbf{S}))}{|\mathbf{S})|} \tag{2.16}$$

where $\dfrac{h(\delta_i, \mathbf{S}))}{|\mathbf{S})|}$ is the probability that a selected instance belongs to class δ_i; $\dfrac{h(\delta_j, \mathbf{S}))}{|\mathbf{S})|}$ is the probability that a selected instance belongs to class δ_j; for $i \neq j$ (Rodriguez-Galiano et al., 2012). Thus the random forest seek to produce a measure of proximity between each pair of instances in the classification tree.

Alternatively, for a given node τ with estimated class probabilities $Prob(j|\tau), j = 1, ..., J$, the node impurity, $I(\tau)$, employing the Gini index is defined as follows:

$$I(\tau) = \sum_{j \neq i}^{J} Prob(j|\tau)Prob(i|\tau). \tag{2.17}$$

The Gini index is minimised when the node is pure (homogeneous) with respect to one of the classes.

2.3.4.3 Conditional Inference Tree

The conditional inference tree (CTree) enables the use of recursive partitioning and tree-structured models in a conditional inference framework. The use of the Gini index to determine the most favourable split induces a selection bias toward covariates with many possible splits and also cannot distinguish between a significant and an insignificant improvement in the information measure. Hothorn et al. (2006) proposed the conditional inference approach tree where the node split is selected based on how good the association is between the response and the covariates. The resulting nodes should provide a high association between the response and the covariates. The significance of the association is investigated by a χ^2 test and the covariate with highest association is selected for splitting. Moreover, multiple test procedures are applied to determine whether no significant association between any of the covariates and the response can be stated and the recursion needs to stop.

In more detail, let $\mathbf{Z} = (Z_1, \cdots, Z_k)$ be the k-dimensional vector of covariates and let $\mathbf{y}$ be a categorical response variable. $\mathbf{Z}$ is taken from a sample space $\mathcal{Z} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_k$. We assume that the conditional distribution of $\mathbf{y}$ given $\mathbf{Z}$ depends on the function f of $\mathbf{Z}$ as follows:

$$D(\mathbf{y}|\mathbf{Z}) = D(\mathbf{y}|Z_1, \cdots, Z_k) = D(\mathbf{y}|f(Z_1, \cdots, Z_k)).$$

Thus, a generic algorithm for recursive binary partitioning for a given learning sample

$$\mathcal{L}_n = (\mathbf{y}_i, Z_{1i}, \cdots, Z_{ki}),\ i = 1, \cdots, n,$$

can be formulated using non-negative integer valued case weights $\mathbf{w} = (w_1, \cdots, w_n)$.

Each node of the tree is represented by a vector of case weights having nonzero elements when the corresponding observations are elements of the node, and are zero otherwise. The following steps implement recursive binary partitioning:

1. Test the global null hypothesis of independence between any covariate $\mathbf{Z}$ and the categorical response variable $\mathbf{y}$ for case weights w. Stop if this hypothesis cannot be rejected. Otherwise, select the j-th covariate Z_j with the strongest association to $\mathbf{y}$.

2. Choose a set $A \subset \mathcal{Z}_j$ to split $\mathcal{Z}_j$ into two disjoint sets of A and A^c. The case weights w_{left} and w_{right} determine the two subgroups with $w_{left,i} = w_i I(Z_{j,i} \in A)$ and $w_{right,i} = w_i I(Z_{j,i} \notin A)$, for all $i = 1, 2, \cdots, m$, where $I(\cdot)$ is the indicator function.

3. Repeat steps 1 and 2 recursively with the different case weights w_{left} and w_{right}, respectively.

2.3.4.4 Conditional Inference Forest

Random forest has been criticised for the bias that results from favouring covariates with many split-points. The conditional inference forest (CForest) is known to correct this bias by separating the procedure for the best covariate to split on from that of the best split point search for the selected covariate. The conditional inference forest is an implementation of the random forest and bootstrap aggregating ensemble algorithms, utilising conditional inference trees as base learners. To determine the variable importance in conditional inference forests, the vector of the predictor variables is randomly permuted and the initial association with the response variable is broken. When the permuted and the non-permuted variables are used to predict the response variable for the out of bag observations, the classification accuracy decreases substantially if the permuted variable is associated with the response. Hence, the variable importance is the difference in the prediction accuracy before and after permutation of the variable average over all trees (Strobl et al., 2008; Das et al., 2009).

Given the out of bag sample $B(\tau)$ for a tree $\tau \in 1, ..., ntree$. The variable importance of a single tree is defined as follows:

$$Var^{Imp}(\tau; Z_j) = \frac{\sum_{t \in B(\tau)} Imp\Big(y_i = \delta(\tau; Z_j)\Big)}{|B(\tau)|} - \frac{\sum_{t \in B(\tau)} Imp\Big(y_i = \delta(\tau; Z_j, \eta_j)\Big)}{|B(\tau)|} \tag{2.18}$$

where Z_j is the j^{th} input or predictor variable; y_i is the response variable at observation i; $\delta(\tau; Z_j)$ represent the predicted classes before the permuting process; $\delta(\tau; Z_j, \eta_j)$ represent the predicted classes after the permuting process (Strobl et al., 2008).

The raw variable importance score for each of the input variables is the mean importance over all trees $\tau \in 1, ..., ntree$, and can be computed as follows:

$$Var^{Imp}(Z_j) = \frac{\sum_{\tau=1}^{ntree} Var^{Imp}(\tau; Z_j)}{ntree} \tag{2.19}$$

where $Var^{Imp}(\tau; Z_j)$ are the individual importance scores, computed from $ntree$ independent bootstrap samples.

That is, the variable importance of any variable is the difference in the prediction accuracy before and after the permuting process of the variable, averaged over all τ trees.

2.3.4.5 Adaptive Boosting

Boosting is an ensemble technique aimed at increasing the strength of a weak learning classifier by improving its accuracy. A boosting algorithm, as proposed by Schapire (1990), seeks to convert a weak learner into a strong learner. The

principle consists of sequentially applying the classifier to adaptively re-weighted versions of the initial dataset $\mathbf{S}_b^*, b = 1, 2, \cdots, B$. In each step, the learning attention is focused on modified versions of the data, where the modifications give more weight, w_t, to misclassified points. Once the process has finished, the single classifiers obtained are combined into a final classifier by weighted majority vote. In boosting, the predictors are made sequentially rather than independently.
For a real adaptive boosting (AdaBoost), the classification score is defined as follows:

$$\hat{\delta}_b(\mathbf{Z}_t) = \frac{1}{2}\ln\Big(\frac{\hat{\rho}_b^*(\mathbf{Z}_t)}{1-\hat{\rho}_b^*(\mathbf{Z}_t)}\Big)$$

where $\hat{\rho}_b^*(\mathbf{Z}_t)$ is the estimated probability in step b, for $t = 1, 2, ..., T$.
From the weights $w_{t,1} = \frac{1}{T}$, for $t = 1, 2, ..., T$; the weights for the next step $b+1$ are updated as follows:

$$w_{i,b+1} = w_{i,b}\exp(-y_t\hat{\delta}_b(\mathbf{Z}_t)), \quad t = 1, 2, ..., T$$

where $\sum_{t=1}^{T} w_{t,b} = 1$ and the corresponding probability estimate for the iteration $b+1$ becomes $\hat{\rho}_{b+1}^*(\mathbf{Z}_t)$ (Hofner et al., 2014; Lai et al., 2009).
The procedure is repeated for $b = 1, 2, ..., B$ until the final prediction is obtained as follows:

$$\hat{\Psi}_{boost}(\mathbf{Z}) = sign\Big[\sum_{b=1}^{B}\hat{\delta}_b(\mathbf{Z}) - \tau_B\Big], \hat{\Psi}_{boost}(\mathbf{Z}) \in \{-1, 1\} \tag{2.20}$$

where the threshold τ_B is a correction term for balanced training sample, which could be zero ($\tau_B = 0$) when a proportion sample is used (Lemmens and Croux, 2006).

Given the test set $\{(Z_1, y_1), (Z_2, y_2), ..., (Z_t, y_t), ..., (Z_T, Z_T)\}$ with individual classifier scores $\hat{\delta}(\mathbf{z}_t)$ and the final classification score $\hat{\Psi}(\mathbf{Z}_t)$ for $t = 1, 2, ..., T$;

then the error rate will be computed as

$$ER = \frac{1}{T}\sum_{t=1}^{T} I\left[\hat{\Psi}(\mathbf{z}_t) \neq y_t\right]$$

where T is the size of the test sample; $I[\cdot]$ is a binary indicator function.

The main steps of the Adaboost algorithm are as follows (Freund and Schapire, 1996; Alfaro et al., 2013):

1. Initialize the observation weights $w_t = \dfrac{1}{T}$ for $t = 1, 2, \cdots, T$.

2. For $b = 1, 2, \cdots, B$:

   (a) Fit a classifier $\hat{\delta}_b(\mathbf{Z})$ to the training data using observation weights w_t.

   (b) Compute the weighted misclassification error for $\hat{\delta}_b(\mathbf{Z})$:
   $err_b = \frac{\sum_{t=1}^{T} w_t I[y_t \neq \hat{\delta}_b(\mathbf{Z}_t)]}{\sum_{t=1}^{T} w_t}$

   (c) Compute $\alpha_b = \frac{1}{2} ln[\frac{1-err_b}{err_b}]$, where α_b is the weight score for $b = 1, 2, ..., B$

   (d) Update the weights $w_t \leftarrow w_t exp(\alpha_b I[y_t \neq \hat{\delta}_b(\mathbf{Z}_t)])$, for $t = 1, 2, \cdots, T$ and normalize them.

3. Output the final classifier $\hat{\Psi}_{boost}(\mathbf{Z}) = sign\left[\sum_{b=1}^{B} \alpha_b \hat{\delta}_b(\mathbf{Z})\right]$, $\hat{\Psi}_{boost}(\mathbf{Z}) \in \{-1, 1\}$.

Other boosted tree models used in this research include the gradient boosting machine (GBM), the generalized linear boosting model (GLMBoost) and the LogitBoost model.

2.3.4.6 Gradient Boosting

Gradient boosting (GBM) seeks to generate a prediction model in the form of an ensemble of weak learners such as decision trees, in order to minimize the resulting classification error.

Let $\mathbf{Z} = \{Z_1, Z_2, ..., Z_k\}$ be k-dimensional set of predictors with target output variable $\mathbf{y} \in \{-1, 1\}$, and a collection of L instances in the form $\{(\mathbf{y}_\ell, \mathbf{Z}_\ell); \ell = 1, 2, ..., L\}$. Then we can model a learning prediction function $\hat{\delta}(\mathbf{Z}) : Z \longrightarrow \mathbf{y}$ that minimizes the expectation of the loss function $B_{loss}(\mathbf{y}, \delta)$ over the joint distribution of all ordered pair $(\mathbf{y}, \mathbf{Z})$. The predictive classifier function is defined as follows:

$$\hat{\delta}(\mathbf{Z}) = \underset{\delta(\mathbf{Z})}{\operatorname{argmin}} E[\mathbf{y}, \mathbf{Z}] B_{loss}\Big(y, \delta(\mathbf{Z})\Big)$$

where $E[\mathbf{y}, \mathbf{Z}]$ is the joint expectation of the input vector $\mathbf{Z}$ and the target output $\mathbf{y}$; $B_{loss}\Big(\mathbf{y}, \delta(\mathbf{Z})\Big)$ is the loss function.
The conditional expectation of $\mathbf{y}$ given $\mathbf{Z}$ is as follows:

$$E[\mathbf{y}|\mathbf{Z}] = \delta(\mathbf{Z}) = \sum_{j=1}^{k} \theta_j Z_j = \sum_{j=1}^{k} \delta_j(Z_j)$$

where $\delta_1(Z_1), \delta_2(Z_2), ..., \delta_k(Z_k)$ are smooth functions.
We can extend the classifier function by introducing additive model with functions $\delta_j(Z_j), j = 1, 2, ...k$ of all the input variables, defined as follows:

$$\delta(\mathbf{Z}) = \sum_{j=1}^{k} \delta_j(Z_j) = \sum_{j=1}^{k} \theta_j \Gamma(Z_j; \alpha_j) \tag{2.21}$$

where $\theta_j \Gamma(Z_j; \alpha_j)$ is a weak learner characterized by a parameter vector $\alpha = (\alpha_1, \alpha_2, ..., \alpha_k)$ and a vector of multiplier $\theta = (\theta_1, \theta_2, ..., \theta_k)$; $\delta_j(Z_j)$ is the weighted majority vote of the individual weak learners (Guelman, 2012; Son et al., 2015). Thus, the resulting objective function can be minimized as follows:

$$\min_{\{\theta_j, \alpha_j\}_{j=1}^{k}} \sum_{\ell}^{L} B_{loss}\Big(y_i, \sum_{j=1}^{k} \theta_j \Gamma(\mathbf{Z}_\ell; \alpha_j)\Big) \tag{2.22}$$

where $B_{loss}\Big(\mathbf{y}, \delta(\mathbf{Z})\Big)$ is the chosen loss function required to estimate a lack of fit.

Friedman (2001, 2002) laid the groundwork for a new generation of boosting algorithms. Assume that we are interested in modelling $Pr(\mathbf{y} = 1|\mathbf{Z} = Z)$ for a

Bernoulli response variable. The idea is to fit a model of the following form:

$$\lambda(\mathbf{Z}) = G_B(\mathbf{Z}) = \sum_{b=1}^{B} g_b(\mathbf{Z}; \gamma_b)$$

where

$$\lambda(\mathbf{Z}) = log\left(\frac{Pr(\mathbf{y} = 1|\mathbf{Z} = Z)}{Pr(\mathbf{y} = 0|\mathbf{Z} = Z)}\right)$$

and γ_b is parameter vector, which for the trees, captures the identity of the split variables, their split values and the constants in the terminal nodes.

The main steps of the gradient boosting algorithm are as follows:

1. Start with $\hat{G}_0(\mathbf{Z}) = 0$, and set the shrinkage parameter $\epsilon > 0$.

2. For $b = 1, 2, \cdots, B$:

   (a) Compute the pointwise negative gradient of the loss function at the current fit as follows:
   $r_t = -\frac{\partial L(y_t, \lambda_t)}{\partial \lambda_t}$

   (b) Approximate the negative gradient by a depth-d tree by solving the following:
   $minimise_\gamma \sum_{t=1}^{T}(r_t - g_b(\mathbf{Z}; \gamma_b))^2$.

   (c) Update $\hat{G}_b(\mathbf{Z}) = \hat{G}_{b-1}(\mathbf{Z}) + \hat{g}_b(\mathbf{Z})$, with $\hat{g}_b(\mathbf{Z}) = \epsilon g(\mathbf{Z}; \hat{\gamma}_b)$.

3. Return the sequence $\hat{G}_b(\mathbf{Z})$, for $b = 1, 2, \cdots, B$.

2.3.4.7 Generalized Linear Boosting

The generalized linear boosting (GLMBoost) fits a tree based model using a boosting algorithm as opposed to maximum likelihood estimation, which trains the data with best cross-valided mstop tuning parameter, performs variable selection and predict future classes. The GLMBoost employs component-wise (generalised) linear models as base-learners (Bühlmann and Yu, 2003; Bühlmann et al., 2007). Let $\mathbf{Z} = (Z_1, Z_2, ..., Z_k)'$ be k-dimensional vector of covariates, from which the categorical binary response variable $y_i \in \{1, ..., c\}$ can be predicted. Then a generalized linear model can be fitted as follows:

$$\ell(\hat{\mu}) = \beta_0 + \beta_1 Z_1 + ... + \beta_k Z_k \tag{2.23}$$

where $\hat{\mu} = E(\mathbf{y}|\mathbf{Z})$ is the conditional expectation of the binary response; ℓ is the link function; β is a vector of unknown parameters.
The boosted generalized linear model additionally performs variable selection and the effects are shrunken toward zero if early stopping (mstop) is applied in the model (Hofner et al., 2014; Alfaro et al., 2013). The GLMBoost fits simple linear models separately for each column of the design matrix to the negative gradient vector, for each boosting iterations, using the best fitting base-learner in the update step.

2.3.4.8 LogitBoost

The LogitBoost is an algorithm used to produce a logistic regression model at every node in the classification tree and each node is able to be split using a suitable splitting criterion (Friedman et al., 2000; Landwehr et al., 2005). It is designed to train the classification algorithm using stumps or one node decision trees as weak learners.
Let $\{(y_i, \mathbf{Z}_i)\}_{i=1}^{N}$ be input dataset set with N samples, $\mathbf{Z}_i \in \mathbf{Z}$, $y_i \in \mathbf{y} \in \{-1, 1\}$. The binomial log-likelihood loss function of a binary logitboost is defined as follows:

$$B_{loss}(\delta) = E\Big[-log(1 + e^{\mathbf{y}\delta(\mathbf{Z})})\Big]$$

which varies directly with the classification error and appears to be less sensitive to noise and outliers.
The weight w_i and the working respond $\mathbf{Z}_i$ in each of the Newton iteration steps

$\ell = 1, 2, ..., L$ is defined as follows:

$$\begin{cases} w_i = Prob(\mathbf{Z}_i)(1 - Prob(\mathbf{Z}_i)) \\ \mathbf{Z}_i = \frac{\frac{(y_i+1)}{2} - Prob(\mathbf{Z}_i)}{w_i} \quad \text{for} \quad i = 1, 2, ..., N \end{cases}$$

with initial values $w_i = \frac{1}{N}$; $Prob(\mathbf{Z}) = Prob(\mathbf{y} = 1|\mathbf{Z}) = \frac{1}{2}$ and $\delta(\mathbf{Z}) = 0$ (Kamarudin et al., 2017; Qi et al., 2011).

We then fit the function $f_\ell(\mathbf{Z})$ by a weighted least squares regression of y_i to $\mathbf{Z}_i$ using weights w_i, and thereafter we update the committee function and the corresponding probability based on the following:

$$\begin{cases} \delta(\mathbf{Z}) = \delta(\mathbf{Z}) + \frac{1}{2} f_\ell(\mathbf{Z}) \\ Prob(\mathbf{Z}) = \frac{e^{\delta(\mathbf{Z})}}{e^{\delta(\mathbf{Z})} + e^{-\delta(\mathbf{Z})}} \end{cases}$$

when all the iterations are exhausted then the model becomes:

$$\delta(\mathbf{Z}) = \frac{1}{2} \sum_{\ell=0}^{L} f_\ell(\mathbf{Z}) = \frac{1}{2}\{0 + f_1(\mathbf{Z}) + f_2(\mathbf{Z}) + ... + f_L(\mathbf{Z})\}$$

and the overall classifier is the resulting decision function:

$$\Psi(\mathbf{Z}) = sign\{\delta(\mathbf{Z})\} = \begin{cases} 1 \quad \text{if} \quad \delta(\mathbf{Z}) > 0 \quad \Longrightarrow \Psi(\mathbf{Z}) \quad \text{belongs to class 1} \\ -1 \quad \text{if} \quad \delta(\mathbf{Z}) \leq 0 \quad \Longrightarrow \Psi(\mathbf{Z}) \quad \text{belongs to class 2} \end{cases} \tag{2.24}$$

(Li, 2012; Feng et al., 2005).

Thus, the Newton steps for optimization of the loss function seeks to build a robust classifier by iteratively adding a weak classifier to improve the classification process.

Alternatively, let $\{(y_i, \mathbf{Z}_i)\}_{i=1}^{N}$ be the input dataset with N samples, $\mathbf{Z}_i \in \mathbf{Z}$, $y_i \in \mathbf{y} \in \{-1, 1\}$, and use the transformation $\mathbf{y}^* = \frac{1+\mathbf{y}}{2}$ to represent the outcome with a 0/1 response, and represent the probability of $\mathbf{y}^* = 1$ with $Prob(\mathbf{Z})$ where

$$Prob(\mathbf{Z}) = \frac{e^{F(\mathbf{Z})}}{e^{F(\mathbf{Z})} + e^{-F(\mathbf{Z})}}.$$

The main steps of the LogitBoost algorithm are as follows:

1. Start with $w_t = 1/T, t = 1, \cdot, T$, $F(\mathbf{Z}) = 0$, and probability estimates $Prob(\mathbf{Z}_i) = \frac{1}{2}$.

2. For $b = 1, 2, \cdots, B$:

   (a) Compute the working response r_i and the weights w_i as follows

   $$\begin{cases} w_i = Prob(\mathbf{Z}_i)(1 - Prob(\mathbf{Z}_i)) \\ r_i = \frac{\mathbf{y}^* - Prob(\mathbf{Z}_i)}{w_i} \end{cases}$$

   (b) Fit the function $f_b(\mathbf{Z})$ by a weighted least-squares regression of r_i to $\mathbf{Z}_i$ using weights w_i.

   (c) Update $F(\mathbf{Z}) \leftarrow F(\mathbf{Z}) + \frac{1}{2} f_b(\mathbf{Z})$, and $Prob(\mathbf{Z}) = \frac{e^{F(\mathbf{Z})}}{e^{F(\mathbf{Z})} + e^{-F(\mathbf{Z})}}$.

3. Return the classifier $sign\left[F(\mathbf{Z})\right] = sign\left[\sum_{b=1}^{B} f_b(\mathbf{Z})\right]$, for $b = 1, 2, \cdots, B$.

2.3.4.9 Recursive Partitioning Algorithm

The recursive partitioning (RPart) algorithm builds a decision tree that attempt to correctly classify elements of the set by splitting it into subsets based on several features. The splitting process continues indefinitely, resulting in newer subsamples and terminates after a specific stopping criterion is attained (Cook and Goldman, 1984).

Let y_t be a conditionally distributed dichotomous response variable given the k predictors, such that the k predictors are elements of a sample space $\Omega = \Omega_1 \times \Omega_2 \times .... \times \Omega_k$. Then, by tree-structured recursive partitioning, the conditional distribution of y_t given $\mathbf{Z}_{t-1}$ depends on the function:

$$\Psi(y_t|\mathbf{Z}_{t-1}) = \Psi(y_t|g(Z_{(t-1)1}, Z_{(t-1)2}, ..., Z_{(t-1)k})) \tag{2.25}$$

from which the p disjoint cells $B_1, B_2, ..., B_p$ partitioning the predictor space

$$\Omega = B_1 \cup B_2 \cup ..... \cup B_p = \cup_{j=1}^{p} B_j$$

are obtained; where $g(\cdot)$ is a function of the k predictors (Hothorn et al., 2006). The fitted model is based on a learning sample with some missing predictors $\mathbf{Z}_{t-1}$, defined by the following:

$$\ell_T = \{y_t; Z_{(t-1)1}, Z_{(t-1)2}, ..., Z_{(t-1)k}; t = 1, 2, ..., T\} \tag{2.26}$$

The recursive algorithm proposed by Zeileis et al. (2008), Hothorn et al. (2006) is as follows:

1. Fit the model to all observations at once in the initial node and estimate the unknown parameters by minimizing the objective function;
2. Evaluate the stability or instability of the estimated parameters with respect to the ordering features;
3. Determine the splitting point that locally optimizes the objective function using a fixed or adaptive number of splits;
4. Split the node into sub-nodes and repeat the procedure recursively until no further splitting is feasible.

2.3.4.10 Linear Discriminant Analysis

The discriminant function was first introduced by Fisher (1936). Linear discriminant analysis (LDA) uses Bayes' theorem to estimate output class probabilities given the input features, using the assumptions that the input data $\mathbf{Z} = (Z_1, Z_2, \cdots, Z_k)$ follow a multivariate Gaussian distribution with a class specific mean vector μ_c and a common covariance matrix $\mathbf{S}_c = \mathbf{S}$ for all c. If $f_c(\mathbf{Z})$ is the class conditional density of the covariates $\mathbf{Z}$, in class $\mathbf{y} = c$, i.e., $f_c(Z) = Prob(\mathbf{Z} = Z|\mathbf{y} = c)$, and ψ_c is the prior probability of class c, then by Bayes' theorem, the class posterior probability is given by the following:

$$Prob(\mathbf{y} = c|\mathbf{Z} = Z) = \frac{f_c(\mathbf{Z})\psi_c}{\sum_{c=1}^{C} f_c(\mathbf{Z})\psi_c}, \quad for \quad c = 1, 2, \cdots, C$$

and $\mathbf{Z}$ has a multivariate Gaussian density for each class given by the following:

$$f_c(\mathbf{Z}) = (2\pi)^{-\frac{p}{2}}|\mathbf{S}_c|^{-\frac{1}{2}} exp\Big(- \frac{1}{2}(\mathbf{Z} - \mu_c)'\mathbf{S}_c^{-1}(\mathbf{Z} - \mu_c)\Big).$$

The LDA classifier assigns an observation given by $\mathbf{Z} = Z$ to the class c given by the following:

$$\Psi_c^{LDA}(\mathbf{Z}) = argmax_c \left\{ \mathbf{Z}'\mathbf{S}^{-1}\mu_c - \frac{1}{2}\mu_c'\mathbf{S}^{-1}\mu_c + log\psi_c \right\}. \tag{2.27}$$

For a proof of the above equation, see (James et al., 2013). The word linear in the LDA classifier stems from the fact that the discriminant function is a linear function of the input features $\mathbf{Z}$.

2.3.4.11 Quadratic Discriminant Analysis

The quadratic discriminant analysis (QDA) classifier separates multi-class measurements by a quadratic surface. Unlike LDA, in the case of the QDA classifier, the input features in each class follow a multivariate Gaussian distribution with a class specific mean vector μ_c and a class specific covariance matrix $\mathbf{S}_c$, owing to the heterogeneity of variance-covariance matrices for the various classes (James et al., 2013; Ou and Wang, 2009).

The QDA classifier is given by the following:

$$\Psi_c^{QDA}(\mathbf{Z}) = argmax_c\Big(\Omega_c(\mathbf{Z})\Big) = argmax_c\left\{-\frac{1}{2}log|\mathbf{S}_c| - \frac{1}{2}(\mathbf{Z}-\mu_c)'\mathbf{S}_c^{-1}(\mathbf{Z}-\mu_c) + log\psi_c\right\}. \tag{2.28}$$

The QDA classifier obtains its name from the fact that the QDA discriminant function is a quadratic function of the input features $\mathbf{Z}$.

2.3.4.12 Regularized Discriminant Analysis

The regularized discriminant analysis (RDA) introduces regularization into the estimates of the covariance matrices and allows the shrinkage of the separate covariance matrices of QDA toward a common covariance, as in LDA. In this sense, RDA is a compromise between LDA and QDA. The regularized covariance matrices have the form:

$$\mathbf{S}_c(\lambda) = \lambda\mathbf{S}_c + (1-\lambda)\mathbf{S}$$

where $\mathbf{S}$ is the pooled covariance matrix used in the LDA; $\mathbf{S}_c$ is the class specific covariance matrix of the input features used in the QDA; and λ is a non-negative tuning parameter that controls the degree of shrinkage of the individual class covariance matrix estimates toward the pooled estimates. Here, $\lambda \in [0,1]$ allows a continuum of models between LDA and QDA and needs to be specified. In practice, λ can be chosen employing cross-validation. Biasing the class covariance matrices toward commonality is not the only way to shrink them. An additional convex combination allows $\mathbf{S}_c$ itself to be shrunk toward a scaled identity matrix, using the shrinkage parameter γ as follows:

$$\mathbf{S}_c(\lambda,\gamma) = (1-\gamma)\mathbf{S}_c(\lambda) + \gamma\frac{1}{d}\,\mathrm{tr}[\mathbf{S}_c(\lambda)]I$$

where $\frac{1}{d}\,\mathrm{tr}[\mathbf{S}_c(\lambda)]$ is the mean of the diagonal elements of $\mathbf{S}_c, (\lambda)$, so it is the mean variance of the class input features. The RDA classifier is given by the following:

$$\Psi_c^{RDA}(\mathbf{Z}) = \big\{(\mathbf{Z}-\bar{\mathbf{Z}})'\mathbf{S}_c^{-1}(\lambda,\gamma)(\mathbf{Z}-\bar{\mathbf{Z}}_k) + log|\mathbf{S}_c(\lambda,\gamma)|\big\} \tag{2.29}$$

where λ is the cross-validated parameter that controls the degree of shrinkage of the individual class covariance matrix estimates toward the pooled estimates and γ is an additional regularization parameter that controls shrinkage toward a multiple of the identity matrix for a given value of λ (Friedman, 1989).

2.3.4.13 Heteroscedastic Discriminant Analysis

The heteroscedastic discriminant analysis (HDA) is a generalized method of the LDA in that its feature space transformation does not require the imposition of equal within-class covariance assumptions as compared to the standard LDA. The HDA classifier is capable of handling different covariance structures of the class distributions (Kumar and Andreou, 1998).

Let $\left\{\mathbf{Z}_i\right\}_{i=1}^{N}$ denote a sequence of k-dimensional feature vectors, with each vector belonging to a single class $j \in \{1, ..., C\}$, and let $\mathbf{y}$ denote a categorical response variable. If N_j, μ_j and Σ_j represent the sample count, mean and covariance, respectively, of the j^{th} class, then the between-class matrix $\mathbf{M}$ can be extracted in the following form:

$$\mathbf{M} = \frac{1}{N}\sum_{j=1}^{C} N_j \mu_j \mu_j' - \mu\mu'$$

where μ_j' is the transpose of μ_j of the j^{th} class; μ is a vector of overall means. The HDA objective function seeks to find a projection matrix, denoted by β, that maximizes the likelihood in the Jacobian transformation space $\mathbf{y} = \beta'\mathbf{Z}$ under the normality assumption, such that the ratio of the determinants:

$$\Omega(\beta) = \frac{|\beta\mathbf{M}\beta'|^{N}}{\prod_{j=1}^{C} |\beta\Sigma_j\beta'|^{N_j}} \tag{2.30}$$

is maximized, where β' is the transpose of β (Huang et al., 2000; Szepannek et al., 2009).

The HDA classifier is then given by the following:

$$\Psi^{HDA}(\beta) = \underset{\beta}{\text{argmax}}\, log\Big\{\Omega(\beta)\Big\} = \underset{\beta}{\text{argmax}}\,\Big\{\sum_{j=1}^{C} -N_j log|\beta\Sigma_j\beta'| + N log|\beta\mathbf{M}\beta'|\Big\} \tag{2.31}$$

where $\mathbf{M}$ is the between-class matrix. See Kumar and Andreou (1998) for further details.

2.3.4.14 Sparse Discriminant Analysis

The sparse LDA introduces projection techniques that imposes zero entries in the feature matrix, aimed at reducing the dimensionality to produce a final parsimonious model. The sparse discriminant function involves the inclusion of an ℓ_1 penalty norm in the optimal scoring problem which results in the optimization problem, follows:

$$max_{\beta_j}\beta_j'\mathbf{S}\beta_j - \eta\beta_{j_1} \text{ subject to } \beta_j'(\mathbf{S}_w + \Omega)\beta_j = 1, \quad \beta_j'(\mathbf{S}_w + \Omega)\beta_m = 0 \text{ for all } m < j \tag{2.32}$$

where β_j is the discriminant vector of class j, Ω is a positive definite matrix; $\mathbf{S}_w$ is the within class covariance matrix. The j^{th} sparse discriminant analysis solution pair (θ_j, β_j) is obtained by solving the problem, as follows:

$$min_{\beta_j,\theta_j}\Big\{\mathbf{y}\theta_j - \mathbf{Z}\beta_j{}^2 + \eta\beta_j'\Omega\beta_j + \lambda\beta_{j_1}\Big\} \tag{2.33}$$

$$\text{subject to} \quad \frac{1}{n}\theta_j'\mathbf{y}'\mathbf{y}\theta_j = 1 \quad \text{and} \quad \theta'\mathbf{y}'\mathbf{y}\theta_m = 0 \quad \text{for all} \quad m < j$$

and the sparse LDA is as follows:

$$\Psi^{SparseLDA}(\theta, \beta) = \underset{\beta_j,\theta_j}{\text{argmin}}\Big\{\frac{1}{n}\|\mathbf{y}\theta_j - \mathbf{Z}\beta_j\|^2 + \eta\beta_j'\Omega\beta_j + \lambda\|\beta_j\|_1\Big\} \tag{2.34}$$

where $\mathbf{y}$ is a vector of dummy variables for the j^{th} classes; θ_j is a j-vector of scores; n is the sample size; η and λ are non-negative tuning parameters (Clemmensen et al., 2011). Thus, the ℓ_1 penalty norm on β_j results in sparsity when the tuning parameter λ is large.

2.3.4.15 High Dimensional Discriminant Analysis

The high dimensional discriminant analysis (HDDA) is another important extension of the LDA most feasible for a dimensionality reduction model involving many features as compared to the sample size, and in which the LDA is weak in performance. Let Γ_i be an orthogonal matrix of eigenvectors of a covariance matrix $\mathbf{S}_i$; let Φ_i be the basis from the eigenvectors of $\mathbf{S}_i$, and assuming the class conditional densities follows Gaussian $\mathcal{N}(\mu_i, \mathbf{S}_i)$ for all $i = 1, ..., c$. Then, the class conditional covariance matrix Ω_i, is defined by the following:

$$\Omega_i = \Gamma_i' \mathbf{S}_i \Gamma_i$$

where Ω_i is diagonal matrix with two distinct eigenvectors u_i and v_i, $u_i > v_i$.
If $\Pi_i(\mathbf{Z}) = \hat{\Gamma}_i \hat{\Gamma}_i'(\mathbf{Z} - \mu_i) + \mu_i$ represents the projection of the input vector $\mathbf{Z}$ on the affine space $_i$, then the cost function will be as follows:

$$\theta_i(\mathbf{Z}) = \frac{\|\mu_i - \Pi_i(\mathbf{Z})\|^2}{u_i} + \frac{\|\mathbf{Z} - \Pi_i(\mathbf{Z})\|^2}{v_i} + d_i \ln u_i + (k - d_i) \ln v_i - 2 \ln \pi_i \quad (2.35)$$

where $u_i = \dfrac{\sigma_i^2}{\alpha_i}$ and $v_i = \dfrac{\sigma_i^2}{1 - \alpha_i}$ with $\alpha_i \in \{0, 1\}$ and $\sigma_i > 0$ for all $i = 1, ..., c$; k is the k-dimensional input vector; d_i is the i^{th} diagonal of Γ_i (Bouveyron et al., 2007).
The posterior probability is defined as follows:

$$Prob(i|\mathbf{Z}) = \frac{e^{-\frac{1}{2}\theta_i(\mathbf{Z})}}{\sum_{j=1}^{c} e^{-\frac{1}{2}\theta_j(\mathbf{Z})}} \quad \text{for} \quad i \neq j \quad (2.36)$$

Thus, the maximum likelihood estimators of u_i and v_i are, respectively, as follows:

$$\hat{u}_i^{MLE} = \frac{1}{d_i} \sum_{j=1}^{d_i} \varpi_{i,j} \quad \text{and} \quad \hat{v}_i^{MLE} = \frac{1}{k - d_i} \sum_{j=d_i+1}^{k} \varpi_{i,j}$$

where $\varpi_{i,1} \geq \varpi_{i,2} \geq ... \geq \varpi_{i,k}$ are the eigenvectors of $\hat{S}_i$.

Following this approach, the maximum likelihood estimators of α_i and σ_i^2 are

$$\hat{\alpha}_i^{MLE} = \frac{\hat{v}_i}{\hat{u}_i + \hat{v}_i} \quad \text{and} \quad (\hat{\sigma}_i^2)^{MLE} = \frac{\hat{u}_i \hat{v}_i}{\hat{u}_i + \hat{v}_i}$$

2.3.4.16 Distance Weighted Discrimination

The distance weighted discrimination (DWD) was introduced by Marron et al. (2007) to tackle high-dimensional datasets and to specifically improve the performance of support vector machines. It employs the concept of maximization, thereby maximizing the existing gap between an ordered pair of classes to make them more separable, introducing harmonic mean of the distances of all data vectors to the separating hyperplane (Huang et al., 2012). Given the training dataset $\{(y_i, \mathbf{Z}_i)\}_{i=1}^{N}$ with k-dimensional vector of covariates $\mathbf{Z}$, $\mathbf{y}$ the binary response variable $\mathbf{y} \in \{-1, +1\}$, let $d_i = (\mathbf{Z}_i'\mathbf{w} + \theta)y_i + \alpha_i$ be the distance of the i^{th} data vector to the separating hyperplane. Then, the DWD is obtained by the following:

$$\underset{\mathbf{w},\theta,\alpha_i}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\frac{1}{d_i} + C(\alpha_i) \right) \text{ subject to } d_i = (\mathbf{Z}_i'\mathbf{w} + \theta)y_i + \alpha_i; \;\; d_i, \alpha_i \geq 0; \; \forall \; ||w||^2 \leq 1 \tag{2.37}$$

where α_i is a positive slack variable included to boost the positivity of d_i; $\mathbf{w}$ is the weight vector (Qiao and Zhang, 2015). The slack variable serves as a correction measure, which corresponds to the amount of misclassification for the i^{th} vector. Thus, the DWD binary linear classification process employs gap minimization to improve the separability of the two classes and the minimization of the misclassification error.

2.3.4.17 k Nearest Neighbour

The k nearest neighbour (kNN) is used for classifying objects based on the closest training instances in the feature space.

Given the training data set $\{(Z_1, y_1), (Z_2, y_2), \cdots, (Z_L, y_L)\}$ in which an object is to be classified based on a majority being assigned to the class most common to its corresponding k nearest neighbours. Then, the Euclidean distance between instances i and j is defined as follows:

$$d_{Euclid}(i,j) = \sqrt{\sum_{\ell}^{L}\left(\frac{|y_{i\ell - j\ell}|}{r_\ell}\right)^2} = \sqrt{\left(\frac{|y_{i1-j1}|}{r_1}\right)^2 + \left(\frac{|y_{i2-j2}|}{r_2}\right)^2 + ... + \left(\frac{|y_{iL-jL}|}{r_L}\right)^2} \tag{2.38}$$

where r_ℓ denotes the maximum range of attribute ℓ; $\ell = 1, 2, ..., L$.

When k nearest neighbours with known classification are picked for an unclassified instance ℓ, then a combined classification approach that combines the classifications from the k nearest neighbours will predict the next class for ℓ, and so on.

The instance ℓ is classified as belonging to class $\mathbf{y}$, using the average distance measure if

$$\frac{1}{k_1}\sum_{i\in\mathbf{y}(\ell,k)} d_{Euclid}(i,\ell) < \frac{1}{k_2}\sum_{i\in N(\ell,k)} d_{Euclid}(i,\ell) \tag{2.39}$$

where $k = k_1 + k_2$; k_1 is the number of instances belonging to class $\mathbf{y}$ in the k nearest neighbours; k_2 is the number of instances belonging to class N in the k nearest neighbours (Huang et al., 2008; Su, 2011)

The k can be chosen by cross-validation, and the kNN model does not depend on the prior probabilities of the classes (Imandoust and Bolandraftar, 2013; Nayak et al., 2015).

2.3.4.18 Naive Bayes

The Naive Bayes classifier combines the Bayes model with a decision rule, and a common rule is to pick the most probable hypothesis, which is known as maximum posterior decision rule (Ou and Wang, 2009).

Given the training set $\{(Z_1, y_1), (Z_2, y_2), \cdots, (Z_T, y_T)\}$ and using the assumption that the features $\mathbf{Z}_c$ are independent given a class $\mathbf{y} = j$ such that $f_j(\mathbf{Z}) = \prod_{c=1}^{p} f_{jc}(\mathbf{Z})$ are functional class labels with kernel smoothing estimates of the function $f_{jc}(.)$ from the training set.

Then the Naive Bayes classifier is as follows:

$$\Psi^{NB}(\mathbf{Z}) = argmax_j \{f_j(\mathbf{Z})\psi_j\} \tag{2.40}$$

where ψ_j is estimated from the sample proportion.

2.3.4.19 Learning Vector Quantization

The learning vector quantization (LVQ) algorithm (Kohonen, 1995; Ripley, 1996), is an artificial neural network designed to enable one to construct a modified training set iteratively. The modified training sets are called codebooks. Let's consider the LVQ1 process based on Kohonen (1995). Assume that a number of codebooks m_i are placed into the input space to approximate various domains of the input vector $\mathbf{Z}$ by their quantized values. Usually several codebook vectors are assigned to each class of $\mathbf{Z}$ values, and $\mathbf{Z}$ is then decided to belong to the same class to which the nearest m_i belongs. Let $c = argmin(||\mathbf{Z} - m_i||)$, define the nearest m_i to $\mathbf{Z}$, denoted by m_c.

Values for the m_i that approximately minimize the misclassification errors in the above nearest-neighbor classification can be found as asymptotic values in the following learning process. Let $\mathbf{Z}(t)$ be a sample of input and let the $m_i(t)$ represent sequences of the m_i in the discrete-time domain. The basic LVQ1 process is defined by:

$$m_c(t+1) = m_c(t) + \alpha(t)[\mathbf{Z}(t) - m_c(t)] \tag{2.41}$$

if $\mathbf{Z}$ and m_c belong to the same class,

$$m_c(t+1) = m_c(t)\alpha(t)[\mathbf{Z}(t) - m_c(t)] \tag{2.42}$$

if $\mathbf{Z}$ and m_c belong to different classes, and

$$m_i(t+1) = m_i(t) \tag{2.43}$$

for i not in c. Here $0 < \alpha(t) < 1$, and $\alpha(t)$ may be constant or decrease monotonically with time.

2.3.4.20 Neural Network

The neural network (NNET) is a system made up of a number of simple highly interconnected processing elements, which process information by their dynamic state response to external inputs. The NNET consists of layers made up of interconnected nodes that contain the activation function (Caudill, 1989; Ripley, 1996). The NNET layers are as follows:

$$Input \longmapsto Hidden \longmapsto Output$$

Given an input vector of covariates $\mathbf{Z}$, and a categorical output $\mathbf{y}$. Then, a neural network can be modelled in the following form:

$$\mathbf{x}_j = \Gamma(\theta_{0,j} + \theta'_j \mathbf{Z}) \quad for \quad j = 1, 2, ..., p$$

$$\hat{y}_k = \Psi(\beta_{0,k} + \beta'_k \mathbf{Z}) \quad for \quad k = 1, 2, ..., q$$

where $\Gamma(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{x}}}$ is the sigmoid activation function, used to introduce a nonlinearity at the hidden layer. The parameters $\theta_{j,l}$ and $\beta_{k,j}$ are known as the weights and define linear combinations of the input vector $\mathbf{Z}$ and hidden unit output $\mathbf{x}$. The intercepts $\theta_{0,j}$ and $\beta_{0,k}$ are known as biases. The function Ψ permits a final transformation of the output and a typical choice for binary classification is the inverse logit function.

Let $L(\theta, \beta)$ be defined as follows:

$$L(\theta, \beta) = \sum_{i=1}^{N} \sum_{j=1}^{k} (y_k^i - \hat{y}_k^i)^2$$

then their respective partial derivatives will be as follows:

$$\frac{\partial L^i}{\partial \beta_{k,j}} = -2(y_k^i - \hat{y}_k^i)\delta_k'(\beta_k'\mathbf{x}^i)\mathbf{x}_j^i$$

and

$$\frac{\partial L^i}{\partial \theta_{j,l}} = -2\sum_{k=1}^{q}(y_k^i - \hat{y}_k^i)\delta_k'(\beta_k'\mathbf{x}^i)\beta_{k,j}\Gamma'(\theta_j'\mathbf{x}^i)\mathbf{x}_l^i$$

where the superscript i is the i^{th} component, for $j = 1, 2, ..., p$ and $k = 1, 2, ..., q$; θ' is the transpose of θ; β' is the transpose of β; N is the number of components. Thus, the gradient updates corresponding to the $(s+1)^{th}$ iteration with learning rate τ_s by back propagation, resulting in the following:

$$\beta_{k,j}^{(s+1)} \Longleftarrow \beta_{k,j}^{s} - \tau_s \sum_{i=1}^{N} \frac{\partial L^i}{\partial \beta_{k,j}^{s}} \tag{2.44}$$

$$\theta_{j,l}^{(s+1)} \Longleftarrow \theta_{j,l}^{s} - \tau_s \sum_{i=1}^{N} \frac{\partial L^i}{\partial \theta_{j,l}^{s}} \tag{2.45}$$

where τ_s is the learning rate (Caudill, 1989; Ou and Wang, 2009).

2.3.5 Statistical and Economic Performance Evaluation

2.3.5.1 Correct Prediction Ratio

In this case, a 2 x 2 square matrix of contingency table for cross-classifying the actual and predicted outcomes in each of the two categorical pairs is constructed before computing the correct prediction ratio or simply the 'hit ratio' for the direction of change in the stock market return.

The correct prediction ratio (CPR) is defined as follows:

$$CPR = CP_{upward} + CP_{downward} = \frac{1}{T}\sum_{t=1}^{T} I(\hat{\tau}_t = \tau_t)$$

where $CP_{upward} = \dfrac{\sum_{t=1}^{T} I(\hat{\tau}_t = 1) I(\tau_t = 1)}{\sum_{t=1}^{T} I(\tau_t = 1)}$ is the proportion of correct predictions for the upward moves; $CP_{downward} = \dfrac{\sum_{t=1}^{T} I(\hat{\tau}_t = 0) I(\tau_t = 0)}{\sum_{t=1}^{T} I(\tau_t = 0)}$ is the proportion of correct predictions for the downward moves; $I(.)$ is a binary indicator function which takes the value 1 when the argument in the parenthesis is true and 0 when it is false, based on the 0.50 threshold; $\hat{\tau}_t$ is the predicted value and τ_t is the actual value (Chevapatrakul, 2013).

The CPR lies between 0 and 1, $(0 \leq CPR \leq 1)$ and it is usually expressed in percentage.

2.3.5.2 The Pesaran-Timmermann Directional Predictability Test

This test was first proposed by Pesaran and Timmermann (1992) and was improved by Granger and Pesaran (2000) for evaluating directional forecasting or predictability performance and market timing. The null hypothesis H_0, which is "No statistically significant directional predictability" against the alternative hypothesis H_A, which is "There is statistically significant directional predictability" can be tested based on the Pesaran-Timmermann test statistic, as follows:

$$PT = \frac{\sqrt{T} KS}{\left(\frac{\bar{\tau}_I (1 - \bar{\tau}_I)}{\bar{I}(1 - \bar{I})} \right)^{0.5}} \overset{asymptotically}{\sim} N(0,1)$$

where $KS = TR - FR$ is the Hanssen-Kuiper skill score; $TR = \dfrac{\hat{I}^{uu}}{\hat{I}^{uu} + \hat{I}^{du}}$ is the true or hit rate; $FR = \dfrac{\hat{I}^{ud}}{\hat{I}^{ud} + \hat{I}^{dd}}$ is the false rate; T is the sample period in months;

and the forecasts' classifications are again obtained from the 2 x 2 contingency table showing:

$$\hat{I}^{uu} = \sum_{t=1}^{T} I(\hat{I}_t = 1, I_t = 1);$$

$$\hat{I}^{ud} = \sum_{t=1}^{T} I(\hat{I}_t = 1, I_t = 0);$$

$$\hat{I}^{du} = \sum_{t=1}^{T} I(\hat{I}_t = 0, I_t = 1);$$

$$\hat{I}^{dd} = \sum_{t=1}^{T} I(\hat{I}_t = 0, I_t = 0);$$

where u is an upward signal ($I_t = 1$) and d is a downward signal ($I_t = 0$); $I(.)$ is an indicator function taking the values 0 and 1; $\bar{I}$ is the sample mean of the sign indicator values I_t computed in the $T - month$ sample period; $\hat{I}_t$ is the predicted excess stock return sign indicator; I_t is the actual or realized excess stock return sign indicator; $\bar{\tau}_I = \bar{I}TR + (1 - \bar{I})FR$ (Nyberg, 2011; Granger and Pesaran, 2000; Bergmeir et al., 2014)

Thus, the PT test statistic as stated above has the asymptotic standard normal distribution under the null hypothesis H_0 of no directional predictability.

2.3.5.3 Confusion Matrix Metrics

The confusion matrix consists of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN). In this study, we use the following met-

	Positive (Predicted)	Negative (Predicted)
Positive (Actual)	TP	FN
Negative (Actual)	FP	TN

Total = TP + TN + FP + FN; where TP = true positives, FN = false negatives, FP = false positives, TN = true negatives

Table 2.1: The Confusion Matrix

rics to evaluate the accuracy and correctness of the classification models:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

Note that the accuracy of prediction equals the correct prediction ratio (CPR).

$$Precision = \frac{TP}{TP + FP}.$$

$$Sensitivity = \frac{TP}{TP + FN}.$$

$$Specificity = \frac{TN}{TN + FN}.$$

$$F_1 Score = \frac{2}{\dfrac{1}{Sensitivity} + \dfrac{1}{Precision}} = \frac{2TP}{2TP + FP + FN}.$$

Kappa Statistic: The kappa statistic, denoted by κ, is computed as follows:

$$\kappa = \frac{p_0 - p_e}{1 - p_e} = 1 - \frac{1 - p_0}{1 - p_e}$$

where p_0 is the relative observed agreement among the raters; p_e is the hypothetical probability of chance agreement, which can be obtained from the following:

$$p_e = \frac{1}{N^2} \sum_m n_{m_1} n_{m_2}$$

for categories m with N items and n_{m_i} is the number of times rater i predicted category m.

McNemar's Test: The McNemar's test, as introduced by McNemar (1947), is used in this paper to investigate the marginal homogeneity between the row and column marginal frequencies in the 2×2 confusion matrix. The null hypothesis of marginal homogeneity (that is the two outcomes are marginally equiprobable) against the alternative hypothesis that they differ in probabilities is defined as follows:
$H_0 : Prob(FN) = Prob(FP)$
$H_A : Prob(FN) \neq Prob(FP)$
The $McNemar's$ test statistic is defined as follows:

$$\chi^2 = \frac{(FN - FP)^2}{FN + FP} \sim \chi_1^2(\alpha).$$

Thus, the McNemar's test statistic is asymptotically chi-square distributed with

1 degree of freedom at the $\alpha\%$ significance level.

2.3.5.4 Economic Performance Evaluation

Evaluating the economic performance of a forecasting model is of great importance to a profit oriented portfolio investor. Consider the following trading strategy: Let $Prob_t(R_{t+1} > 0)$ be the estimated probability of a positive excess stock return for the period $t + 1$. Then the trading strategy or decision rule can be expressed as follows:

If $Prob_t(R_{t+1} > 0) > 0.5$, then purchase the stock index.

Else if $Prob_t(R_{t+1} > 0) \leq 0.5$, then purchase the treasury bill.

The performance of the constructed portfolios is evaluated over the out-of-sample period (1991 to 2016: T=312 months) using a plethora of performance measures. First, we consider the realized returns of the constructed portfolios. Let $r_{p,t+1}$ be the realized return of the portfolio at time $t + 1$. The average or expected return (ER) within the out-of-sample period, the cumulative return at the end of the period, and the volatility of the portfolio can be computed. It is imperative to compare the return per unit of risk by using the Sharpe Ratio.

Sharpe Ratio

Let's consider the Sharpe Ratio (SR) which standardizes the realized returns with the risk of the portfolio. The SR is computed through the following model:

$$SR_p = \frac{E(r_p) - E(rf_t)}{\sqrt{Var(r_p)}},$$

where r_p is the average realized return of the portfolio over the out-of-sample period; rf_t is the risk-free interest investment rate; $Var(r_p)$ is the variance of the portfolio over the out-of-sample period.

Optimally, portfolios with high Sharpe ratios are most preferable to portfolios with low Sharpe ratios, owing to the fact that the higher the Sharpe ratio the higher the return and the lower the volatility.

Maximum Drawdown

A portfolio measure associated with the sustainability of the portfolio losses is the maximum drawdown (MaxDD) which broadly reflects the maximum cumulative loss from a peak to a following bottom. MaxDD is defined as the maximum sustained percentage decline (peak to trough), which has occurred in the portfolio within the period studied. MaxDD up to time T is the maximum of the drawdown over the history of the specific variable under consideration. It is computed as follows:

$$MaxDD_p = \max_{T_0 \leq t \leq T-1} [\max_{T_0 \leq j \leq T-1} (PV_j) - PV_t],$$

where PV denotes the portfolio value; T_0, T denote the beginning and end of the evaluation period, respectively.

Omega Ratio

The Omega ratio, as a risk-return performance measure of a portfolio investment introduced by Keating and Shadwick (2002), gives the probability weighted ratio of gains versus losses for a stipulated threshold return target. We first define the n-th lower partial moment (LPM_n) of the portfolio return and the kappa function K_n, and used the concept to compute Omega, Sortino and the Upside Potential respectively, see (Harlow and Rao, 1989; Sortino and Van Der Meer, 1991; Sortino and Price, 1994) for detail studies. The n-th lower partial moment (LPM_n) of the portfolio return is defined as follows:

$$LPM_n(r_b) = E[((r_b - r_p)_+)^n]$$

where r_b is the benchmark return.

The Kappa function $K_n(r_b)$ is defined as follows:

$$K_n(r_b) = \frac{E(r_p) - r_b}{\sqrt[n]{LPM_n(r_b)}} \quad \text{for} \quad n = 1, 2, ...$$

Thus, the Omega ratio is computed from the following formula:

$$Omega(r_b) = K_1(r_b) + 1$$

Sortino Ratio

Unlike the Sharpe ratio, which penalizes both upside and downside volatility equally, the Sortino ratio penalizes only the returns that fall below a user specified target. The Sortino ratio measures the risk adjusted return of a portfolio investment. It can be computed from the following formula:

$$S(r_b) = K_2(r_b)$$

Like the Sharpe ratio, the higher the Sortino ratio, the better the risk adjusted performance and vice versa.

Upside Potential

Upside Potential is a measure of the return of an investment relative to the minimal acceptable return. The upside potential is calculated as follows:

$$UP(r_b) = \frac{E[(r_p - r_b)^+]}{\sqrt{LMPM_2(r_b)}}$$

The economic importance of the upside cannot be overemphasized. It is not only indicating an investor's potential gain in value but also judges the success of a portfolio manager's performance comparative to a benchmark.

Additionally, I investigate the tail-risk of the different proposed models. A CVaR of $\lambda\%$ at the 100(1-α)% confidence level means that the average portfolio loss measured over 100α% of worst cases is equal to $\lambda\%$ of the wealth managed by the investor. To compute VaR and CVaR, we use the empirical distribution of the portfolio realized returns. VaR and CVaR are calculated at the 95% confidence levels.

In this study, the U.S. 3-month interest rate for the risk-free rate rf_t and for the benchmark rate of return (r_b) necessary for the calculation of *Omega*, *MaxDD* and S were employed.

2.4 Data Analysis and Discussion

2.4.1 Sources of Data and Variables

The data used in this chapter are obtained from Amit Goyal's webpage[1], covering monthly observations ranging from January 1960 to December 2016. These variables, presented in Table 2.2, have been used in the existing literature quite extensively for predictability of the equity premium, see (Rapach et al., 2010; Nyberg, 2011; Meligkotsidou et al., 2014, 2019) among others. The total number of observations is $T = 684$. An out-of-sample period of $T_2 = 312$ monthly observations ranging from January 1991 to December 2016 has been employed for the evaluation of the forecasting performance. The forecast horizon denoted by h is one month ahead for each of the forecasting models.

In the out-of-sample method, the parameters of the forecasting models are estimated recursively using an expanding window of observations, in which the fitted models are estimated using data from the start date of the dataset to the present time and obtain a one month-period-ahead forecast. The procedure is

Table 2.2: The Financial Variables used for the Study

Indicator	**Time Series Variable**
Equity Premium	$EquityPrem$
Default Return Spread	DFR
Excess Stock Return	ESR
Short Term Interest Rate	$\Delta ShortR$
Long Term Yield	$\Delta LongR$
Term Spread	$TermSpr$
Inflation	$\Delta Infl$
Return Spread	$ReturnSpr$
Yield Spread	$YieldSpr$
Book to Market Value	BMV
Net Equity Expansion	NEE
Dividend Price Ratio	DPR
Earning Price Ratio	EPR
Stock Variance	$SVar$

[1]www.hec.unil.ch/agoyal/

repeated iteratively until the end of the forecast sample period is attained. In the CART techniques, each classification model is trained, pre-processed the training dataset in a closed centre and scale form, tuned the parameter(s) of each model by cross-validation and resampling, determining the variable importance before making the out-of-sample forecasts. The resampling approach seeks to determine the values of each of the model parameters (if any) and uses the best tuning parameter(s) based on fitted in-sample accurate measures to produce the out-of-sample forecasts. In each model, the best tuning parameter(s) were used to run the out-of-sample forecasts recursively, and their respective performance evaluation measures were obtained. All computations in this study were obtained using R software and the associated packages (Kuhn et al., 2008; Kuhn, 2012, 2015) and https://topepo.github.io/caret/available-models.html, see Tables 2.3 and 2.4.

2.4.2 Statistical Performance Evaluation Results

The statistical performance evaluation results for the proposed techniques in this chapter, presented in Tables 2.3 and 2.4, are shown to be promising, owing to the empirical evidence of useful predictability. The out-of-sample positive class return forecasts are depicted in Figures 2.1 to 2.6. In the benchmark binary probit models, the predictive accuracy of the static binary probit model involving all covariates appeared to be very low with insignificant evidence of PT directional predictability, and the kappa statistic is extremely poor, indicating a poor inter-rater agreement between the actual and predicted values. Whereas the application of stepwise variable selection by the Akaike information criterion (AIC) on the static model seeks to improve the predictive accuracy, it does not provide statistically significant evidence of directional predictability and the kappa statistic is still low. The dynamic binary probit, which includes the lagged excess stock return indicator together with the other predictor variables, produced a slightly better predictive accuracy as compared to the static probit.

Table 2.3: Statistical Performance Evaluation Results

Model	**Package(s)**	**Method Value**	**CPR/Acc.**	**MCE**	**PT**	**Prec.**	**Spec.**	**Sens.**	**Kappa**	**McN**	F_1S
Binary Probit											
Static BP	stats	glm	0.561	0.439	-0.066	0.794	0.391	0.605	-0.003	0.000	0.686
Static BP StepAIC	MASS	glmStepAIC	0.596	0.404	1.533	0.820	0.477	0.628	0.079	0.000	0.711
Dynamic BP	stats, dyn	glm	0.580	0.420	0.794	0.810	0.452	0.617	0.014	0.000	0.700
Dynamic BP StepAIC	stats, dyn	glmStepAIC	0.596	0.404	1.533	0.820	0.477	0.628	0.079	0.000	0.711
Penalized Binary Probit											
Ridge BP	glmnet	glmnet	0.651	0.349	3.939***	0.947	0.706	0.644	0.163	0.000	0.767
LASSO BP	glmnet	glmnet	0.619	0.381	2.215*	0.894	0.545	0.631	0.101	0.000	0.740
Elastic Net BP	glmnet	glmnet	0.628	0.372	2.744**	0.894	0.574	0.638	0.127	0.000	0.744
Bagging and Boosting											
Bagging	ipred, plyr	treebag	0.606	0.394	2.626**	0.735	0.500	0.656	0.147	0.047	0.693
RPart	rpart	rpart	0.612	0.388	1.674*	0.621	0.138	0.921	0.068	0.000	0.742
RF	randomForest	rf	0.654	0.346	4.339***	0.681	0.577	0.804	**0.240**	0.001	0.738
CTree	party	ctree	0.609	0.391	0.970	0.984	0.571	0.610	0.020	0.000	0.753
CForest	party	cforest	0.612	0.388	1.465	0.610	0.024	0.995	0.023	0.000	0.756
AdaBoost	fastAdaboost	adaboost	0.644	0.356	3.668***	0.963	0.731	0.636	0.136	0.000	0.766
LogitBoost	caTools	LogitBoost	0.583	0.417	1.294	0.627	0.293	0.773	0.070	0.000	0.692
GBM	gbm, plyr	gbm	0.615	0.385	2.001*	0.899	0.537	0.627	0.089	0.000	0.739
GLMBoost	plyr, mboost	glmboost	0.625	0.375	2.408**	0.952	0.625	0.625	0.086	0.000	0.755
Nearest Neighbour											
kNN	class	knn	0.628	0.372	3.219***	0.799	0.542	0.659	0.175	0.000	0.722

Note: All computations in this study are obtained using R version 3.4.3; CPR = correct prediction ratio, Acc. = accuracy, PT = Pesaran-Timmermann test statistic, Prec. = precision, Spec. = specificity, Sens. = sensitivity; NcM = McNemar's p-value, $F_1S = F_1$ score. The "$***$", "$**$" and "$*$" signified 0.1%, 1% and 5% significance, respectively.

Table 2.4: Statistical Performance Evaluation Results Continued

Model	**Package(s)**	**Method Value**	**CPR/Acc.**	**MCE**	**PT**	**Prec.**	**Spec.**	**Sens.**	**Kappa**	**McN**	F_1S
Neural Networks											
NNET	nnet	nnet	0.593	0.407	0.555	0.619	0.195	0.852	0.052	0.000	0.717
LVQ	class	lvq	0.606	0.394	2.626**	0.624	0.187	0.878	0.073	0.000	0.730
Bayesian Models											
Bayes GLM	arm	bayesglm	0.603	0.397	1.739*	0.836	0.492	0.629	0.088	0.000	0.718
Naive Bayes	klaR	nb	0.660	0.340	4.402***	0.937	0.707	0.653	0.195	0.000	0.770
Discriminant Analysis											
LDA	MASS	lda	0.558	0.442	-0.241	0.794	0.381	0.602	-0.012	0.000	0.685
Sparse LDA	sparseLDA	sparseLDA	0.603	0.397	1.533	0.862	0.490	0.625	0.073	0.000	0.724
Step LDA	klaR, MASS	stepLDA	0.603	0.397	0.663	0.952	0.471	0.610	0.021	0.000	0.744
HDA	hda	hda	0.580	0.420	0.407	0.847	0.420	0.611	0.019	0.000	0.710
HDDA	HDclassif	hdda	**0.670**	**0.330**	5.244***	0.894	0.667	0.671	**0.241**	0.000	0.766
QDA	MASS	qda	0.654	0.346	4.131***	0.857	0.609	0.667	**0.215**	0.000	0.750
Step QDA	klaR, MASS	stepQDA	0.603	0.397	0.663	0.952	0.471	0.610	0.021	0.000	0.744
RDA	klaR	rda	0.636	0.365	3.015**	0.926	0.622	0.636	0.029	0.000	0.759
DWD Linear	kerndwd	dwdLinear	0.609	0.391	1.581	0.905	0.514	0.622	0.067	0.000	0.737

Note: All computations in this study are obtained using R version 3.4.3; CPR = correct prediction ratio, Acc. = accuracy,
PT = Pesaran-Timmermann test statistic, Prec. = precision, Spec. = specificity, Sens. = sensitivity; NcM = McNemar's p-value, $F_1S = F_1$ score.
The " $***$ ", " $**$" and " $*$ " signified 0.1%, 1% and 5% significance, respectively.

Again, the application of stepwise variable selection by AIC on the dynamic probit results in a slight increase in the predictive accuracy and the result equals the result of the stepwise static binary probit. The analysis of the static and dynamic binary probit models revealed that a parsimonious approach is preferable to incorporating many predictors in the models. The replication of the static and dynamic binary probit models used in the previous findings, as shown in the existing literature, such as in Nyberg (2011), had confirmed the feasibility of these models for excess stock return directional predictability. Interestingly, the empirical analysis of the static and dynamic binary probit models in this chapter produced predictive accuracy (CPRs) equivalent to the CPRs of these models demonstrated by Nyberg (2008), Nyberg (2011) and investigate other important statistical performance measures, such as the kappa statistic, which determines inter-rater agreement between the actual results and the forecasts, and the McNemar's test for the detection of marginal homogeneity or equiprobability.

Turning to penalized binary probit models, the inclusion of penalty vector norm(s) in the ordinary binary probit models revealed a good improvement in predictive performance of the models. Specifically, the ridge, LASSO and elastic net provide higher predictive accuracy, which outperformed the benchmark binary probit models, with Ridge being statistically significant at 0.1%, EN at 1% and LASSO at 5%, with better inter-rater agreement between the actual results and the forecasts, as judged by the kappa statistic, and McNemar's p_{value} evidence of marginal heterogeneity. The penalized probit models also produced better precision, specificity, sensitivity and F_1 scores compared to the ordinary probit models. The ridge produced a better predictive accuracy and other statistical performance evaluation measures than the LASSO and elastic net, outperforming both the LASSO and the elastic net in this direction. Overall, the presence of the ℓ_1 and ℓ_2 penalty vector norms in the binary probit models appeared to improve the predictive task and the overall performance of the resulting models.

The models employed for forecasting the direction of the U.S. stock market

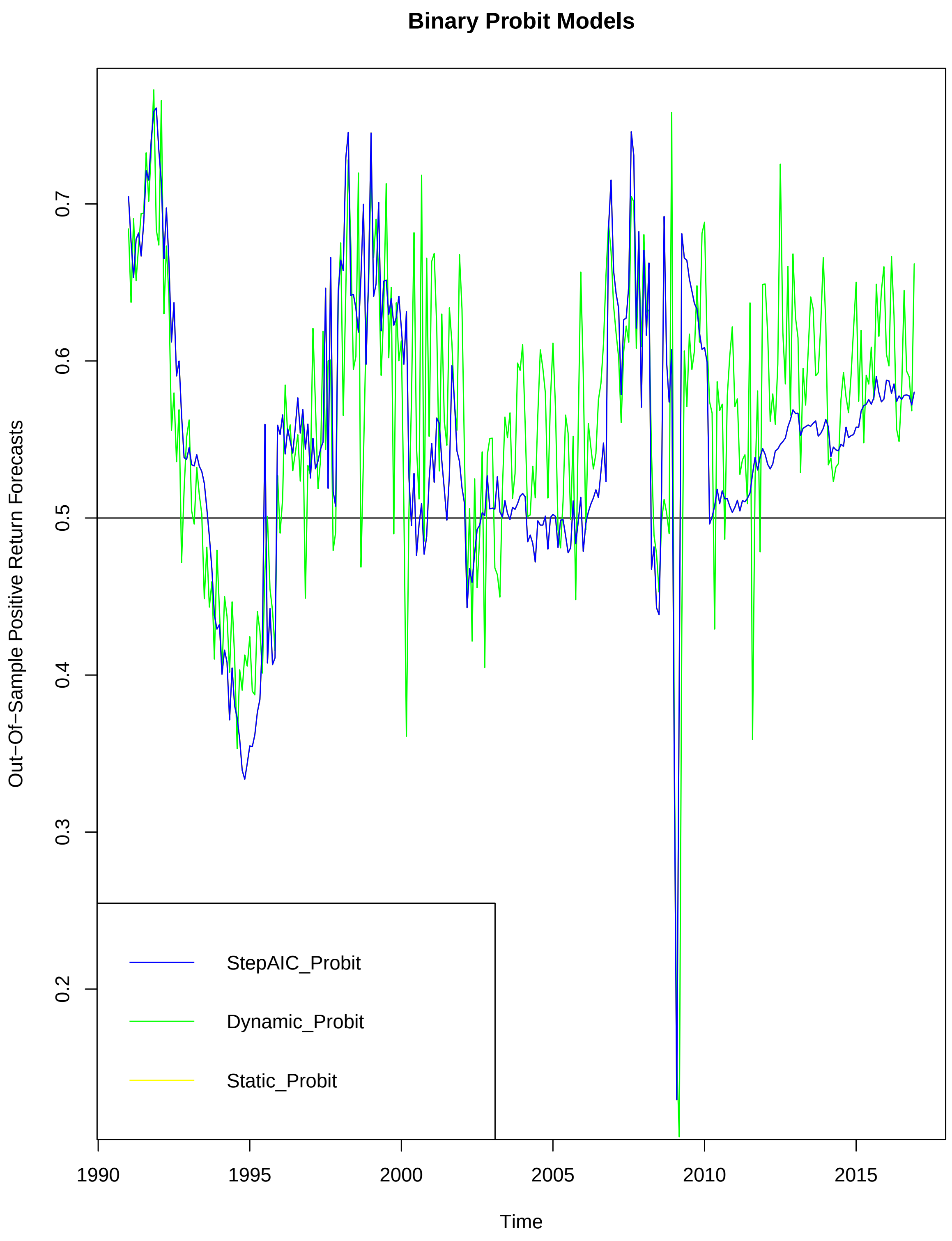


Figure 2.1: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts

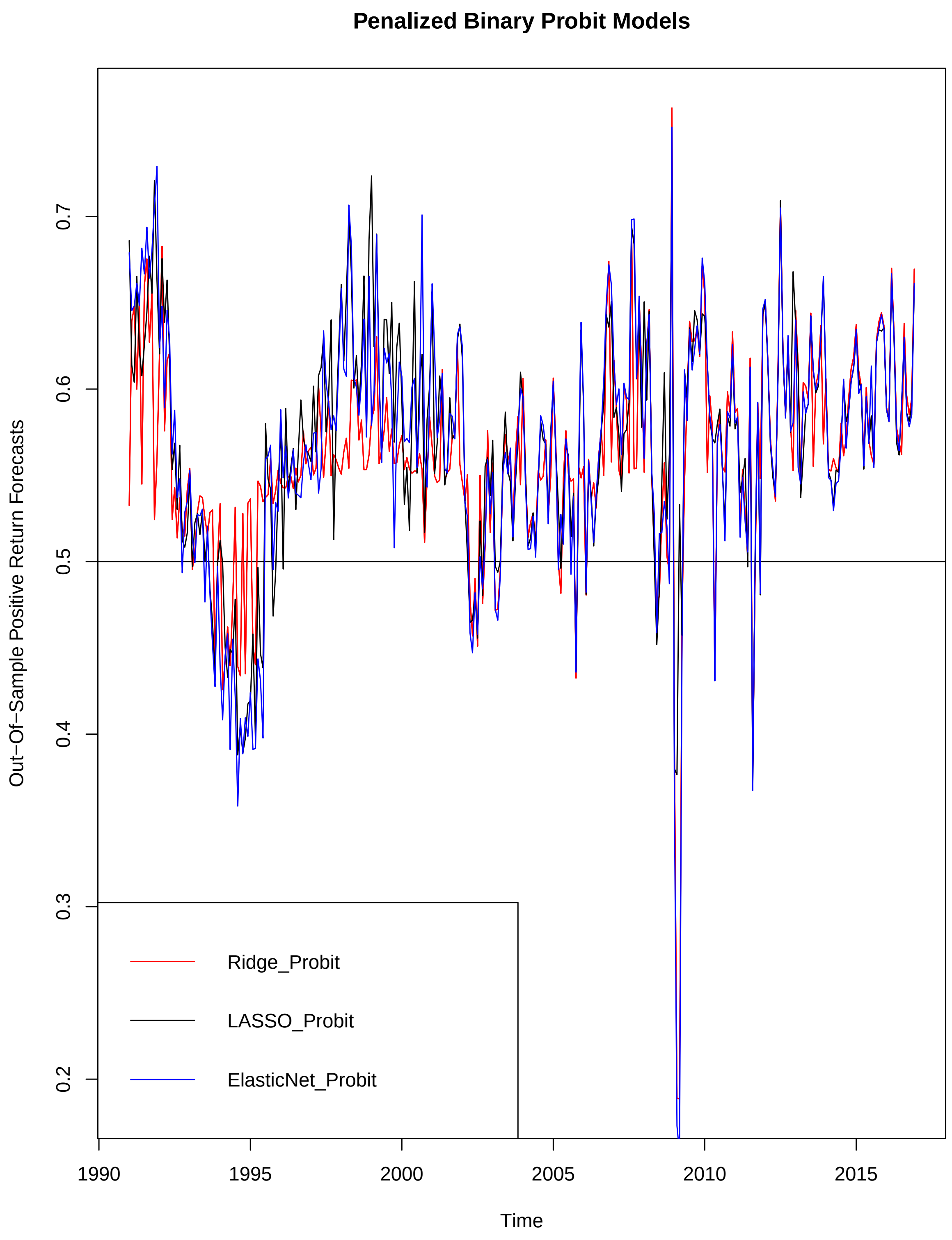


Figure 2.2: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts continued

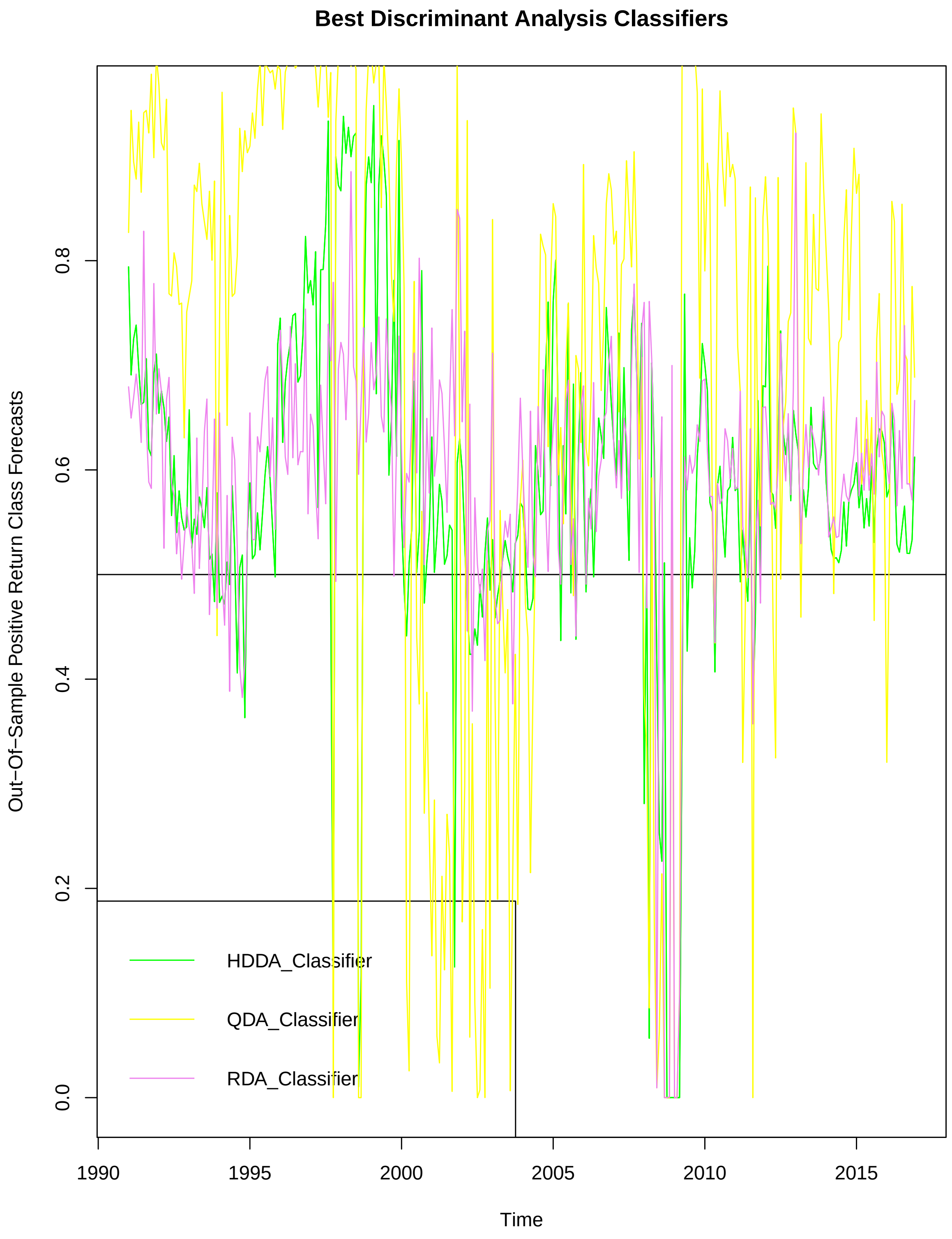


Figure 2.3: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts

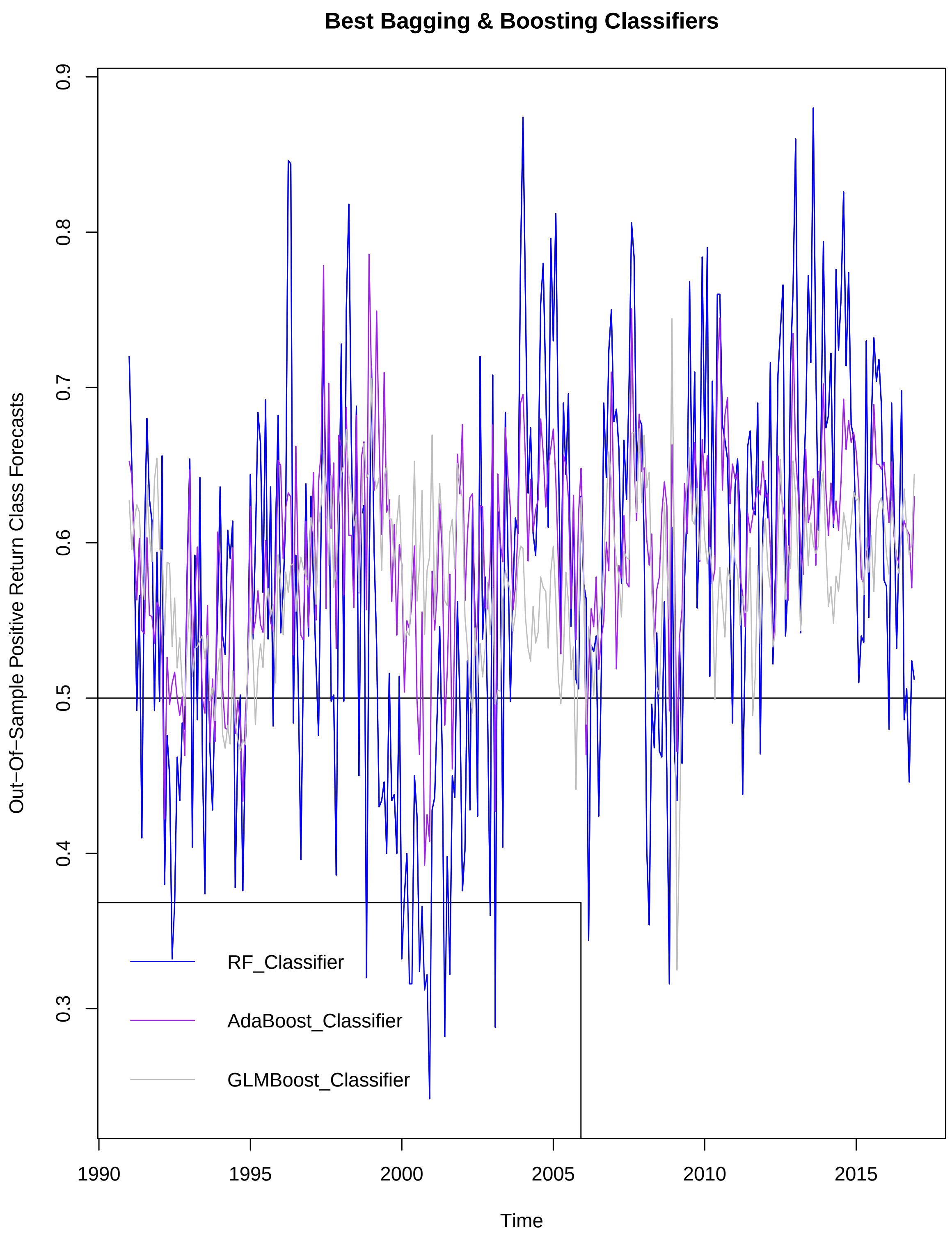


Figure 2.4: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts continued

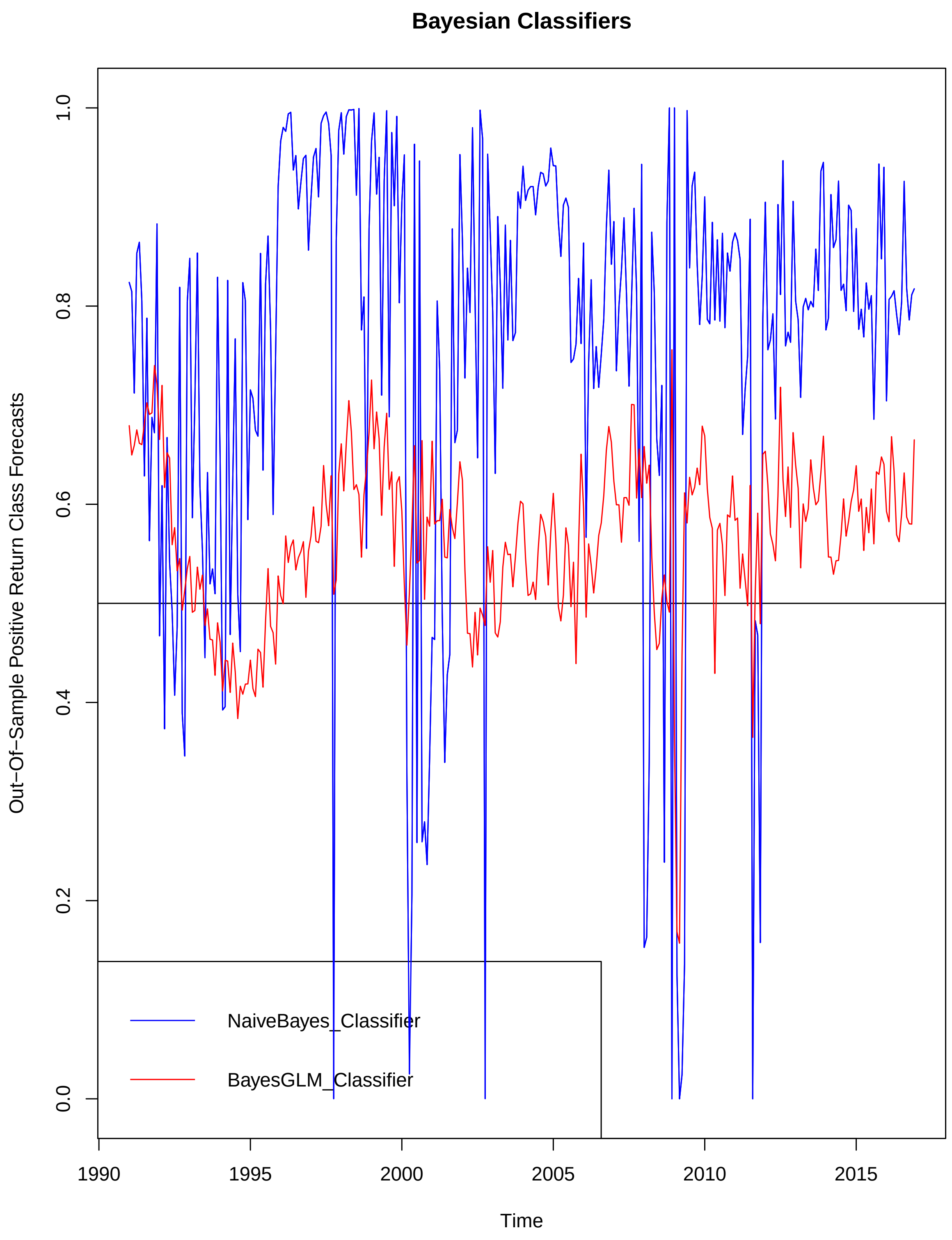


Figure 2.5: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts continued

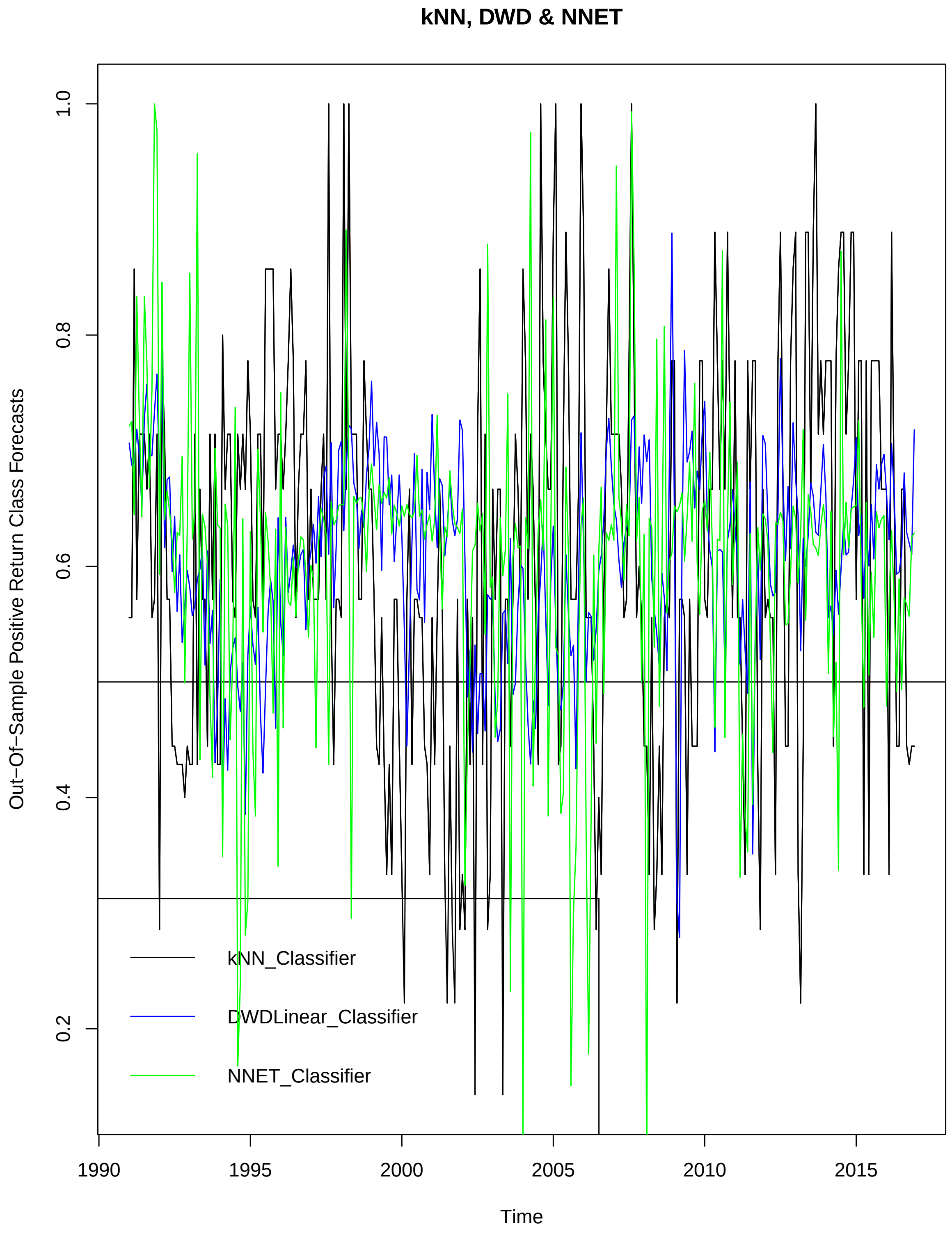


Figure 2.6: Graphical Representation of the Out-of-Sample Positive Class Return Forecasts continued

in this chapter demonstrate both the feasibility of the models and significant evidence of outperformance over the benchmark probit models. With the exception of LogitBoost, neural networks, LDA and HDA, all other classifiers in this chapter are shown to have outperformed the benchmark binary probit models by statistical performance evaluation measures. In more detail, seven of the proposed methods outperform the benchmark probit models at 0.1%, five methods at 1% and four methods at 5%. It is noticeable that the introduction of the stepwise variable selection concept in the LDA model improved the predictive task and the resulting statistical performance of the LDA model, whereas the introduction of the stepwise concept in the QDA model worsened the predictive task and overall performance of the QDA model. The empirical analysis in this study confirmed the superior outperformance of random forest (RF) over other forest based classification models in financial analysis, as shown in Ballings et al. (2015). Bagging and boosting, as demonstrated by Zheng (2006) in other aspects of stock market analysis, also appeared to have outperformed the neural networks in this study. Unlike the benchmark binary probit models, the three sophisticated machine learning classification models, i.e., random forest, HDDA and QDA, provide fair inter-rater agreement between the actual results and the forecasts, as shown by their respective kappa statistic. The HDDA appeared to produce the best out-of-sample statistical performance evaluation results, followed by naive Bayes, and the QDA, with significant evidence of outperformance. Overall, the HDDA is the best model for predicting the direction of the U.S. stock market in terms of statistical measures of predictability.

2.4.3 The Economic Performance Evaluation Results

As in the statistical case, the economic performance evaluation results, presented in Tables 2.5 and 2.6, also revealed that the dynamic binary probit model produced better cumulative returns, Sharpe ratio (SR), MaxDD, Omega, Sortino Ratio and Upside Potential than the static binary probit, and the stepwise variable selection cases by AIC, in each case, appeared to yield even better economic evaluation results than the ordinary case. The penalized binary probit models (ridge, LASSO and elastic net) produced better cumulative returns, SR, MaxDD, Omega, Sortino and Upside than the ordinary binary probit models and, hence, demonstrate economically significant evidence of outperformance over the benchmark binary probit. For example ridge has SR equal to 0.642 while the static probit has SR equal to 0.271. Interestingly, all the penalized binary probit models outperformed the benchmark static and dynamic binary probit models in this chapter. Again, the ridge outperformed the LASSO and elastic net in terms of the economic significance measure and seems to provide better economic information on future investment outcomes to a stock market investor than the LASSO and elastic net. All the CART models that are shown to be promising in terms of statistical predictability in this chapter are also shown to be promising in terms of economic significance to portfolio investors. The effectiveness of Bagging (Bootstrap Aggregating), Boosting, Trees, Forests, Naive Bayes Discriminant Analysis models and other ensembles that were demonstrated to be useful in other concepts of financial analysis are also shown to be useful in forecasting the direction of the U.S. excess stock market returns and providing portfolio investors with better economic significance about the future outcome of investments in the stock market. The Random Forest method produced the highest SR (0.643) among the bagging and boosting models and by far greater than the static probit model (0.271). It is worth noting that a best performing model in terms of the statistical measure may not necessarily reflect the best performance in the economic significance measure.

Table 2.5: Economic Performance Evaluation Results

Model	CumRet	ER	SD	SR	VaR0.05	CVaR0.05	MaxDD	Omega	Sortino	UP
Buy & Hold Strategy	1.914	0.074	0.144	0.326	-0.230	-0.342	34.655	1.274	0.130	0.594
Binary Probit										
Static BP	1.576	0.061	0.125	0.271	-0.215	-0.308	38.973	1.256	0.107	0.525
Static BP StepAIC	2.145	0.083	0.130	0.430	-0.215	-0.308	20.784	1.436	0.177	0.582
Dynamic BP	2.012	0.078	0.126	0.428	-0.215	-0.308	26.342	1.374	0.163	0.574
Dynamic BP StepAIC	2.145	0.083	0.130	0.430	-0.215	-0.308	20.784	1.436	0.177	0.582
Penalized Binary Probit										
Ridge BP	2.796	0.108	0.126	0.642	-0.181	-0.284	15.556	1.669	0.274	0.683
LASSO BP	2.459	0.095	0.130	0.524	-0.204	-0.301	17.362	1.526	0.219	0.630
Elastic Net BP	2.533	0.097	0.127	0.559	-0.189	-0.289	16.755	1.574	0.236	0.647
Bagging and Boosting										
Bagging	2.137	0.082	0.109	0.511	-0.189	-0.253	13.475	1.563	0.223	0.618
RPart	1.655	0.064	0.140	0.265	-0.229	-0.338	38.932	1.231	0.105	0.559
RF	2.605	0.100	0.114	0.643	-0.168	-0.266	9.256	1.737	0.285	0.672
CTree	1.986	0.076	0.143	0.348	-0.229	-0.341	33.527	1.305	0.139	0.594
CForest	1.973	0.076	0.144	0.342	-0.230	-0.342	33.720	1.296	0.136	0.598
AdaBoost	2.383	0.092	0.137	0.475	-0.219	-0.319	21.728	1.450	0.195	0.628
LogitBoost	1.762	0.068	0.124	0.332	-0.201	-0.296	32.488	1.318	0.138	0.570
GBM	2.112	0.081	0.131	0.418	-0.218	-0.295	27.367	1.392	0.175	0.620
GLMBoost	2.161	0.083	0.138	0.409	-0.224	-0.326	27.669	1.376	0.166	0.607
Nearest Neighbour										
kNN	2.266	0.087	0.124	0.487	-0.179	-0.298	17.423	1.519	0.203	0.594

Note: All computations in this study are obtained using R version 3.4.3; CumRet = cumulative return, ER = expected return, SD = standard deviation, SR = Sharpe ratio, MaxDD = maximum draw down, UP = upside potential

Table 2.6: Economic Performance Evaluation Results Continued

Model	**CumRet**	**ER**	**SD**	**SR**	**VaR0.05**	**CVaR0.05**	**MaxDD**	**Omega**	**Sortino**	**UP**
Buy & Hold Strategy	1.914	0.074	0.144	0.326	-0.230	-0.342	34.655	1.274	0.130	0.594
Neural Networks										
NNET	2.004	0.077	0.127	0.396	-0.218	-0.290	27.504	1.367	0.165	0.612
LVQ	1.861	0.072	0.143	0.314	-0.230	-0.342	34.198	1.270	0.124	0.585
Bayesian Models										
Bayes GLM	2.230	0.086	0.124	0.477	-0.189	-0.289	21.103	1.487	0.198	0.603
Naive Bayes	2.807	0.108	0.125	0.653	-0.191	-0.278	15.844	1.673	0.281	0.699
Discriminant Analysis										
LDA	1.481	0.057	0.126	0.240	-0.218	-0.311	38.973	1.223	0.094	0.515
Sparse LDA	1.972	0.076	0.130	0.380	-0.219	-0.309	27.066	1.361	0.152	0.575
Step LDA	2.007	0.077	0.142	0.356	-0.229	-0.338	33.202	1.315	0.143	0.595
HDA	1.927	0.074	0.127	0.375	-0.201	-0.308	25.157	1.360	0.151	0.569
HDDA	3.090	0.119	0.111	0.831	-0.162	-0.225	16.430	1.935	0.396	0.820
QDA	**3.517**	0.135	0.101	**1.077**	-0.125	-0.168	**6.072**	**2.378**	**0.598**	**1.032**
Step QDA	1.863	0.072	0.141	0.319	-0.229	-0.338	34.156	1.281	0.127	0.579
RDA	2.499	0.096	0.124	0.561	-0.204	-0.277	31.126	1.552	0.240	0.676
DWD Linear	1.921	0.074	0.133	0.355	-0.223	-0.322	37.314	1.331	0.141	0.569

Note: All computations in this study are obtained using R version 3.4.3; CumRet = cumulative return, ER = expected return, SD = standard deviation, SR = Sharpe ratio, MaxDD = maximum draw down, UP = upside potential

Contrary to the statistical performance analysis, the HDDA does not correspondingly provide the best economic performance result; instead the QDA produced the highest cumulative return, SR, Omega, Sortino and Upside with a corresponding least MaxDD. QDA gives SR equal to 1.077 (almost four times the SR of static probit), while HDDA produces SR equal to 0.831. Although the HDDA also demonstrates good evidence of economic significance and appeared to have outperformed the other models in terms of some useful economic performance evaluation measures, another suitable benchmark comparative measure of economic significance on portfolio investment by investors is to compare the expected return on portfolio investment produced by the model with a buy and hold trading strategy of the SP500 index. In this case, we see that the simple probit models do not outperformed the buy and hold strategy. However, the penalized probit models and the prominent CART models (for example, HDDA, QDA, RDA and Naive Bayes) outperformed the buy and hold strategy, providing higher risk-adjusted returns.

Interestingly, the prominent CART models used in this chapter have economically outperformed the benchmark binary probit models and the buy and hold trading strategy with a significant margin. Overall, the QDA appeared to be the best economically significant model for forecasting the direction of the U.S. stock market out-of-sample.

2.5 Conclusion

The analysis of the benchmark binary probit models in this chapter corroborates the empirical findings in previous studies, especially in Nyberg (2008), Nyberg (2011). In this chapter, additional statistical and economic performance evaluation measures were introduced to investigate the long-run usefulness of these models in the financial stock market.

The empirical analysis in this chapter revealed that the proposed sophisticated machine learning techniques outperformed the benchmark binary probit models both statistically and economically. In terms of the statistical predictive accuracy, the best penalized binary probit model outperformed the best binary probit model by 5.5% and the best CART model outperformed the best binary probit model by 7.4%.

In terms of statistical performance evaluation measures, the HDDA appeared to be the best model for forecasting the direction of the U.S. stock market in this chapter, owing to its highest predictive accuracy with minimum misclassification error (MCE) and other resulting statistical measures. Adding to the previous analysis in the existing financial and econometric literature, the Kappa statistic was used in this chapter to investigate the inter-rater agreement between the actual values and forecasts produced by the various models. The Kappa statistic revealed that there is no inter-rater agreement between the actual values and the forecasts obtained by the static and the dynamic binary probit models. Interestingly, the RF, QDA and HDDA proposed in this chapter provide evidence of fair inter-rater agreement between the actual values and the forecasts produced by the models. However, the QDA appeared to be the best model in terms of the measures of economic significance in this chapter. The QDA seems to provide more economic value to guarantee the success of a portfolio manager in the stock market than the other models used in this chapter.

Overall, the HDDA is the best model for forecasting the direction of the U.S. stock market out-of-sample in terms of statistical predictability measures, while the QDA is the best economically significant model for a portfolio investor whose utmost goal is to minimise risk and maximize profit, based on the empirical analytical findings in this chapter.

[2]

[2]This chapter is published in a reputable international journal, as follows:
Iworiso, J. & Vrontos, S. (2019). On the Directional Predictability of Equity Premium Using Machine Learning Techniques. Journal of Forecasting, 1-21. https://doi.org/10.1002/for.2632

Chapter 3

Forecasting the U.S. Equity Premium with Regression Training Techniques

3.1 Introduction

The out-of-sample predictability of equity premium is a major confrontational research issue with controversial views among scholars in empirical finance. The challenge about expectation of the stock market delivery to mean-variance investors above the treasury bill rate led to the quest for a meaningful estimate of the equity premium (Campbell, 2008). The historical average model is seen as an old-fashioned efficient market theory for forecasting the equity premium, owing to the inconsistent forecasts comparative to the real-time market setting. Empirical literature have documented that several financial and economic variables used as potential predictors can only forecast the equity premium in-sample but are unable to deliver significantly superior out-of-sample forecasts relative to the benchmark global historical average out-of-sample forecasts, and hence the research question: can anything consistently beat the historical average out-of-sample? (Campbell and Thompson, 2005; Goyal and Welch, 2007). The historical average is used as a benchmark for comparing the performance of any model whose forecasts are estimated via expanding or rolling window in an out-of-sample fashion, from which the relevant statistical measures of forecastability can be evaluated. In this regard, any model whose relevant statistical measures outperformed those obtained with the benchmark historical average is said to beat the historical average out-of-sample.

This chapter focuses mainly on the application of sophisticated regression training (RT) techniques to forecast the U.S. monthly equity premium out-of-sample recursively. It involves the training of a regression model which comprises all relevant financial and economic variables rather than using fewer or individual variables, and then using the resulting sophisticated model together with the best tuning parameters to forecast the equity premium. We employed a broad categories of regression models, which includes, kitchen sink linear model, partial least squares regression, kernel-based regularized least squares, support vector regression, relevance vector regression, regularized or penalized regression, components regression, Gaussian processes regression, regression splines, rule-based regression, nearest neighbour, projection pursuit, and neural networks. A major advantage of the RT techniques is that all predictor variables are regarded as important variables before preprocessing in the course of training the model and the resulting fitted or trained model decides variable importance associated with the final cross-validated model.

The application of these RT forecasting models in this perspective is expected to beat the old-fashioned benchmark historical average. Hopefully, the output of this study will enrich empirical literature and fills the research gap by addressing the controversial arguments between scholars on the predictive ability of financial and economic predictor variables in forecasting the U.S. equity premium out-of-sample relative to the benchmark historical average. In particular, the significant RT out-of-sample forecasting models among these RT models aimed to provide meaning information to a mean-variance portfolio investor in a real-time setting who optimally reallocates a monthly portfolio between equities and risk-free treasury bill.

3.2 Literature Review

The predictability of equity premium is drawing a keen interest in modern financial and econometric research. By definition, the equity premium is the difference between the expected return on the market portfolio (SP500) and the risk free interest rate. It is the return that investors can expect from holding the market portfolio in excess of the return on the 3-month Treasury bills. In finance, it is considered to be the most important concept owing to portfolio allocation decisions and cost of capital estimates. The backbone of investment strategies depends on the ability to predict future returns but the forecastability itself does not necessarily guarantee the investor's profit from the trading strategy based on the resulting forecasts (Campbell and Thompson, 2005; Bai, 2010). Thus the quest for a reasonable precise estimate of the equity risk premium by a number of scholars in the financial business cycle.

Several empirical literature have demonstrated evidence in forecasting equity premium and evaluating their performance in an attempt to determine both statistical and economic significance, see among others (Polk et al., 2006; Goyal and Welch, 2007; Campbell and Thompson, 2007; Della Corte et al., 2010; Kellard et al., 2010; Baur and Löffler, 2015; Aye et al., 2016; Kolev and Karapandza, 2017; Avdis and Wachter, 2017). A notable argument in this perspective, over the years, is the critical examination of any other predictive model that can significantly outperform the so-called benchmark historical average forecasting model. One of the academic debating questions in modern review of financial studies is that: can any other empirical model accurately forecast the equity premium better than the forecasts from the historical mean? Goyal and Welch (2007) have argued previously that no other variable beats a simple forecast based on the historical mean, owing to the fact that in-sample correlations conceal a systematic failure of the financial and economic variables out-of-sample. Contrary to this view, the analysis made by Rapach et al. (2007) reveals that despite the failure of the individual model forecasts to outperform the historical mean forecasts, the combination of the individual model forecasts yield statistically and economically significant gains, relative to the historical mean, consistently over time. Although some of the indicators used as predictors appeared to be good statistically significant predictors of the equity premium in-sample at some specific horizons but are relatively poor in the out-of-sample forecasting ability. Fama and French (2002) on the other hand, verified analytically that the estimates from economic fundamentals especially the dividend growth model, produced lower standard errors resulting to better precision than the estimates from the historical average model. The variety and inconsistency in performance as shown by different scholars could be traceable to the choice of predictor variables, the frequency of the dataset (annually, quarterly, monthly, weekly or daily), the estimation period, the forecast horizon and the specific models used by the researcher.

The notion on poor predictive performance of these variables in existing literature prompt Goyal and Welch (2003) to investigate the cause of the poor predictive ability using dividend ratio as a typical predictor variable. The finding clarifies that the poor predictive ability is mainly caused by instability of parameters in the models. The in-sample tests provide better explanatory power than the counterpart, and hence, a good in-sample performance does not necessarily imply a good out-of-sample performance. Notwithstanding the out-of-sample weak explanatory power, they seems to be economically meaningful for investors. Controversial to some existing literature in U.S. financial studies, Kellard et al. (2010) have demonstrated statistical evidence that the UK dividend ratios possess some predictive ability for equity premium, and that FTSE All-Share dividend ratios have relatively strong forecasting power than the S&P500 dividend ratios over the whole sample period. Campbell and Thompson (2007) argued that the empirical models can produce useful out-of-sample forecasts by restricting the model parameters.

Comparative to this argument, the kitchen sink regression models incorporating the relevant financial and economic variables do not only fail to beat the unconditional benchmark historical mean in a statistically significant approach for over three decades; also underperformed the prevailing forecasting model especially in the out-of-sample case, and concluded that the underperformed models could not guarantee investors with profitable information to time the market (Goyal and Welch, 2007).

The exploration of economic variables versus technical analysis is another crucial discourse in forecasting equity premium. The findings made by Neely et al. (2010) suggests that both economic fundamental and moving average rules provide statistical and economic significant evidence of forecasting gains in different proportions which appeared mostly in the U.S. business cycle recession periods. The resulting evidence for the forecasting gains produced by both techniques seems to be significant to a mean-variance investor. Neely et al. (2014) confirmed that both technical indicators and macroeconomic variables displayed statistically and economically significant evidence of in-sample and out-of-sample forecasting ability, with the technical indicators seemingly outperforming the macroeconomic variables. The empirical analysis suggests that the combination of both technical indicators and macroeconomic variables will significantly improve the equity risk premium forecasts rather than using either of the two information only. As a follow-up to the robustness of this finding, Baetje and Menkhoff (2016) argued that the predictive abilities of both indicators seem to possess similar quality when assessed by their respective long term forecast errors. Unlike the economic indicators that loses predictive ability on a long run, the technical indicators maintain or increase stability over time, and hence, the technical indicators consistently outperformed the economic indicators over time. The empirical findings in Rapach et al. (2010) confirmed that combination of forecasts yields statistically and economically significant out-of-sample gains consistently on a long run, as compared to the benchmark historical average. Thus, the forecasts combination approach maintains a long run statistical and economic stability in this direction.

In an attempt to determine the role of macroeconomic risk to the declining nature of the equity premium, Lettau et al. (2007) argued that a decline in macroeconomic risk leads to a fall in future equity risk premium, resulting to a boom in stock prices in the model economy. The introduction of economic constraints on the sign of coefficients and return forecasts, and the imposition of statistical constraints through shrinkage estimators in the out-of-sample models by Li and Tsiakas (2017) appeared to outperformed both the models conditioning the economic fundamentals and technical indicators. Lee et al. (2015) added that the asymptotic properties constrained local historical mean estimators appeared to minimize the asymptotic variance and the mean squared errors. The substantial nonlinearity is controlled by the local historical mean, and the local positive constraint improves the equity premium out-of-sample forecasts.

The application of bagging (bootstrap aggregation) to smooth parameter restrictions (positivity of the regression coefficients and positivity of the forecasts) by Hillebrand et al. (2009), Hillebrand et al. (2014) produced lower forecast errors than the forecast errors from the simple restricted and benchmark historical average models. In light of the empirical analysis in Rapach et al. (2010), Hillebrand et al. (2014) added that although simple forecasts combination do consistently well, but are not best at all times, owing to the noticeable improvement by introduction of bagging constraints. This led to the recommendation that forecasts combination could be improved by bagging. The outperformance of some notable techniques over the benchmark historical average, as mentioned in the existing literature suggests that the regression training techniques will play enormous role in forecasting the U.S. monthly equity premium. The application of sophisticated regression training techniques in this research is proposed to fill the identifiable gaps in the existing empirical literature, and to yield more consistent recursive out-of-sample forecasts with significant economic gains. Thus, the outcome of this research shall enrich empirical literature on forecastability of future equity premium and to guarantee investor's profit in the financial market.

3.3 Methodology

3.3.1 The Historical Average

Given a univariate time series $\{y_t\}_{t=1}^{T}$, with y_t denoting the monthly equity premium. The historical average (HA) model is defined as follows:

$$y_{t+1} = \beta + \epsilon_{t+1} \tag{3.1}$$

where β is a parameter representing the intercept; ϵ_t is a zero mean disturbance term; $t = 1, 2, ..., T$ (Campbell and Thompson, 2005; Lee et al., 2015). The least squares estimator (LSE) of the historical average is as follows:

$$\hat{\beta}_{LSE}^{HA} = \frac{1}{T}\sum_{t=1}^{T} y_t$$

which implies that the forecast for $\hat{y}_{T+1}$ is given by:

$$\hat{y}_{T+1|T} = \frac{1}{T}\sum_{t=1}^{T} y_t$$

where $\hat{\beta}_{LSE}^{HA}$ is the parametric estimator of β.

3.3.2 The Least Squares Regression Training

3.3.2.1 Kitchen Sink Model

Given a training dataset $\{y_t, x_{t,1}, x_{t,2}, ..., x_{t,k}\}_{t=1}^{T}$ of T statistical units, then a kitchen sink predictive linear model takes the form:

$$y_{t+1} = \beta_0 + \beta_1 x_{t,1} + \beta_2 x_{t,2} + ... + \beta_k x_{t,k} + \epsilon_{t+1} \quad t = 1, 2, ..., T \tag{3.2}$$

where y_{t+1} is the equity premium at $t+1$ (Rapach et al., 2010); $x_{t,1}, x_{t,2}, ..., x_{t,k}$ are the predictor variables available at the end of t used to predict y_{t+1}; β_0 is a constant term representing the intercept; $\beta_1, \beta_2, ..., \beta_k$ are the model coefficients; ϵ_{t+1} is a zero mean disturbance term (Goyal and Welch, 2007).
The above model can be represented in matrix form, as follows:

$$\mathbf{y} = \mathbf{X}\beta + \epsilon \tag{3.3}$$

where $\mathbf{y}$ is a $T \times 1$ vector of observed values; $\mathbf{X}$ is $T \times (k+1)$ matrix of predictor variables; β is $(k+1) \times 1$ dimensional parameter vector; ϵ is $T \times 1$ zero mean vector of disturbances.
If the parameters $\beta = (\beta_0, \beta_1, ..., \beta_k)$ are estimated by OLS, then the linear model (LM) forecasts can be obtained from the resulting kitchen sink predictive model:

$$\hat{y}_{T+h}(\hat{\beta}^{OLS}) = \mathbf{X}'_{T+h-1}\hat{\beta}^{OLS} \tag{3.4}$$

where $\hat{\beta}^{OLS} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$ is the OLS estimate of β.

3.3.2.2 The Partial Least Squares

The partial least squares (PLS) regression finds a set of latent vectors or components that performs a simultaneous decomposition of $\mathbf{X}$ and $\mathbf{y}$ with the constraint that these components explain as much as possible the covariance between $\mathbf{X}$ and $\mathbf{y}$. The latent vectors obtained from $\mathbf{X}$ are used to predict $\mathbf{y}$, where X is $T \times k$ matrix of T inputs and $\mathbf{y}$ is $T \times 1$ vector of response values.
Following the decomposition of $\mathbf{X}$ and $\mathbf{y}$ as a product of common set of orthogonal factors and a set of specific loadings, the predictor variables are decomposed as follows:

$$\mathbf{X} = \mathbf{S}\mathbf{L}' \quad with \quad \mathbf{S}'\mathbf{S} = \mathbf{I} \tag{3.5}$$

where $\mathbf{I}$ is the $T \times T$ identity matrix; $\mathbf{S}$ is the $T \times k$ score matrix; $\mathbf{L}$ is the non-orthogonal loading matrix with column vector of loadings k (Abdi, 2010).

Given two sets of weights $\mathbf{w}$ and $\mathbf{w}^*$, associated with a linear combination of the columns of $\mathbf{X}$ and $\mathbf{y}$, to attain maximum covariance. Our aim is to obtain a first pair of latent vectors $\mathbf{s}$ and $\mathbf{u}$ in the form:

$$\mathbf{s} = \mathbf{X}\mathbf{w} \quad \text{and} \quad \mathbf{u} = \mathbf{y}\mathbf{w}^* \tag{3.6}$$

with constraints $\mathbf{w}'\mathbf{w} = 1$, $\mathbf{s}'\mathbf{s} = 1$ and $\mathbf{s}'\mathbf{u}$ is maximal; where $\mathbf{w}$ denotes the $k \times 1$ weight vector; $\mathbf{s}$ and $\mathbf{u}$ denote the $T \times 1$ latent vectors respectively. When the first latent vector is obtained, it will be subtracted from both $\mathbf{X}$ and $\mathbf{y}$ and the procedure is re-iterated until it becomes a null matrix.

Using the nonlinear iterative partial least squares (NIPALS) algorithm, we create two matrices $\mathbf{E} = \mathbf{X}$ and $\mathbf{F} = \mathbf{y}$, which are then column centred and normalized; where $\mathbf{E}$ is $T \times k$ matrix of T inputs and $\mathbf{F}$ is $T \times 1$ vector of responses. The NIPALS algorithm takes the form (Rosipal and Trejo, 2001; Abdi, 2010):

```
Step 1: randomly initialize $\mathbf{u}$    (estimate $\mathbf{X}$ weights)
Step 2: $\mathbf{w} = \mathbf{E}'\mathbf{u}$    (estimate $\mathbf{X}$ factor scores and normalize)
Step 3: $\mathbf{s} = \mathbf{E}\mathbf{w}$,    $\mathbf{s} \leftarrow \frac{\mathbf{s}}{||\mathbf{s}||}$    (estimate $\mathbf{y}$ weights)
Step 4: $\mathbf{w}^* = \mathbf{F}'\mathbf{s}$    (estimate $\mathbf{y}$ scores)
Step 5: $\mathbf{u} = \mathbf{F}\mathbf{w}^*$,    $\mathbf{u} \leftarrow \frac{\mathbf{u}}{||\mathbf{u}||}$
Step 6: repeat steps 2 and 5 until $\mathbf{s}$ converges
Step 7: deflate matrices $\mathbf{E}$ and $\mathbf{F}$ in the form $\mathbf{E} = \mathbf{E} - \mathbf{s}\ell'$ and $\mathbf{F} = \mathbf{F} - b\mathbf{s}\mathbf{w}^{*\prime}$
```

where $b = \mathbf{s}'\mathbf{u}$ is the value of b used to predict $\mathbf{y}$ from $\mathbf{s}$; $\ell = \mathbf{E}'\mathbf{s}$ represent the factor loadings for $\mathbf{X}$.

The vectors $\mathbf{s}, \mathbf{u}, \mathbf{w}, \mathbf{w}^*$ and ℓ are then stored in the corresponding matrices and the scalar b is stored as a diagonal element of the regression coefficients β.

If $\mathbf{E}$ becomes a null matrix, then the whole set of latent vectors is obtained, otherwise the procedure is re-iterated from step 2 onward.
Thus, the response variable is predicted using the resulting regression model:

$$\hat{\mathbf{y}} = \mathbf{S}\beta\Lambda' = \mathbf{X}\beta_{PLS} \quad with \quad \beta_{PLS} = (\mathbf{L}'^{+})\beta\Lambda' \tag{3.7}$$

where $\mathbf{L}^{+}$ is the $T \times k$ matrix of the Moore-Penrose pseudo-inverse of $\mathbf{L}$ (Abdi, 2010); β_{PLS} is a $k \times 1$ vector of the diagonal matrix of the PLS regression coefficients; $\Lambda = (w_1^*, w_2^*, ..., w_k^*)'$ is a $k \times 1$ vector of weights.

KernelPLS

Consider a nonlinear transformation function of the k-dimensional input variables $\mathbf{x}$ into a feature space F, defined by $\Phi : \mathbf{x} \in R^k \rightarrow \Phi(\mathbf{x}) \in F$, where Φ is an $T \times k$ matrix of regressors. The kernel PLS algorithm is obtained by modifying the NIPALS algorithm using the matrix Φ of mapped input data (Rännar et al., 1994; Rosipal and Trejo, 2001):

Step 1: randomly initialize $\mathbf{u}$

Step 2: $\mathbf{s} = \Phi\Phi'\mathbf{u}, \quad \mathbf{s} \leftarrow \frac{\mathbf{s}}{||\mathbf{s}||}$

Step 3: $\mathbf{w}^* = \mathbf{y}'\mathbf{s}$

Step 4: $\mathbf{u} = \mathbf{y}\mathbf{w}^*, \quad \mathbf{u} \leftarrow \frac{\mathbf{u}}{||\mathbf{u}||}$

Step 5: repeat steps 2 and 5 until $\mathbf{s}$ converges

Step 6: deflate $\Phi\Phi'$ matrix and $\mathbf{y}$ vector respectively in the form: $\Phi\Phi' = (\Phi - \mathbf{s}\mathbf{s}'\Phi)(\Phi - \mathbf{s}\mathbf{s}'\Phi)'$ and $\quad \mathbf{y} = \mathbf{y} - \mathbf{s}\mathbf{s}'\mathbf{y}$.

We can apply the kernel trick $\Phi(x_i)'\Phi(x_j)' = \Gamma(x_i, x_j)$, where $\Phi\Phi'$ represents the $k \times k$ kernel Gram matrix Γ of the cross dot product between all mapped input data points $\{\Phi(x_i), \Phi(x_j)\}_{i,j=1}^k$. The deflation of the $\Phi\Phi' = \Gamma$ matrix after extraction of $\mathbf{s}$ component is as follows:

$$\Gamma \leftarrow (I - \mathbf{ss}')\Gamma(I - \mathbf{ss}') = \Gamma - \mathbf{ss}'\Gamma - \Gamma\mathbf{ss}' + \mathbf{ss}'\Gamma\mathbf{ss}' \tag{3.8}$$

where I is an k-dimensional identity matrix.
The matrix of the regression coefficients will take the form:

$$\mathbf{B} = \Phi' U (S'\Gamma U)^{-1} S'\mathbf{y} \tag{3.9}$$

and the prediction on the test data takes the form:

$$\hat{\mathbf{y}} = \Phi\mathbf{B} = \Gamma\mathbf{U}(S'\Gamma\mathbf{U})^{-1}\mathbf{S}'\mathbf{y} = \mathbf{SS}'\mathbf{y} \tag{3.10}$$

where the last part of the equation follows from the fact that the matrix of the components $\mathbf{S}$ may be expressed as $\mathbf{S} = \Phi\mathbf{R}$ with $\mathbf{R} = \Phi'\mathbf{U}(\mathbf{S}'\Gamma\mathbf{U})^{-1}$; $\mathbf{U}$ is $T \times k$ matrix of extracted components (Rosipal and Trejo, 2001).
Thus, the solution of the kernel PLS regression is as follows:

$$PLS_{kernel}(\mathbf{x}, \pi) = \sum_{i,j=1}^{k} \pi_i \Gamma(x_i, x_j) \tag{3.11}$$

where $\pi_i = \mathbf{U}(\mathbf{S}'\Gamma\mathbf{U})^{-'}\mathbf{S}'\mathbf{y}$ is a $k \times 1$ vector of partial least square estimators; $\Gamma(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{\sigma^2}}$, $||x_i - x_j||$ is the Euclidean distance between x_i and x_j, $\sigma^2 \in R^+$ is the bandwidth of the kernel function.

The wide kernel PLS (WideKernelPLS) takes the same form as the kernel PLS but differs in the number of components (ncomp) used as tuning parameter(s) of the model. The ncomp in the WideKernelPLS is usually less than the ncomp from the kernel PLS.

SparsePLS

The objective function for the first sparse PLS direction vector can be formulated by adding an ℓ_1 penalty norm constraint to the PLS model (3.6), with modification to obtain the following:

$$\max_{\mathbf{w}}(\mathbf{w}'\mathbf{M}\mathbf{w}) \quad \text{subject to} \quad \mathbf{w}'\mathbf{w}=1, \quad |\mathbf{w}| \leq \alpha \tag{3.12}$$

where $\mathbf{M} = \mathbf{X}'\mathbf{y}\mathbf{y}'\mathbf{X}$ is a $T \times T$ matrix and α determines the amount of sparsity. In order to obtain a sufficiently sparse solution, a generalization can be made that achieves the sparsity by imposition of an ℓ_1-penalty norm onto a surrogate of the direction vector $\mathbf{w}^*$, instead of the original direction $\mathbf{w}$, while keeping $\mathbf{w}$ and $\mathbf{w}^*$ close to each other, and it takes the form (Chun and Keleş, 2010):

$$\min_{\mathbf{w},\mathbf{w}^*}\{-\kappa\mathbf{w}'\mathbf{M}\mathbf{w}+(1-\kappa)(\mathbf{w}^*-\mathbf{w})'\mathbf{M}(\mathbf{w}^*-\mathbf{w})+\alpha_1||\mathbf{w}^*||_1+\alpha_2||\mathbf{w}^*||_2^2\} \quad \text{subject to} \quad \mathbf{w}'\mathbf{w}=1 \tag{3.13}$$

where κ is a thresholding parameter which represents the weight factor; α_1 and α_2 are the parameters regulating the amount of sparsity; $||\mathbf{w}^*||_1$ is the ℓ_1-penalty norm which encourages the sparsity on $\mathbf{w}^*$; $||\mathbf{w}^*||_2$ is the ℓ_2-penalty norm which addresses the potential singularity in $\mathbf{M}$ when solving for $\mathbf{w}^*$.

We can obtain a solution of the generalized SPLS by alternatively iterating between solving for $\mathbf{w}$ when $\mathbf{w}^*$ is fixed and solving for $\mathbf{w}^*$ when $\mathbf{w}$ is fixed.

In the first case, the objective function becomes:

$$\min_{\mathbf{w}}\{-\kappa\mathbf{w}'\mathbf{M}\mathbf{w}+(1-\kappa)(\mathbf{w}^*-\mathbf{w})'\mathbf{M}(\mathbf{w}^*-\mathbf{w})\} \quad \text{subject to} \quad \mathbf{w}'\mathbf{w}=1 \tag{3.14}$$

and when $0 < \kappa < \frac{1}{2}$, then we obtain:

$$\min_{\mathbf{w}}\{(\mathbf{Z}'\mathbf{w}-\hat{\kappa}\mathbf{Z}'\mathbf{w}^*)'(\mathbf{Z}'\mathbf{w}-\hat{\kappa}\mathbf{Z}'\mathbf{w}^*)\} \quad \text{subject to} \quad \mathbf{w}'\mathbf{w}=1 \tag{3.15}$$

where $\mathbf{Z} = \mathbf{X}'\mathbf{y}$ is a $T \times 1$ vector and $\hat{\kappa} = \frac{(1-\kappa)}{(1-2\kappa)}$ (Chun and Keleş, 2010).
Thus the resulting constrained least squares problem can be solved via the method of Lagrange multipliers and the solution is as follows:

$$\mathbf{w} = \hat{\kappa}(\mathbf{M} + \alpha^*\mathbf{I})^{-1}\mathbf{M}\mathbf{w}^* \tag{3.16}$$

where the multiplier α^* is the solution of $\mathbf{w}^{*\prime}\mathbf{M}(\mathbf{M} + \alpha\mathbf{I})^{-2}\mathbf{M}\mathbf{w}^* = \hat{\kappa}^2$.
Following Zou et al. (2006), for $\kappa = \frac{1}{2}$, the objective function reduces to $-\mathbf{w}'\mathbf{M}\mathbf{w}^*$ and the solution is $\mathbf{w} = \mathbf{U}\mathbf{v}'$, where $\mathbf{U}$ and $\mathbf{v}$ are obtained from the singular value decomposition of $\mathbf{M}\mathbf{w}^*$.
In the second case in which we solve for $\mathbf{w}^*$ when $\mathbf{w}$ is fixed, the minimization problem becomes:

$$\min_{\mathbf{w}^*}\{(\mathbf{Z}'\mathbf{w}^* - \mathbf{Z}'\mathbf{w})'(\mathbf{Z}'\mathbf{w}^* - \mathbf{Z}'\mathbf{w}) + \alpha_1||\mathbf{w}^*||_1 + \alpha_2||\mathbf{w}^*||_2^2\} \tag{3.17}$$

which is equivalent to the naive elastic net optimization problem proposed by Zou and Hastie (2005). It can be solved efficiently via the least angle regression algorithm proposed by Efron et al. (2004) when the expression $\mathbf{Z}'\mathbf{w}$ replaces $\mathbf{y}$ in the naive elastic net.

3.3.2.3 The Kernel-Based Regularized Least Squares

Let Γ be $T \times T$ symmetric kernel matrix whose i^{th} and j^{th} entry is denoted by $\gamma(\mathbf{x}_i, \mathbf{x}_j)$, measuring the pairwise similarities between each of the k covariate vectors $\mathbf{x}_i, \mathbf{x}_j$. Let $(\beta_1, \beta_2, ..., \beta_T)'$ be $T \times 1$ vector of choice coefficients and let $\mathbf{y} = (y_1, y_2, ..., y_T)'$ be $T \times 1$ vector of the outcome values. Then a mathematical relationship in the training dataset exists in vector form (Hainmueller and Hazlett, 2013):

$$\mathbf{y} = \Gamma\beta = \begin{pmatrix} \gamma(x_1, x_1) & \gamma(x_1, x_2) & \cdots & \gamma(x_1, x_T) \\ \gamma(x_2, x_1) & \gamma(x_2, x_2) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \\ \gamma(x_T, x_1) & \gamma(x_T, x_2) & \cdots & \gamma(x_T, x_T) \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_T \end{pmatrix} \tag{3.18}$$

We can employ the regularization proposed by Tychonoff (1963) to consider both model fit and the model complexity by choosing:

$$\underset{\psi \in \mathcal{H}}{\operatorname{argmin}} \sum_{t=1}^{T} \Big(L(y_t, \psi(\mathbf{x}_t)) \Big) + \lambda \mathcal{R}(\psi) \tag{3.19}$$

where $L(y_t, \psi(\mathbf{x}_t))$ is a loss function which computes how wrong the function is at each observation; $\mathcal{H}$ is hypothesis space of possible functions; $\mathcal{R}$ is a regularizer measuring the complexity of the function ψ; and $\lambda \in R^+$ is a parameter that determines the trade off between the model fit and complexity.

By choosing L as the squared loss function, and the regularizer $\mathcal{R}$ as the square of the ℓ_2-norm, we obtain:

$$\langle \psi, \psi \rangle_{\mathcal{H}} = ||\psi||_{\Gamma}^2 = \sum_i \sum_j \beta_i \beta_j \gamma(\mathbf{x}_i, \mathbf{x}_j) = \beta' \Gamma \beta \tag{3.20}$$

(Schölkopf et al., 2002; Ferwerda et al., 2017).

The hypothesis space $\mathcal{H}$ is the space of functions defined by $\mathbf{y} = \Gamma\beta$, and the optimization problem results to:

$$\beta^* = \underset{\beta \in R^k}{\operatorname{argmin}} (\mathbf{y} - \Gamma\beta)'(\mathbf{y} - \Gamma\beta) + \lambda \beta' \Gamma \beta \tag{3.21}$$

for which $\mathbf{y}^* = \Gamma\beta^*$ gives the fitting approximation, and the conditional expectation function $E[\mathbf{y}|X, \lambda]$ holds for a fixed λ (Ferwerda et al., 2015, 2017). This minimization is equivalent to a ridge regression.

Thus a closed-form solution is obtained from the optimization problem by taking the first derivative of the objective function with respect to β, and solving the resulting first order conditions, which eventually gives:

$$\beta^* = (\Gamma + \lambda I)^{-\prime}\mathbf{y} \tag{3.22}$$

where I is a $T \times T$ identity matrix.

The radial basis and polynomial kernel functions of the KRLS are as follows:

$$\Gamma(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} e^{-\dfrac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\eta^2}}, & \text{radial basis function} \\ (\mathbf{x}_i \cdot \mathbf{x}_j' + 1)^d, & \text{polynomial function} \end{cases} \tag{3.23}$$

where η denotes the kernel parameter; d denotes the kernel degree for the polynomial.

3.3.3 The Support Vector Regression

The support vector regression (SVR) is an extension of support vector machines (SVM) proposed by Boser et al. (1992).
Let $\{(\mathbf{x}_t, y_t)\}_{t=1}^{T}$ be a training data set with $\{(\mathbf{x}_t)\}_{t=1}^{T} \in R^k$ as observation samples of k predictor variables, and $\{(y_t)\}_{t=1}^{T} \in R$ as observation samples of the response variable, for $t = 1, 2, ..., T$. Then there exists a linear relationship of the form:

$$f(\mathbf{x}) = \mathbf{w}'\varphi(\mathbf{x}) + b^* \tag{3.24}$$

where $\mathbf{w}$ is the weight vector; $\mathbf{x}$ is a vector of regressors; b^* is a bias constant term (Kazem et al., 2013).
The task is to train the model to learn the data using a training sample and obtain the final model that can generalize the entire population. In this sense, the

parameters of the SVM model are estimated by minimizing:

$$\underset{b^* \in R}{min} \left\{ \frac{1}{2} \|\mathbf{w}\|^2 + \lambda \sum_{t=1}^{T} (\tau_t + \tau_t^*) \right\}$$

$$\text{subject to} \begin{cases} y_t - (\mathbf{w}'\varphi(\mathbf{x}) + b^* \leq \epsilon + \tau_t \\ \text{and} \\ (\mathbf{w}'\varphi(\mathbf{x}) + b^* - y_t \leq \epsilon + \tau_t^*; \tau_t, \tau_t^* \geq 0 \end{cases}$$

where $\|\mathbf{w}\|$ is the Euclidean weight vector norm; λ is the regularization constant determining the trade off between the training error and model complexity; τ_t, τ_t^* are slack variables controlling the model via the penalty factor, λ; ϵ is the tube size of the SVM model (Nalbantov et al., 2006; Plakandaras et al., 2015).
The minimization of the objective function of the SVR model can be represented as a maximization problem in dual form:

$$max \left\{ -\frac{1}{2} \sum_{t,s=1}^{T} (\alpha_t - \alpha_t^*)(\alpha_s - \alpha_s^*)\Gamma(\mathbf{x}_t, \mathbf{x}_s) + \sum_{t=1}^{T} (\alpha_t - \alpha_t^*) y_t - \epsilon \sum_{t=1}^{T} (\alpha_t + \alpha_t^*) \right\}$$

subject to $0 \leq \alpha_t, \alpha_t^* \leq \lambda; \sum_{t=1}^{T} (\alpha_t - \alpha_t^*) = 0, 1 = 1, 2, 3, ...T$ for the inequalities.
where $\Gamma(\mathbf{x}_t, \mathbf{x}_s)$ is the kernel function; α_t and α_t^* are non zero Lagrangian multipliers representing the solution to dual maximization problem (Nalbantov et al., 2006). The kernel function results to an inner product, which can be extended to other kernel functional forms.
Thus, the minimization of the objective function results to the following regression estimates:

$$f(\mathbf{x}) = \sum_{t=1}^{T} (\alpha_t^* - \alpha_t)\Gamma(\mathbf{x}_t, \mathbf{x}_s) + b^* \tag{3.25}$$

The minimization of the objective function can be done via the sequential minimal optimization (SMO) algorithm, see (Wen et al., 2018):

```
The SMO Algorithm for Support Vector Regression
Input $\Longrightarrow$ a training set with $T$ data point
Output $\Longrightarrow$ an optimal weight vector $\mathbf{w}$
Step 1: Initialize $w_t$ and $\mathbf{F}_t$,    for $t \leftarrow 1$ to $T$
Step 2: $w_t \leftarrow 0$,    $\mathbf{F}_t \leftarrow y_t\lambda$
Step 3: Repeat 1 and 2
Step 4: Search for $\mathbf{F_u}$ and determine $\mathbf{u}$
Step 5: Compute kernel values $\Gamma_\mathbf{u}$
Step 6: Search for $\mathbf{F_v}$ and determine $\mathbf{v}$
Step 7: Compute kernel values $\Gamma_\mathbf{v}$
Step 8: Update $\mathbf{w_u}$ and $\mathbf{w_v}$
Step 9: Update $\mathbf{F}$
Step 10: Search for $\mathbf{F}_{max}$
Step 11: Until $\mathbf{F_u} < \mathbf{F}_{max}$
```

The distinct kernels for the SVM are as follows:

$$\Gamma(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} \mathbf{x}_i'\mathbf{x}_j, & \text{linear function} \\ e^{-}\psi\,\|\mathbf{x}_i - \mathbf{x}_j\|^2, & \text{radial basis function} \\ (\psi\mathbf{x}_i'\mathbf{x}_j + \eta)^d, & \text{polynomial function} \end{cases} \tag{3.26}$$

where ψ, η are kernel parameters; d is the degree of the polynomial in the polynomial kernel function (Plakandaras et al., 2015).

3.3.4 Relevance Vector Regression

The relevance vector machine (RVM) applied multi kernel functions to a sparse linear model employing Bayesian inference to obtain parsimonious solutions for regression and probabilistic classification.

Let $\{(\mathbf{x}_1, \Im_1), (\mathbf{x}_2, \Im_2), ..., \mathbf{x}_t, (\Im_t), ..., (\mathbf{x}_T, \Im_T)\}$ be a training data set for RVM learning process, where $\{\mathbf{x}_t\}_{t=1}^{T} \in R^k$ denotes the k-dimensional input vector of predictors and $\{\Im_t\}_{t=1}^{T} \in R$ denotes the target outputs. The regression function of the RVM consisting of linear combination of the weighted kernel functions is as follows:

$$\Im = \mathbf{y}(\mathbf{x}; \mathbf{w}) = \sum_{t=1}^{T} w_t \Gamma(\mathbf{x}, \mathbf{x}_t) + w_0 \tag{3.27}$$

where $\mathbf{w}$ is a $T \times 1$ vector of weight parameters; $\Gamma(\mathbf{x}, \mathbf{x}_t)$ is a kernel basis function; w_0 is a bias term (Cummins et al., 2015; Lou et al., 2016).

The likelihood of all the data is obtained from the following product:

$$Pr(\Im|\sigma^2) = (2\pi\sigma^2)^{\frac{T}{2}} exp\left(\frac{-||\Im - \Phi\mathbf{w}||^2}{2\sigma^2}\right) \tag{3.28}$$

where $\mathbf{w} = (w_0, w_1, w_2, ..., w_T)'$; $\Phi = (\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), ..., \phi(\mathbf{x}_T))'$; and
$\phi(\mathbf{x}_j) = [1, \Gamma(\mathbf{x}_j, \mathbf{x}_1), \Gamma(\mathbf{x}_j, \mathbf{x}_2), ..., \Gamma(\mathbf{x}_j, \mathbf{x}_T)]'$.
Let α_t be separate hyperparameter for each of the weight parameters w_t, so that the weight prior will be defined as follows:

$$Pr(\mathbf{w}|\alpha) = \prod_{t=0}^{T} \frac{\alpha_t}{\sqrt{2\pi}} exp\left(\frac{-\alpha_t w_t^2}{2}\right) \quad \text{for} \quad \alpha = (\alpha_1, \alpha_2, ..., \alpha_T) \tag{3.29}$$

The posterior distribution over all the unknown parameters can be obtained from the Bayes' rule:

$$Pr(\mathbf{w}, \alpha, \sigma^2|\Im) = \frac{Pr(\Im|\mathbf{w}, \alpha, \sigma^2)Pr(\mathbf{w}, \alpha, \sigma^2)}{Pr(\Im)} \tag{3.30}$$

Let $\mathbf{x}_*$ be a new test vector of predictor, then we can obtain a prediction for the corresponding target $\Im_*$ in terms of the predictive distribution, defined as follows:

$$Pr(\Im_*|\Im) = \int Pr(\Im_*|\mathbf{w}, \alpha, \sigma^2)Pr(\mathbf{w}, \alpha, \sigma^2|\Im)d\mathbf{w}d\alpha d\sigma^2 \tag{3.31}$$

The posterior $Pr(\mathbf{w}, \alpha), \sigma^2|\Im$, can be decomposed as follows:

$$Pr(\mathbf{w}, \alpha), \sigma^2|\Im = Pr(\mathbf{w}|\Im, \alpha, \sigma^2)Pr(\alpha, \sigma^2|\Im) \tag{3.32}$$

The posterior distribution over the weights is defined as follows:

$$Pr(\mathbf{w}|\Im, \alpha, \sigma^2) = \frac{Pr(\Im|\mathbf{w}, \sigma^2)Pr(\mathbf{w}|\alpha)}{\int Pr(\Im|\mathbf{w}, \sigma^2)Pr(\mathbf{w}|\alpha)} = (2\pi)^{-\left(\dfrac{T+1}{2}\right)}|\Sigma|^{-\dfrac{1}{2}} exp\left(\frac{-(\mathbf{w}-\mu)'\Sigma^{-1}(\mathbf{w}-\mu)}{2}\right) \tag{3.33}$$

where $\mu = \sigma^{-2}\Sigma\Phi'\Im$ is the posterior mean; $\Sigma = (D + \sigma^{-2}\Phi'\Phi)'$ is the posterior covariance; $D = diag(\alpha_0, \alpha^1, ..., \alpha_T)$ (Tipping, 2001; Huang and Wu, 2008; Lou et al., 2016; Zhou et al., 2017). Thus, the estimated value of RVM model weights is the maximum posterior (MP) estimate of the weights.

The marginal likelihood for the hyperparameter can be obtained by integrating the weights, as follows:

$$Pr(\Im|\alpha, \sigma^2) = (2\pi)^{-\dfrac{T}{2}}|\Sigma|^{-\dfrac{1}{2}} exp\left(\frac{-\Im'\Sigma^{-1}\Im}{2}\right) \tag{3.34}$$

where $\Sigma = \sigma^2 I + \Phi D^{-1}\Phi$.

Our aim is to maximize the likelihood for the hyperparameters with respect to α and σ^2. For a new vector of predictors $\mathbf{x}_*$, the predictive distribution can be computed as follows:

$$Pr(\alpha_{MP}, \sigma^2_{MP}) = \arg\max_{\alpha, \sigma^2} Pr(\alpha, \sigma^2|\Im) \tag{3.35}$$

and the probability distribution for the corresponding output is:

$$Pr(\Im_*|\Im, \alpha_{MP}, \sigma^2_{MP}) = \int Pr(\Im_*|\mathbf{w}, \alpha_{MP}, \sigma^2_{MP})Pr(\mathbf{w}, \alpha_{MP}, \sigma^2_{MP}|\Im)d\mathbf{w} \tag{3.36}$$

where both terms in the integrand are Gaussian; and MP is maximum posterior (Lou et al., 2016).

Thus, we obtained the final predictions as follows:

$$Pr(\Im_*|\Im) \approx N(\Im_*|\mathbf{y}_*, \sigma_*^2) \quad \text{and} \quad \mathbf{y}_* = \mu'\Phi(\mathbf{x}_*) \tag{3.37}$$

where $\sigma_*^2 = \sigma_{MP}^2 + \Phi(\mathbf{x}_*)'\Sigma\Phi(\mathbf{x}_*)$; and $\Phi(\mathbf{x}_*) = [1, \Gamma(\mathbf{x}_*, \mathbf{x}_1), \Gamma(\mathbf{x}_*, \mathbf{x}_2), ..., \Gamma(\mathbf{x}_*, \mathbf{x}_T)]$.

The multiple kernel functions of the RVM are as follows:

$$\Gamma(\varphi_i(\mathbf{x}), \varphi_j(\mathbf{x})) = \begin{cases} \langle \varphi_i(\mathbf{x}_t), \varphi_j(\mathbf{x}_t) \rangle, & \text{linear function} \\ e^{-\dfrac{\|\varphi_i(\mathbf{x}_t) - \varphi_j(\mathbf{x}_t)\|^2}{\eta^2}}, & \text{radial basis function} \\ (\varphi_i(\mathbf{x}_t) \cdot (\varphi_j(\mathbf{x}_t))')^d, & \text{polynomial function} \end{cases} \tag{3.38}$$

where η denotes the kernel parameter; d denotes the kernel degree for the polynomial.

The algorithm for the RVM training and forecasting procedures suggested by Nicolaou et al. (2012) can be summarized as follows:

The Algorithm for Relevance Vector Regression

Training Stage: training set $\{(\mathbf{x}_t, y_t)\}_{t=1}^{T}$, $t = 1, 2, ..., T$

Step 1: Obtain output features $\mathbf{y}_t^{\mathbf{v}}$

Step 2: Construct basis matrix $\Omega_{\mathbf{w},\mathbf{u}} = \left(\Omega_{\mathbf{w}}|\Omega_{\mathbf{u}}^{\mathbf{v}}\right)$

(a) Apply kernel $\Gamma_{\mathbf{w}}$ to obtain Ω_{ω} for input feature $\mathbf{x}$

(b) Apply kernel $\Gamma_{\mathbf{u}}$ to obtain $\Omega_{\mathbf{u}}^{\mathbf{v}}$ for output feature $y^{\mathbf{v}}$

Step 3: Using marginal likelihood maximization

(a) Determine the hyperparameters (θ, μ, σ^2)

(b) $\mu = \sigma^2\Sigma\Omega_{\mathbf{w},\mathbf{u}}'S$ and $\Sigma = \left(\sigma^2\Omega'\Omega + D\right)^{-\prime}$

Forecasting Stage: validation set

Step 4: Obtain output features $\mathbf{y}_*^{\mathbf{v}}$

Step 5: Forecast and estimate the variance

(a) $S_* = \mu'_{\mathbf{w},\mathbf{u}}\Big[\Omega_{\mathbf{w}}(\mathbf{x}_*)|\Omega_{\mathbf{u}}(\mathbf{y}^{\mathbf{v}}_*)\Big]$

(b) $\sigma^2_* = \sigma^2 + \Big[\Omega_{\mathbf{w}}(\mathbf{x}_*)|\Omega_{\mathbf{u}}(\mathbf{y}^{\mathbf{v}}_*)\Big]'\Sigma\Big[\Omega_{\mathbf{w}}(\mathbf{x}_*)|\Omega_{\mathbf{u}}(\mathbf{y}^{\mathbf{v}}_*)\Big]$

3.3.5 The Regularized or Penalized Regression

3.3.5.1 The Ridge

The ridge regression model aimed to reduce multicollinearity owing to the presence of ℓ_2-penalty norm and the cross-validated tuning parameter, λ_1, which regulates the amount of shrinkage imposed on the model coefficients.

Using the training set $\{(x_{t,1}, y_1), (x_{t,2}, y_2), ..., (x_{T,k}, y_T)\}$, and by imposition of ridge constraints, the model parameter estimates will be obtained by minimizing the objective function

$$\sum_{t=1}^{T-1}(y_{t+1} - \beta_0 - \sum_{j=1}^{k}\beta_j x_{t,j})^2 \text{ subject to} \sum_{j=1}^{k}\beta_j^2 \leq s_1 \; for \; s_1 \in \lambda_1 \tag{3.39}$$

which is a convex optimization problem, hence the solution has a closed form (James et al., 2013).

The ridge model parameter estimates will be:

$$\hat{\beta}^{Ridge} = \underset{\beta \in R^{k+1}}{argmin}\left\{\sum_{t=1}^{T-1}(y_{t+1} - \beta_0 - \sum_{j=1}^{k}\beta_j x_{t,j})^2 + \lambda_1\sum_{j=1}^{k}\beta_j^2\right\}$$

$$\implies \hat{\beta}^{Ridge} = (\mathbf{X}'\mathbf{X} + \lambda_1\mathbf{I}_k)^{-1}\mathbf{X}'\mathbf{y}$$

which is always invertible, and hence non-singular (Ahn et al., 2012); where $\mathbf{X}$ is the $T \times k$ matrix of covariates; $\lambda_1\sum_{j=1}^{k}\beta_j^2 = \lambda_1 \|\beta\|_2^2$ is the shrinkage penalty; $\sum_{j=1}^{k}|\beta_j^2| = \|\beta\|_2^2$ is the ℓ_2-norm of the vector β; $\lambda_1 > 0$ is the ridge tuning parameter; β_0 is the intercept; $\beta_1, \beta_2, ..., \beta_k$ are the ridge coefficients; I_k is a $k \times k$ identity matrix; k is the number of parameters to be estimated; T is the sample size; $j = 1, 2, ..., k$.

Thus, the ridge forecasts are obtained from the resulting forecasting model:

$$\hat{\mathbf{y}}_{t+1|T}^{Ridge}(\hat{\beta}^{Ridge}) = \mathbf{X}_t' \hat{\beta}^{Ridge} \tag{3.40}$$

where

$$\hat{\beta}^{Ridge} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \mathbf{X}_t' \beta)^2 + \lambda_1 \sum_{j=1}^{k} \beta_j^2 \right\}$$

β is $(k+1) \times 1$ vector of unknown parameters, including the intercept; T is the sample size.

The ridge forecasts converges to the sample mean for large values of tuning parameter, λ_1:

$$\hat{\mathbf{y}}_{t+1|T}^{Ridge} \rightarrow \frac{1}{T-1} \sum_{t=2}^{T} y_t \quad \text{when} \quad \lambda_1 \rightarrow \infty.$$

The Forward - Backward Ridge

We also employed the forward-backward (FOBA) ridge which is an extension of the ridge model. The FOBA is a regularization model that implements the forward and backward or both directional sparse learning algorithms for the ridge regression model. In this case, $\varrho \in (0, 1)$ controls how likely the steps are to be taken, determing either addition or deletion of a variable in the rdige model. The FOBA method takes a backward step when the ridge penalized risk increase is less than ϱ times the ridge penalized risk reduction in the corresponding forward step, and vice versa.

3.3.5.2 The Least Absolute Shrinkage and Selection Operator

The least absolute shrinkage and selection operator (LASSO) as introduced by Tibshirani (1996) aimed to shrink the model coefficients toward zero, hence performing both variable selection and model interpretability, owing to the presence of ℓ_1-penalty norm. The LASSO does not admit a closed-form solution because of the ℓ_1-penalty which makes it nonlinear in the $\mathbf{y}_t's$; hence the constraint minimization of the LASSO is a quadratic programming problem whose solution can be obtained by efficient approximation.
The LASSO model parameter estimates are obtained by minimizing the objective function:

$$\sum_{t=1}^{T-1}(y_{t+1} - \beta_0 - \sum_{j=1}^{k}\beta_j x_{t,j})^2 \text{ subject to} \sum_{j=1}^{k}|\beta_j| \leq s_2 \; for \; s_2 \in \lambda_2 \tag{3.41}$$

where $\lambda_2 > 0$ is the LASSO tuning parameter; β_0 is the intercept; $\beta_1, \beta_2, ..., \beta_k$ are the LASSO coefficients; $\sum_{j=1}^{k}|\beta_j| = \|\beta\|_1$ is the ℓ_1-norm of the vector β (Sagaert et al., 2018).

The LASSO model parameter estimates will be:

$$\hat{\beta}^{LASSO} = \underset{\beta \in R^{k+1}}{argmin}\left\{\sum_{t=1}^{T-1}(y_{t+1} - \beta_0 - \sum_{j=1}^{k}\beta_j x_{t,j})^2 + \lambda_2\sum_{j=1}^{k}|\beta_j|\right\}$$

where $\lambda_2 \sum_{j=1}^{k}|\beta_j| = \lambda_2 \|\beta\|_1$ is the shrinkage penalty; and $\hat{\beta}^{LASSO} \to \hat{\beta}^{OLS}$ as $\lambda_2 \to \infty$

Thus, the LASSO forecasts are obtained from the resulting LASSO forecasting model:

$$\hat{\mathbf{y}}_{t+1|T}^{LASSO}(\hat{\beta}^{LASSO}) = \mathbf{X}_t'\hat{\beta}^{LASSO} \tag{3.42}$$

where

$$\hat{\beta}^{LASSO} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \mathbf{X}_t' \beta)^2 + \lambda_2 \sum_{j=1}^{k} |\beta_j| \right\}$$

where $\mathbf{X}$ is the $T \times k$ matrix of covariates; β is $(k+1) \times 1$ vector of unknown parameters, including the intercept; T is the sample size; λ_2 controls the amount of shrinkage (Elliott et al., 2013; MartíNez-MartíNez et al., 2011).

3.3.5.3 The Relaxed Least Absolute Shrinkage and Selection Operator

The relaxed least absolute shrinkage and selection operator (RELAXO) is a generalization of the LASSO for linear regression.
Let λ and α be two separate parameters for controlling model selection and shrinkage estimation. The RELAXO estimator can be defined for $\lambda \in [0, \infty)$ and $\alpha \in (0, \infty]$ as follows:

$$\hat{\beta}^{RELAXO} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \mathbf{X}_t' \{\beta \cdot \mathbf{1}_{\mathcal{S}_\lambda}\})^2 + \alpha\lambda \left|\beta\right|_1 \right\}$$

where $\mathcal{S}_\lambda$ is the set of predictor variables selected by LASSO estimator; $\mathbf{1}_{\mathcal{S}_\lambda}$ is the indicator function on the set of predictor variables; $\alpha\lambda \left|\beta\right|_1$ is the shrinkage penalty for the RELAXO (Meinshausen, 2007). It can be expressed as follows:

$$\{\mathbf{1}_{\mathcal{S}_\lambda}\}_k = \begin{cases} 0, & \text{k} \notin \mathcal{S}_\lambda \\ \beta_k, & \text{k} \notin \mathcal{S}_\lambda \end{cases}$$

Let $\mathcal{L}(\beta)$ be the negative log-likelihood under the parameter β, then the generalized RELAXO estimator takes the form (Meinshausen, 2007):

$$\hat{\beta}^{RELAXO} = \underset{\beta \in \mathcal{S}_\lambda}{argmin} \left\{ \mathcal{L}(\beta) + \alpha\lambda \left|\beta\right|_1 \right\}$$

Thus, the RELAXO forecasts are obtained from the resulting RELAXO forecasting model:

$$\hat{\mathbf{y}}_{t+1|T}^{RELAXO}(\hat{\beta}^{RELAXO}) = \mathbf{X}_t'\hat{\beta}^{RELAXO} \tag{3.44}$$

where

$$\hat{\beta}^{RELAXO} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \mathbf{X}_t'\{\beta \cdot \mathbf{1}_{\mathcal{S}_\lambda}\})^2 + \alpha\lambda \left|\beta\right|_1 \right\}$$

where β is $(k+1) \times 1$ vector of unknown parameters, including the intercept; $\mathcal{S}_\lambda$ is the set of predictor variables selected by LASSO estimator; $\mathbf{1}_{\mathcal{S}_\lambda}$ is the indicator function on the set of predictor variables.

3.3.5.4 The Elastic Net

The elastic net, as proposed by Zou and Hastie (2005) combines both the ℓ_1 and ℓ_2 penalty vector norms, and tends to eliminate extreme solutions. Thus the elastic net model parameter estimates are obtained by minimizing the objective function that includes the ridge and LASSO shrinkage penalties subject to both constraints, which results to:

$$\hat{\beta}^{ElNet} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \beta_0 - \sum_{j=1}^{k} \beta_j x_{t,j})^2 + \lambda_2 \Big(\sum_{j=1}^{k} (1-\lambda_1)\beta_j^2 + \lambda_1|\beta_j| \Big) \right\}$$

where λ_1 is the ridge tuning parameter; λ_2 is the LASSO tuning parameter (Wu and Yang, 2014).
It is worth noting that the Elastic Net is Ridge if $\lambda_1 = 0$; it is LASSO if $\lambda_1 = 1$ and it is strictly convex if $\dfrac{\lambda_2}{\lambda_1 + \lambda_2} > 0$ (Bai and Ng, 2008)

Therefore, the elastic net forecasts are obtained from the elastic net forecasting model:

$$\hat{\mathbf{y}}^{ElNet}_{t+1|T}(\hat{\beta}^{ElNet}) = \mathbf{X}'_t \hat{\beta}^{ElNet} \tag{3.45}$$

where

$$\hat{\beta}^{ElNet} = \underset{\beta \in R^{k+1}}{argmin} \left\{ \sum_{t=1}^{T-1} (y_{t+1} - \mathbf{X}'_t \beta)^2 + \lambda_2 \Big(\sum_{j=1}^{k} (1 - \lambda_1)\beta_j^2 + \lambda_1 |\beta_j| \Big) \right\}$$

β is $(k+1) \times 1$ vector of unknown parameters including the intercept; and $\mathbf{X}$ is the $T \times k$ matrix of covariates.

3.3.5.5 The Least Angle Regression

The least angle regression (LARS), introduced by Efron et al. (2004) is a machine learning model selection algorithm for fitting linear regression models to high dimensional data. There is similarity between the LARS algorithm and forward stepwise regression, and the parameter estimates are increasing in an equiangular direction to each of the corresponding correlations associated with the model residuals.
The LARS algorithm adapted from Alfons et al. (2016) and Efron et al. (2004) is summarized as follows:

- Begin with all coefficients $\beta_j = 0$;
- Determine the predictor X_t most correlated with the response variable y_t;
- Increase the coefficient β_j in the direction of the sign of its correlation;
- Increase (β_j, β_m) in their joint LS direction until another predictor X_s has as much correlation with the residual, where β_m is a coefficient for $j \neq m$ and $j, m = 1, 2, ..., k$; and k is the number of covariates;
- Continue until all predictors are in the model.

The LARS2 is a special improved case of the LARS that uses *step* as the tuning parameter instead of *fraction*. The LARS2 adopts an infinitesimal forward stagewise regression approach in that each of the covariate is examined from the set of covariates in the model. The *step* as a model tuning parameter serves as a step-by-step tool, determining the inclusion or exclusion of each covariate, and hence the stepwise selection approach. Suppose the response variable is to be determined by a linear combination of a subset of potential covariates, then the LAR2 algorithm will provide a suitable means of producing an estimate about the covariate(s) to be included together with their respective coefficients.

3.3.6 Components Regression

3.3.6.1 The Principal Component Regression

The principal component regression (PCR) is based on principal component analysis aimed at dimensionality reduction of a given multivariate dataset.
Let $\mathbf{y} = \{y_t\}_{t=1}^{T}$ be a $T \times 1$ vector of observed outcomes and $X = \{x_{t,j}\}_{t=1,j=1}^{t=T,j=k}$ be a $T \times k$ data matrix of observed covariates. Then a PCR can be performed from a principal component analysis (PCA) on the centered data matrix X.
Let $X = P\Delta Q'$ represent the singular value decomposition of X;

$\Delta_{k\times k} = diag(\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_k)$ with $\mathbf{w}_1 \geq \mathbf{w}_2 \geq ... \geq \mathbf{w}_k \geq 0$ representing the non-negative singular values of X, such that the columns of $P_{k\times k} = (p_1, p_2, ..., p_k)$ and $Q_{k\times k} = (q_1, q_2, ..., q_k)$ are orthonormal sets of vectors respectively, representing the left and right singular vectors of X (Ince and Trafalis, 2007; Mor-Yosef and Avron, 2019).

Then $Q\Lambda Q'$ gives a spectral decomposition of $X'X$; $\Lambda_{k\times k} = diag(\vartheta_1, \vartheta_2, ..., \vartheta_k)$ equal to

$diag(\mathbf{w}_1^2, \mathbf{w}_2^2, ..., \mathbf{w}_k^2)$=$\Delta^2$; $\vartheta_1 \geq \vartheta_2 \geq ... \geq \vartheta_k \geq 0$ represent the non-negative eigenvalues of $X'X$; and $column(Q)$ represent the corresponding orthonormal set of eigenvectors.

We can obtain the j^{th} principal component represented by Xq_j and the j^{th} PCA loading respectively, corresponding to the j^{th} largest principal value ϑ_j for each $j \in \{1, 2, ..., k\}$.

If $\hat{\theta}_k = (Z_k' Z_k)^{-1} Z_k' \mathbf{y} \in R^k$ is the OLS vector of estimated coefficients with $Z_k = XQ_k = (Xq_1, Xq_2, ..., Xq_k)$, then for any $m \in \{1, 2, ..., k\}$, the final PCR estimator of β based on the first m principal components is obtained as follows:

$$\hat{\beta}_m = Q_m \hat{\theta}_m \in R^k \tag{3.46}$$

(Olivieri, 2018).

The *super principal component regression* (superpc) is a special case of the principal component regression (PCR) that uses a selected subset of the covariates based on connectivity with the target outcome.

3.3.6.2 Independent Component Regression

Independent component regression (ICR) is a useful model for separating a multivariate signal into additive subcomponents.

Given a multivariate data matrix $X = (x_{t,1}, x_{t,2}, ..., x_{t,k})'$ with latent components $\mathbf{c} = (c_1, c_2, ..., c_T)'$, $T \geq k$. Then the task is to systematically transform the observed multivariate data X, using an unknown mixing matrix Ψ, in the form:

$$\mathbf{c} = \Psi X$$

into an observable vector of maximal statistically independent components $\mathbf{c}_t$, measured by some independent functions $(c_1, c_2, ..., c_T)$, and assuming linear noiseless holds. Then the components of the observed multivariate data matrix X are generated as the sum of the independent components $\mathbf{c}_s, s = 1, 2, ..., T$ weighted by the mixing weights $\alpha_{t,s}$ such that:

$$\mathbf{x}^{(t)} = \Phi\mathbf{c} = \sum_{s=1}^{T} \alpha_s \mathbf{c}_s = \alpha_1 c_1 + \alpha_2 c_2 + ... + \alpha_s c_s + ... + \alpha_T c_T \tag{3.47}$$

where $\alpha_s = (\alpha_1, \alpha_2, ..., \alpha_T)'$ represent the $T \times 1$ basis vector; Φ represent the $T \times k$ mixing matrix whose entries obtained from the basis vector α_s; $\mathbf{c} = (c_1, c_2, ..., c_T)'$ represent the $T \times 1$ latent components (Matilainen, 2018).

Generally, the independent components are obtained by the product of the unknown mixing matrix Ψ and the multivariate data matrix X, resulting to the model:

$$\mathbf{y} = \Psi X$$

where $\mathbf{y} = (y_t), t = 1, 2, ..., T$ is $T \times 1$ response values and y_t represent the statistically independent components; Ψ is the unknown mixing matrix obtained by the inverse of mixing matrix Φ: $\Psi = \Phi^{-1}$; X is the $T \times k$ matrix of covariates (Lu et al., 2009; Tang, 2019).

Thus, the non-Gaussianity of the independent components is defined by the negentropy:

$$J(\mathbf{y}) = S(\mathbf{y}_{Gaussian}) - S(\mathbf{y}) \tag{3.48}$$

where $\mathbf{y}_{Gaussian}$ is the Gaussian random vector such that $Cov(\mathbf{y}_{Gaussian}) = Cov(\mathbf{y})$; S is the entropy of a random vector $\mathbf{y}$ with density function $\xi(\mathbf{y})$, defined by:

$$S(\mathbf{y}) = -\int_{-\infty}^{\infty} \xi(\mathbf{y}) \log \xi(y) d\mathbf{y}$$

Using the approximation proposed in Lu et al. (2009), the neg-entropy above results to:

$$J(\mathbf{y})[E(G(\mathbf{y})) - E(G(\mathbf{u}))]^2 \tag{3.49}$$

where $G(\mathbf{y})$ is an exponential function of $\mathbf{y}$, defined by $G(\mathbf{y}) = e^{-\frac{1}{2}\mathbf{y}^2}$; $\mathbf{y}$ is a standard Gaussian distributed variable; $\mathbf{u}$ is a random variable having the same standard Gaussian distribution form; $E(\cdot)$ and $E(\cdot)$ denote the expected value of $\mathbf{y}$ and $\mathbf{u}$ respectively.

3.3.7 Gaussian Processes Regression

Let $\mathbf{x} = (x_1, x_2, ..., x_k)'$ be input vector of covariates and y denotes the response variable. Then the Gaussian process can be modelled as a distribution over a function Γ, which maps the input space , to $\Re$:

$$\Gamma :\longrightarrow \Re$$

such that for any finite subset $S \subset$, its marginal distribution $\Phi(\Gamma(x_1), \Gamma(x_2), ..., \Gamma(x_k))$ is a multivariate normal distribution (Rasmussen, 2004).

By parameterization, let $\mu_{\mathbf{x}}$ denotes the mean function and $\Omega(\mathbf{x}_i, \mathbf{x}_j)$ denotes the covariance function, such that:

$$\Gamma(\mathbf{x}) \sim Gaussian(\mu_{\mathbf{x}}, \Omega(\mathbf{x}_i, \mathbf{x}_j)) \Rightarrow \Gamma|X \sim N(\mu_{\mathbf{x}}, \Omega(\mathbf{x}_i, \mathbf{x}_j))$$

where Γ is a vector valued function; $\mu_{\mathbf{x}}$ is the mean function; $\Omega(\mathbf{x}_i, \mathbf{x}_j)$ is the $k \times k$ covariance matrix, for $\Omega_{i,j} = \Omega(\mathbf{x}_i, \mathbf{x}_j) = Cov(\mathbf{x}_i, \mathbf{x}_j)$.

If y denote an observation with a Gaussian distribution noise $\epsilon \sim N(0, \sigma_k^2)$. Then the resulting Gaussian process regression model is as follows:

$$y = \Gamma(\mathbf{x}) + \epsilon; \quad \epsilon \sim N(0, \sigma_k^2) \tag{3.50}$$

The joint distribution of the training and test outputs denoted respectively by y and Γ^*, with zero mean function will be:

$$\begin{pmatrix} y \\ \Gamma^* \end{pmatrix} \sim N\left(0, \begin{pmatrix} \Omega(X,X) + \sigma_k^2 I_k & \Omega(X,X^*) \\ \Omega(X^*,X) & \Omega(X^*,X^*) \end{pmatrix}\right)$$

where X is a design matrix for the training data; X^* is a design matrix for the test data (Singh, 2016; Sheng et al., 2018).

The predictive distribution can be obtained by conditioning Γ^* on X, y and X^*:

$$\Gamma^* | X, y, X^* \sim N(\hat{\mu}_{\Gamma^*}, \hat{\sigma}_{\Gamma^*})$$

where $\hat{\mu}_{\Gamma^*} = \Omega(X^*,X)[\Omega(X,X) + \sigma_k^2 I_k]^{-1} y$ is the estimated mean; $\hat{\sigma}_{\Gamma^*} = \Omega(X^*,X^*) - \Omega(X^*,X)[\Omega(X,X) + \sigma_k^2 I_k]^{-1}\Omega(X,X^*)$ is the estimated variance. Thus, the mean prediction is a linear combination of the noisy observation.

The kernel functions of the Gaussian processes regression (GPR) are as follows:

$$\Gamma(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} \sum_{m=1}^{M} \sigma_m^2 x_{i,m} x_{j,m}, \text{linear function} \\ e^{-\frac{1}{2}\left(\frac{r}{m}\right)^2}, \text{radial basis function} \\ (\mathbf{x}_i \cdot \mathbf{x}_j' + \sigma_0^2)^d, \text{polynomial function} \end{cases} \tag{3.51}$$

where $m > 0$ is a length-scale parameter for $i, j, m = 1, 2, ..., k$; r denotes a radius vector with the same or different number of dimensions as the input covariates $\mathbf{x}$;

d denotes the kernel degree for the polynomial function (Rasmussen, 2006).

3.3.8 Regression Splines

The multivariate adaptive regression splines (MARS), as introduced by Friedman (1991) is a non-parametric regression technique for handling non-linearities and interactions between covariates in a regression model.

Given a k-dimensional vector of covariates $\mathbf{x} = (x_1, x_2, ..., x_k)'$ and a response variable y, related in the form:

$$y = \phi(\mathbf{x}) + \epsilon \tag{3.52}$$

where ϵ is a normally distributed error term. A flexible model can be estimated by reflected pairs, known as piecewise linear basis functions:

$$(x-\tau)_+ = \begin{cases} x-\tau, & \text{if } x > \tau \\ 0, & otherwise \end{cases}$$

and

$$(\tau-x)_+ = \begin{cases} \tau-x, & \text{if } x < \tau \\ 0, & otherwise \end{cases}$$

where $+$ denotes the positive part; τ denotes the breaking or knot point (Kooperberg, 2006; Koc and Bozdogan, 2015).

Let T be the sample size for the training set, then a reflected pairs for each predictor variable $\{x_j\}_{j=1}^{k}$ with breaking points at each observed values $\{x = x_{t,j}\}_{i=1,j=1}^{t=T,j=k}$ of the specific variable can be formed, and the resulting set of all possible reflected pairs with their respective breaking points is defined as:

$$\mathbf{S} = \Big\{(x_j - \tau)_+, \quad (\tau - x)_+ | \tau \in \{x_{t,j}\}_{t=1,j=1}^{t=T,j=k}\Big\}$$

The MARS employed the functional covariates in the set **S**, and their product (for multivariate case), instead of the original covariates, to form the approximate model:

$$\phi(x) = \delta_0 + \sum_{\ell=1}^{p} \delta_\ell \mathbf{W}_\ell(x), \quad \ell \in \{1, 2, ..., p\} \tag{3.53}$$

where $\mathbf{W}_\ell(x)$ is a weighted sum of basis function from **S**; ℓ is the number of basis functions in the resulting model; δ_ℓ represent the constant coefficients; δ_0 represent the intercept (Zhang and Goh, 2016).

For the avoidance of trademark infringements, some open source implementations of MARS are referred to as *Earth*, with alternative usage method values *bagEarth* for bagging MARS, and *gcvEarth* for generalized cross-validated MARS used as a form of regularization to trade off goodness of fit against the model complexity.

3.3.9 Cubist Regression

The cubist regression, as introduced by Quinlan et al. (1992) is a rule-based regression model in which a tree is grown where the terminal leaves contain linear regression, and extensive models are built based on covariates used during previous splits.

Given the regression model at each terminal node of the regression tree, then the covariates can be made recursively and the tree is reduced to a set of decision rules drawn from the top to bottom of the regression tree, see Kuhn et al. (2016) for the computational details. Thus the learning and decision rules can be eliminated by pruning or pairwise combination.

Model committees are usually created by generating a sequence of rule-based models similar to boosting approach. The training set outcome is adjusted based on the prior model fit and then builds a new set of rules using pseudo response (Boston and Kuhn, 2000; Zhou et al., 2019):
The $\mathbf{k}^{th}$ committee model uses an adjusted response defined by:

$$y_t^{(\mathbf{k})} = 2y_t^{(\mathbf{k}-1)} - \hat{y}_t^{(\mathbf{k}-1)}$$

New samples can be predicted using each model once the full set of committee models are created. Thus the final rule-based prediction is the simple average of the individual model predictions.

For the neighbour based adjustment approach, the **k** most similar neighbours are determined from the training set when predicting a new sample, thus resulting to:

$$\hat{\mathbf{y}} = \frac{1}{\mathbf{k}} \sum_{\kappa=1}^{\mathbf{k}} \omega_\kappa \Big[(s_\kappa - \hat{s}_\kappa) + \hat{\mathbf{y}} \Big] \tag{3.54}$$

where s_κ is the observed outcome for a training set neighbour; $\hat{s}_\kappa$ is the model prediction of that neighbour; ω_κ is a weight calculated using the distance of the training set neighbours to the new sample (Boston and Kuhn, 2000).
The nearest neighbours are determined using Manhattan distances and neighbours are only included if they are not over the average distance, and are from the prediction sample.

3.3.10 k Nearest Neighbour

Using the ordering training pairs from the set of data points
$(y, \mathbf{x}) = \{(y_1, x_1), (y_2, x_2), ..., (y_T, x_T)\}$, with $\mathbf{x} = (x_1, x_2, ..., x_k)'$ representing a vector of covariates and $y = (y_1, y_2, ..., y_T)'$ representing a vector of values for the

response variable. Then the nearest neighbour estimator is the mean function value of the nearest neighbours, obtained as follows:

$$\hat{y}(\mathbf{x}) = \frac{1}{k_T}\sum_{t=1}^{k_T} y_t(\mathbf{x}) \tag{3.55}$$

where y_t is the response variable at t; $\mathbf{x}$ is the vector of covariates; $\{(y_1, x_1), (y_2, x_2), ..., (y_T, x_T)\}$ is a reordering of the data points according to increasing distances $||\mathbf{x}_i - \mathbf{x}_j||$ of the $\mathbf{x}'s$ to $\mathbf{x}_j$; k_T is the number of neighbours from the data points in $N(\mathbf{x})$ (Biau et al., 2012; Altman, 1992).

It is worth noting that kNN involves the introduction of weighted average, in which the weight of each neighbour depeneds on its closeness to a query point, as follows:

$$\hat{y}(\mathbf{x}) = \frac{1}{\zeta}\sum_{\mathbf{x}_j \in N(\mathbf{x})} d(\mathbf{x}_i, \mathbf{x}_j) \cdot y(\mathbf{x}_j) \tag{3.56}$$

where ζ is a normalization factor; $d(\mathbf{x}_i, \mathbf{x}_j)$ is the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$.

The weights can be assessed by computing the inverse of the squared Euclidean distance, as follows (Leon and Curteanu, 2015; Luken et al., 2019):

$$\mathbf{w}_d(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{d(\mathbf{x}_i, \mathbf{x}_j)^2} = \frac{1}{||\mathbf{x}_i - \mathbf{x}_j||^2} \tag{3.57}$$

Thus, the nearest neighbour estimator is obtained as follows:

$$\hat{y(\mathbf{x})} = \frac{\sum_{\mathbf{x}_j \in N(\mathbf{x})} \mathbf{w}_d(\mathbf{x}_i, \mathbf{x}_j) y(\mathbf{x}_j)}{\sum_{\mathbf{x}_j \in N(\mathbf{x})} \mathbf{w}_d(\mathbf{x}_i, \mathbf{x}_j)} \tag{3.58}$$

where $\mathbf{w}_d(\mathbf{x}_i, \mathbf{x}_j)$ is the weight function.

The kNN algorithm uses a the weighted average of k nearest neighbours, weighted by the inverse of their distances. Thus, the algorithm can be summarized as follows:

1. Compute the Euclidean distance from the query instance to the labelled instances;
2. Reoder the labelled instances by increasing distance;
3. Determine a heuristically optimal k of nearest neighbours based on mean squared error, by cross-validation;
4. Compute the inverse distance weighted average with the k nearest neighbours.

3.3.11 Projection Pursuit Regression

The projection pursuit regression (PPR), developed by Friedman and Stuetzle (1981), is an extensive additive model whose first task is to project the data matrix of covariates in the optimal direction before the application of smoothing functions to the covariates.
Let $(y, \mathbf{x}) = \{(y_i, \mathbf{x}_t)\}_{t=1}^{T}$ be a training data set in a finite dimensional space and let $\Phi : R \longrightarrow R$ be a collection of smooth hyperparameter functions which maps R to R. Then y_t and $\mathbf{x}_t$ are related in the form (Lingjaerde and Liestøl, 1998; Matilainen, 2018):

$$y_t = \theta_0 + \sum_{j=1}^{q} \Phi_j(\beta_j' \mathbf{x}_t) + \epsilon_t \tag{3.59}$$

where y_t is the response variable; $\mathbf{x}_t$ is a k-dimensional vector of covariates; Φ_j is a collection of q initially unknown smooth functions; β_j is a vector of q unknown parameters; q is a hyper parameter.

Thus, the PPR estimators can be obtained by minimizing the error function:

$$\underset{\Phi_j, \beta_j}{min} \xi(\epsilon_t) = \sum_{t=1}^{T} \left\{ \left(y_t - \sum_{j=1}^{q} \Phi_j(\beta_j' \mathbf{x}_t) \right)^2 \right\}$$

If we consider each pair (Φ_j, β_j) individually, and fixing all other parameters, to dtermine the residual, then the unaccounted variance of the output of those parameters is defined as follows:

$$r_t = y_t - \sum_{\ell \neq j} \Phi_\ell(\beta'_\ell \mathbf{x}_t) \tag{3.60}$$

The minimization of the error function becomes as follows:

$$\underset{\Phi_j,\beta_j}{min} \xi(\epsilon_t) = \sum_{t=1}^{T} \left\{ \Big(r_t - \Phi_j(\beta'_j \mathbf{x}_t) \Big)^2 \right\}$$

where r_t is an approximation function used to transform the optimization problem to a closed-form (Matilainen, 2018).

Thus, the resulting PPR estimator of the optimal β_j is as follows:

$$\hat{\beta}_j^{PPR} \simeq \underset{\beta_j}{argmin} \left\| \hat{\mathbf{A}} - X\theta_j \right\|_\Delta^2 = (X'\Delta X)^{-1} X'\Delta\hat{\mathbf{A}} \tag{3.61}$$

where Δ is diagonal matrix; $\hat{\mathbf{A}}$ is a vector of stack target observations; X is the $T \times q$ matrix of covariates.

3.3.12 Neural Networks Regression

Let $(X, \mathbf{y})$ be a training data set in a finite dimensional space, such that a function Υ maps X to $\mathbf{y}$:

$$\Upsilon : X \longrightarrow \mathbf{y}$$

according to a particular learning rule. Then $\Upsilon(\mathbf{x})$ is a neuron network, expressed as a composition of a decomposable function $\eta_j(\mathbf{x})$, with nonlinear weights ω_j, in the form:

$$\Upsilon(\mathbf{x}) = \Gamma\left\{\sum_{j=1}^{k}\omega_j\eta_j(\mathbf{x})\right\} = \Gamma\Big(\omega_1\eta_1(\mathbf{x}), \omega_2\eta_2(\mathbf{x}), ..., \omega_k\eta_k(\mathbf{x})\Big)$$

where $\Gamma(\cdot)$ is the activation function; $\omega = (\omega_1, \omega_2, ..., \omega_k)$ is a vector of weighted parameters; $\eta(\mathbf{x}) = (\eta_1(\mathbf{x}), \eta_2(\mathbf{x}), ..., \eta_k(\mathbf{x}))$ is a vector of composition functions (Samarasinghe, 2016).

According to Specht (1991), the aim of generalized neural networks is to enhance data splitting into training and test sets, such that every training sample represents a sample mean to a radial basis neuron. For the purpose of model training and forecasting in this study, the most extensively used neural network is the single hidden-layer feed-forward networks.
The predictive model for the generalized neural network is as follows:

$$\mathbf{y}(\mathbf{x}) = \frac{\sum_{j=1}^{k}\omega_j\Gamma(\mathbf{x}, \mathbf{x}_j)}{\sum_{j=1}^{k}\Gamma(\mathbf{x}, \mathbf{x}_j)} \tag{3.62}$$

where $\mathbf{y}(\mathbf{x})$ is the predicted value for the input vector $\mathbf{x}$; ω_j is the activation for the pattern layer neuron at j; $\Gamma(\mathbf{x}, \mathbf{x}_j)$ is the radial basis function with Gaussian kernel $exp(-\frac{\Theta_j}{2\sigma^2})$, for $\Theta_j = (\mathbf{x}, \mathbf{x}_j)'(\mathbf{x} - \mathbf{x}_j)$ is the squared Euclidean distance between the training samples $\mathbf{x}_j$ and the input $\mathbf{x}$. See Liu et al. (2015) for further detail.

3.3.12.1 Quantile Regression Neural Network

Let $X_i(t)$ be a vector of predictor variables and let $\mathbf{y}(t)$ be the response variable outputs from a quantile regression neural network. The output from ℓ^{th} hidden layer node denoted by $\eta_\ell(t)$ can be deduced by applying a hyperbolic tangent with a sigmoid transfer function to the inner product between $X_i(t)$ and the hidden layer weights $\omega_{i,\ell}^{(h)}$ together with the hidden layer bias $\beta_\ell^{(h)}$, in the form:

$$\eta_\ell(t) = tanh\Big(\sum_{i=1}^{I} X_i(t)\omega_{i,\ell}^{(h)} + \beta_\ell^{(h)}\Big) \tag{3.63}$$

and the estimate of the conditional τ-quantile is defined as:

$$\hat{\mathbf{y}}_\tau(t) = \Gamma\Big(\sum_{\ell=1}^{L} \eta_\ell(t)\omega_\ell^{(\otimes)} + \beta^{(\otimes)}\Big) \tag{3.64}$$

where $\omega_\ell^{(\otimes)}$ are the output layer weights; $\beta^{(\otimes)}$ is the output layer bias; $\Gamma(\cdot)$ is the output layer transfer function.

Thus the linear quantile regression parameter estimates of a quantile regression neural network with a single layer node can be obtained from the following:

$$\hat{\mathbf{y}}_\tau(t) = \Gamma\Big(\sum_{i=1}^{I} \omega_1^{(\otimes)}\omega_{i,1}^{(h)} X_i(t) + \beta_1^{(h)}\omega_1^{(\otimes)} + \beta^{(\otimes)}\Big) \tag{3.65}$$

(Pradeepkumar and Ravi, 2017; Cannon, 2011).

If the number of predictor variables I and the number of hidden layers nodes L in a quantile regression neural network appeared to be complex thereby overfitting the training data, then the concept of weight decay regularization will be introduced to prevent overfitting. The weight decay regularization seek to penalize large weights ($\omega \longrightarrow \infty$) in the input hidden layer of the model by inclusion of a quadratic penalty term to the error function:

$$\xi_\tau^{(\alpha)} = \frac{1}{T}\sum_{t=1}^{T} \rho_\tau^{(\alpha)}\Big(\mathbf{y}(t) - \hat{\mathbf{y}}_\tau(t)\Big) + \lambda\frac{1}{I, L}\sum_{i=1}^{I}\sum_{\ell=1}^{L}\Big(\omega_{i,\ell}^{(h)}\Big)^2 \tag{3.66}$$

where $\rho_\tau^{(\alpha)}$ is a tilted absolute function; $\lambda > 0$ is a constant that controls the amount of regularization in the weight decay term (Cannon, 2011).

The neural networks (NNET) can be applied in different forms. In this study, the feasible NNET regression were implemented using the caret built packages for regression training (RT). The method value *avNNet* implements the feed-forward neural network RT model in which the multiple networks are averaged (AVNNET); the method value *pcaNNet* implements the neural networks RT model applied with principal component (PCANNET) for dimensionality reduction.

3.3.13 Statistical and Economic Performance Evaluation

3.3.13.1 Traditional Measures of Forecast Accuracy

The traditional measures used to determine the forecasting accuracy of each model include the *mean squared error* (MSE), *root mean squared error* (RMSE), *mean absolute error* (MAE) and *mean absolute percentage error* (MAPE) respectively. They can be computed as follows:

$$MSE = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}$$

$$MAE = \frac{1}{T}\sum_{t=1}^{T}|y_t - \hat{y}_t|$$

$$MAPE = \frac{1}{T}\sum_{t=1}^{T}\left|\frac{y_t - \hat{y}_t}{y_t}\right|$$

where T is the out-of-sample forecasting period; y_t is the actual value at specific time t; $\hat{y}_t$ is the forecast value at specific time t.

The MSE is employed to compute the out-of-sample mean squared forecast error for each RT model among these traditional measures of forecast accuracy in this study. The MSE is chosen because our interest is to minimise risk and to advice a mean-variance portfortolio investor to invest on a portfolio which gives the minimum volatility. In this case, an RT model with lower MSE is preferable to a model with higher one, because the higher the volatility, the more riskier the security/investment and vice versa.

3.3.13.2 Out-of-Sample Forecast Evaluation: The R^2_{OOS} Statistic

The out-of-sample statistical goodness of fit used to measure the performance of individual equity premium forecasting model, suggested by Campbell and Thompson (2007) for evaluating the overall performance of any competing model forecasts in terms of proportional error minimization, relative to the benchmark historical average forecast is defined as follows:

$$R^2_{OOS} = 1 - \frac{\sum_{t=1}^{T}(y_{T_0+t} - \hat{y}_{T_0+t})^2}{\sum_{t=1}^{T}(y_{T_0+t} - \bar{y}_{T_0+t})^2}$$

where $R^2_{OOS} > 0$ implies that the MSE of the forecasting model is less than the MSE of the benchmark forecasts based on historical average; $\hat{y}_{T_0+t}$ represents an equity premium forecast based on a specific competing model; and $\bar{y}_{T_0+t}$ represents the equity premium forecast based on the historical average, obtained by either expanding or rolling window method.

3.3.13.3 Diebold-Mariano Test

Another suitable measure of individual model forecasts accuracy as compared to the benchmark historical average forecasts accuracy is based on the outcome of Diebold-Mariano (DM) test, whose assumptions relies on the forecast error loss differential function (Diebold and Mariano, 2002; Diebold, 2015). Let $\epsilon_{1,t}$ and $\epsilon_{2,t}$ denote the forecast errors associated with the loss functions $L(\epsilon_{1,t})$ and $L(\epsilon_{2,t})$ for forecasts 1 and 2 respectively. The time-t loss differential between forecasts 1 and

2 is defined as follows:

$$d_{1,2(t)} = L(\epsilon_{1,t}) - L(\epsilon_{2,t})$$

The DM hypothesis of equal forecast accuracy, also known as equal expected loss, corresponds to the zero mean assumption of $d_{1,2(t)}$, i.e., $E(d_{1,2(t)}) = 0$; where $E(\cdot)$ denotes the mean value. Thus, the null hypothesis of equal forecast accuracy against the alternative hypothesis of unequal forecast accuracy between forecasts 1 and 2, based on monthly forecast horizon $h = 1$, can be tested using the DM test statistic as follows (Diebold, 2015):

$$DM_{1,2} = \frac{\bar{d}_{1,2}}{\hat{\sigma}_{\bar{d}_{1,2}}} \overset{asymptotically}{\sim} N(0,1)$$

where $\bar{d}_{1,2} = \dfrac{1}{T}\sum_{t=1}^{T} d_{1,2(t)}$ is the sample mean loss differential and $\hat{\sigma}_{\bar{d}_{1,2}}$ is a consistent estimate of the standard deviation of $\bar{d}_{1,2}$. Thus, the DM test statistic has the asymptotic standard normal distribution under the null hypothesis of equal forecast accuracy. In this study, the forecast errors of each RT model are compared with the forecast errors from the benchmark historical average.

3.3.13.4 Sharpe Ratio

In finance, the Sharpe Ratio (SR) is required to examine the performance of an investment by simply adjusting for the risk associated with it. The SR is a measure of excess return per unit of deviation in an investment asset or trading strategy (Sharpe, 1994). In this study, we use the SR which standardizes the realized returns with the risk of the portfolios. It is computed as follows:

$$SR_p = \frac{E(R_p) - E(R_f)}{\sqrt{Var(R_p)}}$$

where $E(R_p)$ is the average realized return of the portfolio over the out-of-sample period; $E(R_f)$ is the average risk-free treasury bill rate; $Var(R_p)$ is the variance of the portfolio over the out-of-sample period. A portfolio with a higher SR is considered to be superior relative to its counterpart.

The cumulative return (CR) of the portfolio, was also employd to determine the aggregate amount that an investment has gained or lost, independent of the time period involved. It can be computed as follows:

$$CR = \sum_{t=1}^{T} R_t \tag{3.67}$$

where R_t is the return on month t; T is the number of months in the out-of-sample period.

3.3.13.5 Utility Gain

The utility gain (UG) is an economic performance evaluation measure for a mean-variance portfolio investor on a real-time basis. The UG describes the portfolio management fee that an investor would be willing to pay in order to have access to the additional available information in a specific RT forecasting model relative to the sole information in the historical average. A mean-variance investor who forecasts the monthly equity premium using the global historical average will decide at the end of time t to allocate risky weights as share of her portfolio to equities in time $t + 1$, in form:

$$\omega_{0,t} = \lambda^{-1}\Big(\frac{\bar{R}_{t+1}}{\hat{\sigma}^2_{R,t+1}}\Big)$$

where $\omega_{0,t}$ represent the portfolio risky weights, constrained to lie between 0% and 150%, i.e., $\omega_{0,t}$ equals zero if $\omega_{0,t} < 0$ and $\omega_{0,t}$ equals 1.5 if $\omega_{0,t} > 1.5$, in accordance with the techniques used in Campbell and Thompson (2007); λ is the risk aversion parameter; $\bar{R}_{t+1}$ represent the equity premium forecasts based on the benchmark global historical average; $\hat{\sigma}^2_{R,t+1}$ is the expanding window estimate of variance of stock returns. The rationale for using portfolio weights that lie between

0% and 150% is to impose realistic portfolio constraints to prevent the investor from shorting stocks or taking more than 50% leverage, and hence, confining the portfolio weight on stocks to lie between 0.0 and 1.5.
The investor realizes an average utility over the out-of-sample period, given by the following:

$$\hat{U}_0 = \hat{\mu}_{0,p} - \frac{1}{2}\lambda\hat{\sigma}^2_{0,p}$$

where $\hat{\mu}_{0,p}$ is the sample mean over the out-of-sample period for the benchmark portfolio formed using the equity premium forecasts based on the global historical average; $\lambda\hat{\sigma}^2_{0,p}$ is the sample variance over the out-of-sample period for the benchmark portfolio formed using the equity premium forecasts based on the global historical average.

The next step is to compute the average utility for the same investor when she forms forecasts of the equity premium using an individual RT model. In this case, the weight risky equity share can be chosen by the following:

$$\omega_{j,t} = \lambda^{-1}\Big(\frac{\hat{R}_{j,t+1}}{\hat{\sigma}^2_{R,t+1}}\Big)$$

Then the investor realizes an average utility, defined as follows:

$$\hat{U}_j = \hat{\mu}_{j,p} - \frac{1}{2}\lambda\hat{\sigma}^2_{j,p}$$

where $\hat{\mu}_{j,p}$ is the sample mean over the out-of-sample period for the return on the portfolio formed using forecasts of the equity premium based on an individual RT model; $\lambda\hat{\sigma}^2_{j,p}$ is the sample mean over the out-of-sample period for the return on the portfolio formed using forecasts of the equity premium based on an individual RT model (Rapach et al., 2010; Goyal and Welch, 2007).

Thus, the utility gain (UG) can be computed as follows:

$$UG = \hat{U}_j - \hat{U}_0$$

for each of the RT out-of-sample forecasting models used in this study for forecasting the U.S. monthly equity premium.

3.4 The Empirical Results and Discussion

3.4.1 Data, Variables and Forecasting Method

In this study, the monthly data are obtained from Amit Goyal's webpage[1], each covering monthly observations from January 1960 to December 2016, which results to a total of $T = 684$. Most of these variables, presented in Table 3.1, were previously used in financial econometric literature by other scholars to forecast the U.S. equity premium on either monthly, quarterly or annually. In accordance with the benchmark method suggested by Goyal and Welch (2003), Goyal and Welch (2007), Campbell and Thompson (2007) in which the performance of any forecasting model comparative to the historical average method when forecasting equity premium should be based on either expanding window or rolling window out-of-sample forecasting approach, all forecasts in this study are obtained using expanding window out-of-sample forecasting method for the various RT forecasting models. The out-of-sample period consists of monthly observations from 1991M1 to 2016M12. In this case, the parameters of the forecasting models are estimated recursively using an expanding window of observations, with data point from the start date to the present time and obtain a one month-period-ahead forecast, $y_{1|T}$. The procedure is repeated until the last forecast, $y_{312|T}$, is obtained. In this study, we trained each RT model, preprocessed the training dataset in a centred and scaled form, tuned the RT parameter(s) of each

[1]Available at: www.hec.unil.ch/agoyal/docs/PredictorData2016.xlsx

model by cross-validation and resampling all, and eventually used the best tuning parameters to run the out-of-sample forecasts recursively. The resampling approach seek to figure out the values of each model parameters, providing the best tuning parameter(s) from the validation set that can be used to produce the out-of-sample forecasts. All computations in this study were obtained using R software and the associated packages (Kuhn et al., 2008; Kuhn, 2012, 2015) and https://topepo.github.io/caret/available-models.html, see Tables 3.3 and 3.4 for detail specification of the packages and the their values. The forecast horizon denoted by h is one month ahead ($h = 1$) for all the RT forecasting models used in this study. All tables and graphics are depicted appropriately.

Table 3.1: Data & Description of Variables: 1960M01 to 2016M12

Data & Description of Variables: 1960M01 to 2016M12	
Variable	Description
Stock Index, SP_t	Is the Standard&Poor 500 U.S stock index.
Excess Stock Return, r_t	The difference between the expected return on the market portfolio (SP500) and the risk-free treasury bill rate.
Dividend Price Ratio (log), DPR_t	The dividends over the past year divided by the current stock index value.
Dividend Yield (log), DY_t	Is the difference between the log of dividends and the log of lagged prices.
Earnings Price Ratio (log), EPR_t	The earnings over the past year divided by the current stock index value.
Realized Stock Variance, RSV_t	Is the sum of squared daily returns on the $S\&P500$ index within one month.
Book to Market Value, BMV_t	Is the ratio of book value to market value for the Dow Jones Industrial Average.
Net Equity Expansion, NEE_t	Is the ratio of 12-month moving sums of net issues by New York Stock Exchange (NYSE) listed stocks to total end of year market capitalization of the NYSE stocks.
Treasury Bill Rate, TBR_t	Is the interest rate on a 3-month treasury bill, secondary market.
Long Term Yield, LTY_t	Is the long term government bond yield, constant maturity.
Long Term Return, LTR_t	Is the return on long term government bonds.
Term Spread, TS_t	Is the difference between the long term yield (LTY_t) and the treasury bill rate (TBR_t).
Default Yield Spread, DYS_t	Is the difference between the BAA and AAA rated corporate bond yields.
Default Return Spread, DRS_t	Is the difference between the long term corporate bond and long term government bond returns.
Inflation, INF_t	Is computed from the consumer price index (CPI) for all urban consumers.

Table 3.2: Descriptive Statistics for the Time Series Variables:1960M01 to 2016M12

Variable	**Min.**	**Max.**	**Mean**	**Median**	**1st Qu**	**3rd Qu**	**Std. Dev.**	**Kurtosis**	**Skewness**
$EquityPrem_t$	-0.2508	0.1443	0.0014	0.0056	-0.02186	0.0294	0.0429	2.4373	-0.6794
DPR_t (log)	-4.524	-2.753	-3.588	-3.520	-3.912	-3.333	0.3943	-0.6929	-0.23305
DY_t (log)	-4.531	-2.751	-3.538	-3.516	-3.903	-3.331	0.3942	-0.6661	-0.23848
EPR_t (log)	-4.836	-1.899	-2.840	-2.873	-3.049	-2.644	0.4281	3.0463	-0.69115
RSV_t	0.000072	0.0709	0.0021	0.0011	0.00065	0.00214	0.00435	136.8135	10.27581
BMV_t	0.1205	1.2065	0.5024	0.4592	0.3059	0.6518	0.2571	-0.2844	0.7461
NEE_t	-0.0577	0.05116	0.0113	0.01419	0.00197	0.02519	0.0194	0.5966	-0.7923
TBR_t	-0.0462	0.0261	-0.000058	0.0000	-0.0010	0.0012	0.0043	27.6286	-1.81099
LTY_t	-0.2123	0.1838	0.0000149	0.00135	-0.0211	0.0227	0.0405	3.1476	-0.20287
TS_t	-0.0365	0.0455	0.0182	0.0182	0.0072	0.0301	0.01462	-0.1894	-0.3218
DYS_t	0.0032	0.0338	0.01019	0.0090	0.0073	0.0121	0.00448	4.2387	1.7339
DRS_t	-0.0975	0.0737	0.00013	0.0006	-0.0055	0.0057	0.01453	6.6048	-0.3974
$INFL_t$ (lag)	-0.0192	0.0181	0.0031	0.0029	0.000746	0.005098	0.00359	3.0357	0.02165

Note: All computations in this study are obtained using R version 3.4.3;
Min. = Minimum, Max. = Maximum, Qu = Quartile, Std. Dev. = Standard Deviation

The descriptive statistics for the time series variables are as shown in Table 3.2. Among these descriptive statistics, the kurtosis described the tailedness or sharpness of the peak of the probability distribution of each variable relative to a normal distribution. As a rule of thumb, any variable whose excess kurtosis is positive has fatter tails than a normal distribution, signifying a leptokurtic distribution while a negative excess kurtosis indicate skinnier tails than a normal distribution, and hence, platykurtic in distribution. A zero excess kurtosis indicate an exact normal bell-shaped curve matching that of a normal distribution and is said to be mesokurtic in distribution. The skewness is a symmetric measure of data distribution. Following a rule of thumb, a negative skewness indicates that the mean of the dataset is less than the median resulting to a left-skewed distribution while a positive skewness indicates its counterpart resulting to a right-skewed distribution. A perfectly symmetric dataset has skewness equals zero resulting to a zero skewed normal distribution.

From the descriptive statistics, the $EquityPrem_t$ have a positive kurtosis and a negative skewness, with mean less than its median and the resulting distribution is skewed towards the left. It is leptokurtic in distribution which appeared to have higher probability of future negative excess stock return realizations than normal.

3.4.2 Results and Discussion

The empirical analysis in this study is splitted into two perspectives, namely the statistical performance of each RT model and the corresponding economic significance relative to the benchmark historical average. Rather than fitting simple linear model for the individual variables or in ordinary kitchen sink form as shown in the existing literature, each sophisticated RT model in this study is trained by first preprocessing the training dataset in a centred and scaled form, tuned the RT parameter(s) of each model by cross-validation and resampling all, and eventually used the best tuning parameters to run the forecasts recursively until the whole out-of-sample periods are obtained. The performance of the various RT models are summarized in distinct panels as shown in Tables 3.3 and 3.4. Using the benchmark procedures for statistical predictability of the equity premium in Goyal and Welch (2003), Campbell and Thompson (2005), any of the RT models which gives an $R^2_{OOS} > 0$ with corresponding mean squared forecast error (MSE) less than the mean squared error of the benchmark historical average obtained using either expanding or rolling window is said to have outperformed or consistently beat the historical average. In the regularized/shrinkage models panel, each of the models produced $R^2_{OOS} > 0$ with corresponding MSE less than the one obtained from historical average out-of-sample. The ridge model outperformed the other regularized/shrinkage models out-of-sample, in terms of statistically predictive power, owing to its smallest MSE and highest R^2_{OOS}. The forward-backward ridge ($FOBA$) underperformed the ridge model while the relaxed LASSO ($RELAXO$) appeared to have outperformed the LASSO in this direction. Thus, the concept of bias-variance trade off in sophisticated RT models is a useful approach for forecasting the U.S. monthly equity premium out-of-sample with significant predictive power. In the component regression models panel, the ICR, the PCR and the Super PC models evidently beat the benchmark historical average, with the Super PC outperforming the ICR and PCR respectively, in a statistically justifiable approach. As for the neural network models panel, the $NNET$, the $AVNNET$ and the $QRNN$ significantly beat the historical average whereas the PCANNET does not outperform the historical average (i.e. $R^2_{OOS} < 0$) with corresponding ($MSE_{PCANNET} < MSE_{GHA}$). The quantile regression neural network ($QRNN$) has by far outperformed the other neural network models in this study in terms of statistical predictability of the U.S. equity premium out-of-sample. In the kitchen sink panel, the linear model (LM) underperformed the benchmark historical average, and hence, it corroborates the empirical results in previous findings in which the ordinary linear regression is unable to consistenly beat the benchmark historical average out-of-sample.

Table 3.3: The Statistical and Economic Performance Evaluation Results

RT Model	**Package**	**Method Value**	**MSE**	**DM pValue**	$R^2_{OOS}(\%)$	**CR**	**SR**	**AU**	**UG (%)**
$\bar{R}_f$ **= 0.220%, RAP = 3**									
Kitchen Sink									
Linear Model	stats	lm	0.00174	0.50377	-0.06017	1.49629	0.32575	1.99436	0.82514
Partial Least Squares									
PLS	pls	pls	0.00157	0.01813	9.59804	2.45642	0.6608	3.81281	2.64359
KernelPLS	pls	kernelpls	0.00154	0.01847	11.05486	2.6179	0.7418	3.8473	2.67808
WideKernelPLS	pls	widekernelpls	0.00154	0.01654	11.33454	2.88296	0.81821	4.03877	2.86955
SparsePLS	spls	spls	0.00162	0.11667	6.61718	2.31099	0.58328	2.49708	1.32786
KRLS									
KRLS with RBF	KRLS	krlsRadial	0.0017	0.00029	1.77888	1.96438	0.35924	2.34361	1.17439
KRLS with PBF	KRLS	krlsPoly	0.00172	0.0013	0.59912	2.03051	0.36198	1.59615	0.42693
Support Vector									
SVM with LBF	kernlab	svmLinear	0.00162	0.09857	6.79678	2.57787	0.6612	3.58143	2.41221
SVM with RBF	kernlab	svmRadial	0.00162	0.00576	6.40262	2.84065	0.70234	4.97097	3.80175
SVM with PBF	kernlab	svmPoly	0.00158	0.01112	9.1643	2.49997	0.5472	4.37425	3.20503
Relevance Vector									
RVM with LBF	kernlab	rvmLinear	0.00161	0.05829	7.19702	1.98248	0.55241	2.84276	1.67354
RVM with RBF	kernlab	rvmRadial	0.00163	0.02674	6.21632	2.87415	0.8531	4.70084	3.53162
RVM with PBF	kernlab	rvmPoly	0.00154	0.00618	11.09931	3.11029	0.90064	4.14468	2.97546
Regularized/Shrinkage									
Ridge	elasticnet	ridge	0.0016	0.08386	7.69347	2.31496	0.64853	3.40287	2.23365
FOBA	elasticnet	foba	0.00163	0.07888	6.07579	2.17786	0.59993	2.34728	1.17806
LASSO	elasticnet	lasso	0.00163	0.04438	6.06115	2.85118	0.8028	2.63033	1.46111
RELAXO	elasticnet	relaxo	0.00161	0.07062	7.29405	1.59623	0.38087	2.23256	1.06334
Elastic Net	elasticnet	enet	0.00166	0.07797	4.33349	2.36669	0.58797	2.27395	1.10473
LARS	lars	lars	0.00162	0.08571	6.35447	2.51866	0.64666	2.0767	0.90748
LARS2	lars	lars2	0.00162	0.04263	6.75833	2.3292	0.55959	2.61761	1.44839

Note: All computations and graphics in this study are obtained using R version 3.4.3; RT = Regression Training; RAP = risk aversion parameter; AU = average utility; UG = utility gain; SR = Sharpe ratio; $\bar{R}_{free}$ = average risk free treasury bill rate; GPR= Gaussian processes regression; LBF = linear basis kernel function, RBF = radial basis kernel function, PBF = polynomial basis kernel function; $<$ imply 'less than', AVNNET = neural network averaging; PCANNET = neural network with principal component analysis; QRNN = quantile regression neural network

Table 3.4: The Statistical and Economic Performance Evaluation Results Continued

RT Model	**Package**	**Method Value**	**MSE**	**DM pValue**	$R^2_{OOS}(\%)$	**CR**	**SR**	**AU**	**UG (%)**
$\bar{R}_f$ **= 0.220%, RAP = 3**									
Components Regression									
ICR	caret	icr	0.00172	0.19598	0.80778	1.99623	0.44288	1.7568	0.58758
PCR	pls	pcr	0.00173	0.25933	0.49903	2.13503	0.54208	1.66309	0.49387
Super PC	superpc	superpc	0.00168	0.10208	3.27224	2.01792	0.5342	1.93761	0.76839
Gaussian Processes									
GPR with LBF	kernlab	gausspprLinear	0.00168	0.31375	3.01506	1.75855	0.43143	2.47114	1.30192
GPR with RBF	kernlab	gaussprRadial	0.00161	0.01266	7.22282	2.89427	0.80479	4.99488	3.82566
GPR with PBF	kernlab	gaussprPoly	0.00157	0.00943	9.23239	2.70457	0.73388	3.0933	1.92408
Regression Splines									
MARS	earth	mars/earth	0.0017	0.40458	2.27343	2.40596	0.569	1.83558	0.66636
Bagging Earth	caret	bagEarth	0.00166	0.23374	4.26272	2.74979	0.72847	2.46889	1.29967
GCVEarth	earth	gcvEarth	0.00193	0.81927	-11.1736	2.56834	0.78389	4.15575	2.98653
Rule-Based									
Ruled-Based	Cubist	cubist	0.00173	0.40997	0.34439	2.19061	0.4857	2.14786	0.97864
Nearest Neighbour									
kNN	caret	knn	0.0017	0.32469	1.94313	2.39165	0.60636	3.91829	2.74907
Projection Pursuit									
PPR	stats	ppr	0.0017	0.41881	2.15366	1.79607	0.43649	2.31606	1.14684
Neural Networks									
NNET	nnet	nnet	0.00173	0.41451	0.12744	0.90394	0.22669	1.13505	-0.03417
AVNNET	caret	avNNet	0.00173	0.2246	0.38861	1.35237	0.52122	1.3136	0.14438
PCANNET	caret	pcaNNet	0.00174	0.69196	-0.12218	0.88953	0.28713	1.0238	-0.14542
QRNN	qrnn	qrnn	0.00164	0.16154	5.50255	2.76775	0.67345	4.27376	3.10454

Note: All computations and graphics in this study are obtained using R version 3.4.3; RT = Regression Training; RAP = risk aversion parameter; AU = average utility; UG = utility gain; SR = Sharpe ratio; $\bar{R}_{free}$ = average risk free treasury bill rate; GPR= Gaussian processes regression; LBF = linear basis kernel function, RBF = radial basis kernel function, PBF = polynomial basis kernel function; $<$ imply 'less than', AVNNET = neural network averaging; PCANNET = neural network with principal component analysis; QRNN = quantile regression neural network

All the partial least squares models demonstrate statistically significant evidence of outperformance over the benchmark historical average, each of which produced lower MSE corresponding to a higher R^2_{OOS}. It is worth noting that the partial least squares models with kernel functions appeared to produced lower MSE and corresponding higher R^2_{OOS} as compared to the ordinary PLS. The PLS with wide kernel function ($WideKernelPLS$) has the minimum MSE corresponding to the highest R^2_{OOS}, and hence, outperforming the other PLS models in this study. In the kernel-based regularized least squares ($KRLS$) models panel, both the radial basis kernel and the polynomial basis kernel functions significantly beat the benchmark historical average, and the radial basis kernel function outperformed the polynomial basis kernel function, in terms of statistical predictability. As for the regression splines panel, the multivariate adaptive regression splines ($MARS$) and the bagging MARS ($BagEarth$) showed statistically significant evidence of outperformance over the benchmark historical average; whereas the splines with generalized cross-validation (GCVEarth) appeared to have underperformed the benchmark historical average. The MARS model applied with bagging or bootstrap aggregating seems to yield lower mean squared prediction error which corresponds to a higher R^2_{OOS}, and hence, the concept of bootstrap aggregating or bagging the MARS seek to improve the predictive task of the MARS model in this direction. All the kernel functions of the support vector regression (SVR) panel, relevance vector machine (RVM) regression panel, and the Gaussian processes regression (GPR) panel revealed statistically significant evidence of superior performance over the benchmark historical average; the polynomial basis kernel functions of the SVR, RVM and GPR respectively beat the linear and radial basis kernel functions. The SVR, RVM and GPR models that are shown to produced useful statistical information in other aspects of economic and financial stock market analysis in empirical literature, do not only reflect usefulness in this study but also significantly beat the benchmark historical average and many other RT models in terms of forecasting the U.S. monthly equity premium out-of-sample.

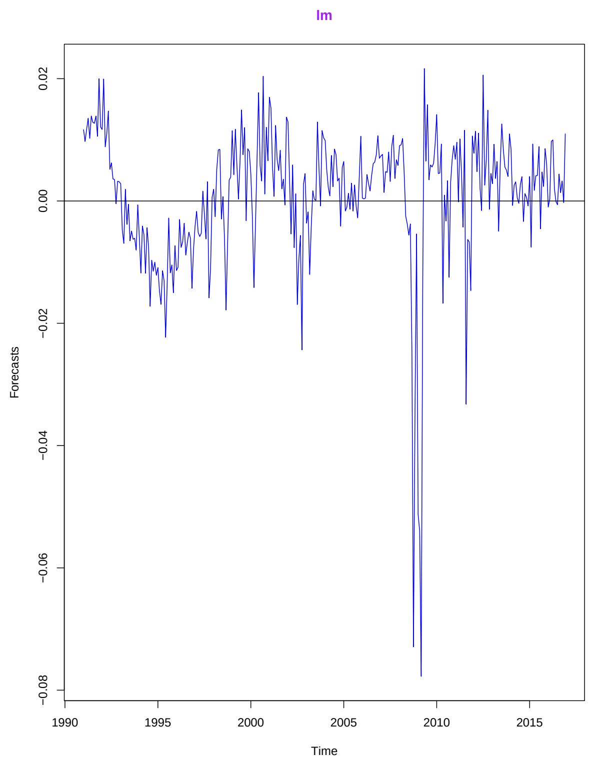
The kNN as an instant-based model also demonstrate statistically significant evidence of beating the benchmark historical average. In the ruled-based model panel, the Cubist significantly beats the benchmark historical average. The projection pursuit model also produced statistically significant evidence of beating the benchmark historical average.

Unlike in other existing papers in which the predictive performance of any model relative to the benchmark historical average rely solely on R^2_{OOS} and MSE, we employed the Diebold-Mariano (DM) as confirmative tests to further investigate whether each RT model demonstrate statistically significant evidence of beating the benchmark historical average, using the residuals of the comparative models together with suitable loss differential function in a one-tailed test of statistical significance. The pValues drawn from the standard Z scores based on the asymptotic normal property are shown in Tables 3.3 and 3.4. The analytical findings in this study revealed that the best performing RT models that are shown to outperformed the benchmark historical average based on the R^2_{OOS} and MSE measures are also confirmed by the DM one-tailed tests to significantly produce better forecast accuracies than the forecast accuracy from the benchmark historical average, at a 5% significance level. However, the DM results seems to be controversial in that some RT models that beat the benchmark historical average based on other statistical measures, do not seems to produce better forecast accuracies than the benchmark historical average. This corroborates the argument in Rapach et al. (2007) which explained that the Diebold and Mariano (2002) statistical tests may be severely undersized when comparing forecasts from nested models, especially for tests involving small difference in predictive accuracies among the competing models, and seems to be inconsistent in some cases. The outcome on the statistical measure of predictability of any model relative to the benchmark historical average depends on the specific statistical tool and method used in the performance evaluation, and hence, the need to specify the standard statistical measures employed.

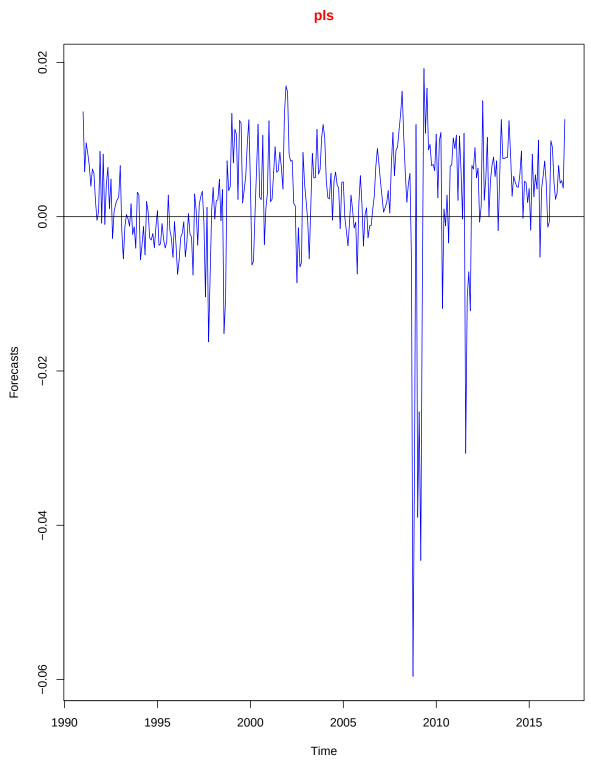
The graphical approach was aslo employed to depict the statistical predictive performance of each RT forecasting model, serially arranged in the order they appeared in the methodology, and in Tables 3.3 and 3.4. The graphical representation of the various forecasts produced by the RT models corroborates the statistical performance evaluation results (see figures 3.1 to 3.9). Like in Rapach et al. (2007) and Rapach et al. (2010), we compute the difference in benchmark historical average forecast cumulative square prediction error and each of the individual RT model forecast cumulative square prediction error (DCSPE), and the outcome is depicted graphically (see figures 3.10 to 3.18). It is noticeable that the DCSPE time-plots of some of the RT models slopes mostly above the threshold (zero), which indicates few outliers produced by the RT model relative to the benchmark historical average, while in some other cases, it slopes mostly below the threshold, which indicates many outliers produced by the RT model relative to the benchmark historical average.

Overall, the wide kernel PLS (WideKernelPLS) produced the minimum MSE and highest R^2_{OOS} among the other RT models in terms of statistical predictability, and it shows statistically significant evidence of producing better forecast accuracy than the benchmark historical average. Thus, the WideKernelPLS is the best RT model for forecasting the U.S. monthly equity premium out-of-sample, among all RT models tested statistically in this study.

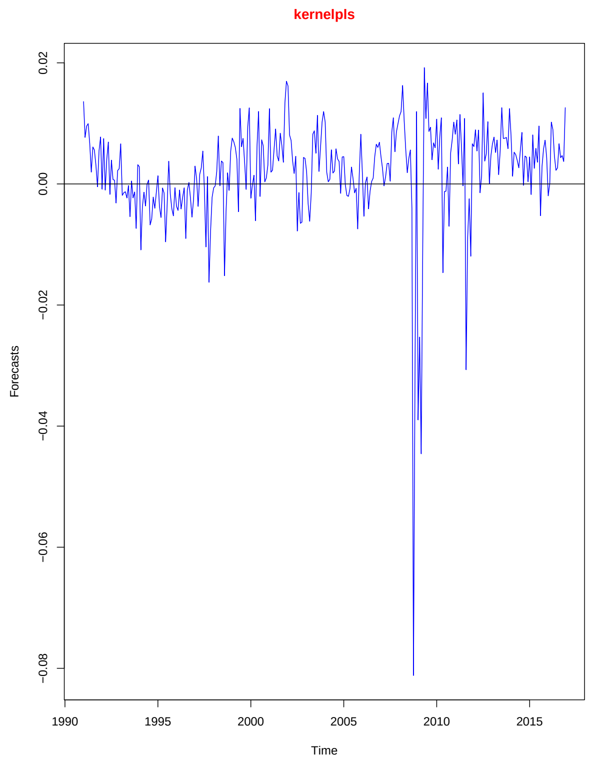
Suffice it to say that the R^2_{OOS} and MSE statistically significant evidence of beating the benchmark historical average in terms of statistical predictive performance does not necessarily reflect economic significance in real terms. A given model may provide evident of useful statistical predictability relative to the benchmark historical average, but it may provide a weak Sharpe ratio and utility gain respectively, underscoring a portfolio held on a risk-free treasury bill rate in some worst case scenarios. The findings in Campbell and Thompson (2007) led to the argument that even very low positive $R^2_{OOS} \geq 0.50\%$ values for monthly data can produce a meaningful economic evidence of equity premium predictability in



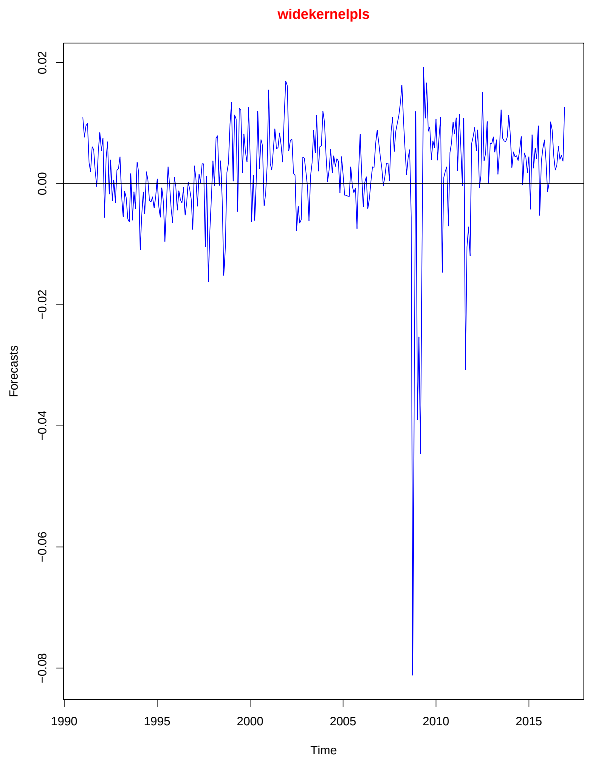
(a) Linear Model



(b) Partial Least Squares

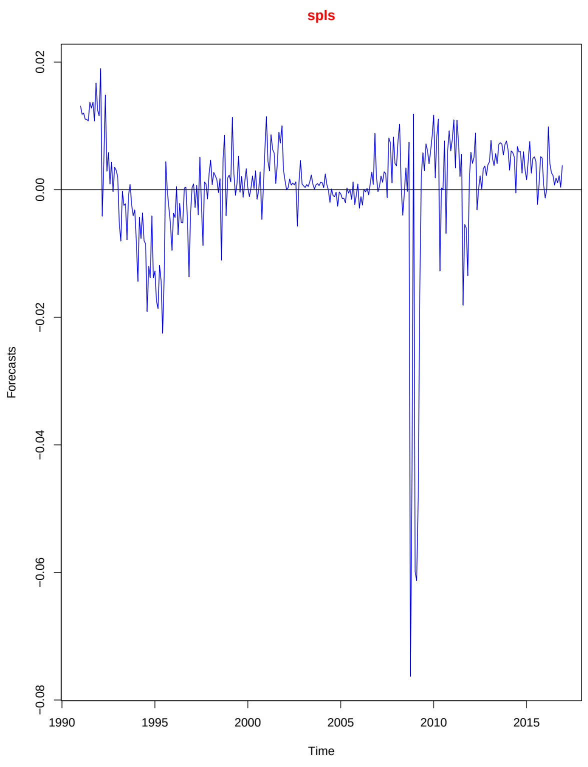


(c) Kernel PLS

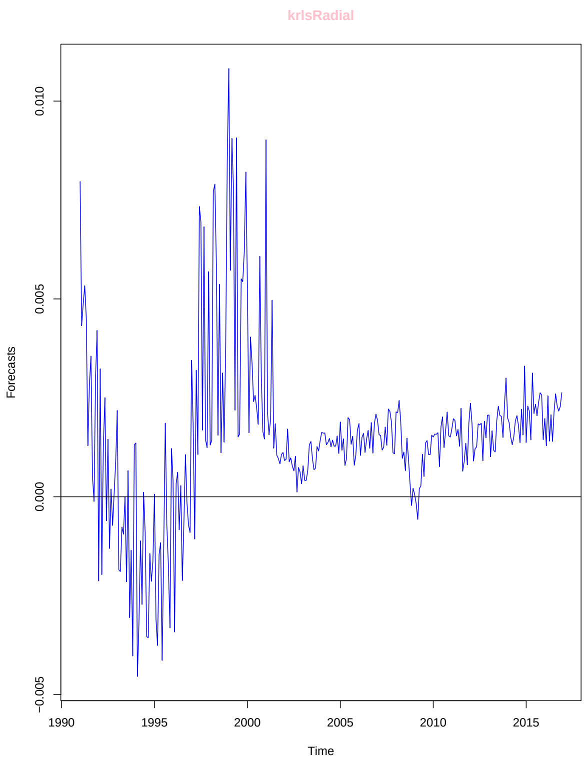


(d) Wide Kernel PLS

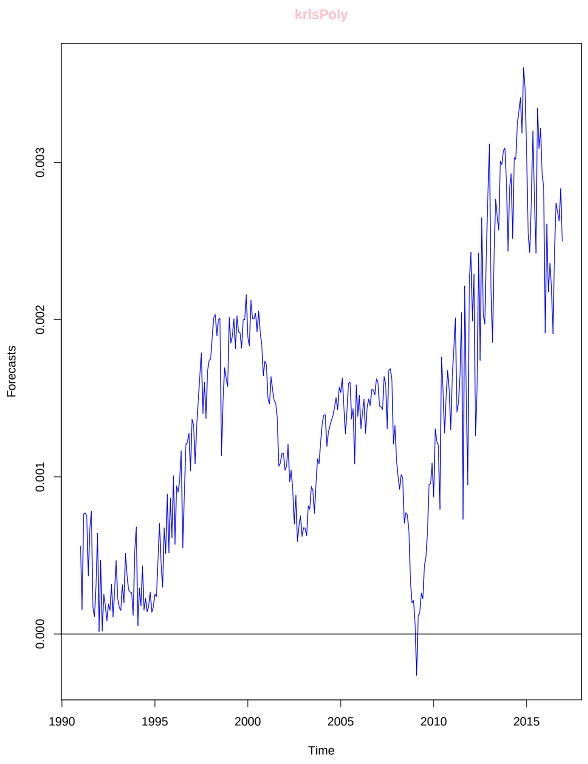
Figure 3.1: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models



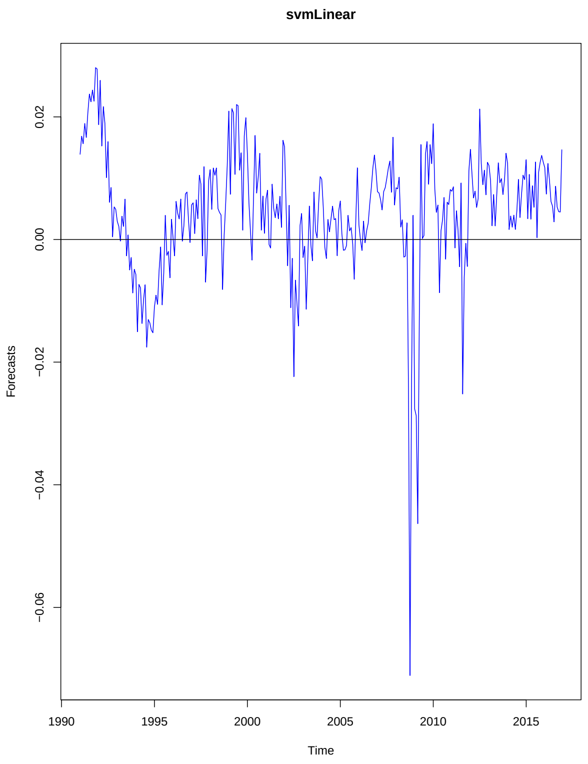
(a) Sparse PLS



(b) KRLS with RBF

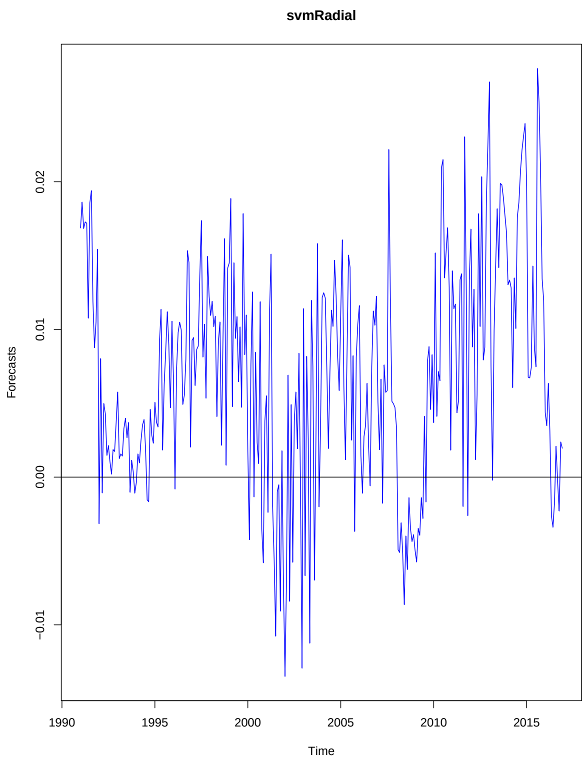


(c) KRLS with PBF

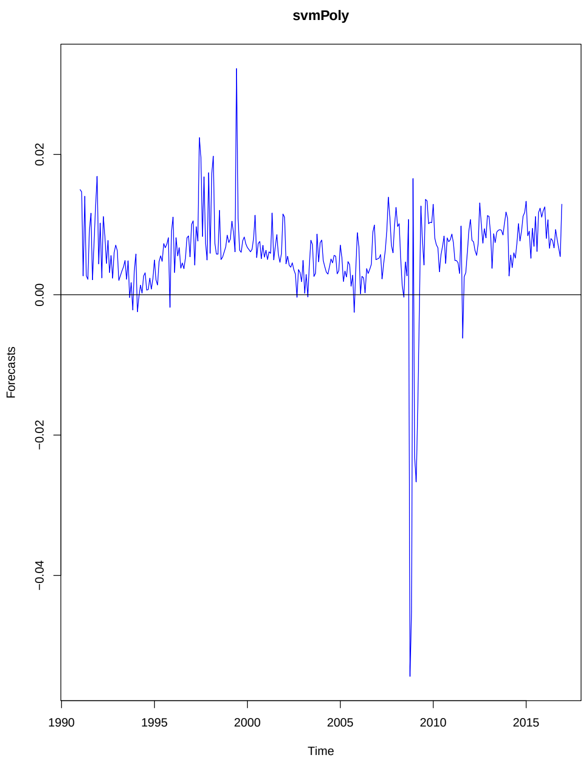


(d) SVM with LBF

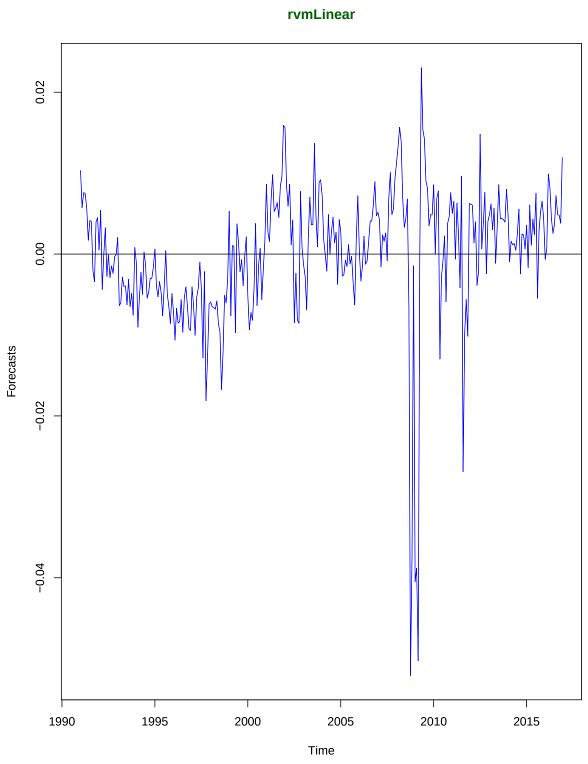
Figure 3.2: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



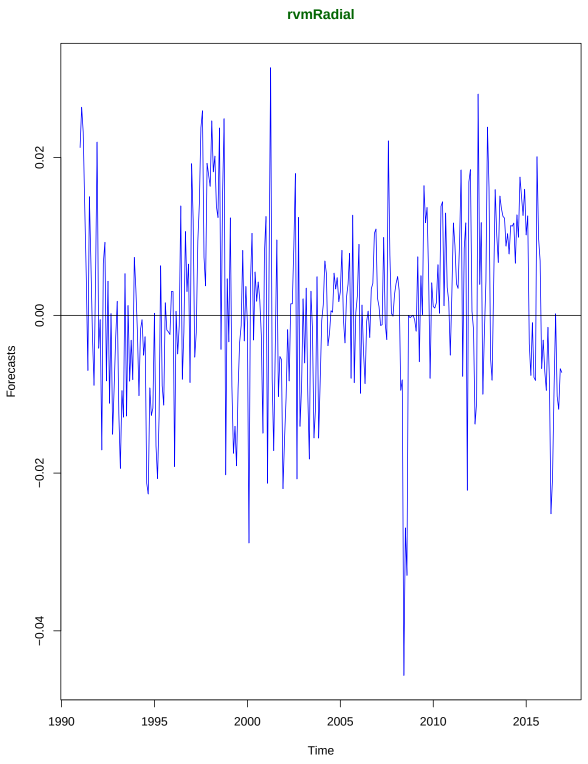
(a) SVM with RBF



(b) SVM with PBF

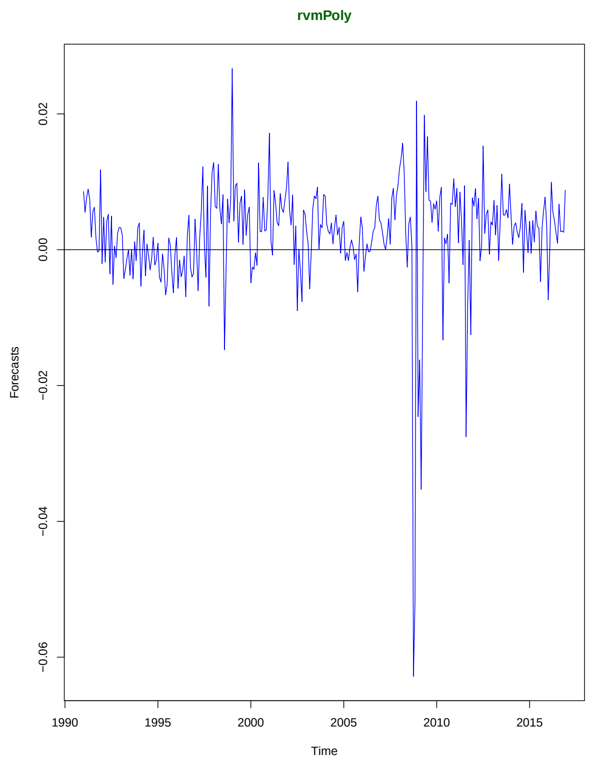


(c) RVM with LBF

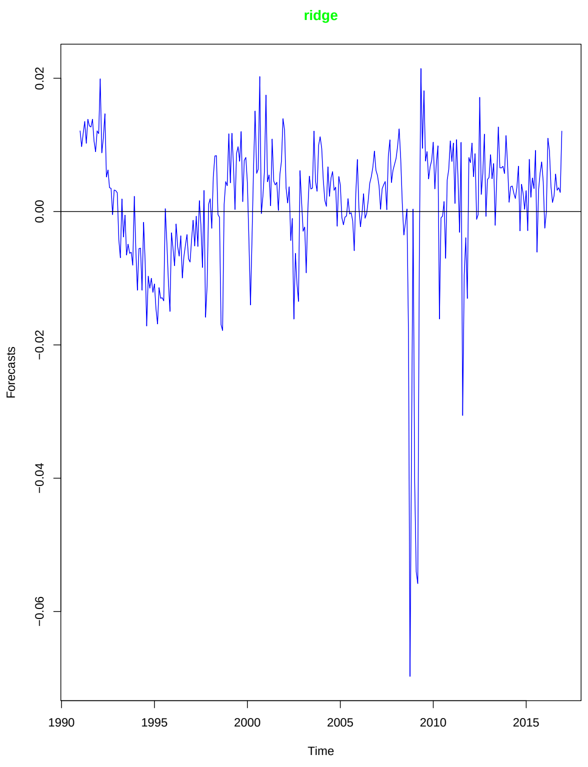


(d) RVM with RBF

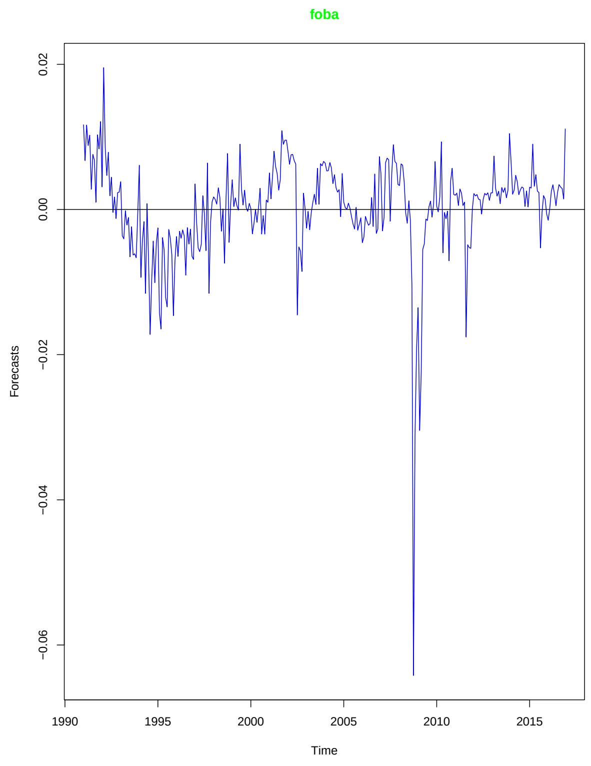
Figure 3.3: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



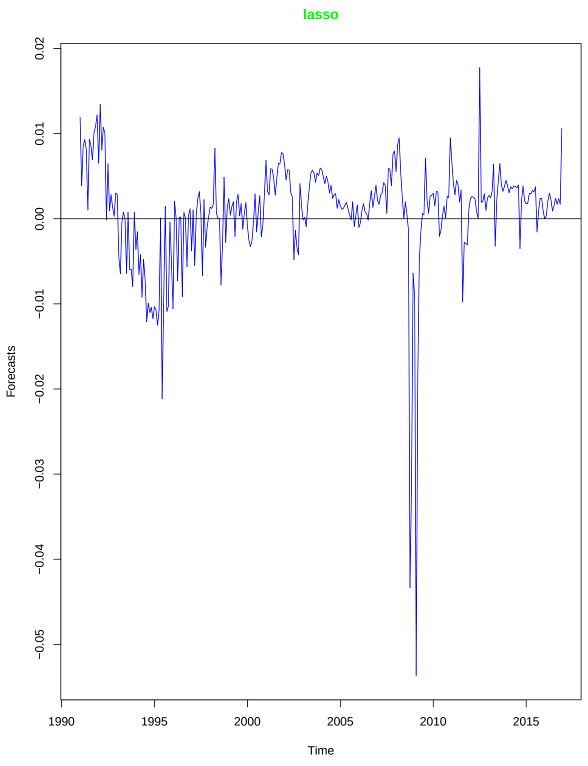
(a) RVM with PBF



(b) Ridge

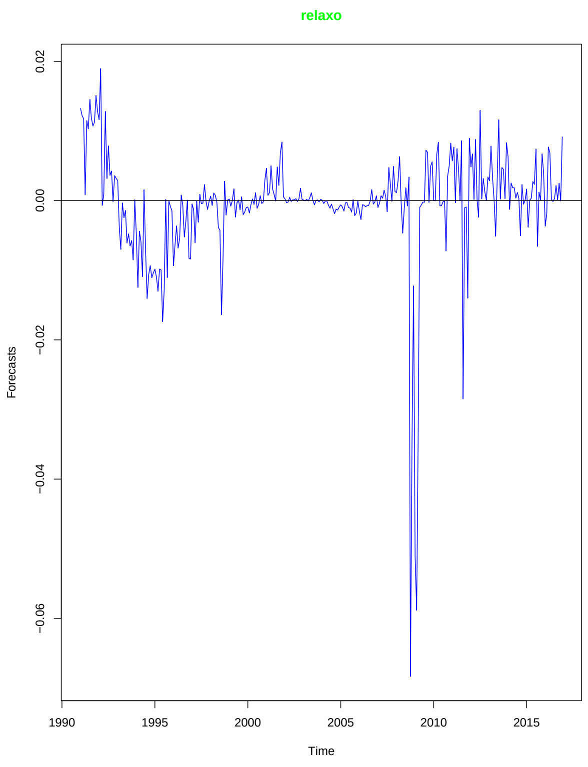


(c) FOBA

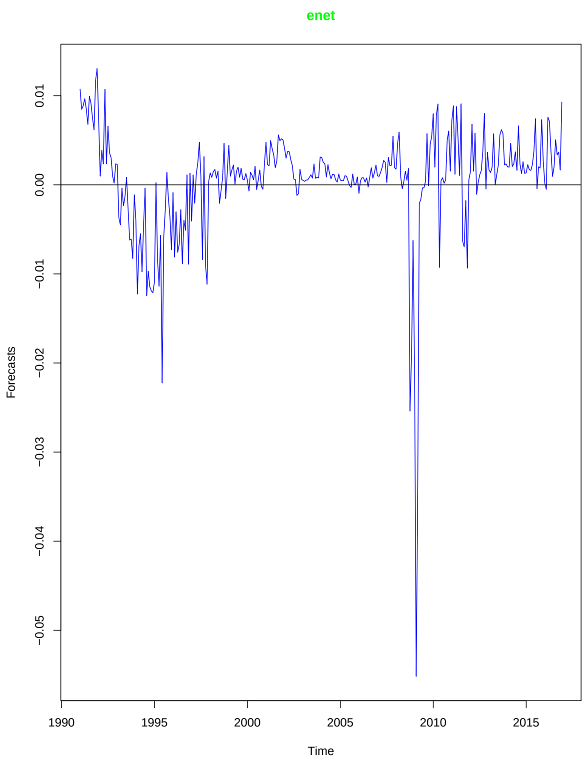


(d) LASSO

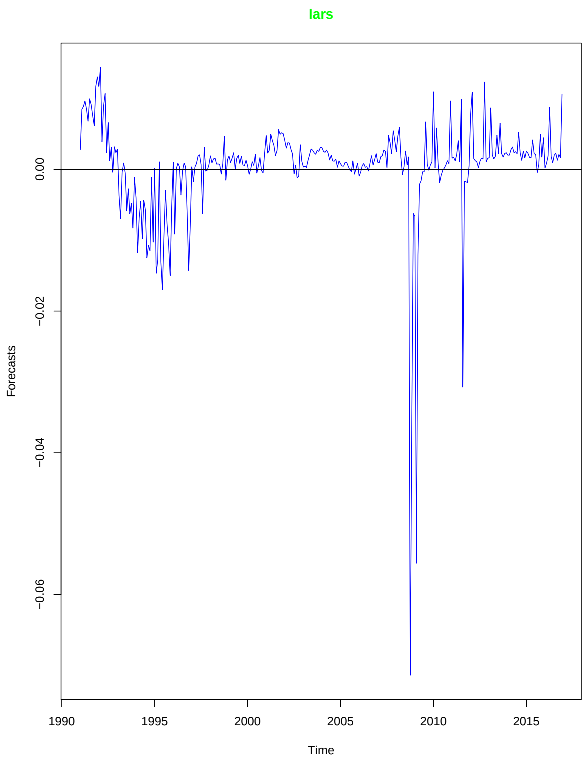
Figure 3.4: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



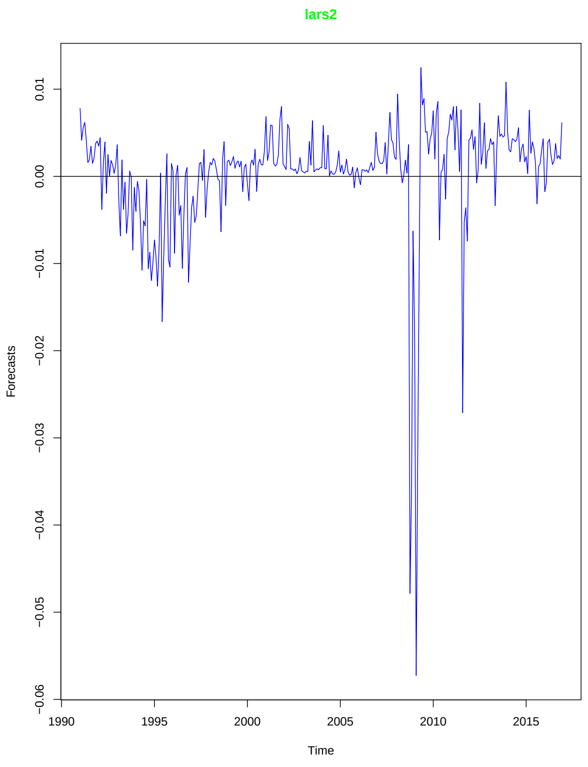
(a) RELAXO



(b) Elastic Net

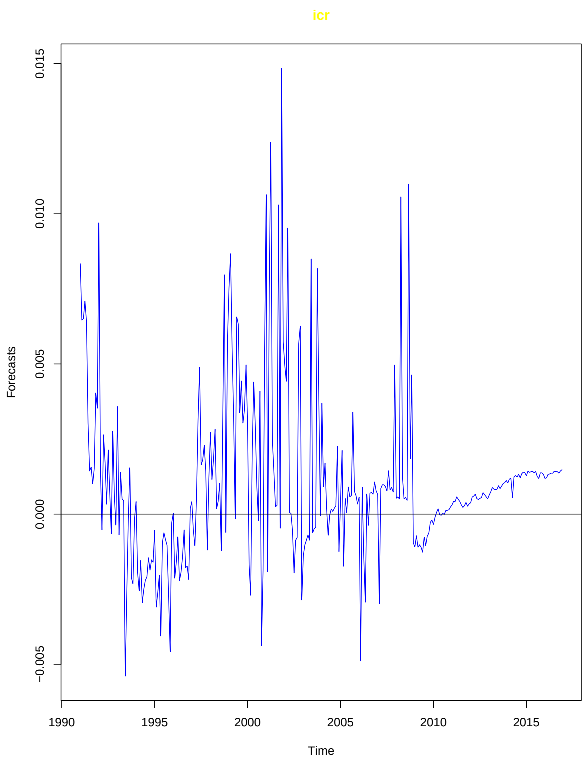


(c) LARS

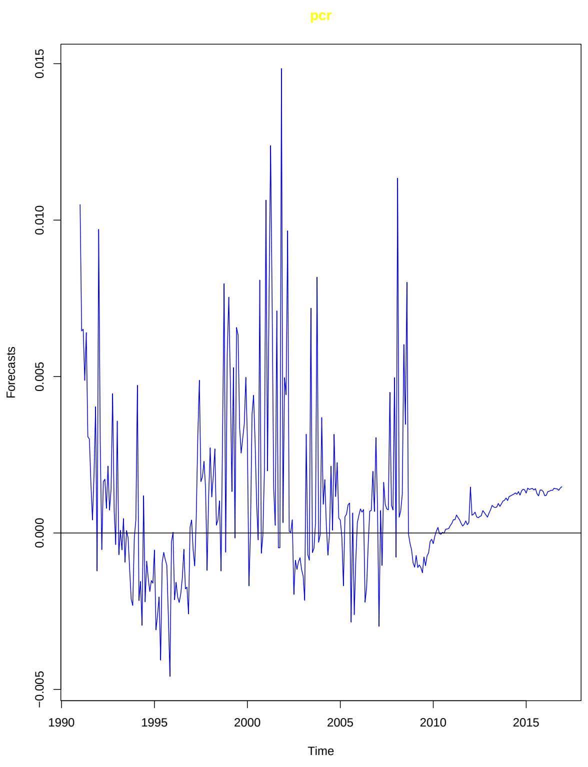


(d) LARS2

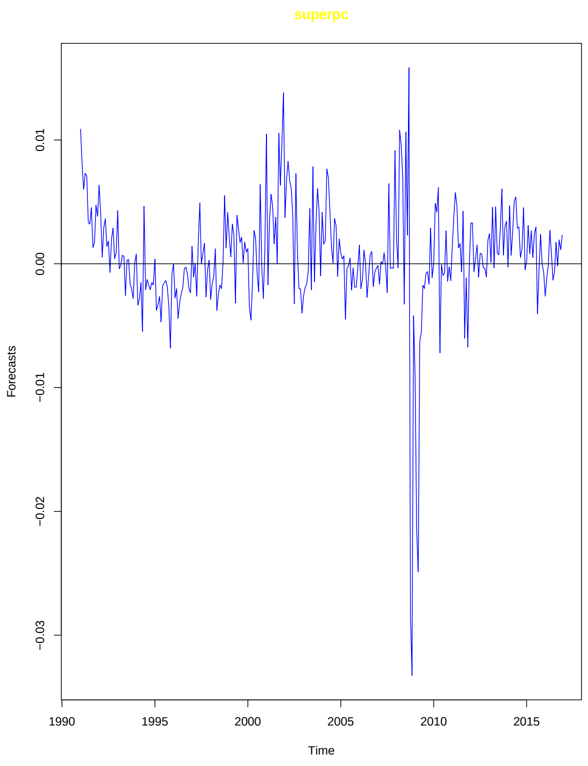
Figure 3.5: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



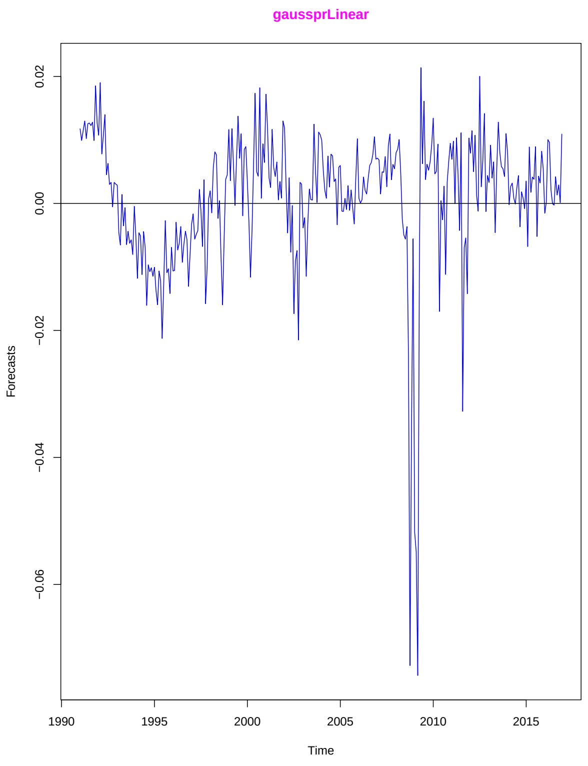
(a) ICR



(b) PCR

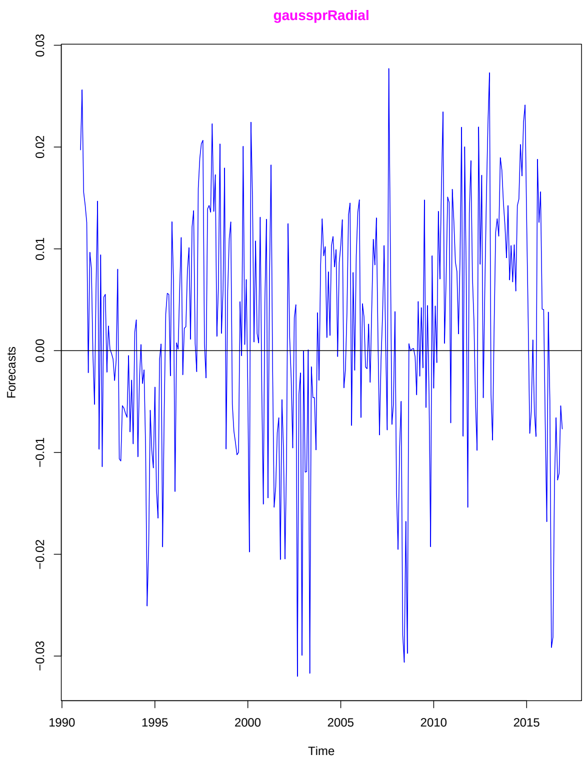


(c) Super PC

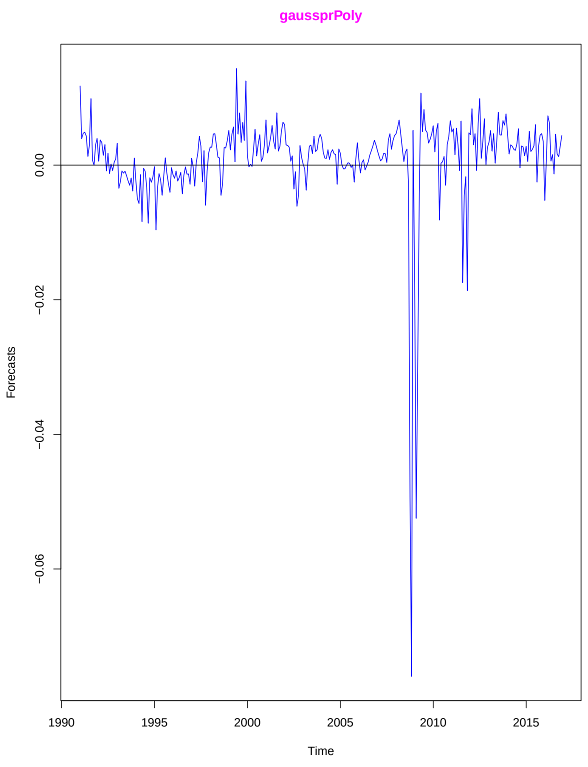


(d) GPR with LBF

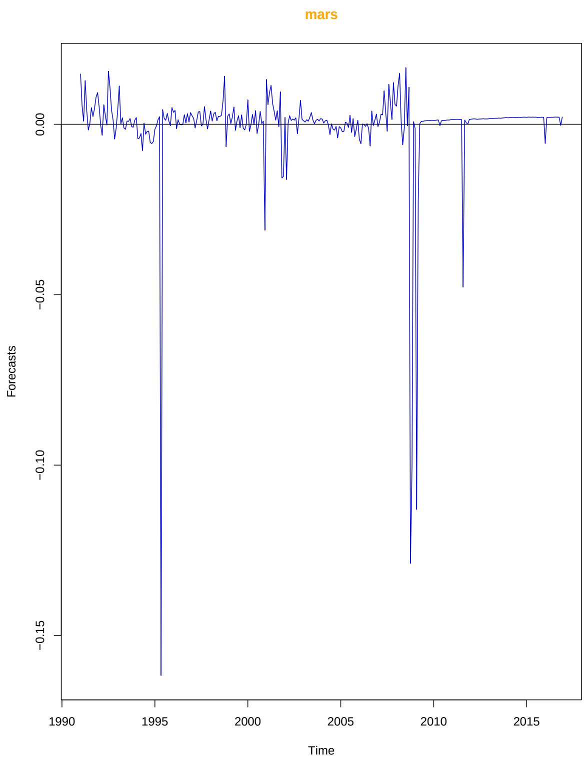
Figure 3.6: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



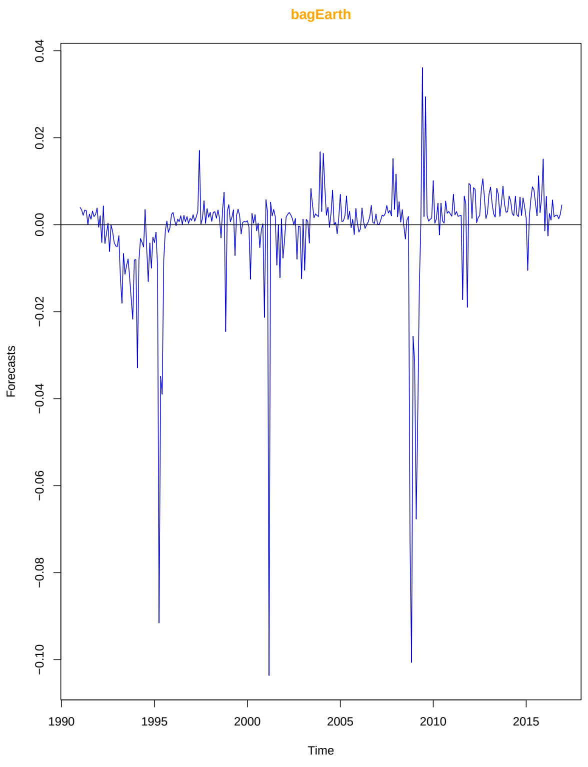
(a) GPR with RBF



(b) GPR with PBF

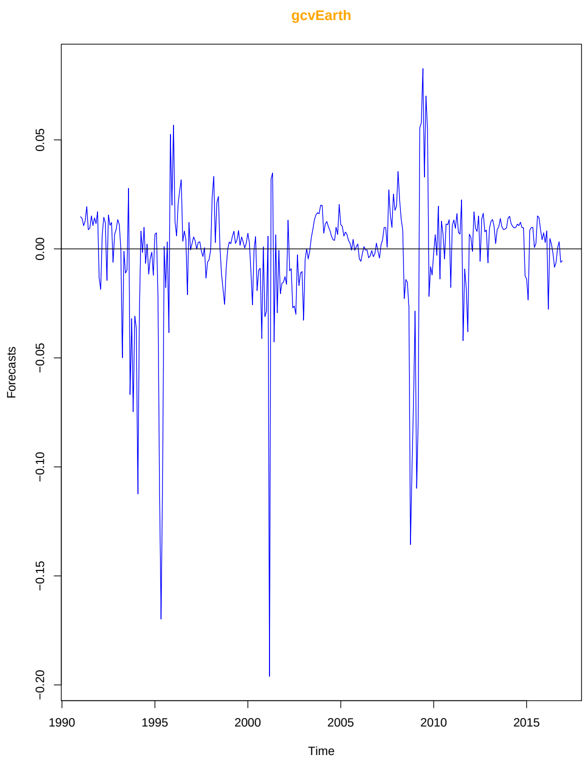


(c) MARS

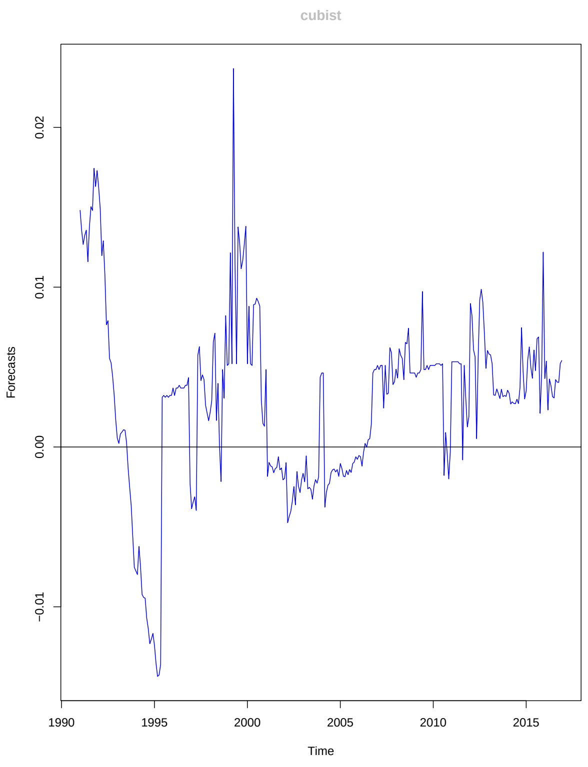


(d) Bagging Earth

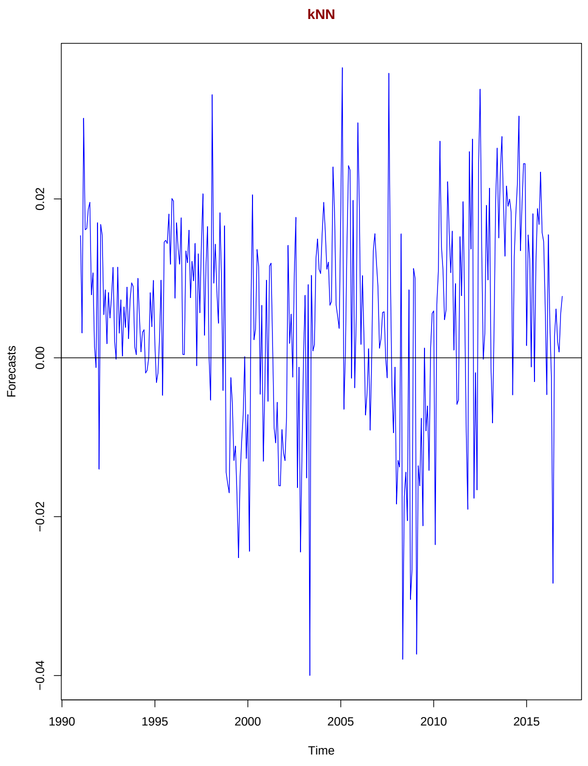
Figure 3.7: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



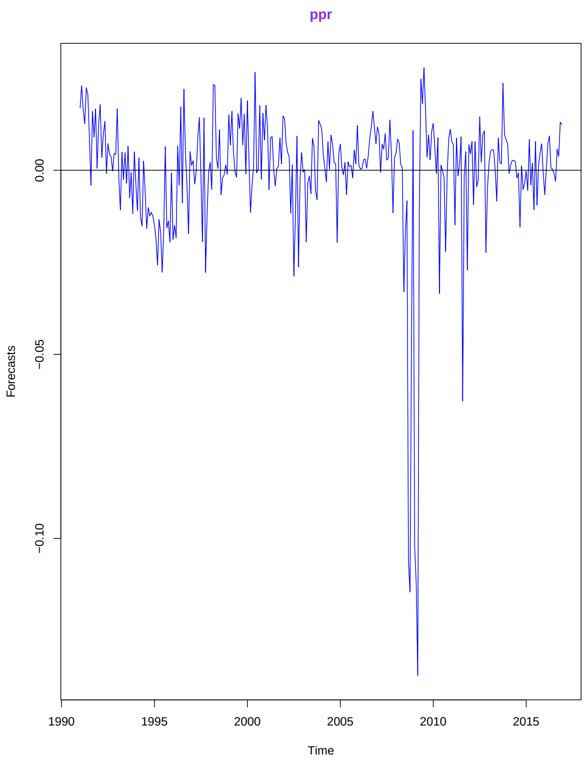
(a) GCV Earth



(b) Cubist

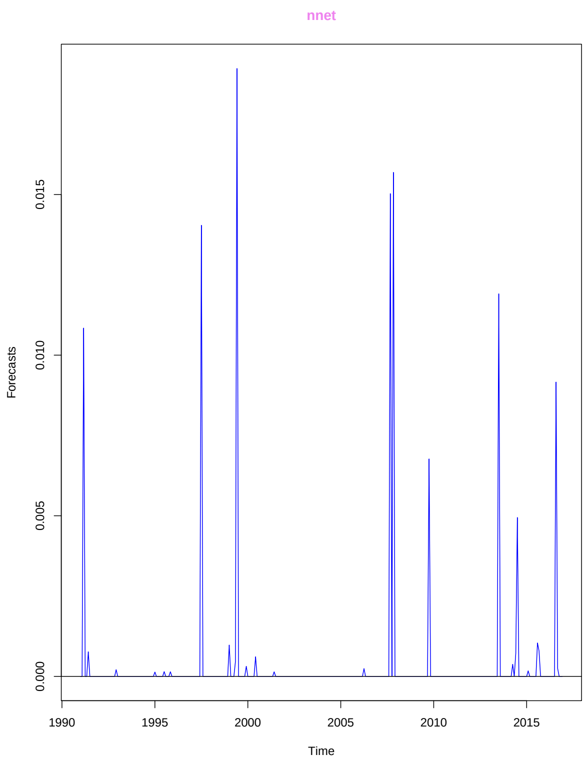


(c) kNN

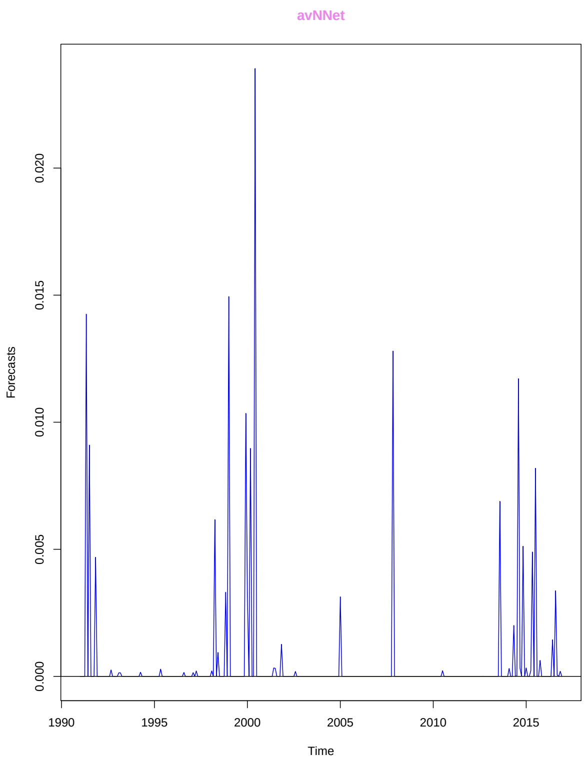


(d) PPR

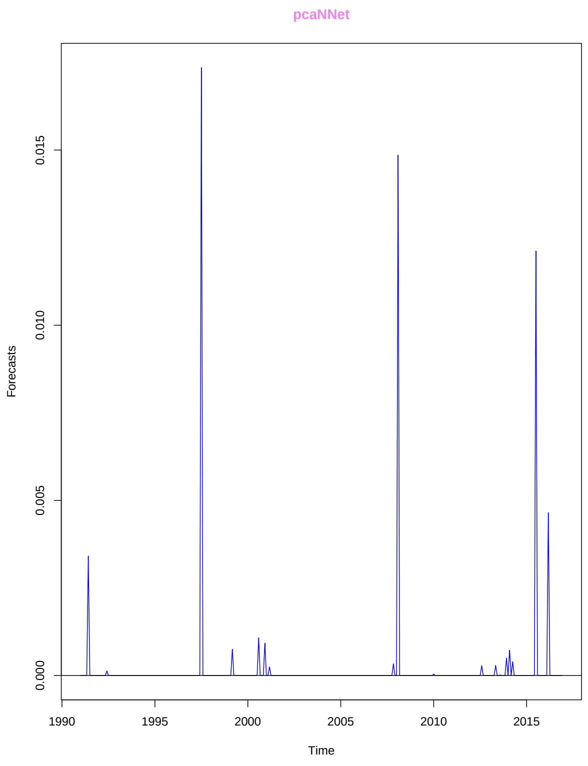
Figure 3.8: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



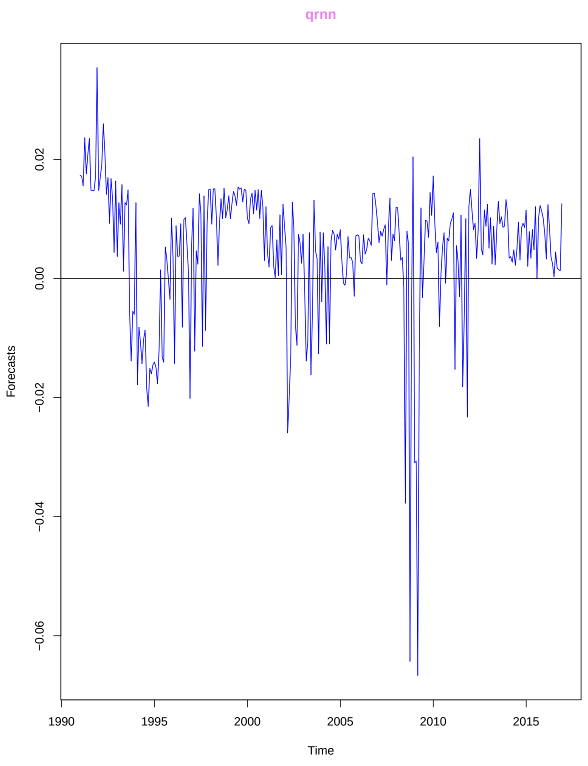
(a) NNET



(b) AVNNET

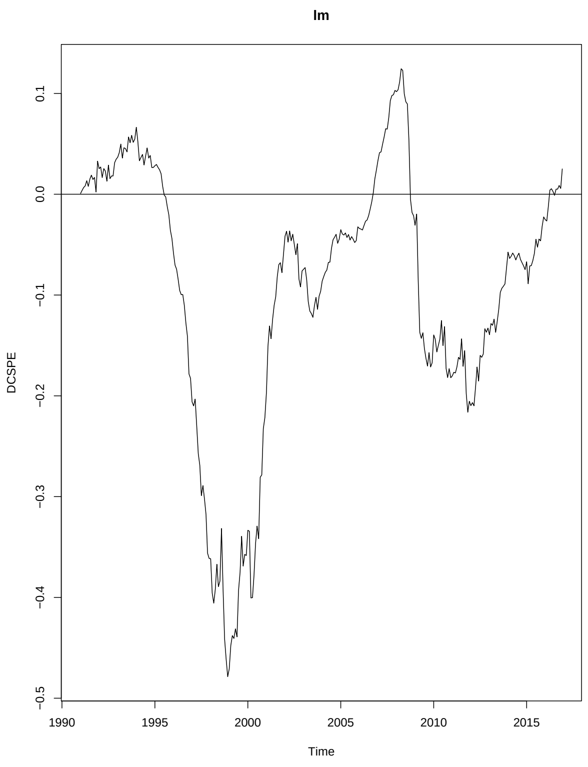


(c) PCANNET

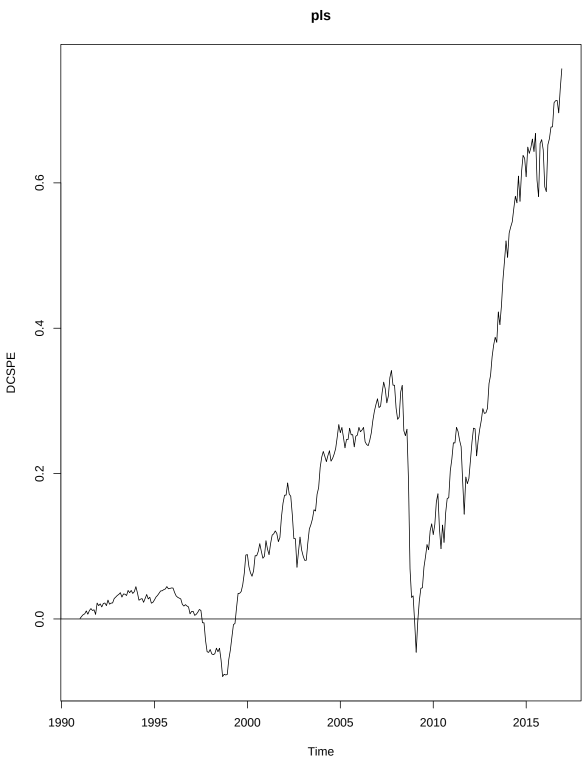


(d) QRNN

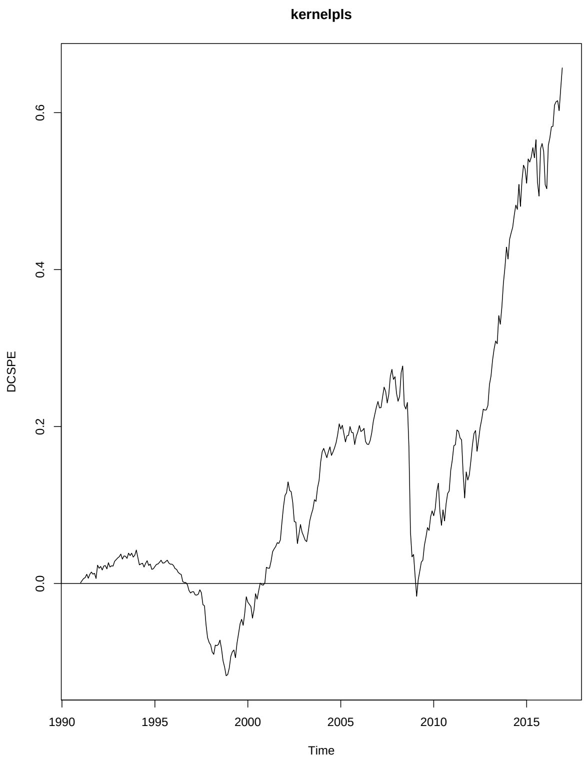
Figure 3.9: Out-of-Sample U.S Monthly Equity Premium Forecasts produced by the RT Models (Continued)



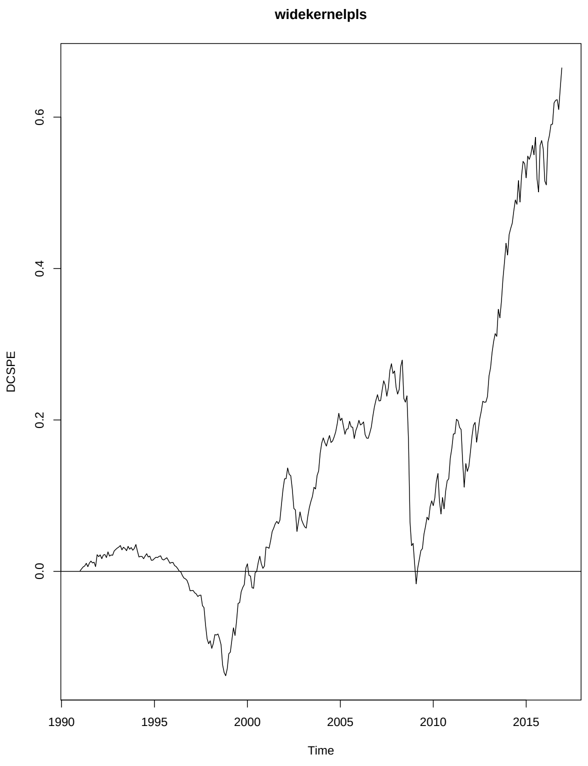
(a) Linear Model



(b) Partial Least Squares

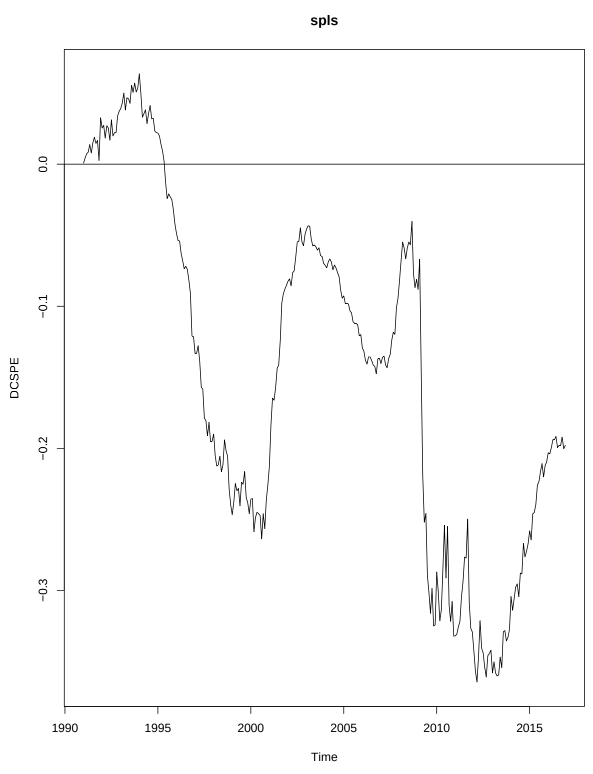


(c) Kernel PLS

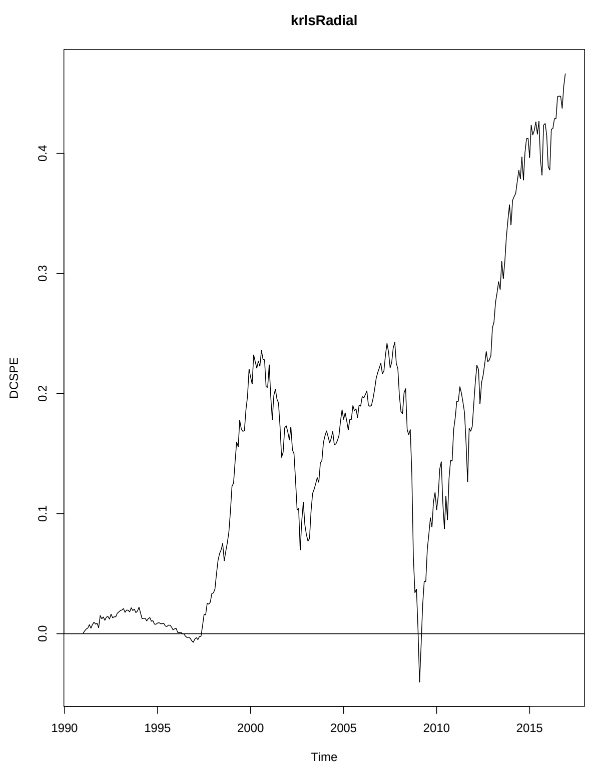


(d) Wide Kernel PLS

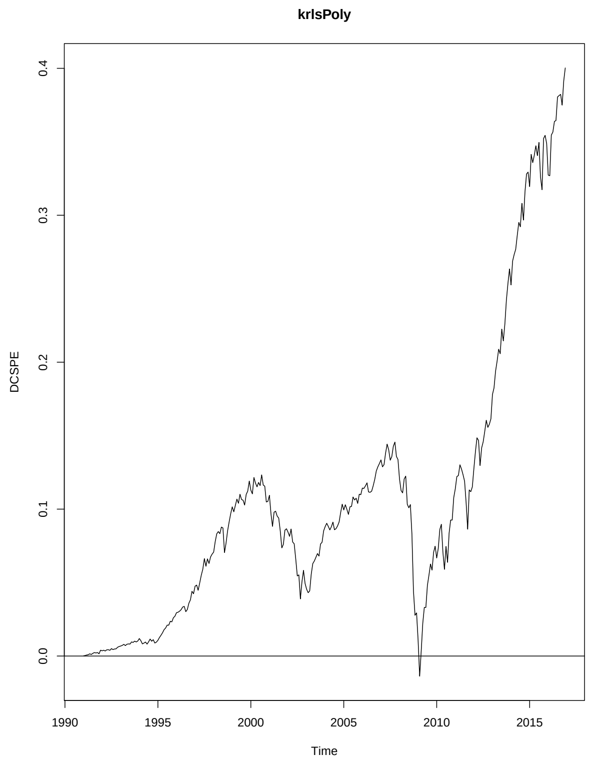
Figure 3.10: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE)



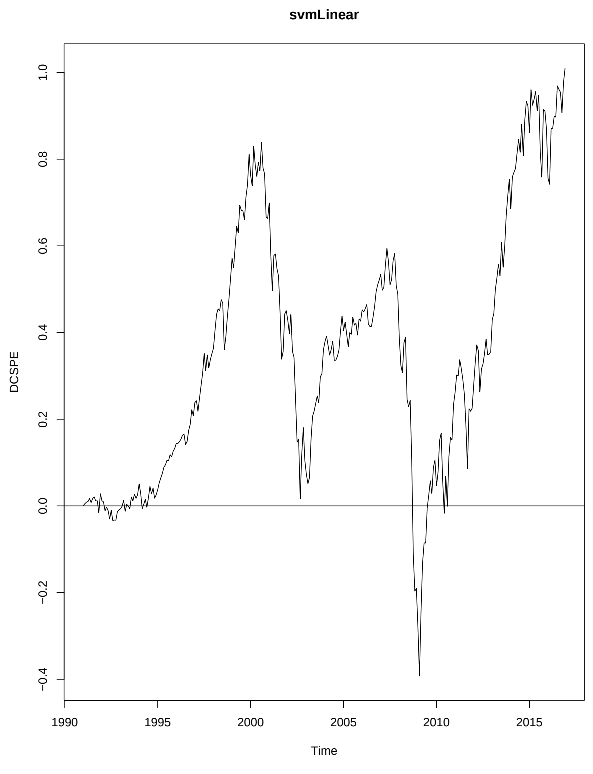
(a) Sparse PLS



(b) KRLS with RBF

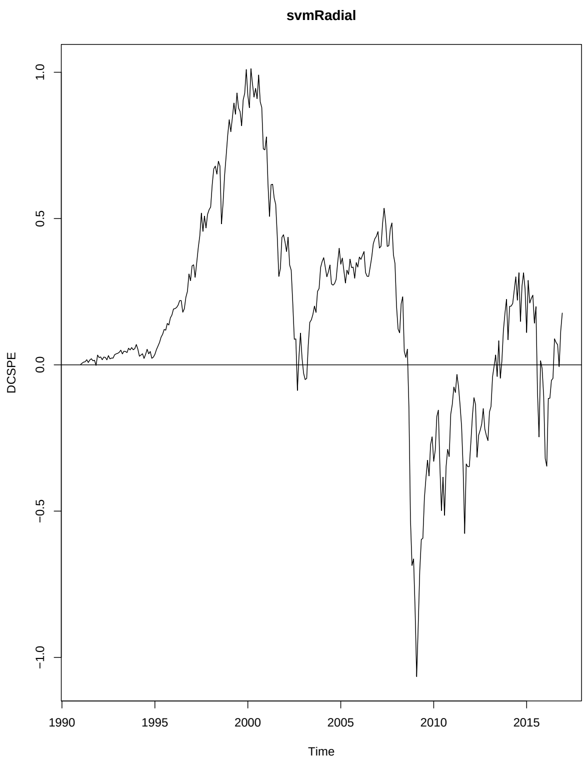


(c) KRLS with PBF

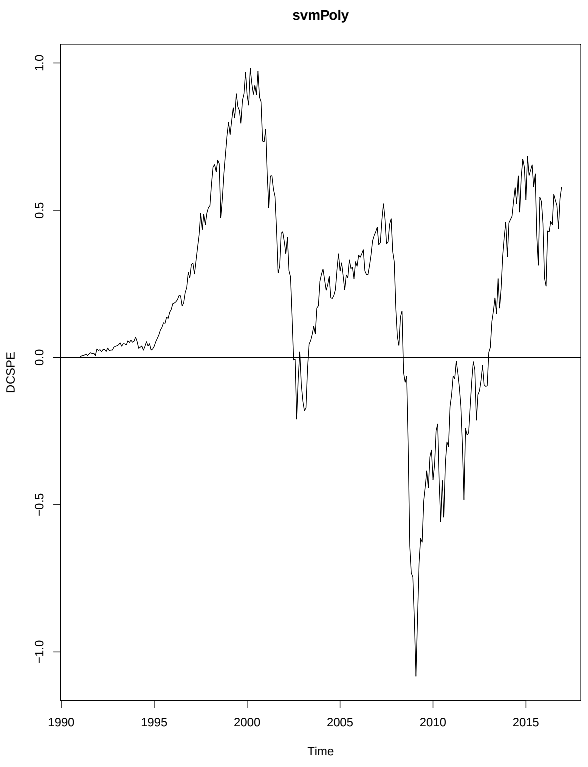


(d) SVM with LBF

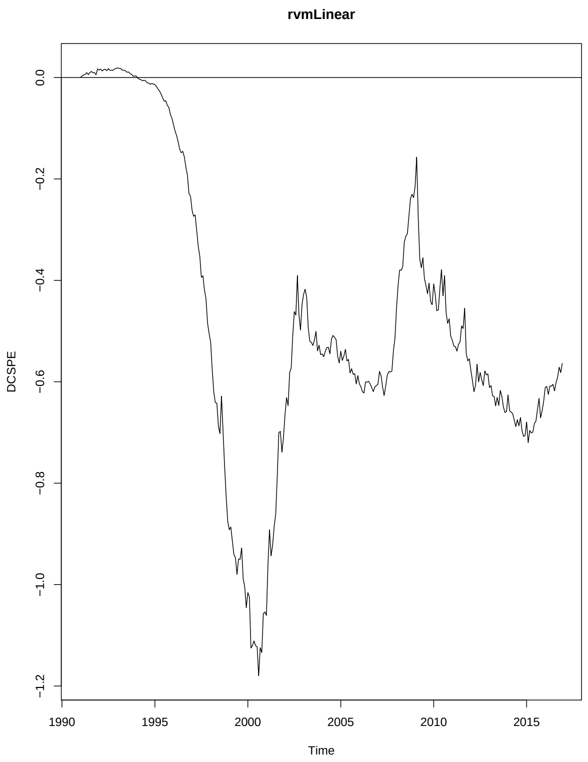
Figure 3.11: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) continued



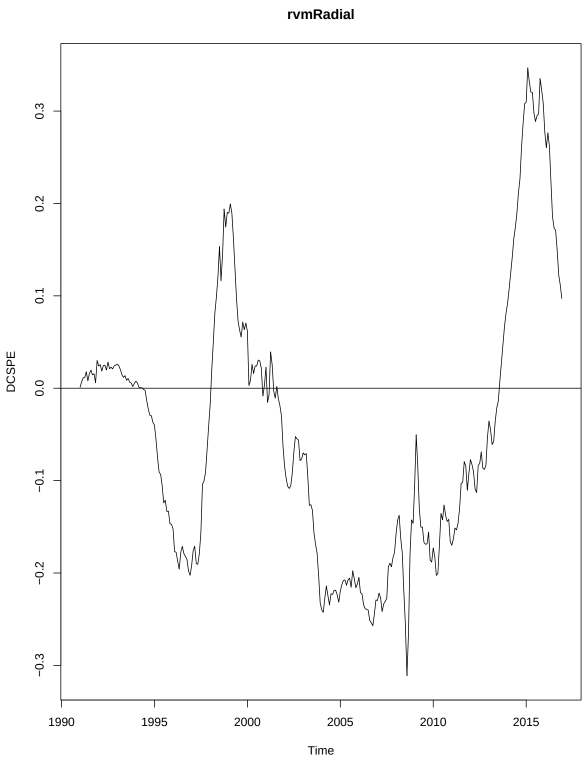
(a) SVM with RBF



(b) SVM with PBF

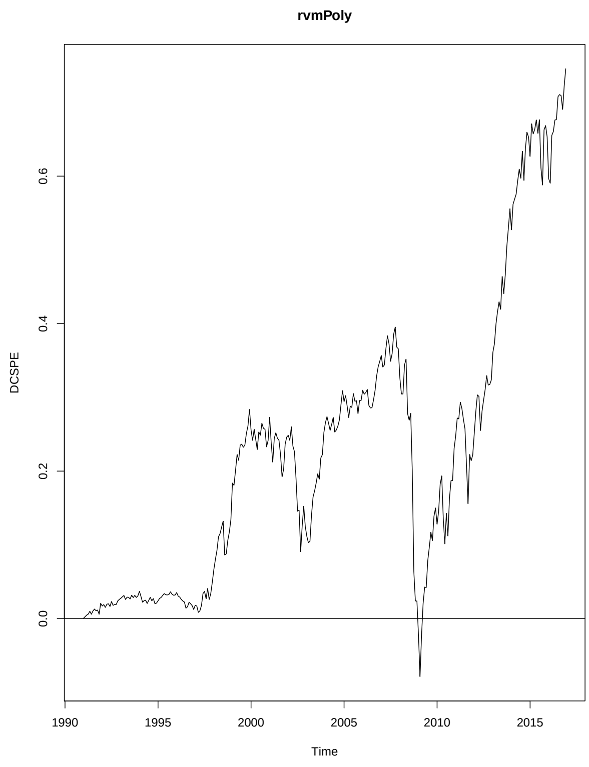


(c) RVM with LBF

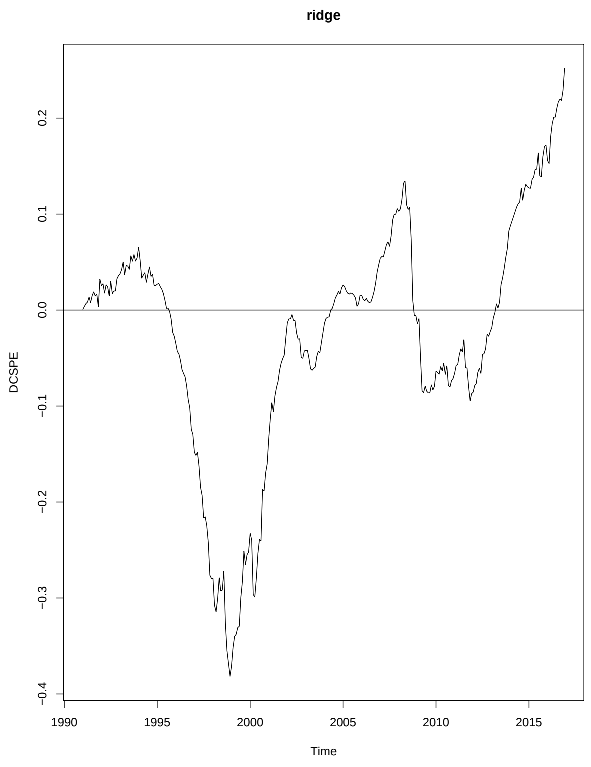


(d) RVM with RBF

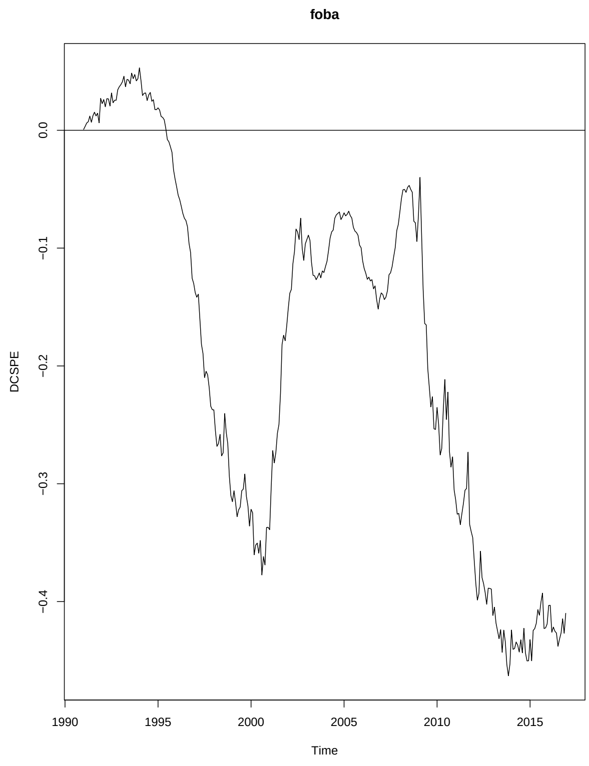
Figure 3.12: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) continued



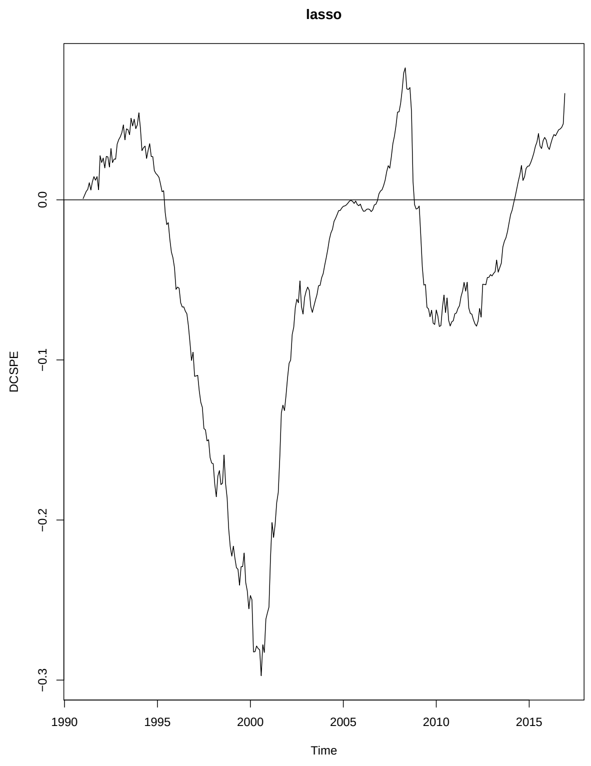
(a) RVM with PBF



(b) Ridge

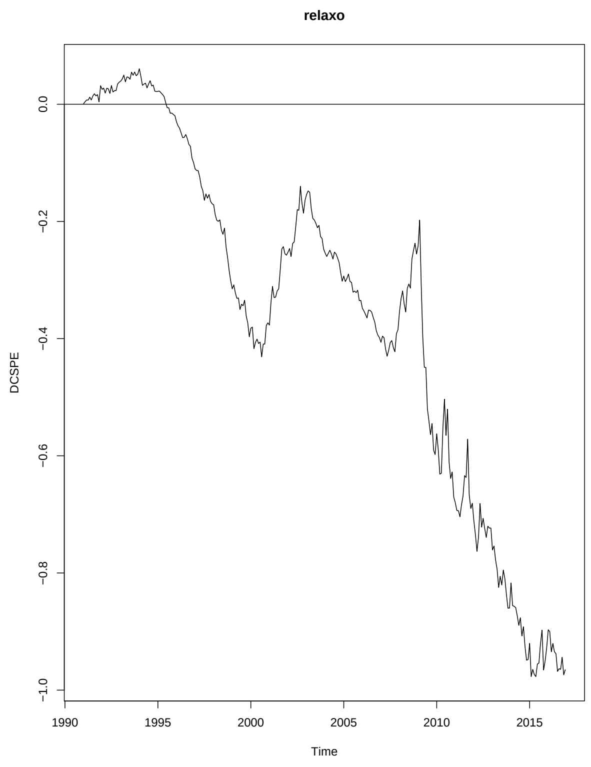


(c) FOBA

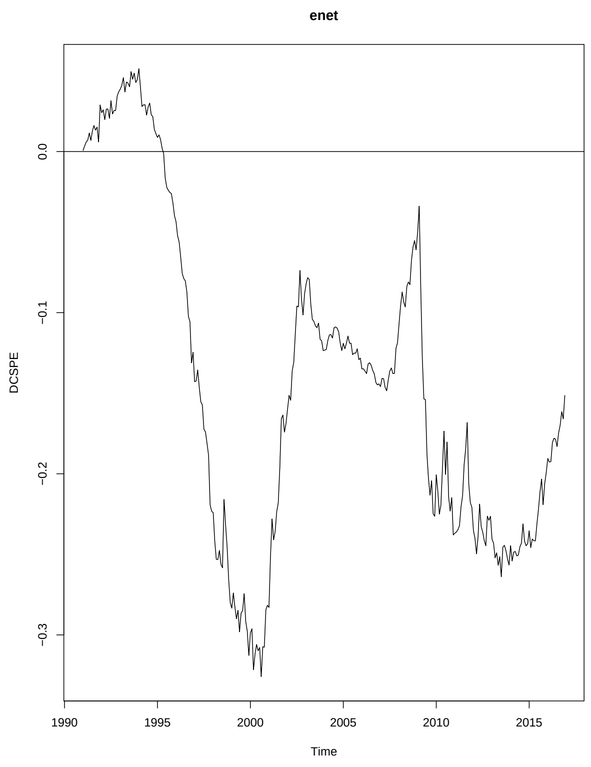


(d) LASSO

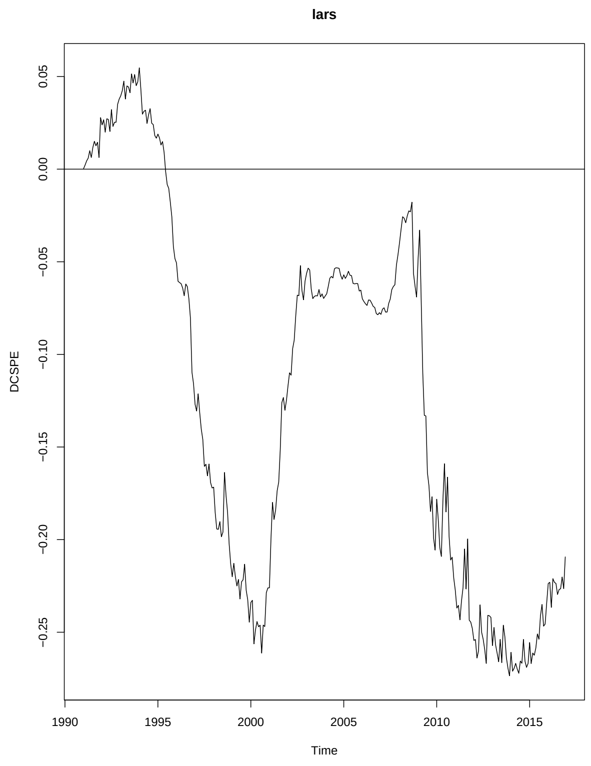
Figure 3.13: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) continued



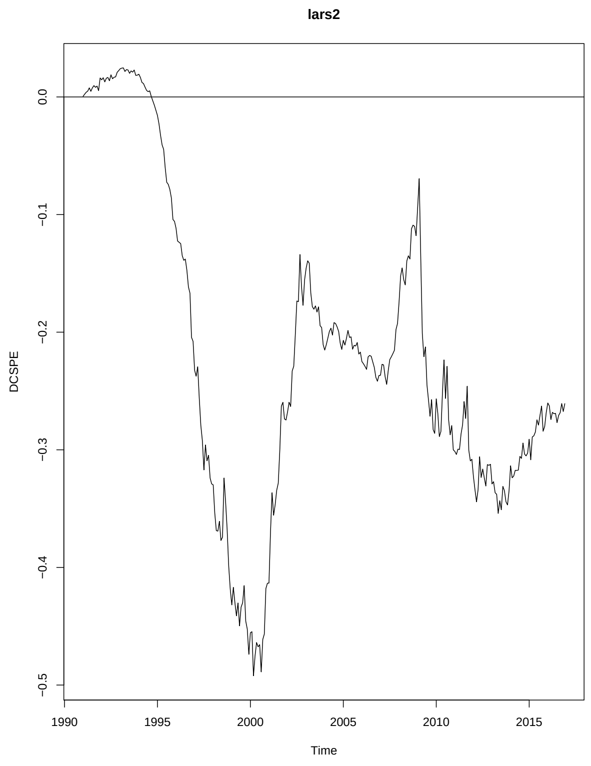
(a) RELAXO



(b) Elastic Net

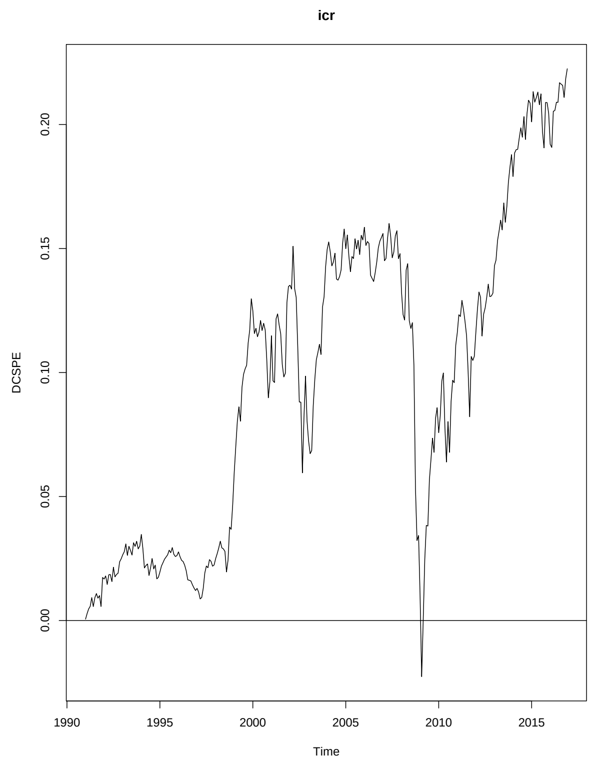


(c) LARS

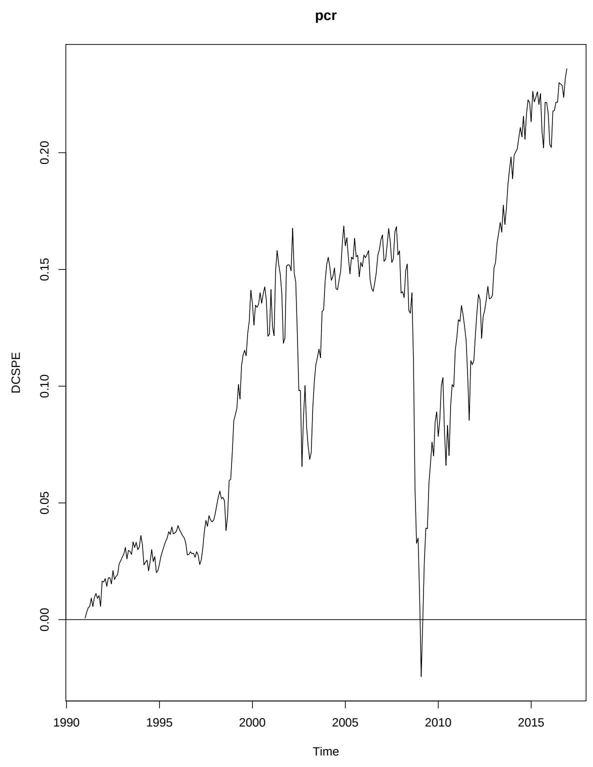


(d) LARS2

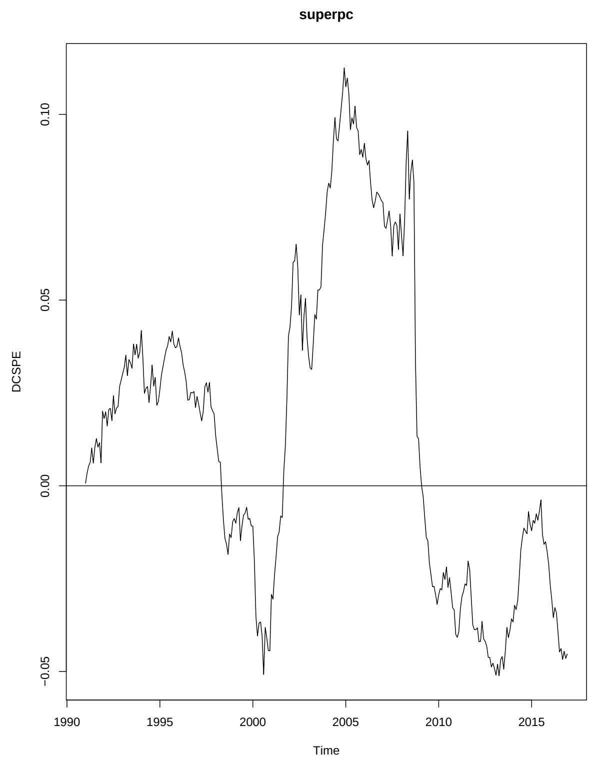
Figure 3.14: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) continued



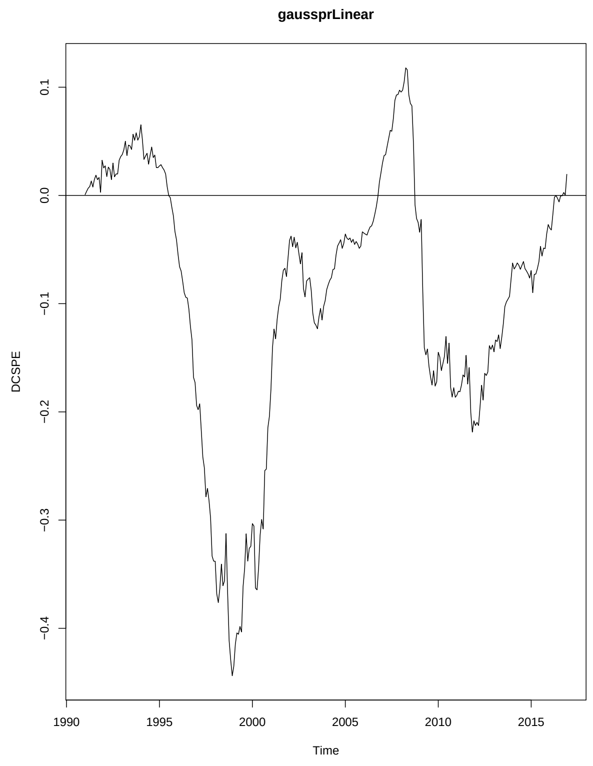
(a) ICR



(b) PCR

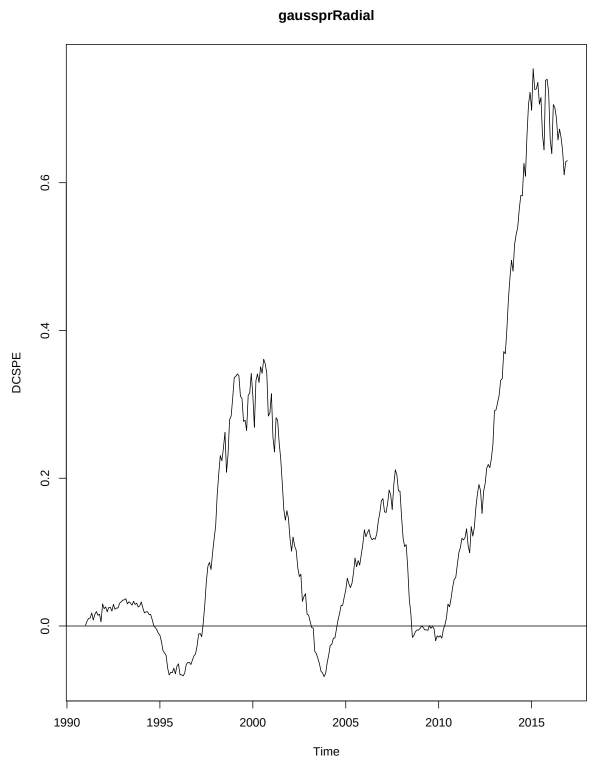


(c) Super PC

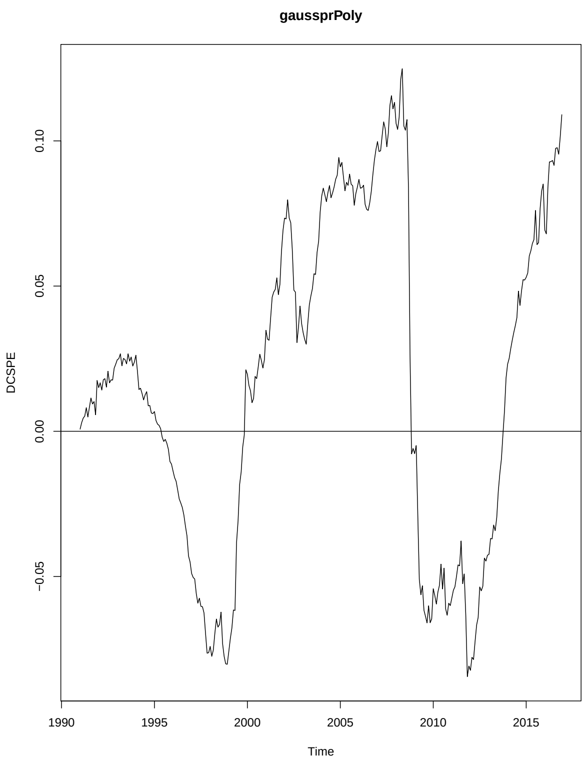


(d) GPR with LBF

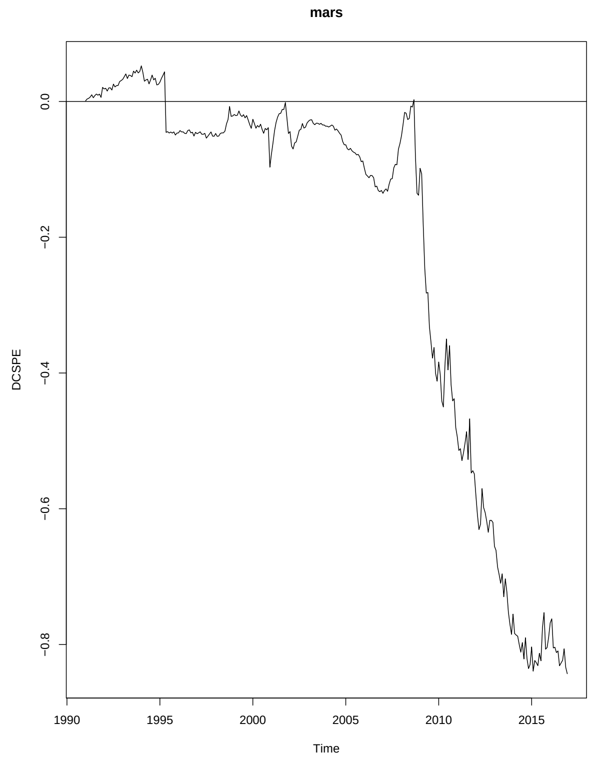
Figure 3.15: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) Continued



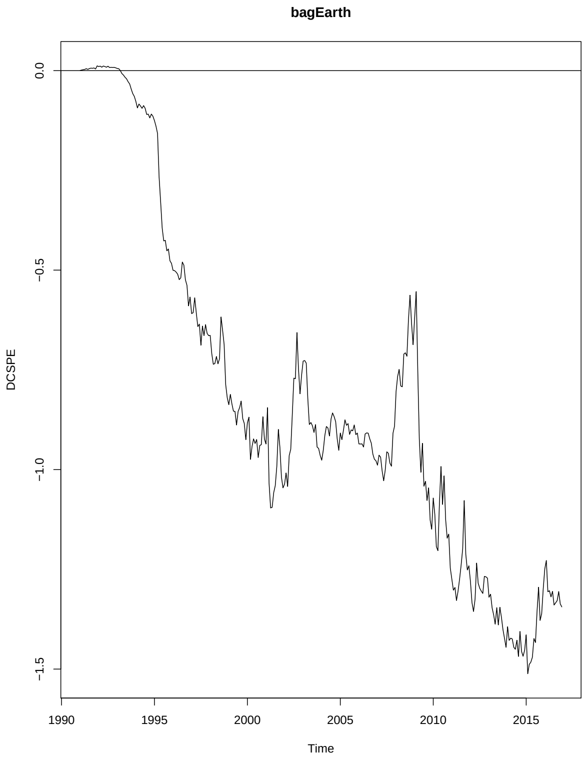
(a) GPR with RBF



(b) GPR with PBF

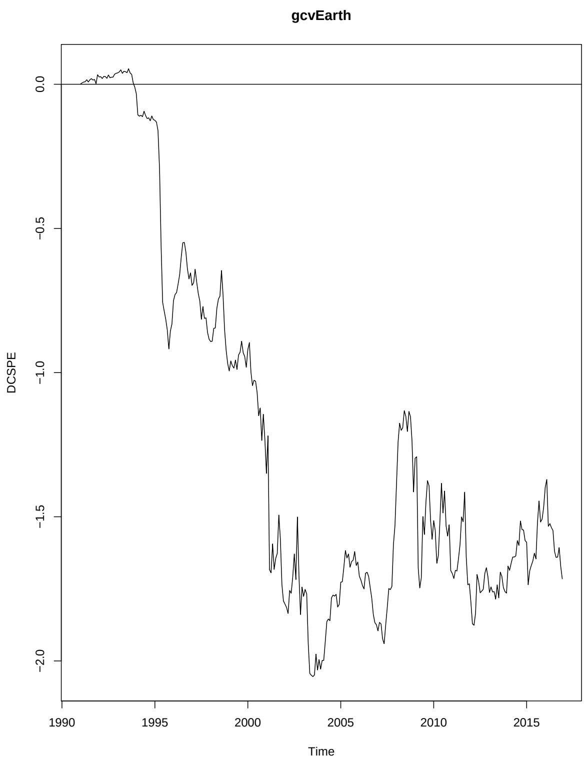


(c) MARS

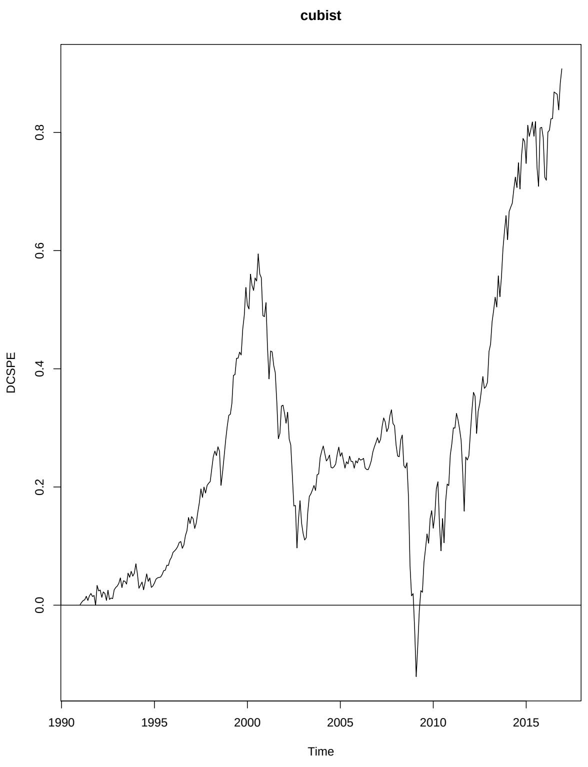


(d) Bagging Earth

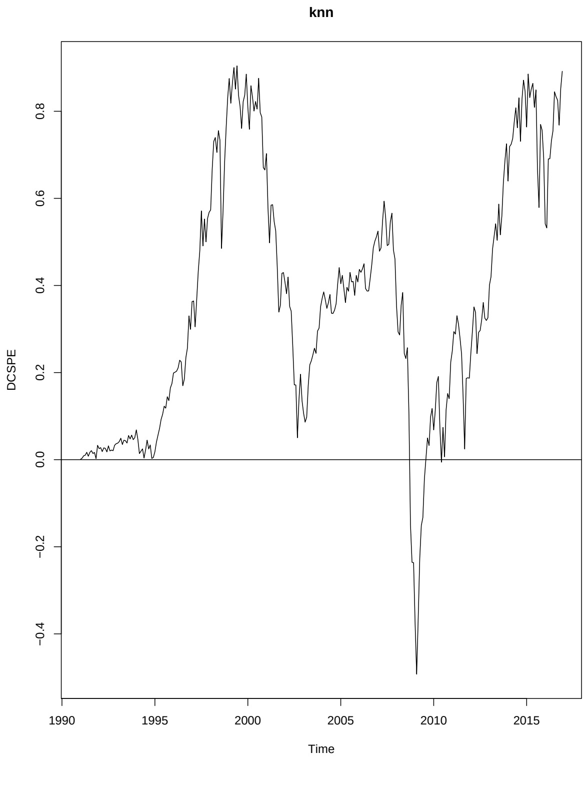
Figure 3.16: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) Continued



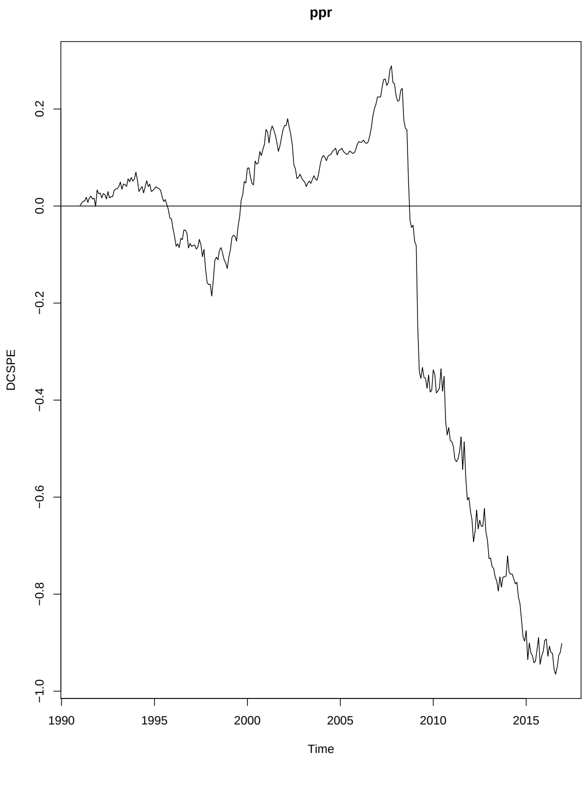
(a) GCV Earth



(b) Cubist

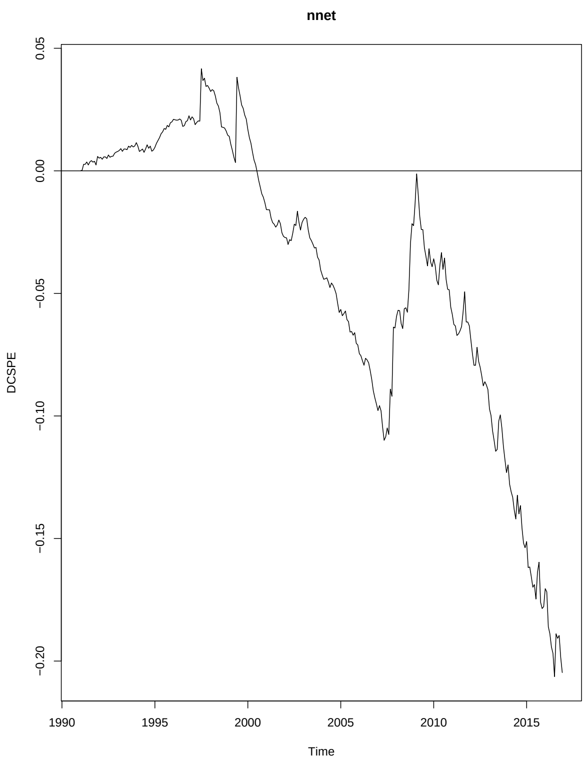


(c) kNN

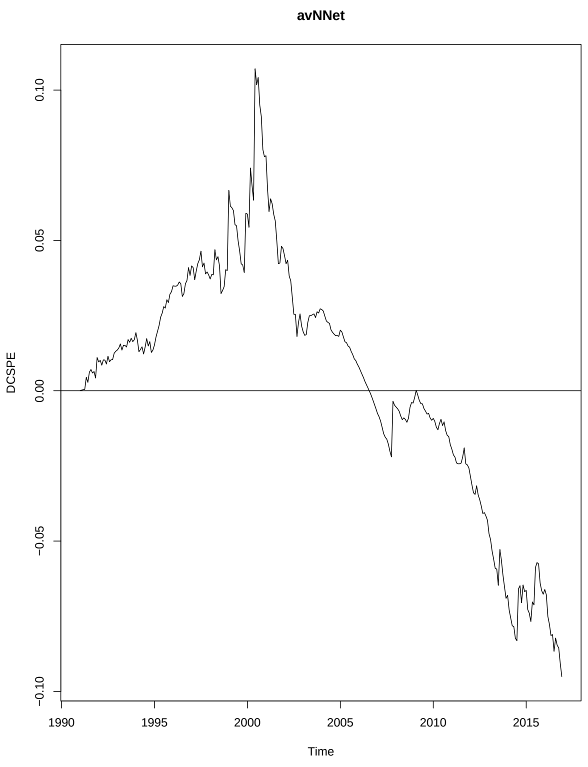


(d) PPR

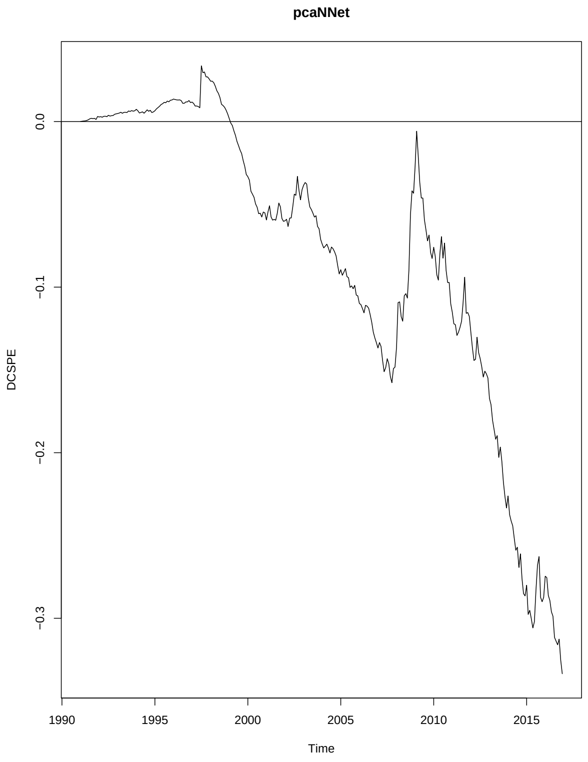
Figure 3.17: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) Continued



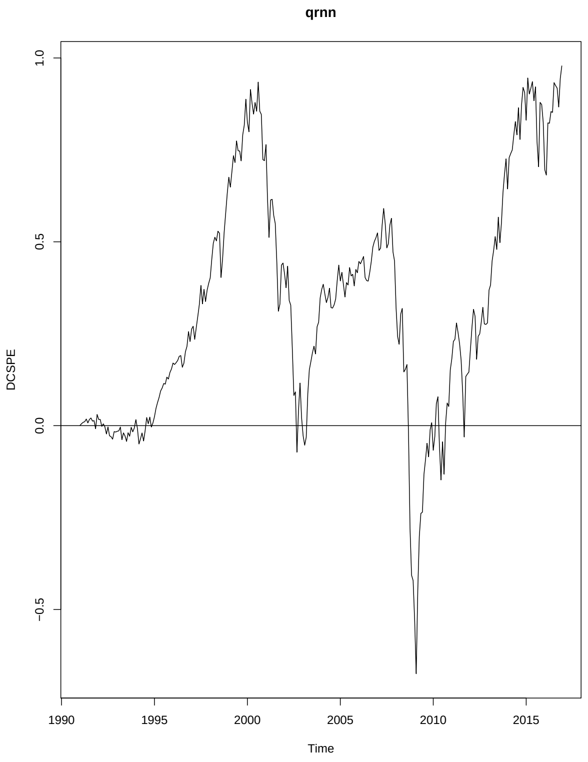
(a) NNET



(b) AVNNET



(c) PCANNET



(d) QRNN

Figure 3.18: Difference in benchmark historical average forecast cumulative square prediction error and the individual RT model forecast cumulative square prediction error (DCSPE) Continued

terms of increased annual portfolio returns for a mean-variance investor. However, the R^2_{OOS} does not explicitly account for an investor's risk over the out-of-sample period, and hence the quest to evaluate the corresponding Sharpe ratios (SR) and utility gains (UG) based on the out-of-sample period. Reconciling the statistical and economic evidence in an attempt to guarantee the future expectation of a mean-variance portfolio investor is another crucial issue in this study. It is noticeable that all the RT models demonstrate useful evidence of economic prodictability, owing to their cumulative returns (CRs) and corresponding positive Share ratios (SRs). It is also importatnt to note that a model may underperformed the benchmark historical avergae, but it may still possess good statistical and economic predictive power, providing useful information to a mean-variance portfolio investor. As suggested by Campbell and Thompson (2007), a mean-variance investor can increase her monthly portfolio return by computing a proportional factor $\dfrac{R^2_{OOS}}{(SR)^2}$, where SR is the Sharpe ratio. The average risk-free treasury bill rate in this study is computed to be $\bar{R}_{free} = 0.220\%$ and the risk aversion parameter $\lambda = 3$. Following Rapach et al. (2007) and Rapach et al. (2010), the utility gains (UGs) in this study are expressed in the form of average annualized percentage returns, also known as certainty equivalent returns. The UG is important in that it provides useful economic information on the portfolio management fee that an investor would be willing to pay in order to have access to the additional available information in the RT forecasting model relative to the sole information in the historical equity premium. A model that produced a UG based on the out-of-sample periods greater than the average risk-free treasury bill is preferable to its counterpart, for a mean-variance portfolio investor. If risk is equal, then it is more profitable to invest in treasury bills than in the portfolio based on the model. In this study, 33 out of the 36 RT models produced positive UGs that are greater than the average risk-free treasury bill, suggesting better alternatives to a mean-variance portfolio manager than the risk-free treasury bill. The $NNET$ and $PCANNET$ that underperformed the benchmark historical average in terms of statistical predictability, also provide corresponding weak economic predictability, suggesting future portfolio investments to a mean-variance investor on the risk-free treasury bill than the specific models. It is worth noting that the RT models that statistically beat the historical average, also economically beat the average risk-free treasury bill, providing better UGs in each case, except for the AVNNET. Contrary to the statistically predictive results, the GCVEarth that underperformed the becnhmark historical average, appeared to produce better economically predictive results, outperforming the average risk-free treasury bill. In terms of ranking the performance of the RT models, it is noticeable that some of the RT models that produced smaller $MSEs$, do not provide corresponding higher UGs by comparison. The WideKernelPLS produced the smallest MSE with the highest CR and SR, while the Gaussian processes regression with radial basis kernel function (GPR with RBF) produced the highest average utility (AU) with a corresponding highest UG in this direction. Thus, the findings agreed with the notion in the litearture in which a model may be weak in statistical predictive power, but it may provide useful economic information in a real-time setting.

The empirical findings in this study revealed that the sophisticated RT models significantly beat the benchmark historical average both statistically and economically, producing smaller $MSEs$, compared to the methods used by many scholars in the literature. Thus, the sophisticated RT models used in this study appeared to guarantee a mean-variance investor in a real-time setting who optimally reallocates a monthly portfolio between equities and risk-free treasury bill using equity premium forecasts at minimal risk.

3.5 Conclusion

The research has fills the gap on the controversial arguments between different scholars about the use of individual or collective variables to significantly forecast the U.S. equity premium out-of-sample relative to the benchmark historical average. Rather than individual financial variables, the sophisticated RT models incorporate all covariates in the kitchen sink form, regardless of variable importance at initial stage, to train the various models and use the resulting model together with the cross-validated tuning parameters to run the forecasts recursively. The replication of the benchmark historical average in this chapter have corroborates the empirical analysis made by scholars in the existing literature. The superior performance of any model depends on the modelling and forecasting techniques used, and not necessarily the dataset under consideration.

The sophisticated RT forecasting models used in this study produced significant evidence of statistical predictability with economic significance out-of-sample relative to the benchmark historical average. The RT models generally produced smaller mean squared prediction errors MSE and better R^2_{OOS}, compared to the analysis shown in the literature in which individual indicators were used to forecast the U.S. equity premium out-of-sample. The results showed that the collective variables provide statistically and economically useful forecasts of the U.S equity premium for investors in real time setting, and demonstrate significant evidence of consistently beating the benchmark historical average out-of-sample. Interestingly, the concept of training and validating a kitchen-sink regression model associated with the underlying sophisticated techniques in this study is evidently considered to be statistically and economically significant approach, adding enormous impact to enrich modern econometric and financial literature to boost the expectation of investors on a long run. However, the statistical predictive superiority of an RT model does not necessarily guarantee a corresponding economic superiority in this direction.

Overall, the WideKernelPLS model produced the best result in terms of statistical predictability of the U.S. monthly equity premium out-of-sample while the Gaussian processes regression with radial basis kernel function (GPR with RBF) produced the highest average utility (AU) with a corresponding highest utility gain UG, indicating the best economic significant result. They provide meaningful economic information on mean-variance portfolio investment for investors who are timing the market to earn future gains at minimal risk, compared to the other RT out-of-sample forecasting models. Therefore, the best performing sophisticated RT out-of-sample forecasting models appeared to guarantee a mean-variance investor in a real-time setting who optimally reallocates a monthly portfolio between equities and risk-free treasury bill using equity premium forecasts at minimal risk.

[2]

[2]This chapter has been submitted for publication, and it is currently on peer review, as follows:
Iworiso, J. & Vrontos, S. (2019). Forecasting the U.S. Equity Premium with Regression Training Techniques. Journal of Empirical Finance. Unpublished (peer review in progress).

Chapter 4

Deep Learning Techniques for Stock Market Statistical Predictability with Economic Significance

4.1 Introduction

Stock market predictability is an open-ended debatable research issue with conceptualization in this modern age of rapid technological growth. A number of methods have been proposed by several scholars as shown in existing literature to analyse different financial data such as bonds, exchange rates, microeconomic variables etc. The researcher's proposed methodology could be promising when a specific dataset is used (i.e., financial, macroeconomic, medical etc.), but not necessarily promising in all datasets, and hence, the quest for robustness over the proposed methodologies.

This chapter proposes a more sophisticated techniques known as deep learning techniques which have the ability to extract features from a large raw dataset without relying on prior knowledge of predictors. Day and Lee (2016) described *deep learning* as a deep neural network, which is a more sophisticated aspect of machine learning. It is a form of machine learning technique that involved the use of data to train a model or recognize pattern(s) or label instances in order to make predictions from new data in a more sophisticated manner (Heaton et al., 2017). The various aspects of the deep learning techniques in this study include the deep neural network (DNN), stacked autoencoder (SAE), $H2O$ deep learning $H2ODL$, long-short-term-memory (LSTM), and the fusion of some of these techniques. The activation function and dropout approach were also introduced. Deep learning techniques has demonstrated useful evidence of feasibility and effectiveness in many fields of study, in computer science, biology, medicine, linguistics, physics, and is currently attracting rapid attention in economics and finance.

The output of this study shall provide extensive knowledge on the effectiveness of deep learning techniques with robustness of the proposed methodologies in predictability of equity premium. It shall intensify investors target by providing remedy to curb the identifiable research ills in prospective profit driven portfolio investment at minimal risk. It shall enrich empirical literature and provide a basis for further research work.

4.2 Review of Relevant Literature

Prediction of stock market behaviour is a challenging issue in financial analysis. Empirical literature have shown several attempts made by researchers to improve the predictive performance in stock market analysis with application of sophisticated econometric, statistical or machine learning techniques over the old-fashioned financial time series models (Di Persio and Honchar, 2016; Tsantekidis et al., 2017). In recent time, the feasibility of artificial neural networks in stock market forecasting with desirable predictive performance had led to the introduction of a more sophisticated learning technique known as *deep learning.* Over the years, artificial neural networks as well as deep belief networks are shown to be useful techniques for handling advanced linear and nonlinear relationships among the data. They offer a great deal of successes in time series analysis, natural language processing, pattern mining and image classification among others (Hinton et al., 2006; Akita et al., 2016; Yoshihara et al., 2014; Dixon et al., 2016; Vargas et al., 2017; Deng et al., 2017; Shen et al., 2018; Candel et al., 2016; Dixon et al., 2015; Sirignano et al., 2016; Pradeepkumar and Ravi, 2017; Hosein and Hosein, 2017; Yoshihara et al., 2015).

A review of unsupervised feature learning and deep learning for time series modelling in Längkvist et al. (2014) revealed that the deep learning techniques provide better representation and classification on a multitude of time series problems when properly configured and trained, compared to the shallow approaches. It is worth noting to give less attention on the preprocessing pipeline for a specific time series problem and focus more on learning better feature representations for a general purpose algorithm for data structured, regardless of the underlying application. The unsupervised feature extraction methods using principal component analysis, autoencoder and restricted Boltzmann machine on a deep learning network to predict future market behaviour by Chong et al. (2017) has demonstrated practical insights and usefulness of deep learning techniques in stock market forecasting with significant improvement in performance evaluation. A comparative analysis on predicting the trend of stock market using recurrent deep neural networks compared to support vector machines and deep belief networks in Yoshihara et al. (2014) indicate the effectiveness of the approach, statistically significant improvement and outperformance over the later models especially when the analytical process is focused on specific period after a known significant event in financial domain. The analysis is suggesting a controversial superiority of recurrent neural networks over the deep belief network in this direction. A notable comparative approach in financial literature is to compare the predictive performance of any model with the benchmark buy and hold trading strategy. The application of deep learning ensemble approach with stacked denoising autoencoders for modelling and forecasting crude oil prices showed statistically significant evidence of superiority of the deep learning techniques over the bootstrap aggregation and other machine learning techniques (Zhao et al., 2017). The introduction and implementation of a hybrid genetic-neural architecture for stock indexes forecasting with consideration of realistic trading commission by Armano et al. (2005) proved to be promising in the selected application task and the overall results shows evidence of superior outperformance over the benchmark buy and hold strategy for a large test dataset. The empirical results in Hu et al. (2018), Feuerriegel and Fehrer (2016), Heaton et al. (2016) also confirmed that the application of deep learning techniques in financial analysis seek to outperformed both the standard methods in finance and the conventional machine learning techniques. Contrary to these analyses, is the empirical results in Krauss et al. (2017) in which a random forest outperformed both deep neural networks and gradient boosted trees in the investigation of statistical arbitrage on $S\&P500$, although a further investigation by hyper-parameter optimization to yield improved results for the deep neural networks is required as an area of future research. The combination of the base learners: deep neural networks, boosted shallow trees and decorrelated trees of high depth into a simple ensemble outperformed the various models in their individual form. It could also be plausible to use the restricted Boltzmann machine to stack autoencoders that can extract features from low signal to noisy time series dataset with appropriately preprocessed inputs, as demonstrated in Takeuchi and Lee (2013), in which deep learning technique was applied to enhance momentum trading strategy in stocks. However, the notion of stacking the network layers in deep learning is proven to be better in performance than the use of shallow structures (Gamboa, 2017) and are evidently seen to be promising with underlying improvement in predictive performance subject to further investigation in future studies.

Some scholars adopted a concept known as long short-term memory (LSTM) units associated with deep learning techniques for stock return predictability. The empirical results in Fischer and Krauss (2018) in financial stock market predictions by deep learning techniques with LSTM seems to have outperformed other approaches such as the random forest, deep neural net and logistic regression. It is worth noting that the inclusion of an LSTM network helps to extract meaningful information effectively from noisy financial time series data. The combination of stacked autoencoders and LSTM in deep learning framework for financial time series in Bao et al. (2017) also demonstrate a superior performance in predictive accuracy and profitability measures over the artificial neural networks and support vector regression. The findings in Xiong et al. (2015) employing a similar approach also give a superior outperformance over the ridge, least absolute shrinkage and selection operator (LASSO) and generalized autoregressive conditional heteroskedastic (GARCH) benchmark models by about 31% margin. The activation function in the deep learning and neural network models were found to be useful and promising with better predictive performance for predicting the stock market behaviour. The use of deep neural networks and transfer learning for decision support from financial disclosures with the aim of predicting stock market movement by Kraus and Feuerriegel (2017) has again confirmed the superiority of the LSTM over all traditional machine learning models based on bag of words data analytical approach, owing to its higher directional accuracy as compared to the traditional machine learning techniques. Traditional machine learning techniques includes clustering, ensembles, dimensionality reduction, reinforecement learning, artificial neural networks etc. Li et al. (2017) added that the adoption of LSTM neural network made deep learning potentially stable in time series analysis even in the presence of strong noise. Nelson et al. (2017) concluded that the LSTM based models do not only outperformed other employed approaches in existing literature but also offer less risks as compared to other strategies. In contrast to the LSTM approach, the analytical findings made by Hiransha et al. (2018) in predicting the National Stock Exchange of India and the New York Stock Exchange using their respective day-wise closing prices as dataset, the convolutional neural network appeared to produce the best prediction accuracy, outperforming the LSTM, multilayer perceptron and recurrent neural network among the deep learning techniques. Thus the convolutional neural network potentially captured the abrupt changes in the system owing to the use of specific window for predicting the future outcome in the stock market.

The use of market indicators: technical versus fundamental or the hybridized approach is a pending confrontational discourse in financial analysis, which require further investigation by various approaches including the deep learning techniques to affirm robustness. The hybridized approach in Adebiyi et al. (2012) showed remarkable results in predicting the future price of stock with better improvement over the use of technical variables alone, suggesting a useful guide for investors in making optimal business decisions in the financial market. The empirical findings in Lee et al. (2017) involving an unsupervised learning phase and a fine-tuning phase (back-propagation algorithm), in which the learning phase uses the restricted Boltzmann machine also confirmed useful evidence of corporate performance predictability, using the company's financial and patent indicators as predictor variables. Although the proposed deep belief network model shows good statistical evidence in predicting a company's performance in terms of technical capability, however there is need to examine the hybridized form and compare results to investigate robustness in further studies.

4.3 Research Methodology

4.3.1 Excess Stock Return Predictability

Let r_t be an excess stock return at time t, and let $\mathbf{Z}_t$ be a k-dimensional covariates, required as predictor variables to predict r_{t+1} at time point $t+1$. Then a kitchen sink predictive model can be obtained in the form (Rapach et al., 2010; Phan et al., 2015):

$$r_{t+1} = \theta_0 + \theta'\mathbf{Z}_t + u_{t+1}; \quad u_{t+1} \sim N(0, \sigma) \tag{4.1}$$

where θ_0 is the intercept; θ is a k-column vector of unknown coefficients; $\mathbf{Z}_t$ is a k-dimensional vector of covariates; u_{t+1} is a disturbance term; $t = 1, 2, ..., T$.

4.3.2 Historical Average

Let r_t be a univariate time series representing the monthly excess stock market return at time t. Following Rapach et al. (2010) and Lee et al. (2015), the future historical average (HA) can be modelled as follows:

$$r_{t+1} = \theta + u_{t+1} \tag{4.2}$$

where θ is the model parameter representing the intercept; u_{t+1} is a zero mean disturbance term; $t = 1, 2, ..., T$. The least squares estimator of the historical average is as follows:

$$\hat{\theta}_{HA} = \frac{1}{T}\sum_{t=1}^{T} r_t \tag{4.3}$$

where $\hat{\theta}_{HA}$ is the parametric estimator of θ.

Thus, the benchmark HA forecasts, denoted by $\hat{r}_{T+1}$ can be obtained as follows:

$$\hat{r}_{T+1|T} = \frac{1}{T}\sum_{t=1}^{T} r_t \tag{4.4}$$

4.3.3 Deep Neural Network

Typically, a single-layer neural network gives the nonlinear connection between the variables h_l, h_{l+1} via a network function, defined as follows:

$$h_{l+1} = H(\mathbf{w}h_l + \mathbf{B}) \tag{4.5}$$

where H is an activation function; $\mathbf{w}$ is $L \times 1$ vector of weight parameters; $\mathbf{B}$ is $L \times 1$ vector of biases; h_l and h_{l+1} are network variables.

Given a predictor function $\mathbf{r} = g(\mathbf{Z})$, then a deep neural network can be constructed by serially stacking the network functions in the form:

$h_1 = H_1(w_1\mathbf{Z} + B_1)$

$h_2 = H_2(w_2 h_1 + B_2)$

$\vdots$

$r = H_L(w_L h_{L-1} + B_L)$

where $\mathbf{Z}$ is $L \times 1$ vector of input covariates; L is the number of layers, $l = 1, 2, ..., L$ (Chong et al., 2017; Heaton et al., 2017).

If $\{(\mathbf{Z}^n, \tau^n)\}_{n=1}^{N}$ represent a dataset of inputs and targets with an error function $\varepsilon(\mathbf{r}^n, \tau^n)$ measuring the difference between the output $\mathbf{r}^n = g(\mathbf{Z}^n)$ and the target τ^n, then the model parameters for the overall network $\phi = \{w_1, w_2..., w_L; B_1, \beta_2..., B_L\}$ can be obtained by minimizing the sum of the errors in the form:

$$\arg\min_{\phi} \left[S = \sum_{n=1}^{N} \varepsilon(\mathbf{r}^n, \tau^n) \right] \tag{4.6}$$

where $n = 1, 2, ..., N$ represent the input units

The solution of the objective function for the above minimization problem can be obtained by gradient descent approach, resulting to (Chong et al., 2017):

$$S = \frac{1}{N} \sum_{n=1}^{N} ||\mathbf{r}^n - \tau^n||^2 + \alpha \cdot \sum_{l=1}^{L} ||w_l||_2 \tag{4.7}$$

where $|| \cdot ||$ denotes the Euclidean vector norm; $|| \cdot ||_2$ is an ℓ_2 vector norm, which serves as a regularizer to eliminate overfitting; α is a user defined coefficient. Such a multi-layer neural network is referred to as a deep neural network (DNN).

4.3.4 Stacked Autoencoder

An autoencoder (AE) involves a deep learning routine which trains the architecture to approximate the covariate $\mathbf{Z}$ by itself ($\mathbf{Z} = \mathbf{r}$) via a bottleneck structure. A bottleneck structure implies a network communication approach in which all inputs flow are fully utilized in a defined relationship.

Let Γ be a function which maps $\mathbf{Z}$ unto $\mathbf{r}$, defined as follows:

$$\Gamma \longrightarrow \mathbf{Z} : \mathbf{r}$$

with an input-output mapping $\mathbf{r} = \Gamma(\mathbf{Z})$ and $\mathbf{Z} = (Z_1, Z_2, ...Z_L)$. Then the solution under an ℓ_2-loss function is as follows:

$$\arg\min_{\mathbf{w},\mathbf{B}} ||\Gamma_{\mathbf{w},\mathbf{B}}(\mathbf{Z}) - \mathbf{r}||_2^2 \tag{4.8}$$

subject to a regularization penalty on the weights and offsets;
where $\mathbf{w} = (w_1, w_2, ..., w_L)$ and $\mathbf{B} = B_1, B_2, ..., B_L$ (Heaton et al., 2016).
Let $\mathbf{Z}$ be input covariate vector in an AE, and setting the target output as $r_t = \mathbf{Z}_t$. Then we obtain a static AE with two layers akin to a traditional model in the form of deep learners, defined by the following:
$h_2 = w_1\mathbf{Z} + B_1$
$\alpha_2 = \gamma_2(h_2)$
$h_3 = w_2\alpha_2 + B_2$
$\mathbf{r} = \Gamma_{\mathbf{w},\mathbf{B}}(\mathbf{Z}) = \alpha_3 = \gamma_3(h_3)$
where α_2, α_3 are activation levels, with initial setting $\alpha_1 = \mathbf{Z}$.
A two-layer deep learning model is obtained when $\{w_l\}_{l=1}^2$ are simultaneously estimated based on the training input covariates $\mathbf{Z}$.

In a dynamic single-layer AE, we need to find the weight vectors $\mathbf{w_Z}$ and $\mathbf{w_r}$, so that the state model encodes while the $\mathbf{w} = (\mathbf{w_z}, \mathbf{w_r})'$ decodes the r_t vector into its lag r_{t-1} and the current state $\mathbf{Z}_t$. Thus, a single-layer dynamic AE for a financial time series r_t can be represented in a coupled system, as follows:

$$r_t = \mathbf{w_Z}\mathbf{Z}_t + \mathbf{w_r}r_{t-1} \quad \text{where} \quad \begin{pmatrix} Z_t \\ r_{t-1} \end{pmatrix} = \mathbf{w}r_t \tag{4.9}$$

(Heaton et al., 2017)

Clearly the AE model is directly in predictive form, and hence we do not need to model the variance-covariance matrix explicitly, given the nonlinear combination of deep learners.

A stacked autoencoder (SAE) consists of multiple layers of sparse autoencoders in which the output of each layer is connected to the inputs of the successive layers. Let $w_{(m,1)}, w_{(m,2)}, B_{(m,1)}, B_{(m,1)}$ represent the parameters $w_1, w_2, \beta_1, \beta_2$ for m^{th} autoencoder. Then for a stacked autoencoder with L layers, the encoding step of the SAE is obtained by running the encoding step of each layer in forward order, defined as follows:

$$\alpha_l = \Gamma(\mathbf{Z}_{(l)}) \tag{4.10}$$

$$\mathbf{Z}_{l+1} = \mathbf{w}_{(l,1)}\alpha_l + B_{(l,1)} \tag{4.11}$$

The decoding step is obtained by running the decoding stack of each autoencoder in reverse order, defined by the following:

$$\alpha_{(L+l)} = \Gamma(\mathbf{Z}_{(L+l)}) \tag{4.12}$$

$$\mathbf{Z}_{(L+l+1)} = \mathbf{w}_{(L-l,2)}\alpha_{(L+l)} + B_{(L-l,2)}$$

where $\alpha_{(L)}$ is the activation of the deepest layer of hidden units in the network model.

4.3.5 H_2O Deep Learning

In a neural network model, the weighted combination, defined by:

$$\alpha = \sum_{l=1}^{L} w_l \mathbf{Z}_l + \mathbf{B} \tag{4.13}$$

of input signals is usually aggregated, so that an output signal, defined by:

$$\Gamma(\alpha) = \Gamma\Big(\sum_{l=1}^{L} w_l \mathbf{Z}_l + \mathbf{B}\Big) \tag{4.14}$$

can be transmitted by the connected neurons in the model; where $\Gamma(\alpha)$ is the nonlinear activation function used throughout the network; β is a bias term representing the neuron's activation threshold (Arora et al., 2015).

The multilayer feedforward neural networks consist of many layers of interconnected neuron units, starting with an input layer to match the feature space, followed by multiple layers of nonlinearity and resulting to a linear regression layer to match the output space.

The objective is to minimize the loss function for each training example j:

$$Loss\big(\mathbf{w}, \mathbf{B}|j\big) = Loss\Bigg(\big\{w_l\big\}_{l=1}^{L-1}, \big\{B_l\big\}_{l=1}^{L-1}|j\Bigg) \tag{4.15}$$

where $\mathbf{w}$ is $L-1$ weight vector connecting layer l to $l+1$ for a network of L layers; $\mathbf{B}$ is $L-1$ column vector of biases for layer $l+1$.

The H_2O follows the model of multilayer feedforward neural networks for predictive modelling. Thus the H_2O deep learning uses a purely supervised training protocol with specification of the training frame together with the tuning parameters for the regression task.

4.3.6 Long Short Term Memory

The long stort term memory (LSTM) refers to a specific form of recurrent neural network which seek to provide a solution by incorporating memory units, allwoing the network to learn when to forget previous hidden states and when to update hidden states given new information. This is done by inclusion of an input gate, a forget gate, an input modulation gate and a memory cell in addition to the hidden unit.

In traditional recurrent neural networks (RNN), the networks can learn complex temporal dynamics via a set of deep recurrence models, defined as follows:

$h_t = \gamma(\mathbf{w}_{\mathbf{Z},h}\mathbf{Z}_t + w_{h,h} + B_{\mathbf{Z}})$

$r_t = \gamma(\mathbf{w}_{h,\mathbf{z}}\mathbf{Z}_t + B_{\mathbf{Z}})$

where $\mathbf{Z}_t$ is the input vector of covariates; h_t is the hidden layer with n hidden units; r_t is the corresponding output at time t, for input sequence of length T in which the updates a computed sequentially.

The architecture for an LSTM model added a hidden state C_t and a sigmoid function $\sigma(\cdot)$ resulting to:

$$\Gamma_t = \sigma(\mathbf{w}'_\Gamma[h_{t-1}, \mathbf{Z}_t] + B_\Gamma) \tag{4.16}$$

$$I_t = \sigma(\mathbf{w}'_I[h_{t-1}, \mathbf{Z}_t] + B_I) \tag{4.17}$$

$$\hat{C}_t = tanh(\mathbf{w}'_C[h_{t-1}, \mathbf{Z}_t] + B_C) \tag{4.18}$$

$$C_t = \Gamma_t \oplus C_{t-1} + I_t \oplus \hat{C}_t \tag{4.19}$$

$$h_t = O_t \oplus tanh(C_t) \tag{4.20}$$

where $\Gamma_t \oplus C_{t-1}$ is the forget gate, which allows to throw away some data from previous cell state; $I_t \oplus \hat{C}_t$ is the input gate, which decides the values of the cell state to be updated by an input signal in the network model; $[h_{t-1}, \mathbf{Z}_t]$ is a pair-wise vector such that the new cell state is a sum of the previous cell state passing through the forget gate selected components; $O_t \oplus tanh(C_t)$ is the output gate, which returns $tanh$ applied to the hidden state with the removal of some elements (Vargas et al., 2017; Heaton et al., 2016).

The LSTM provides a mechanism for dropping irrelevant information from the previous states and adding relevant information from the current time step, improving the predictors by utilizing data from the previous to memorize volatility patterns from previous periods. Thus the designated model aimed to automate the identification of the temporal connections in the dataset at the cost of larger sets of untrained parameters.

4.3.7 Dropout Approach

The dropout approach is a selection technique to eliminate overfitting in the training process of a deep learning model. In this case, the network is defined as follows:
$r_i^l = \gamma(h_i^l)$
$h_i^l = w_i^l \mathbf{Z}^l + B_i^l$
is replaced with the dropout architecture, defined by the following:
$\mathbf{D}_i^l \sim Bernoulli(\pi)$
$\hat{r}_i^l = \mathbf{D}^l \oplus \mathbf{Z}^l$
$r_i^l = \gamma(h_i^l)$
$h_i^l = w_i^l \mathbf{Z}^l + B_i^l$
where $\mathbf{D} \oplus \mathbf{Z}$ replaces $\mathbf{Z}$ in the previous model; $\oplus$ is the element-wise product; $l = 1, 2, ..., L$; $\mathbf{D}$ is L-dimensional vector of independent Bernoulli distributed random variables with parameter π; $\mathbf{Z}$ is $L \times 1$ vector of covariates.
Using the loss function optimization concept, then we marginalize the problem over the randomness to obtain the objective function:

$$\arg\min_{\mathbf{w}} E(\mathbf{D} \sim Bernoulli(\pi)) ||\mathbf{r} - \mathbf{w}(\mathbf{D} \oplus \mathbf{Z})||_2^2 \simeq \arg\min_{\mathbf{w}} ||\mathbf{r} - \pi \mathbf{w} \mathbf{Z}||_2^2 + \pi(1-\pi)||\Sigma \mathbf{w}||_2^2 \tag{4.21}$$

which can be viewed as a Bayesian ridge regression model with a **g**-prior, where $\Sigma = (diag(\mathbf{Z}'\mathbf{Z}))^{\frac{1}{2}}$; E denotes the expectation.

4.3.8 Activation Functions

The activation functions used in this study include *Sigmoid*, *Hyperbolic Tangent*, *Maximum Output*, *Rectifier Linear Units*, *Softmax* and some combinations with the dropout approach. See (Arora et al., 2015; Candel et al., 2016; Nwankpa et al.,

2018) for detail.

The *Sigmoid* is a commonly used nonlinear function in feedforward neural net-

Table 4.1: The Activation Functions

Function	**Formula**
Sigmoid, **Sigm**	$\Gamma(\alpha) = \frac{1}{1+e^{-\alpha}}; \quad \Gamma(\cdot) \in [0,1]$
Hyperbolic Tangent, **Tanh**	$\Gamma(\alpha) = \frac{e^{\alpha}-e^{-\alpha}}{e^{\alpha}+e^{-\alpha}}; \quad \Gamma(\cdot) \in [-1,1]$
Maximum Output, **Maxout**	$\Gamma(\alpha_1, \alpha_2) = max(\alpha_1, \alpha_2); \quad \Gamma(\cdot) \in R$
Rectifier Linear Units, **ReLU**	$\Gamma(\alpha) = max(0, \alpha); \quad \Gamma(\cdot) \in R^{+}$
Softmax	$\Gamma(\alpha) = \dfrac{e^{\alpha_j}}{\sum_{j=1}^{k} e^{\alpha_j}} \in (0,1) \quad \text{for} \quad \alpha = (\alpha_1, \alpha_2, ..., \alpha_k).$

where $\alpha, \alpha_1, \alpha_2, ..., \alpha_k$ represent the weighted combinations.

work models. It is a monotonically bounded continuous real input valued differentiable function with positive derivatives everywhere in its domain. The $Tanh$ is commonly used in multilayer neural networks, which seeks to produce smoother zero-centred output to improve the back propagation process in the model. The $MaxOut$ which is a generalization of the *rectifier linear* activation in which each neuron picks the largest output of k separate channels and each channel has its own wights and biases. The $Softmax$ takes an input vector of α real numbers and normalizes into a probability distribution consisting of α probabilities proportional to the exponentials of the input numbers. The aim of normalization is to ensure that the sum of the components of the output vector $\Gamma(\alpha)$ is equal to 1. The H_2O deep learning allows optional specification of adaptive learning rate for fast convergence, annealing and momentum. The regularization options, dropout and model averaging are special concepts introduced to prevent model overfitting.

4.3.9 Statistical and Economic Performance Evaluation

In this study, the evaluation metric employed to compare the statistical performance of the various models is the mean squared forecast error (MSFE). It is computed as

$$MSFE = \frac{1}{T}\sum_{t=1}^{T}(r_t - \hat{r}_t)^2 \tag{4.22}$$

where T denotes the out of sample periods; r_t denotes the actual at specific time point t; $\hat{r}_t$ is the forecast at specific time point t.
The smaller the $MSFE$ the better the model, and hence the model which gives the minimum $MSFE$ is the best statistically predictive model.

The cumulative return (CR) and the Sharpe ratio (SR) are the two metrics used for the economic performance evaluation in this study.

Sharpe Ratio and Cumulative Return

According to Sharpe (1994), the Sharpe ratio (also known as reward to volatility ratio) is a measure of the additional amount of return that an investor receives per unit of increase in risk. Mathematically, it is defined as follows:

$$SR_p = \frac{E(r_p) - E(r_f)}{\sqrt{Var(r_p)}}$$

where $E(r_p)$ is the average realized return of the portfolio over the out-of-sample period; $E(r_f)$ is the average risk-free treasury bill rate; $Var(r_p)$ is the variance of the portfolio over the out-of-sample period. In this study, we use the SR which standardizes the realized returns with the risk of the portfolios.

The cumulative return (CR) of the portfolio, is computed as follows:

$$CR = \sum_{t=1}^{T} r_t \tag{4.23}$$

where r_t is the return on month t; and T is the number of months in the out-of-sample period.

4.4 Data Analysis & Discussion

The dataset and variables used in this chapter are obtained from Amit Goyal's webpage[1], each covering monthly observations from *1960M1 to 2016M12*, and the sample size is $T = 684$. The three distinct out-of-sample periods in this study consists of monthly observations from *January 1981* to *December 2016* ($T^{OOS} = 432$), *January 1991* to *December 2016* ($T^{OOS} = 312$), and *January 2001* to *December 2016* ($T^{OOS} = 192$). Each deep learning model was fitted and the parameters were estimated recursively using an expanding window of observations, with data point from the start time to the present time and obtain a one month-period-ahead forecast. The procedure is repeated until the last forecast is obtained, for the various out of sample periods. The forecast horizon is one month ahead, for all the deep learning models used in the study.

The empirical analysis in this chapter provide useful evidence of statistical predictability by the deep learning techniques, each producing smaller mean squared forecast errors (MSFEs), which implies that they possess statistical predictive power in financial stock market analysis.

In terms of statistical predictability, all the deep leaning models demonstrate useful evident of statistical predictability. Following the concept of statistical predictability, a model which gives a smaller $MSFE$ is preferable in this direction, and hence the smaller the $MSFE$, the better the predictive performance of the model. Considering the statistical predictive performance of the deep learning models in their isolated form, the *H2O* deep learning ($H2ODL$) gives the smallest $MSFE$ in each of the out-of-sample periods. Specifically, the $H2ODL$ with *Rectifier* used as the activation function produced the smallest $MSFE$, and hence, outperformed the other individual models in the chapter. Indeed, the superiority of $H2ODL$ over the other individual models is robust for both smaller and larger out-of-sample priods. Furtherance to corroborating or refuting the claims made by previous scholars about the fusion of the models in the literature, we investigate

[1]Available at: www.hec.unil.ch/agoyal/docs/PredictorData2016.xlsx

Table 4.2: Data & Description of Variables:1960M01 to 2016M12

Data & Description of Variables:1960M01 to 2016M12	
Variable	Description
Stock Index, SP_t	Is the Standard&Poor 500 U.S stock index.
Excess Stock Return, r_t	The difference between the expected return on the market portfolio (SP500) and the risk-free treasury bill rate.
Dividend Price Ratio (log), DPR_t	The dividends over the past year divided by the current stock index value.
Dividend Yield (log), DY_t	Is the difference between the log of dividends and the log of lagged prices.
Earnings Price Ratio (log), EPR_t	The earnings over the past year divided by the current stock index value.
Realized Stock Variance, RSV_t	Is the sum of squared daily returns on the $S\&P500$ index within one month.
Book to Market Value, BMV_t	Is the ratio of book value to market value for the Dow Jones Industrial Average.
Net Equity Expansion, NEE_t	Is the ratio of 12-month moving sums of net issues by New York Stock Exchange (NYSE) listed stocks to total end of year market capitalization of the NYSE stocks.
Treasury Bill Rate, TBR_t	Is the interest rate on a 3-month treasury bill, secondary market.
Long Term Yield, LTY_t	Is the long term government bond yield, constant maturity.
Long Term Return, LTR_t	Is the return on long term government bonds.
Term Spread, TS_t	Is the difference between the long term yield (LTY_t) and the treasury bill rate (TBR_t).
Default Yield Spread, DYS_t	Is the difference between the BAA and AAA rated corporate bond yields.
Default Return Spread, DRS_t	Is the difference between the long term corporate bond and long term government bond returns.
Inflation, INF_t	Is computed from the consumer price index (CPI) for all urban consumers.

the statistical predictive performance of the SAE fused with $H2O$ at various activation functions, and the results are promising. The fusion of the SAE-with-$H2O$ using $Maxout$ activation function produced the smallest $MSFE$, compared to every other model, for the various out-of-sample periods. The superior performance of the SAE-with-$H2O$ over the individual deep learning models has corroborates the hybrid form, demonstrated in Armano et al. (2005) in the literature. Thus, the SAE-with-$H2O$ gives the best statistical predictive results among all the models tested, and it is robust with the *Maxout* activation function in all out-of-sample periods. In order to demonstrate further evidence of statistical predictability, the forecasts produced by each model is plotted vertically against their respective periods, comparative to the actual values to ease the illustration. Again, the graphical illustration of each predictive model is seen to be promising. The graphical representation of the actual versus the forecasts produced by the SAE-with-$H2O$ also revealed the best relationship between the forecasts and actuals than the other models. Thus, the SAE fused with $H2O$ is the best statistically predictive deep leaning model among all the models used in this chapter, owing to its smallest $MSFE$ with robustness.

Table 4.3: Statistical and Economic Performance: 1981M1 to 2016M12: T^{OOS} = 432

Model	**Package(s)**	**Usage**	**Activation Function**	**MSFE**	**CR**	**SR**
HA	-	-	-	0.00191	2.4149	0.2127
DNN	deepnet	dbn.dnn.train	Tanh	0.00212	1.9190	0.1051
SAE	deepnet, autoencoder	dnn, sae.dnn.train	Sigmoid	0.00194	3.1460	0.4208
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	Maxout	0.00073	7.3792	1.7924
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00144	5.1578	0.9737
H_2O DL	deepnet, h_2o	h_2o.deeplearning	Rectifier	0.00145	4.9854	0.9466
H_2O DL	deepnet, h_2o	h_2o.deeplearning	RectifierWithDropout	0.00174	5.0351	0.9082
H_2O DL	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00153	4.8451	0.8748
H_2O DL	deepnet, h_2o	h_2o.deeplearning	TanhWithDropout	0.00174	4.6416	0.7351
LSTM	tensorflow, keras, mxnet	lstm	Softmax	0.00192	1.6270	0.0218

Note: All computations and graphics in this study are obtained using R version 3.4.3;
MSFE = mean squared forecast error; CR = cumulative return; SR = Sharpe ratio.

Table 4.4: Statistical and Economic Performance: 1991M1 to 2016M12: T^{OOS} = 312

Model	**Package(s)**	**Usage**	**Activation Function**	**MSFE**	**CR**	**SR**
HA	-	-	-	0.00174	1.5649	0.2503
DNN	deepnet	dbn.dnn.train	Tanh	0.00178	2.1176	0.5267
SAE	deepnet, autoencoder	dnn, sae.dnn.train	Sigmoid	0.00175	1.3889	0.2545
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	Maxout	0.00107	4.2412	1.5026
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00140	3.3078	0.9511
H_2O DL	deepnet, h_2o	h_2o.deeplearning	Rectifier	0.00122	3.6606	1.2357
H_2O DL	deepnet, h_2o	h_2o.deeplearning	RectifierWithDropout	0.00158	3.0778	0.9068
H_2O DL	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00145	3.4623	1.1002
H_2O DL	deepnet, h_2o	h_2o.deeplearning	TanhWithDropout	0.00161	2.7019	0.6455
LSTM	tensorflow, keras, mxnet	lstm	Softmax	0.00177	0.7221	0.0097

Note: All computations and graphics in this study are obtained using R version 3.4.3;
MSFE = mean squared forecast error; CR = cumulative return; SR = Sharpe ratio.

Table 4.5: Statistical and Economic Performance: 2001M1 to 2016M12: T^{OOS} = 192

Model	**Package(s)**	**Usage**	**Activation Function**	**MSFE**	**CR**	**SR**
HA	-	-	-	0.00188	0.4734	0.1085
DNN	deepnet	dbn.dnn.train	Tanh	0.00216	0.2772	0.0297
SAE	deepnet, autoencoder	dnn, sae.dnn.train	Sigmoid	0.00202	0.1108	-0.0635
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	Maxout	0.00068	2.6987	1.6908
SAE-with-H_2O	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00144	1.8619	0.9707
H_2O DL	deepnet, h_2o	h_2o.deeplearning	Rectifier	0.00131	1.8474	1.0677
H_2O DL	deepnet, h_2o	h_2o.deeplearning	RectifierWithDropout	0.00165	1.9337	1.0244
H_2O DL	deepnet, h_2o	h_2o.deeplearning	MaxoutWithDropout	0.00144	1.8836	0.9936
H_2O DL	deepnet, h_2o	h_2o.deeplearning	TanhWithDropout	0.00167	1.3846	0.6208
LSTM	tensorflow, keras, mxnet	lstm	Softmax	0.00191	-0.3624	-0.2909

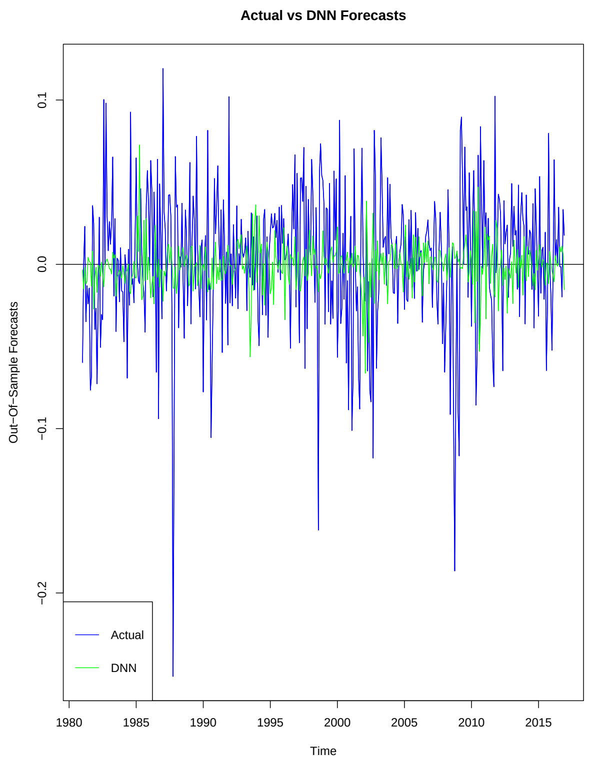
Note: All computations and graphics in this study are obtained using R version 3.4.3;
MSFE = mean squared forecast error; CR = cumulative return; SR = Sharpe ratio.

One notable approach for determining the statistical and economic predictive power of a resulting model is to compare the performance with the performance of the benchmark historical average (HA). It is imperative to note that the HA out-of-sample forecasts are obtained by recursive window forecasting approach. Following the benchmark approach, any deep learning model which yields $MSFE$ smaller than the $MSFE$ from the HA is said to beat the benchmark HA in terms of statistically predictive perspective. From the empirical findings, the $H2ODL$ applied with different activation functions have all consistently beat the HA, owing to smaller $MSFE$ across all out-of-sample periods. The SAE-with-$H2O$ using the *Maxout* and $MaxoutWithDropout$ respectively, have consistently beat the HA in all out-of-saple periods. However, the $LSTM$ and DNN do not outperform the benchmark HA. Thus, the $H2ODL$ and SAE-with-$H2O$ appeared to provide less volatility in future portfolio investment than the benchmark HA in this perspective.

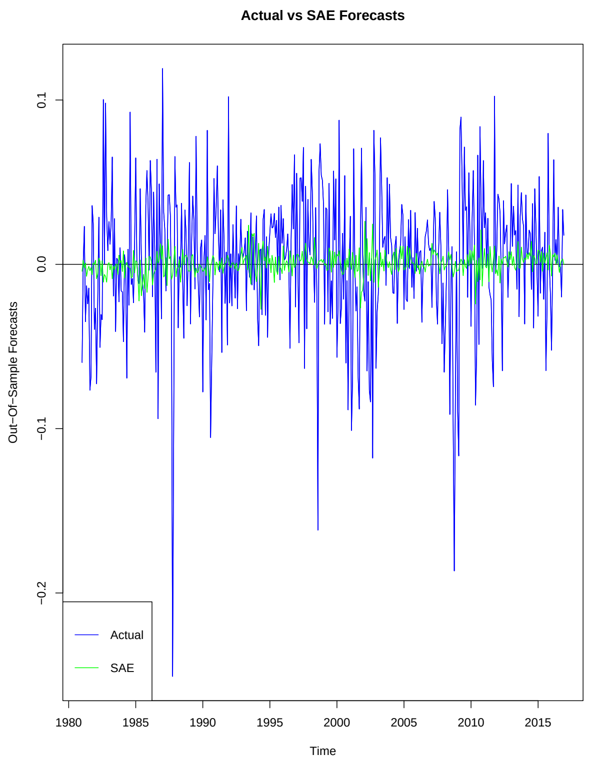
The economic performance evaluation results produced by the various deep learning models generally appeared to be promising. Most of the deep learning models produced positive cumulative return (CR) and a corresponding positive Sharpe ratio (SR), which signifies future investment gains at lower volatility. Interestingly, the good performing deep learning models that provide evidence of statistical predictability, are economically significant in the study. Contrary to the statistically predictive analysis, in which the $H2ODL$ using $Rectifier$ activation function consistently producing the smallest $MSFE$, does not consistently produce the highest CR and SR. In the economic performance evaluation, the $H2ODL$ using $Rectifier$ activation function does not consistently produced highest CR and SR across all the out-of-sample periods, in the isolated form. The $H2ODL$ using the $RectifierWithDropout$ activation function produced higher CR and SR respectively, than the $H2ODL$ with $TanhWithDropout$ activation function, in two disticnt out-of-sample periods. Notwithstanding, the $H2ODL$ appeared to produce the best economic significant results among all the deep learning models tested in their isolated form. Again, the fusion results for the economic performance evaluation reveals that the SAE-with-$H2O$ using the *Maxout* activation function generally yields the highest CR and SR respectively, among all the deep learning models, in the distinct out-of-sample periods. It is worth noting that the higher the CR, the higher the SR and the lower the $MSFE$, which concords with a rule of thumb in an ideal market situation. Thus, the SAE-with-$H2O$ using the *Maxout* activation function consistently yields the best economically significant results among all the deep learning models tested, across all out-of-sample periods in this direction.

In terms of economic performance, the $H2ODL$ and the SAE-with-$H2O$ using the various activation functions, have all produced CR and SR which are by far higher than those from the HA, and they are robust across all out-of-sample periods. Contrary to the statistcally predictive results, the HA do not consistently beat the DNN and SAE economically, across the various out-of-sample periods. In some cases, the SAE and DNN economically beat the HA. The $LSTM$ which was shown to be promising for classification task in the literature, also demonstrate evidence of statistical and economic predictability for the higher out-of-sample periods, but do not beat the benchmark HA, in regression perpective. The outperformance of $H2ODL$ and SAE-with-$H2O$ over the benchmark HA, seems to provide better strategy on future protfolio investments with less volatility than the old-fashioned HA method.

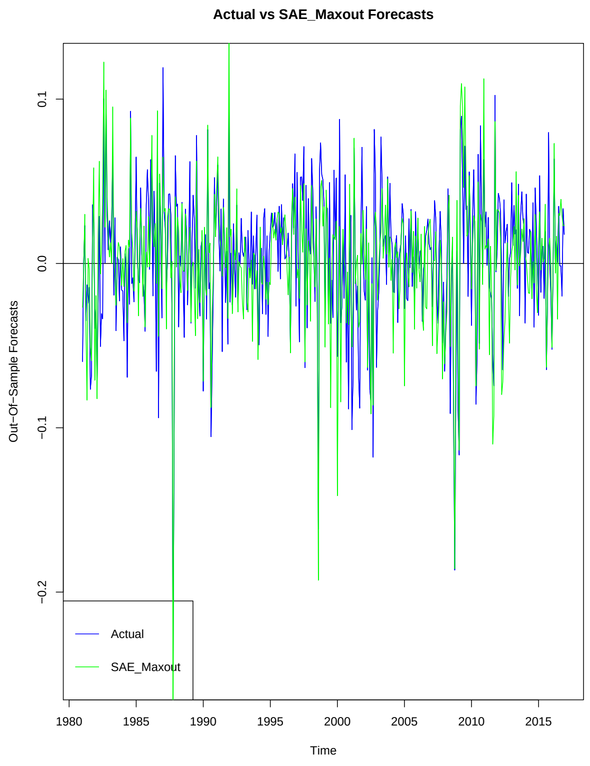
Indeed, the introduction of $H2O$ in equity premium data in this chapter has demonstrated useful evidence of both statistical and economic predictability, and the fusion with any model suggestively butress the predictive performance. Overall, the SAE-with-$H2O$ using *Maxout* activation function consistently produced the best statistically predictive and economically significant results among all the deep learning models tested in the various out-of-sample periods. Therefore it is imperative to introduce $H2O$ and its fusion or hybridized form in deep learning techniques when forecasting financial stock market data in order to improve the



(a) DNN

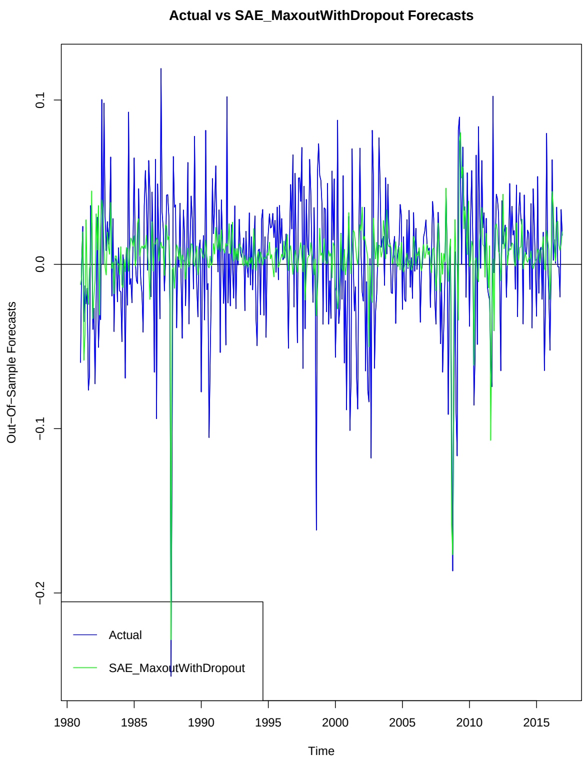


(b) SAE

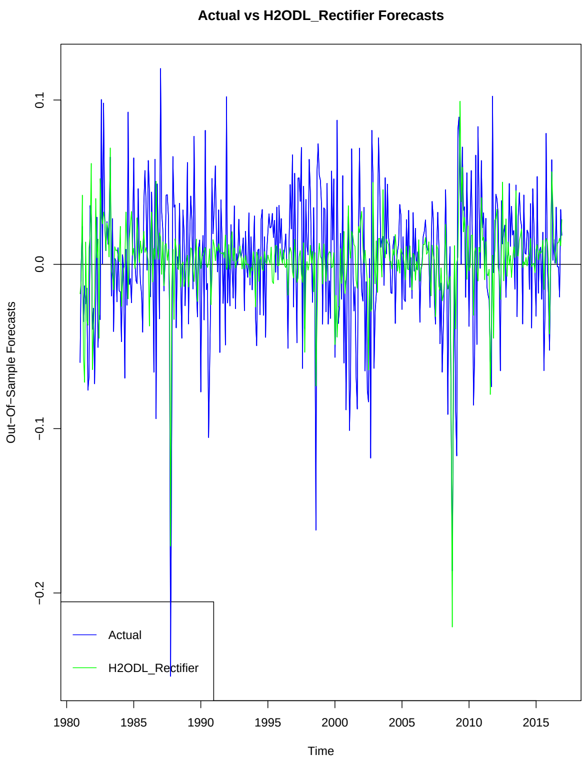


(c) SAE-Maxout

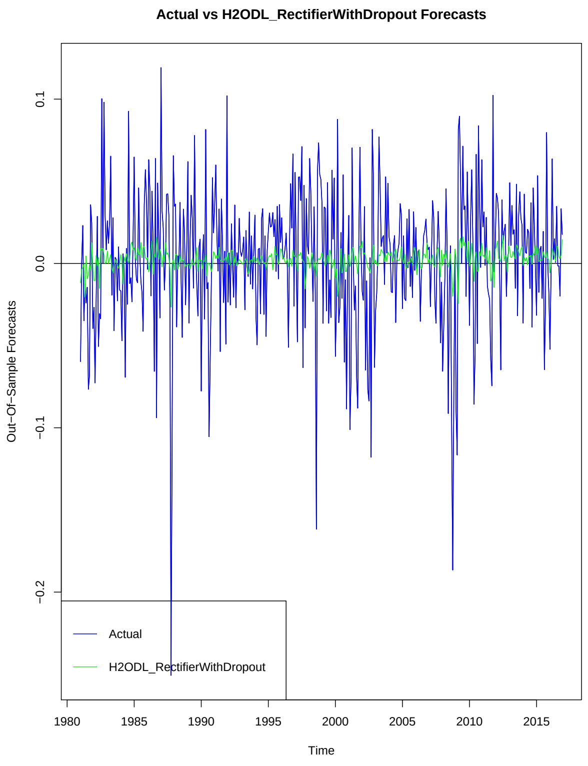
Figure 4.1: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1981 to December 2016



(a) SAE-MaxoutWithDropout

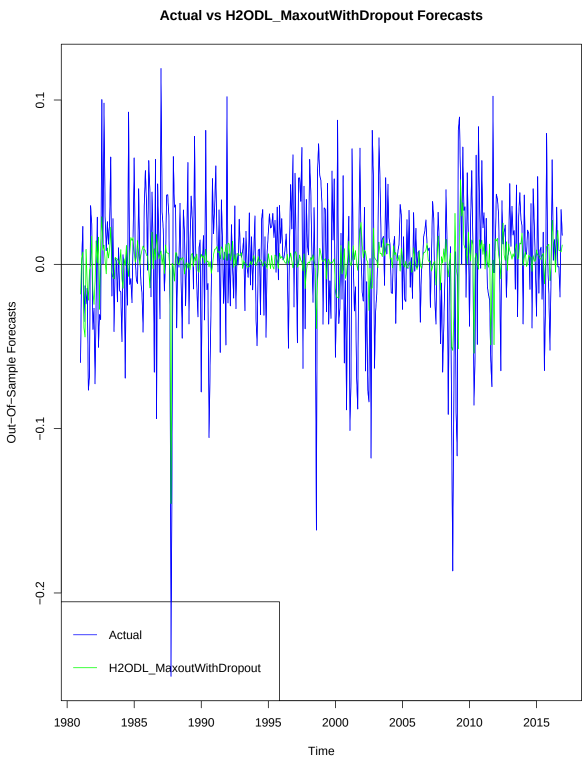


(b) H2ODL-Rectifier

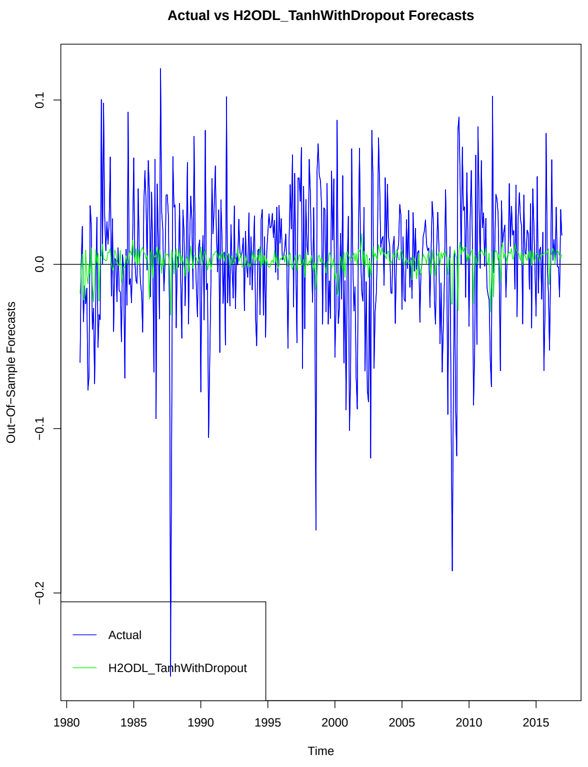


(c) H2ODL-RectifierWithDropout

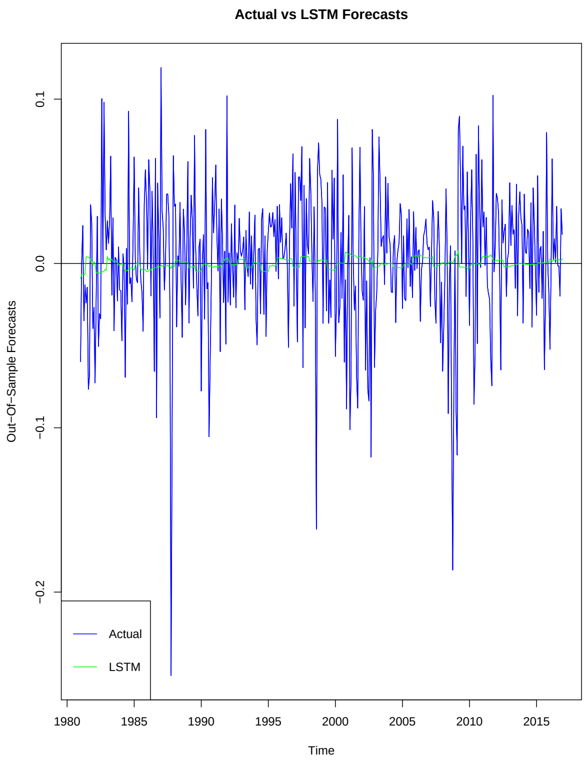
Figure 4.2: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1981 to December 2016 (continued)



(a) H2ODL-MaxoutWithDropout

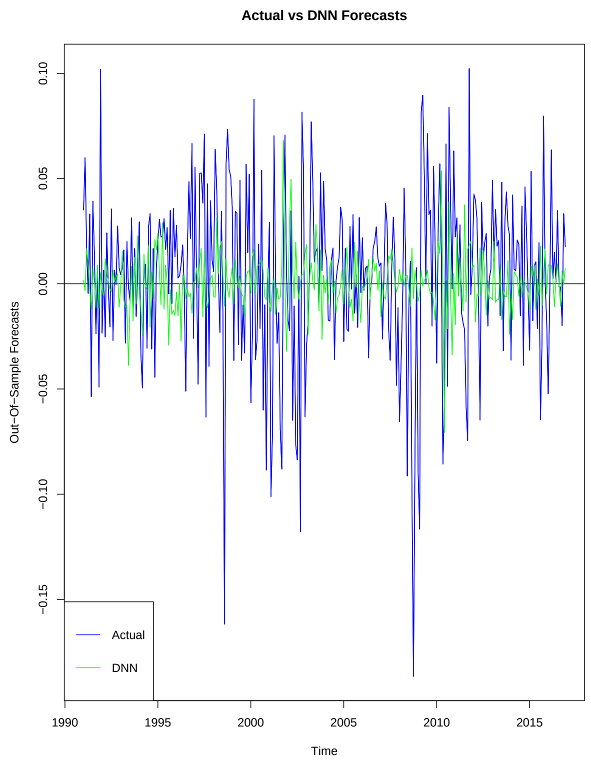


(b) H2ODL-TanhWithDropout

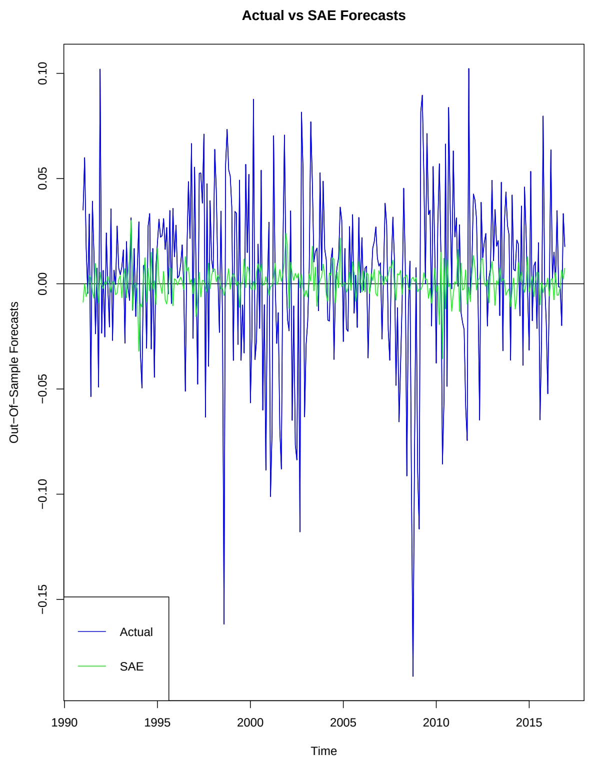


(c) LSTM

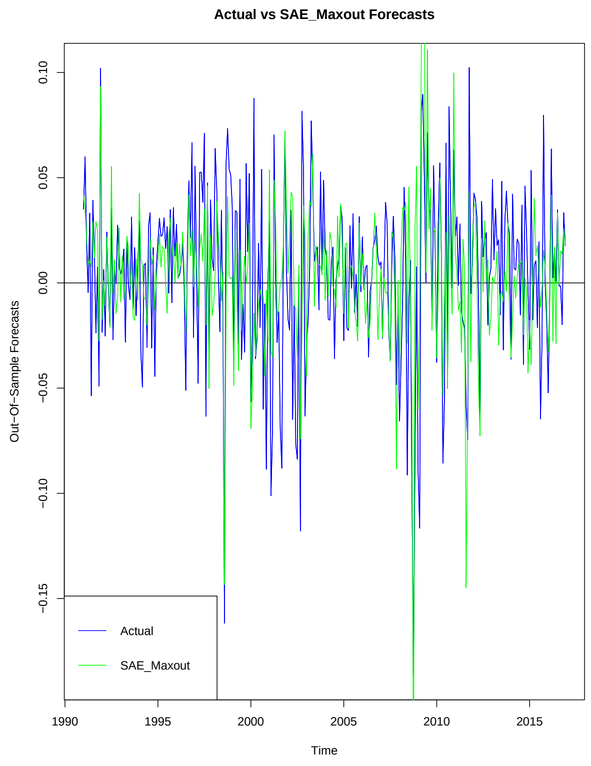
Figure 4.3: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1981 to December 2016 (continued)



(a) DNN

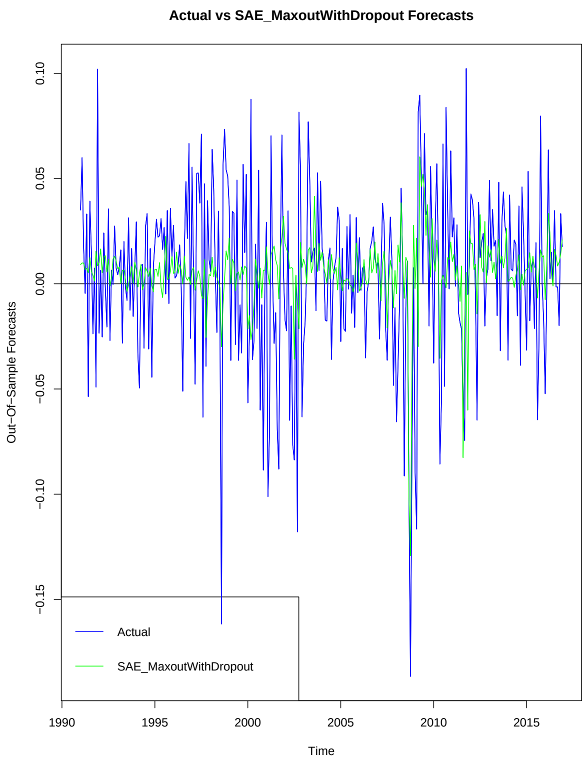


(b) SAE

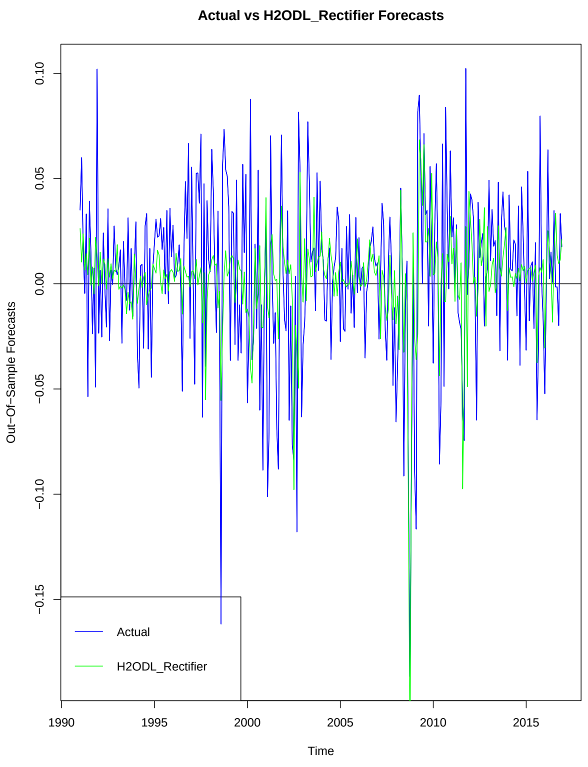


(c) SAE-Maxout

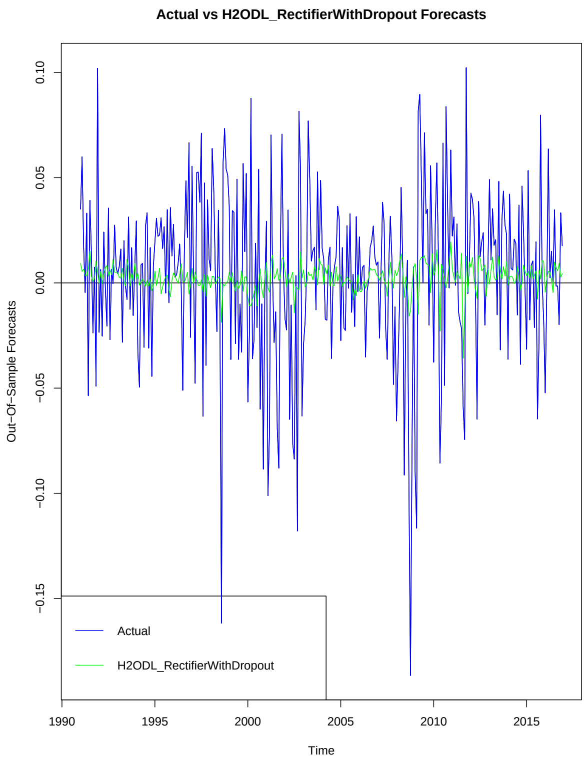
Figure 4.4: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1991 to December 2016



(a) SAE-MaxoutWithDropout

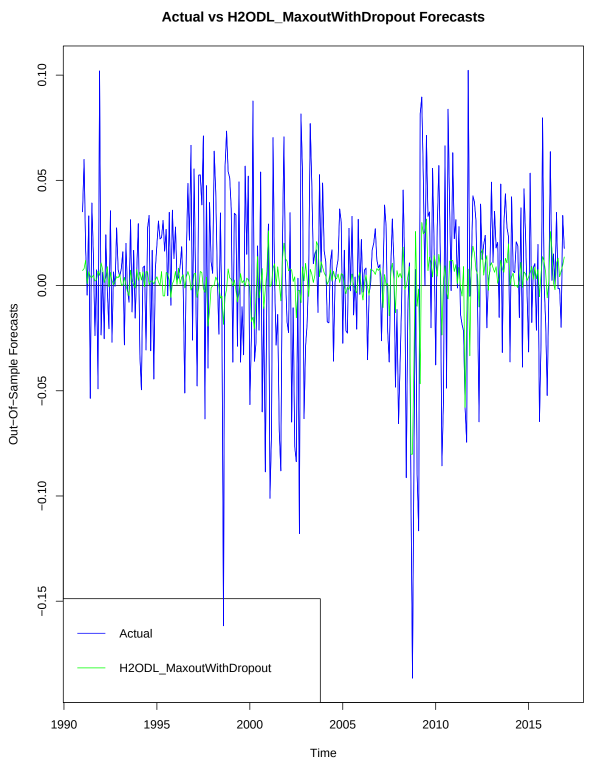


(b) H2ODL-Rectifier

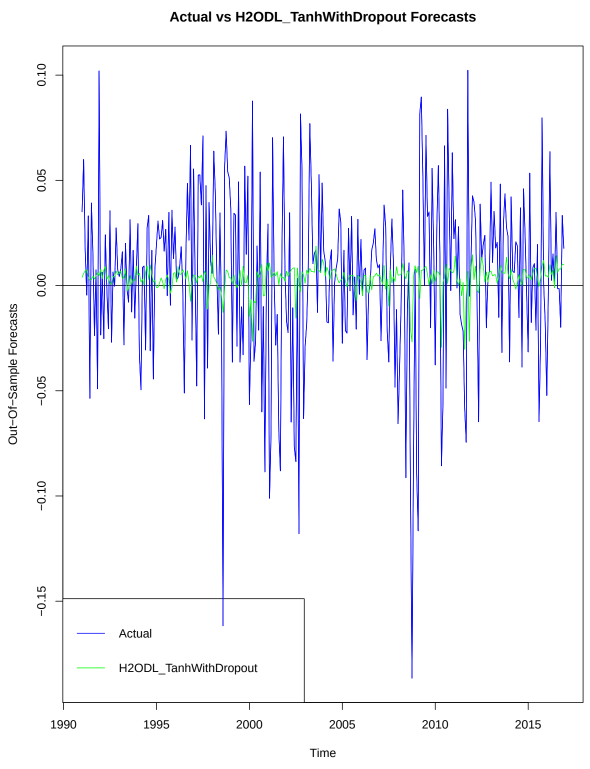


(c) H2ODL-RectifierWithDropout

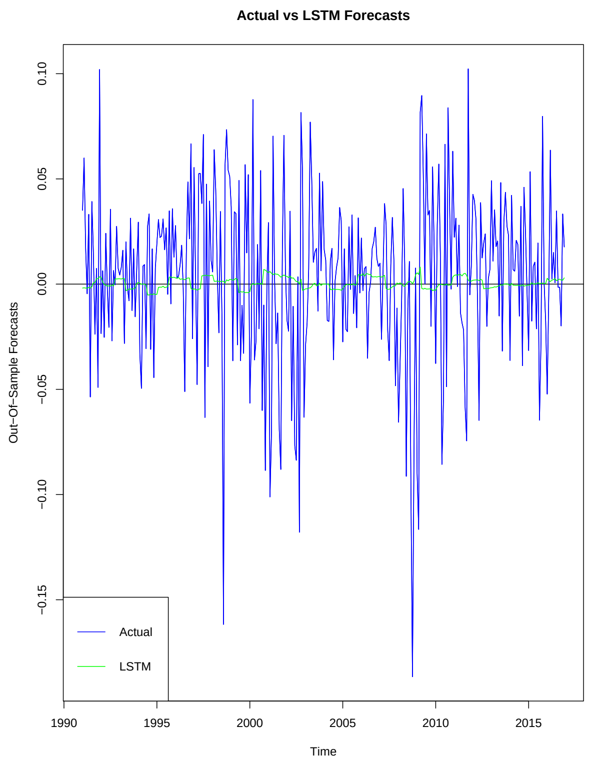
Figure 4.5: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1991 to December 2016 (continued)



(a) H2ODL$_{MaxoutWithDropout}$

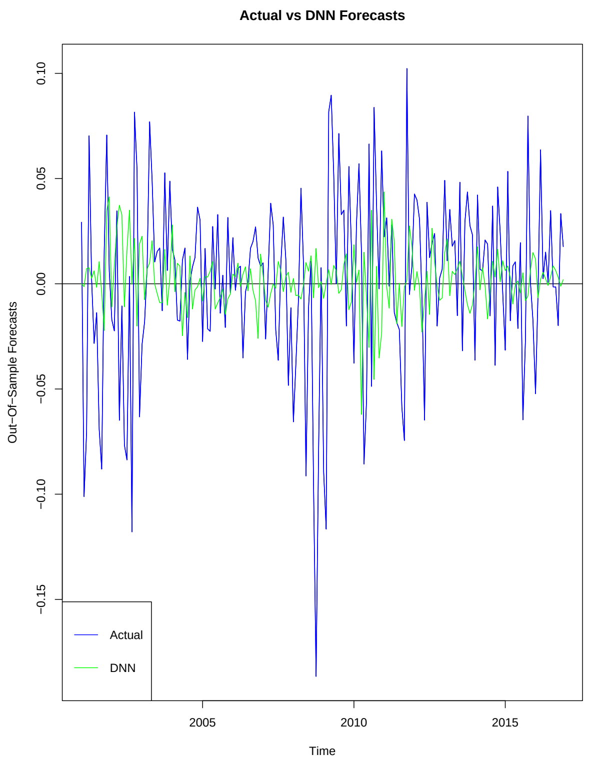


(b) H2ODL$_{TanhWithDropout}$

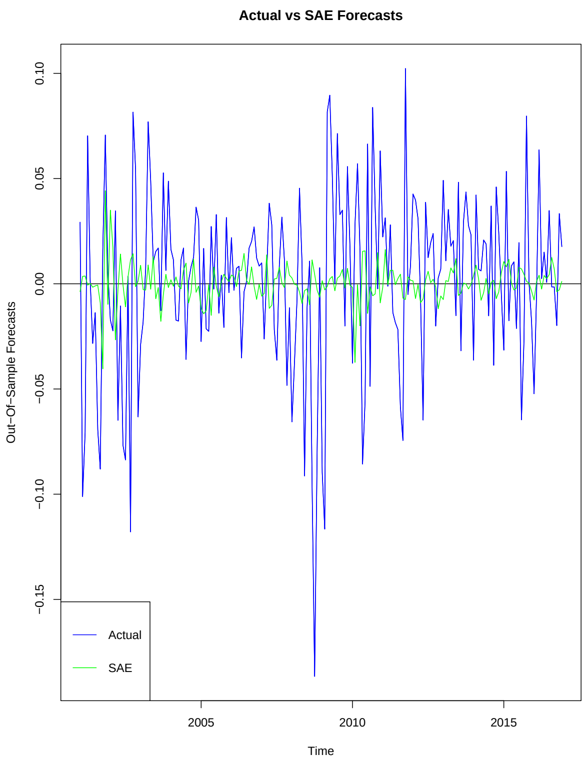


(c) LSTM

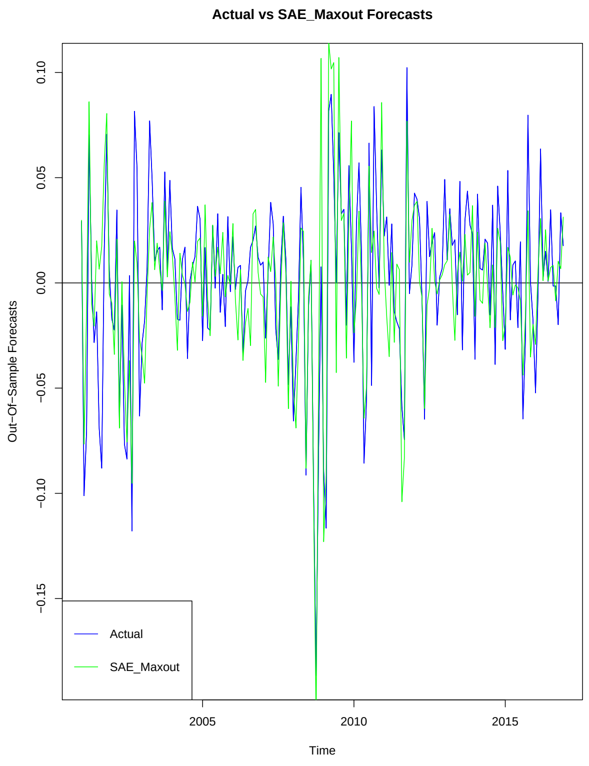
Figure 4.6: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 1991 to December 2016 (continued)



(a) DNN

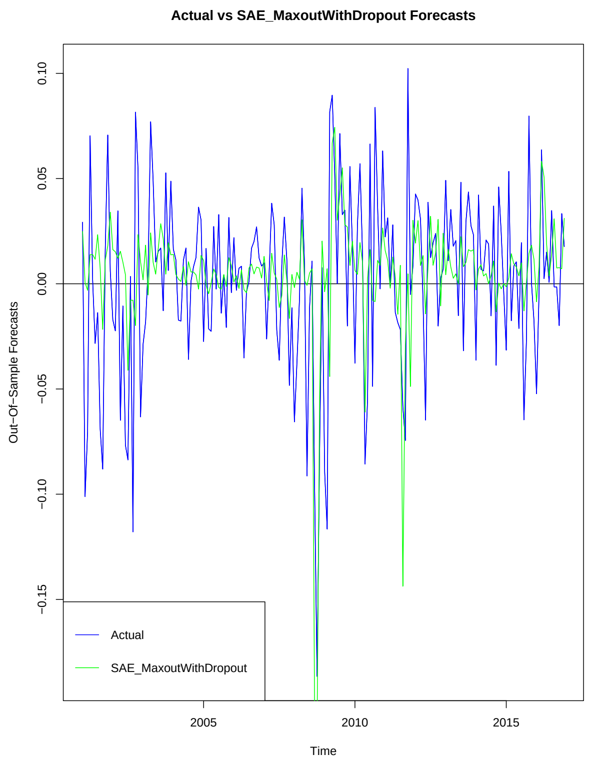


(b) SAE

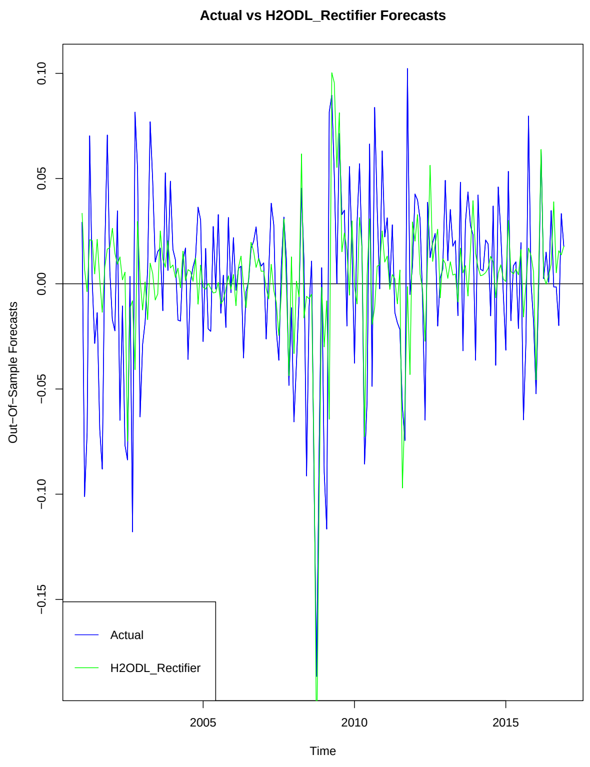


(c) SAE-Maxout

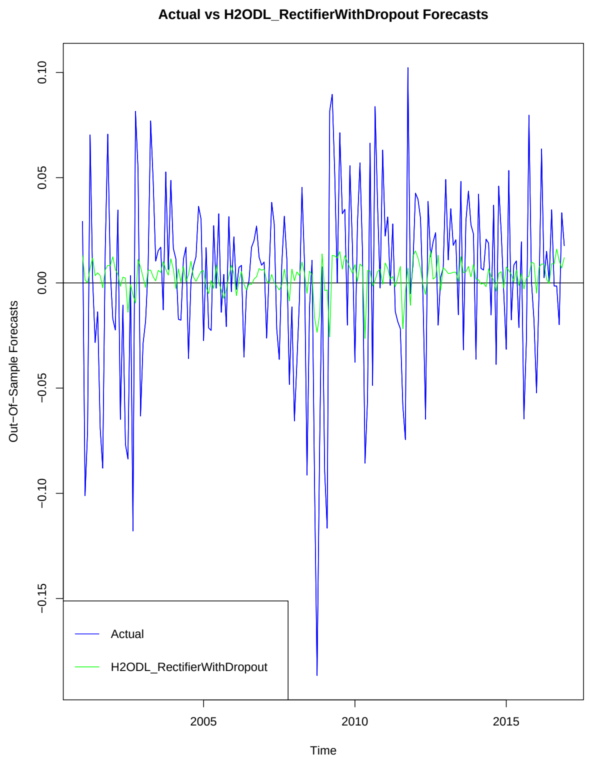
Figure 4.7: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 2001 to December 2016



(a) SAE-MaxoutWithDropout

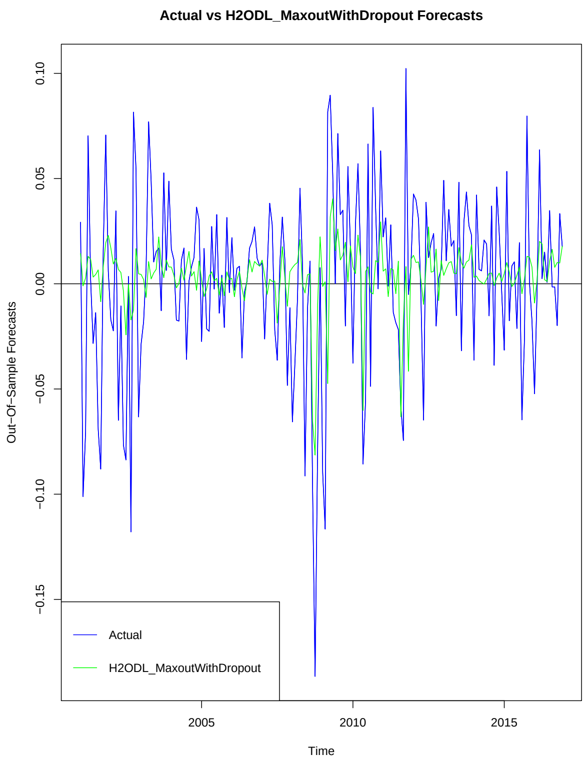


(b) H2ODL-Rectifier

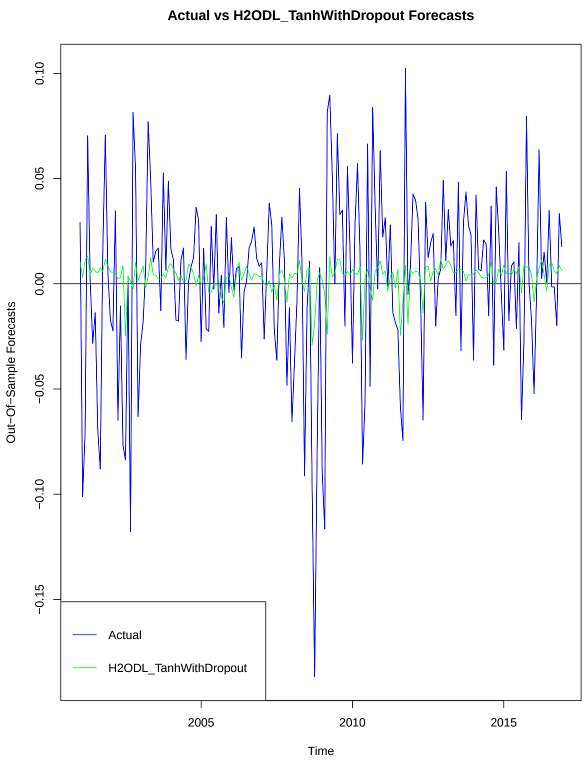


(c) H2ODL-RectifierWithDropout

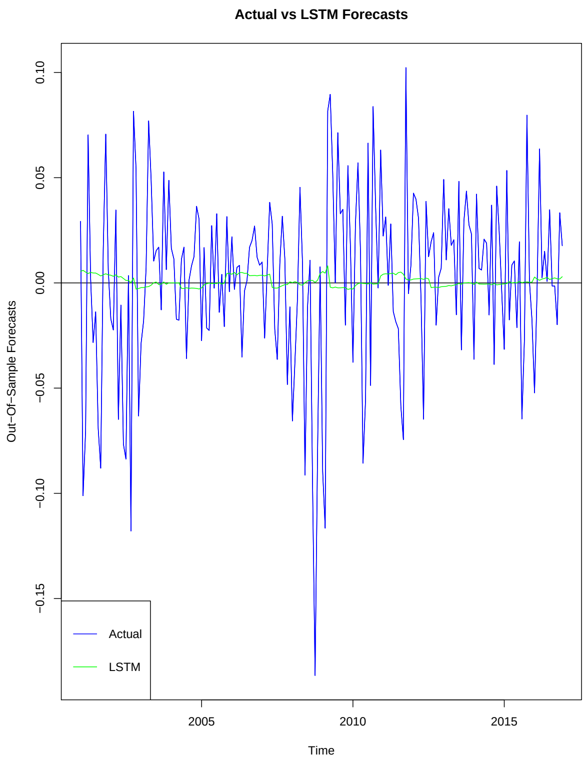
Figure 4.8: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 2001 to December 2016 (continued)



(a) H2ODL-MaxoutWithDropout



(b) H2ODL-TanhWithDropout



(c) LSTM

Figure 4.9: Out-of-Sample U.S Monthly Equity Premium Forecasts by Deep Learning Models: January 2001 to December 2016 (continued)

statistically predictive task of the resulting model(s), with economic significance.

4.5 Conclusion

Deep learning techniques are proven to be useful in many fields of study including in medicine for gene expression data, in linguistics for speech recognition and in physics for digital image classification; but are rarely used in finance, and hence the quest for the feasibility with statistical predictability and economic significance. The objective was to investigate the predictive performance of deep learning techniques on monthly financial stock market data, which has been achieved with feasibility and promising empirical results. This chapter investigates the statistically predictive power and economic significance of financial stock market data by deep learning techniques. In particular, we use the equity risk premium, also known as *excess stock market return* as the response variable and the other variables as the predictors.

Interestingly, most of the deep learning techniques provide useful evidence of statistical predictability and economic significance. The empirical findings in this chapter reveals that the $H2ODL$ and the SAE-with-$H2O$ using various activation functions have consistently outperformed the benchmark HA both statistically and economically, and they are robust across all out-of-sample periods. It is worth noting that the introduction of fusion, in which the SAE is fused with $H2O$ appeared to produce better statistically predictive results with economic significance, than the results obtained from their isolated forms. The deep learning forecasting models that demonstrates evidence of statistical predictability, which beat the benchmark HA, also demonstrates a corresponding evidence of economic significance.

Overall, the empirical analysis in this chapter revealed that the SAE-with-$H2O$ using *Maxout* activation function produced the best statistically predictive and economically significant results with robustness across all the out-of-sample periods. The introduction of fusion in this study has contributed immensely to boost the statistical predictive task of the SAE model, better than its isolated form. Thus, the deep learning techniques seek to intensify investors target by providing remedy to curb the identifiable research ills in prospective profit driven portfolio investment at minimal risk.

[2]

[2]This chapter has been submitted for publication, and it is currently on peer review, as follows:
Iworiso, J. & Vrontos, S. (2019). On the Predictability of Equity Premium Using Deep Learning Techniques. International Journal of Forecasting. Unpublished (peer review in progress).

Chapter 5

Summary, Conclusion and Further Research

5.1 Summary

The objective of this thesis was to explore sophisticated machine learning and deep learning techniques to model financial stock market data in order to make predictions and evaluate their performances both statistically and economically with robustness, and to demonstrate superior outperformance of the proposed methodologies over the benchmark approaches used in the existing literature.

Chapter two applies a plethora of machine learning techniques to forecast the direction of the U.S. equity premium. The techniques include benchmark binary probit models, penalized binary probit models, classification and regression trees (CART), discriminant analysis classifiers, Bayesian classifiers and neural networks. The study begins with replication of existing methods as shown in chapter two, specifically the static and dynamic binary probit models for directional forecasting proposed by Nyberg (2011), and the empirical results corroborate the results shown in the paper, confirming the weak predictive power of the models. The empirical analysis reveals that the sophisticated machine learning techniques significantly outperformed the benchmark binary probit forecasting models, both statistically and economically. Overall, the discriminant analysis classifiers are ranked first among all the models tested. Specifically, the high dimensional discriminant analysis (HDDA) classifier ranks first in terms of statistical performance, while the quadratic discriminant analysis (QDA) classifier ranks first in economic performance. The kNN, Bayesian classifiers, prominent CART and the penalized likelihood binary probit models (Least Absolute Shrinkage and Selection Operator, Ridge, Elastic Net) also outperformed the benchmark binary probit models, providing significant alternatives to portfolio managers. The proposed machine learning techniques provide better investment alternatives to portfolio managers than the buy and hold ($B\&H$) trading strategy used as a decision rule to time the market.

Chapter three focuses mainly on the application of regression training (RT) techniques to forecast the U.S. monthly equity premium out-of-sample recursively with expanding window method. It employed a broad categories of regression models, which includes, the kitchen sink linear model, partial least squares regression, kernel-based regularized least squares, support vector regression, relevance vector regression, regularized regression, components regression, Gaussian processes regression, regression splines, rule-based regression, nearest neighbour, projection pursuit, and neural networks. Interestingly, the RT models demonstrate significant evidence of equity premium predictability both statistically and economically relative to the benchmark historical average, delivering significant utility gains (UGs). The empirical findings revealed that the RT models significantly beat the benchmark historical average. Overall, the partial least squares regression with wide kernel ($WideKernelPLS$) produced the best result in terms of statistical predictability while the the Gaussian processes regression with radial basis kernel function ($GPR\ with\ RBF$) produced the highest average utility (AU) with a corresponding highest utility gain (UG), indicating the best economic significant result, among all the RT models. The results showed that the collective variables provide statistically and economically useful forecasts of the U.S equity premium for investors in real time setting, and demonstrate significant evidence of consistently beating the benchmark historical average out-of-sample. They seek to provide meaningful economic information on mean-variance portfolio investment for investors who are timing the market to earn future gains at minimal risk, compared to the other RT forecasting models. Therefore, the RT models appeared to guarantee a mean-variance investor in a real-time setting who optimally reallocates a monthly portfolio between equities and risk-free treasury bill using equity premium forecasts at minimal risk.

The feasibility and superiority of the machine learning techniques (statistically and economically) over other existing methods led to the proposal of more sophisticated techniques known as deep learning techniques for further investigation on monthly financial stock market data, which has been achieved with feasibility and promising empirical results. Chapter four investigates the statistical predictive power and economic significance of financial stock market data by deep learning techniques. The deep learning techniques used in the chapter include, the deep neural network (DNN), stacked autoencoder (SAE), $H2O$ deep learning $H2ODL$, long-short-term-memory (LSTM), and the fusion of some of these techniques. The activation function and dropout approach were also introduced. In particular, we use the equity risk premium as the response variable and other economic and financial variables as the predictors. The deep learning techniques used in this study provide useful evidence of statistical predictability and economic significance. Considering the statistical predictive performance of the deep learning models in their isolated form, the $H2O$ deep learning ($H2ODL$) gives the smallest mean squared forecast error ($MSFE$), with corresponding highest cumulative return (CR) and Sharpe ratio (SR) respectively, in each of the out-of-sample periods. Specifically, the $H2ODL$ with *Rectifier* used as the activation function, outperformed the other models used in the chapter. In the fusion results, the SAE-with-$H2O$ using *Maxout* activation function yields the smallest $MSFE$ with corresponding highest CR and SR in all the out-of-sample periods. It is worth noting that the higher the CR, the higher the SR, and the lower the $MSFE$ which concords with a rule of thumb. Overall, the empirical analysis in this study revealed that the *SAE*-with-*H2O* using *Maxout* activation function produced the best statistically predictive and economic significant results with robustness across all the out-of-sample periods. Thus, the deep learning techniques seek to intensify investors target by providing remedy to curb the identifiable research ills in portfolio investment at minimal risk.

5.2 Conclusion

The objective of this thesis has been achieved, as the machine learning and deep learning techniques proposed in the study have demonstrate significant evidence of outperforming the benchmark existing methods in the relevant literature, both statistically and economically. The techniques are proven to be robust both statistically and economically when forecasting the monthly equity premium out-of-sample using recursive (expanding) window method. However, reconciling the statistical and economic evidence in an attempt to guarantee the future expectation of a portfolio investor was a crucial issue in this thesis. It is worth noting, from the empirical findings in the thesis, that the superiority of a model in terms of statistical predictability among the machine learning techniques does not necessarily guarantee superiority in economic significance in this direction. Notwithstanding, the best model among the deep learning techniques was shown to be superior both statistically and economically. Thus, they provide better investment alternatives to portfolio managers who are timing the market to earn future profits at minimal risk rather than the conventional *B&H* trading strategy.

The new major contributions in this study/thesis include the following:

1. The use of sophisticated machine learning classifiers involving model training, data preprocessing, resampling/cross-validation and fine-tuning the parameters in forecasting the sign or direction of the U.S. stock market recursively using expanding window has greatly outperformed the previous

methods used in the literature, both statistically and economically, suggesting better investment alternatives to portfolio managers.

2. The best machine learning classifier in this study outperformed the best binary probit model statistically in the literature by 7.4%, see Nyberg (2011).

3. Unlike the previous studies shown in the literature in which only few performance evaluation measures were used, this study introduces a wide range of statistical performance evaluation measures (example Kappa statistic, McNemar etc.) and economic performance evaluation measures (example Sortino ratio, Maximum Draw-down, VaR, CVaR, Upside potential) to boost the empirical analysis of the study.

4. The sophisticated regression training techniques/models which incorporate all covariates as predictors in this study produced smaller mean squared errors, better Sharpe ratios and utility gains respectively than the use of individual variables in ordinary regression models with restrictions by other scholars shown in the literature. Thus the sophisticated regression training techniques appeared to provide a better investment alternative to a mean-variance portfolio investor who is timing the market to earn future profit at minimal risk/volatility, rather than the conventional techniques used in the literature.

5. Unlike some previous works reviewed in the literature, the empirical findings in this study appeared to be robust in that each model performance is considered both statistically and economically.

6. Adequate performance evaluation measures were used to investigate the deep learning techniques/models in this study, in the context of finance, compared to previous studies in the literature.

7. Comparatively, the deep learning techniques in this study appeared to produce the best model in both statistically and economically justifiable manner

when forecasting the U.S. stock market recursively with expanding window, compared to the traditional machine learning techniques and the conventional techniques in finance used in the literature.

5.3 Further Research

The following areas are left pending for further research:

- To investigate the statistical predictability and economic significance of the machine learning and deep learning models used in this thesis at different length(s) of forecast horizon, for example quarterly ($h = 3$), semi-annually ($h = 6$) and annually ($h = 12$).
- To investigate the feasibility and predictability of these sophisticated techniques in forecasting other relevant financial and economic variables such as industrial production, economic recession, economic growth, exchange rate etc.

References

Abdi, H. (2010). Partial least squares regression and projection on latent structure regression (PLS Regression). *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):97–106.

Adebiyi, A., Ayo, C., Adebiyi, M. O., and Otokiti, S. (2012). Stock price prediction using neural network with hybridized market indicators. *Journal of Emerging Trends in Computing and Information Sciences*, 3(1):1–9.

Ahn, J. J., Byun, H. W., Oh, K. J., and Kim, T. Y. (2012). Using ridge regression with genetic algorithm to enhance real estate appraisal forecasting. *Expert Systems with Applications*, 39(9):8369–8379.

Akita, R., Yoshihara, A., Matsubara, T., and Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. In *Computer and Information Science (ICIS), 2016 IEEE/ACIS 15th International Conference on*, pages 1–6. IEEE.

Alfaro, E., Gamez, M., Garcia, N., et al. (2013). Adabag: An R package for classification with boosting and bagging. *Journal of Statistical Software*, 54(2):1–35.

Alfons, A., Croux, C., and Gelper, S. (2016). Robust groupwise least angle regression. *Computational Statistics & Data Analysis*, 93:421–435.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185.

Anatolyev, S. and Gospodinov, N. (2010). Modeling financial return dynamics via decomposition. *Journal of Business & Economic Statistics*, 28(2):232–245.

Armano, G., Marchesi, M., and Murru, A. (2005). A hybrid genetic-neural architecture for stock indexes forecasting. *Information Sciences*, 170(1):3–33.

Arora, A., Candel, A., Lanford, J., LeDell, E., and Parmar, V. (2015). Deep learning with h2o. *H2O. ai, Mountain View.*

Avdis, E. and Wachter, J. A. (2017). Maximum likelihood estimation of the equity premium. *Journal of Financial Economics*, 125(3):589–609.

Aye, G. C., Deale, F. W., and Gupta, R. (2016). Does debt ceiling and government shutdown help in forecasting the us equity risk premium? *Panoeconomicus*, 63(3):273–291.

Baetje, F. and Menkhoff, L. (2016). Equity premium prediction: Are economic and technical indicators unstable? *International Journal of Forecasting*, 32(4):1193–1207.

Bai, J. (2010). Equity premium predictions with adaptive macro indexes. *Staff Report, Federal Reserve Bank of New York, No. 475.*

Bai, J. and Ng, S. (2008). Forecasting economic time series using targeted predictors. *Journal of Econometrics*, 146(2):304–317.

Ballings, M., Van den Poel, D., Hespeels, N., and Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, 42(20):7046–7056.

Bao, W., Yue, J., and Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, 12(7):e0180944.

Baur, D. G. and Löffler, G. (2015). Predicting the equity premium with the demand for gold coins and bars. *Finance Research Letters*, 13:172–178.

Bergmeir, C., Costantini, M., and Benítez, J. M. (2014). On the usefulness of cross-validation for directional forecast evaluation. *Computational Statistics & Data Analysis*, 76:132–143.

Biau, G., Devroye, L., Dujmović, V., and Krzyżak, A. (2012). An affine invariant k-nearest neighbor regression estimate. *Journal of Multivariate Analysis*, 112:24–34.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth Annual Workshop on Computational Learning Theory*. ACM.

Boston, R. and Kuhn, M. (2000). Rules Rules Rules! Cubist Regression Models. *Boston R User Group*.

Bouveyron, C., Girard, S., and Schmid, C. (2007). High-dimensional discriminant analysis. *Communications in StatisticsTheory and Methods*, 36(14):2607–2623.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and Regression Trees*. CRC press.

Bühlmann, P., Hothorn, T., et al. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4):477–505.

Bühlmann, P. and Yu, B. (2003). Boosting with the l 2 loss: Regression and Classification. *Journal of the American Statistical Association*, 98(462):324–339.

Campbell, J. Y. (2008). Estimating the equity premium. *Canadian Journal of Economics*, 41(1):1–21.

Campbell, J. Y. and Thompson, S. B. (2005). Predicting the equity premium out of sample: can anything beat the historical average? Technical report, National Bureau of Economic Research.

Campbell, J. Y. and Thompson, S. B. (2007). Predicting excess stock returns out of sample: Can anything beat the historical average? *The Review of Financial Studies*, 21(4):1509–1531.

Candel, A., Parmar, V., LeDell, E., and Arora, A. (2016). Deep learning with h2o. *H2O. ai Inc.*

Cannon, A. J. (2011). Quantile regression neural networks: Implementation in R and application to precipitation downscaling. *Computers & Geosciences*, 37(9):1277–1284.

Caudill, M. (1989). Neural networks primer, part viii. *AI Expert*, 4(8):61–67.

Chen, J. (2016). The Chen-Tindall system and the LASSO operator: improving automatic model performance. *Occasional Paper*, 16:01.

Chen, M.-Y. (2011). Predicting corporate financial distress based on integration of decision tree classification and logistic regression. *Expert Systems with Applications*, 38(9):11261–11272.

Chen, Y. and Hao, Y. (2017). A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction. *Expert Systems with Applications*, 80:340–355.

Chevapatrakul, T. (2013). Return sign forecasts based on conditional risk: Evidence from the UK stock market index. *Journal of Banking & Finance*, 37(7):2342–2353.

Chong, E., Han, C., and Park, F. C. (2017). Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Systems with Applications*, 83:187–205.

Christoffersen, P. F. and Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science*, 52(8):1273–1287.

Chun, H. and Keleş, S. (2010). Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25.

Clemmensen, L., Hastie, T., Witten, D., and Ersbøll, B. (2011). Sparse discriminant analysis. *Technometrics*, 53(4):406–413.

Cook, E. F. and Goldman, L. (1984). Empiric comparison of multivariate analytic techniques: advantages and disadvantages of recursive partitioning analysis. *Journal of Chronic Diseases*, 37(9):721–731.

Creamer, G. G. (2009). Using random forests and logistic regression for performance prediction of Latin American ADRS and banks.

Cummins, N., Sethu, V., Epps, J., and Krajewski, J. (2015). Relevance vector machine for depression prediction. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Das, A., Abdel-Aty, M., and Pande, A. (2009). Using conditional inference forests to identify the factors affecting crash severity on arterial corridors. *Journal of Safety Research*, 40(4):317–327.

Day, M.-Y. and Lee, C.-C. (2016). Deep learning for financial sentiment analysis on finance news providers. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 1127–1134. IEEE.

de Oliveira, F. A., Nobre, C. N., and Zárate, L. E. (2013). Applying artificial neural networks to prediction of stock price and improvement of the directional prediction index–case study of PETR4, Petrobras, Brazil. *Expert Systems with Applications*, 40(18):7596–7606.

Della Corte, P., Sarno, L., and Valente, G. (2010). A century of equity premium predictability and the consumption–wealth ratio: An international perspective. *Journal of Empirical Finance*, 17(3):313–331.

Deng, Y., Bao, F., Kong, Y., Ren, Z., and Dai, Q. (2017). Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3):653–664.

Di Persio, L. and Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, 10:403–413.

Diebold, F. X. (2015). Comparing predictive accuracy, twenty years later: A personal perspective on the use and abuse of diebold–mariano tests. *Journal of Business & Economic Statistics*, 33(1):1–1.

Diebold, F. X. and Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & Economic Statistics*, 20(1):134–144.

Ding, T., Fang, V., and Zuo, D. (2013). Stock market prediction based on time series data and market sentiment. *URL http://murphy. wot. eecs. northwestern. edu/˜ pzu918/EECS349/final_dZuo_tDing_vFang. pdf*.

Dixon, M., Klabjan, D., and Bang, J. H. (2015). Implementing deep neural networks for financial market prediction on the intel xeon phi. In *Proceedings of the 8th Workshop on High Performance Computational Finance*, page 6. ACM.

Dixon, M., Klabjan, D., and Bang, J. H. (2016). Classification-based financial markets prediction using deep neural networks. *Algorithmic Finance*, (Preprint):1–11.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of Statistics*, 32(2):407–499.

Elliott, G., Gargano, A., and Timmermann, A. (2013). Complete subset regressions. *Journal of Econometrics*, 177(2):357–373.

Estrella, A. and Mishkin, F. S. (1998). Predicting US recessions: Financial variables as leading indicators. *Review of Economics and Statistics*, 80(1):45–61.

Fama, E. F. and French, K. R. (2002). The equity premium. *The Journal of Finance*, 57(2):637–659.

Feng, K.-Y., Cai, Y.-D., and Chou, K.-C. (2005). Boosting classifier for predicting protein domain structural class. *Biochemical and Biophysical Research Communications*, 334(1):213–217.

Ferwerda, J., Hainmueller, J., and Hazlett, C. (2015). KRLS: A stata package for kernel-based regularized least squares. *Journal of Statistical Software*, 55(2).

Ferwerda, J., Hainmueller, J., Hazlett, C. J., et al. (2017). Kernel-based regularized least squares in R (KRLS) and stata (KRLS). *Journal of Statistical Software*, 79(3):1–26.

Feuerriegel, S. and Fehrer, R. (2016). Improving decision analytics with deep learning: the case of financial disclosures. In *ECIS*, pages Research–in.

Fischer, T. and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *In: Thirteenth International Conference on ML*. Citeseer.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Special invited paper. additive logistic regression: A statistical view of boosting. *Annals of statistics*, pages 337–374.

Friedman, J. H. (1989). Regularized discriminant analysis. *Journal of the American Statistical Association*, 84(405):165–175.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378.

Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823.

Gamboa, J. C. B. (2017). Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*.

Goyal, A. and Welch, I. (2003). Predicting the equity premium with dividend ratios. *Management Science*, 49(5):639–654.

Goyal, A. and Welch, I. (2007). A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies*, 21(4):1455–1508.

Granger, C. W. and Pesaran, M. H. (2000). A decision theoretic approach to forecast evaluation. *Statistics and Finance: An Interface*, pages 261–278.

Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications*, 39(3):3659–3667.

Hainmueller, J. and Hazlett, C. (2013). Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach. *Political Analysis*, 22(2):143–168.

Hajek, P., Olej, V., and Myskova, R. (2014). Forecasting corporate financial performance using sentiment in annual reports for stakeholders decision-making. *Technological and Economic Development of Economy*, 20(4):721–738.

Harlow, W. V. and Rao, R. K. (1989). Asset pricing in a generalized mean-lower partial moment framework: Theory and evidence. *Journal of Financial and Quantitative Analysis*, 24(3):285–311.

Heaton, J., Polson, N., and Witte, J. H. (2016). Deep learning in finance. *arXiv preprint arXiv:1602.06561*.

Heaton, J., Polson, N., and Witte, J. H. (2017). Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12.

Hillebrand, E., Lee, T., and Medeiros, M. (2009). Bagging constrained forecasts with application to forecasting equity premium. *JSM Proceedings for Business and Economic Statistics*.

Hillebrand, E., Lee, T.-H., and Medeiros, M. C. (2014). *Bagging constrained equity premium predictors, "Essays in Nonlinear Time Series Econometrics", Festschrift in Honor of Timo Tersvirta, edited by Niels Haldrup, Mika Meitz, and Pentti Saikkonen.* Oxford University Press. Chapter 14, pages 330-356.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Hiransha, M., Gopalakrishnan, E., Menon, V. K., and Soman, K. (2018). Nse stock market prediction using deep-learning models. *Procedia Computer Science*, 132:1351–1362.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12(1):69–82.

Hofner, B., Mayr, A., Robinzonov, N., and Schmid, M. (2014). Model-based boosting in R: A hands-on tutorial using the R package mboost. *Computational Statistics*, 29(1-2):3–35.

Hong, Y. and Chung, J. (2003). Are the directions of stock price changes predictable? Statistical theory and evidence. *Manuscript, Cornell University.*

Hosein, S. and Hosein, P. (2017). Load forecasting using deep neural networks. In *Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), 2017 IEEE*, pages 1–5. IEEE.

Hothorn, T., Hornik, K., and Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674.

Hsu, N.-J., Hung, H.-L., and Chang, Y.-M. (2008). Subset selection for vector autoregressive processes using LASSO. *Computational Statistics & Data Analysis*, 52(7):3645–3657.

Hu, Z., Liu, W., Bian, J., Liu, X., and Liu, T.-Y. (2018). Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 261–269. ACM.

Huang, C.-J., Yang, D.-X., and Chuang, Y.-T. (2008). Application of wrapper approach and composite classifier to the stock trend prediction. *Expert Systems with Applications*, 34(4):2870–2878.

Huang, H., Lu, X., Liu, Y., Haaland, P., and Marron, J. (2012). R/DWD: distance-weighted discrimination for classification, visualization and batch adjustment. *Bioinformatics*, 28(8):1182–1183.

Huang, J., Kingsbury, B., Mangu, L., Padmanabhan, M., Saon, G., and Zweig, G. (2000). Performance improvement in voicemail transcription. In *Proceedings of DARPA Speech Transcription Workshop*. Citeseer.

Huang, S.-C. and Wu, T.-K. (2008). Combining wavelet-based feature extractions with relevance vector machines for stock index forecasting. *Expert Systems*, 25(2):133–149.

Imandoust, S. B. and Bolandraftar, M. (2013). Application of k-nearest neighbor (kNN) approach for predicting economic events: Theoretical background. *International Journal of Engineering Research and Applications*, 3(5):605–610.

Ince, H. and Trafalis, T. B. (2007). Kernel principal component analysis and support vector machines for stock price prediction. *IIE Transactions*, 39(6):629–637.

Inoue, A. and Kilian, L. (2008). How useful is bagging in forecasting economic time series? A case study of US consumer price inflation. *Journal of the American Statistical Association*, 103(482):511–522.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 6. Springer.

Kamarudin, M. H., Maple, C., Watson, T., and Safa, N. S. (2017). A logitboost-based algorithm for detecting known and unknown web attacks. *IEEE Access*, 5:26190–26200.

Kauppi, H. and Saikkonen, P. (2008). Predicting US recessions with dynamic binary response models. *The Review of Economics and Statistics*, 90(4):777–791.

Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., and Hussain, O. K. (2013). Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing*, 13(2):947–958.

Keating, C. and Shadwick, W. F. (2002). An introduction to omega. *AIMA Newsletter*.

Kedem, B. and Fokianos, K. (2005). *Regression models for time series analysis*, volume 488. John Wiley & Sons.

Kellard, N. M., Nankervis, J. C., and Papadimitriou, F. I. (2010). Predicting the equity premium with dividend ratios: Reconciling the evidence. *Journal of Empirical Finance*, 17(4):539–551.

Khaidem, L., Saha, S., and Dey, S. R. (2016). Predicting the direction of stock market prices using random forest. *arXiv preprint arXiv:1605.00003*.

Kim, H. H. and Swanson, N. R. (2014). Forecasting financial and macroeconomic variables using data reduction methods: New empirical evidence. *Journal of Econometrics*, 178:352–367.

Koc, E. K. and Bozdogan, H. (2015). Model selection in multivariate adaptive regression splines (mars) using information complexity as the fitness function. *Machine Learning*, 101(1-3):35–58.

Kohonen, T. (1995). Learning vector quantization. In *Self-Organizing Maps*, pages 175–189. Springer.

Kolev, G. I. and Karapandza, R. (2017). Out-of-sample equity premium predictability and sample split–invariant inference. *Journal of Banking & Finance*, 84:188–201.

Kooperberg, C. (2006). Multivariate adaptive regression splines. *Wiley StatsRef: Statistics Reference Online*.

Kraus, M. and Feuerriegel, S. (2017). Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems*, 104:38–48.

Krauss, C., Do, X. A., and Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500. *European Journal of Operational Research*, 259(2):689–702.

Kuhn, M. (2012). The caret package. *R Foundation for Statistical Computing, Vienna, Austria. URL https://cran. r-project. org/package= caret.*

Kuhn, M. (2015). A short introduction to the caret package. *R Found Stat Comput*, 1.

Kuhn, M. et al. (2008). Caret package. *Journal of Statistical Software*, 28(5):1–26.

Kuhn, M., Weston, S., Keefer, C., and Coulter, N. (2016). *Cubist Models For Regression.* R package Vignette R package.

Kumar, M. and Thenmozhi, M. (2006). Forecasting stock index movement: A comparison of support vector machines and random forest.

Kumar, N. and Andreou, A. G. (1998). Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition. *Speech* C*ommunication*, 26(4):283–297.

Lai, R. K., Fan, C.-Y., Huang, W.-H., and Chang, P.-C. (2009). Evolving and clustering fuzzy decision tree for financial time series data forecasting. *Expert Systems with Applications*, 36(2):3761–3773.

Landwehr, N., Hall, M., and Frank, E. (2005). Logistic model trees. *Machine Learning*, 59(1-2):161–205.

Längkvist, M., Karlsson, L., and Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24.

Lee, J., Jang, D., and Park, S. (2017). Deep learning-based corporate performance prediction model considering technical capability. *Sustainability*, 9(6):899.

Lee, T.-H., Tu, Y., and Ullah, A. (2015). Forecasting equity premium: global historical average versus local historical average and constraints. *Journal of Business & Economic Statistics*, 33(3):393–402.

Leitch, G. and Tanner, J. E. (1991). Economic forecast evaluation: Profits versus the conventional error measures. *The American Economic Review*, pages 580–590.

Lemmens, A. and Croux, C. (2006). Bagging and boosting classification trees to predict churn. *Journal of Marketing Research*, 43(2):276–286.

Leon, F. and Curteanu, S. (2015). Evolutionary algorithm for large margin nearest neighbour regression. In *Computational Collective Intelligence*. Springer.

Lettau, M., Ludvigson, S. C., and Wachter, J. A. (2007). The declining equity premium: What role does macroeconomic risk play? *The Review of Financial Studies*, 21(4):1653–1687.

Leung, M. T., Daouk, H., and Chen, A.-S. (2000). Forecasting stock indices: A comparison of classification and level estimation models. *International Journal of Forecasting*, 16(2):173–190.

Li, J., Bu, H., and Wu, J. (2017). Sentiment-aware stock market prediction: A deep learning method. In *Service Systems and Service Management (ICSSSM), 2017 International Conference on*, pages 1–6. IEEE.

Li, J. and Chen, W. (2014). Forecasting macroeconomic time series: LASSO-based approaches and their forecast combinations with dynamic factor models. *International Journal of Forecasting*, 30(4):996–1015.

Li, J. and Tsiakas, I. (2017). Equity premium prediction: The role of economic and statistical constraints. *Journal of Financial Markets*, 36:56–75.

Li, P. (2012). Robust logitboost and adaptive base class (ABC) logitboost. *arXiv preprint arXiv:1203.3491*.

Lin, F. Y. and McClean, S. (2001). A data mining approach to the prediction of corporate failure. *Knowledge-Based Systems*, 14(3):189–195.

Lingjaerde, O. C. and Liestøl, K. (1998). Generalized projection pursuit regression. *SIAM Journal on Scientific Computing*, 20(3):844–857.

Liu, Y., Lee, S.-M., Kwon, O.-M., and Park, J. H. (2015). New approach to stability criteria for generalized neural networks with interval time-varying delays. *Neurocomputing*, 149:1544–1551.

Lou, J., Jiang, Y., Shen, Q., Shen, Z., Wang, Z., and Wang, R. (2016). Software reliability prediction via relevance vector regression. *Neurocomputing*, 186:66–73.

Lu, C.-J., Lee, T.-S., and Chiu, C.-C. (2009). Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems*, 47(2):115–125.

Luken, K. J., Norris, R. P., and Park, L. A. (2019). Preliminary results of using k-nearest-neighbor regression to estimate the redshift of radio-selected data sets. *Publications of the Astronomical Society of the Pacific*, 131(1004):108003.

Marron, J. S., Todd, M. J., and Ahn, J. (2007). Distance-weighted discrimination. *Journal of the American Statistical Association*, 102(480):1267–1271.

MartíNez-MartíNez, J. M., Escandell-Montero, P., Soria-Olivas, E., MartíN-Guerrero, J. D., Magdalena-Benedito, R., and GóMez-Sanchis, J. (2011). Regularized extreme learning machine for regression problems. *Neurocomputing*, 74(17):3716–3721.

Matilainen, M. (2018). *On Independent Component Analysis and Supervised Dimension Reduction for Time Series.* Available at: https://www.utupub.fi/handle/10024/144382.

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis*, 52(1):374–393.

Meligkotsidou, L., Panopoulou, E., Vrontos, I. D., and Vrontos, S. D. (2014). A quantile regression approach to equity premium prediction. *Journal of Forecasting*, 33(7):558–576.

Meligkotsidou, L., Panopoulou, E., Vrontos, I. D., and Vrontos, S. D. (2019). Out-of-sample equity premium prediction: A complete subset quantile regression approach. *The European Journal of Finance*, pages 1–26.

Mor-Yosef, L. and Avron, H. (2019). Sketching for principal component regression. *SIAM Journal on Matrix Analysis and Applications*, 40(2):454–485.

Moreno, D. and Olmeda, I. (2007). Is the predictability of emerging and developed stock markets really exploitable? *European Journal of Operational Research*, 182(1):436–454.

Nalbantov, G., Bauer, R., and Sprinkhuizen-Kuyper, I. (2006). Equity style timing using support vector regressions. *Applied Financial Economics*, 16(15):1095–1111.

Nayak, R. K., Mishra, D., and Rath, A. K. (2015). A naïve SVM-kNN based stock market trend reversal analysis for indian benchmark indices. *Applied Soft Computing*, 35:670–680.

Neely, C. J., Rapach, D. E., Tu, J., and Zhou, G. (2010). Out-of-sample equity premium prediction: Fundamental vs. technical analysis. *Unpublished working paper, Washington University in St. Louis.*

Neely, C. J., Rapach, D. E., Tu, J., and Zhou, G. (2014). Forecasting the equity risk premium: the role of technical indicators. *Management Science*, 60(7):1772–1791.

Nelson, D. M., Pereira, A. C., and de Oliveira, R. A. (2017). Stock market's price movement prediction with lstm neural networks. In *Neural Networks (IJCNN), 2017 International Joint Conference on*, pages 1419–1426. IEEE.

Nicolaou, M. A., Gunes, H., and Pantic, M. (2012). Output-associative RVM regression for dimensional and continuous emotion prediction. *Image and Vision Computing*, 30(3):186–196.

Nwankpa, C., Ijomah, W., Gachagan, A., and Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378*.

Nyberg, H. (2008). Forecasting the direction of the US stock market with dynamic binary probit models. *Discussion Paper*, (27).

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models. *International Journal of Forecasting*, 27(2):561–578.

Nyberg, H. (2013). Predicting bear and bull stock markets with dynamic binary time series models. *Journal of Banking & Finance*, 37(9):3351–3363.

Nyberg, H. and Pönkä, H. (2016). International sign predictability of stock returns: The role of the United States. *Economic Modelling*, 58:323–338.

Olivieri, A. C. (2018). Principal component regression. In *Introduction to Multivariate Calibration*, pages 73–86. Springer.

Ou, P. and Wang, H. (2009). Prediction of stock market index movement by ten data mining techniques. *Modern Applied Science*, 3(12):28.

Pahwa, N. S., Khalfay, N., Soni, V., and Vora, D. (2017). Stock prediction using machine learning: A Review Paper. *International Journal of Computer Applications*, 163(5).

Park, H. and Sakaori, F. (2013). Lag weighted LASSO for time series model. *Computational Statistics*, 28(2):493–504.

Patel, J., Shah, S., Thakkar, P., and Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259–268.

Pesaran, M. H. (2015). *Time series and panel data econometrics.* Oxford University Press.

Pesaran, M. H. and Timmermann, A. (1992). A simple nonparametric test of predictive performance. *Journal of Business & Economic Statistics*, 10(4):461–465.

Pesaran, M. H. and Timmermann, A. (1995). Predictability of stock returns: Robustness and economic significance. *The Journal of Finance*, 50(4):1201–1228.

Phan, D. H. B., Sharma, S. S., and Narayan, P. K. (2015). Stock return forecasting: some new evidence. *International Review of Financial Analysis*, 40:38–51.

Plakandaras, V., Gupta, R., Gogas, P., and Papadimitriou, T. (2015). Forecasting the U.S. real house price index. *Economic Modelling*, 45:259–267.

Polk, C., Thompson, S., and Vuolteenaho, T. (2006). Cross-sectional forecasts of the equity premium. *Journal of Financial Economics*, 81(1):101–141.

Pönkä, H. (2016). Real oil prices and the international sign predictability of stock returns. *Finance Research Letters*, 17:79–87.

Pradeepkumar, D. and Ravi, V. (2017). Forecasting financial time series volatility using particle swarm optimization trained quantile regression neural network. *Applied Soft Computing*, 58:35–52.

Qi, Z., Xu, Y., Wang, L., and Song, Y. (2011). Online multiple instance boosting for object detection. *Neurocomputing*, 74(10):1769–1775.

Qiao, X. and Zhang, L. (2015). Distance-weighted support vector machine. *Stat*, 1050:8.

Quinlan, J. R. et al. (1992). Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, volume 92, pages 343–348. Singapore.

Rännar, S., Lindgren, F., Geladi, P., and Wold, S. (1994). A PLS kernel algorithm for data sets with many variables and fewer objects. part 1: Theory and algorithm. *Journal of Chemometrics*, 8(2):111–125.

Rapach, D. E., Strauss, J. K., and Zhou, G. (2007). Out-of-sample equity premium prediction: Consistently beating the historical average. *Review of Financial Studies*.

Rapach, D. E., Strauss, J. K., and Zhou, G. (2010). Out-of-sample equity premium prediction: Combination forecasts and links to the real economy. *The Review of Financial Studies*, 23(2):821–862.

Rasmussen, C. (2006). CKI *Williams* G*aussian processes for machine learning*. MIT Press Cambridge, MA, USA.

Rasmussen, C. E. (2004). Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*. Springer.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge University Press.

Rodriguez-Galiano, V. F., Ghimire, B., Rogan, J., Chica-Olmo, M., and Rigol-Sanchez, J. P. (2012). An assessment of the effectiveness of a random forest classifier for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 67:93–104.

Rosipal, R. and Trejo, L. J. (2001). Kernel partial least squares regression in reproducing kernel Hilbert space. *Journal of Machine Learning Research*, 2(Dec):97–123.

Roy, S. S., Mittal, D., Basu, A., and Abraham, A. (2015). Stock market forecasting using LASSO linear regression model. In *Afro-European Conference for Industrial Advancement*, pages 371–381. Springer.

Sagaert, Y. R., Aghezzaf, E.-H., Kourentzes, N., and Desmet, B. (2018). Tactical sales forecasting using a very large set of macroeconomic indicators. *European Journal of Operational Research*, 264(2):558–569.

Samarasinghe, S. (2016). *Neural networks for applied sciences and engineering: from fundamentals to complex pattern recognition.* Auerbach Publications.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

Schölkopf, B., Smola, A. J., Bach, F., et al. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT Press.

Sermpinis, G., Tsoukas, S., and Zhang, P. (2017). Modelling market implied ratings using LASSO variable selection techniques.

Shahpazov, V., Doukovska, L., and Karastoyanov, D. (2014). Artificial intelligence neural networks applications in forecasting financial markets and stock prices. In *Proc. of the International Symposium on Business Modeling and Software Design–BMSD*, volume 14, pages 24–26.

Sharpe, W. F. (1994). The sharpe ratio. *The Journal of Portfolio Management*, 21(1):49–58.

Shen, G., Tan, Q., Zhang, H., Zeng, P., and Xu, J. (2018). Deep learning with gated recurrent unit networks for financial sequence predictions. *Procedia computer science*, 131:895–903.

Shen, W., Wang, J., and Ma, S. (2014). Doubly regularized portfolio with risk minimization. In *AAAI*, pages 1286–1292.

Sheng, H., Xiao, J., Cheng, Y., Ni, Q., and Wang, S. (2018). Short-term solar power forecasting based on weighted gaussian process regression. *IEEE Transactions on Industrial Electronics*, 65(1):300–308.

Singh, S. (2016). *Gaussian Process Regression*. Lecture Notes from North Carolina State University.

Sirignano, J., Sadhwani, A., and Giesecke, K. (2016). Deep learning for mortgage risk. *arXiv preprint arXiv:1607.02470*.

Son, J., Jung, I., Park, K., and Han, B. (2015). Tracking-by-segmentation with online gradient boosting decision tree. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3056–3064.

Sortino, F. A. and Price, L. N. (1994). Performance measurement in a downside risk framework. *The Journal of Investing*, 3(3):59–64.

Sortino, F. A. and Van Der Meer, R. (1991). Downside risk. *Journal of Portfolio Management*, 17(4):27–31.

Specht, D. F. (1991). A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576.

Stock, J. H. and Watson, M. W. (2012). Generalized shrinkage methods for forecasting using many predictors. *Journal of Business & Economic Statistics*, 30(4):481–493.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, 9(1):307.

Su, M.-Y. (2011). Using clustering to improve the kNN-based classifiers for online anomaly network traffic identification. *Journal of Network and Computer Applications*, 34(2):722–730.

Swanson, N. R. and White, H. (1997). A model selection approach to real-time macroeconomic forecasting using linear models and artificial neural networks. *The Review of Economics and Statistics*, 79(4):540–550.

Szepannek, G., Harczos, T., Klefenz, F., and Weihs, C. (2009). Extending features for automatic speech recognition by means of auditory modelling. In *2009 17th European Signal Processing Conference*, pages 1235–1239. IEEE.

Takeuchi, L. and Lee, Y.-Y. A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In *Technical Report*. Stanford University.

Tang, Y. (2019). Independent component analysis employing exponentials of sparse antisymmetric matrices. *Neurocomputing*, 325:172–181.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1(Jun):211–244.

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., and Iosifidis, A. (2017). Forecasting stock prices from the limit order book using convolutional neural networks. In *Business Informatics (CBI), 2017 IEEE 19th Conference on*, volume 1, pages 7–12. IEEE.

Tychonoff, A. (1963). *Solution of incorrectly formulated problems and the regularization method,(Russian)* D*oklady* A*kademii* N*auk* SSSR*, Vol. 151.*

Vargas, M. R., de Lima, B. S., and Evsukoff, A. G. (2017). Deep learning for stock market prediction from financial news articles. In *Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), 2017 IEEE International Conference on*, pages 60–65. IEEE.

Wen, Z., Zhang, R., Ramamohanarao, K., and Yang, L. (2018). Scalable and fast SVM regression using modern hardware. *World Wide Web*, 21(2):261–287.

Wu, L. and Yang, Y. (2014). Nonnegative elastic net and application in index tracking. *Applied Mathematics and Computation*, 227:541–552.

Xiong, R., Nichols, E. P., and Shen, Y. (2015). Deep learning stock volatility with google domestic trends. *arXiv preprint arXiv:1512.04916*.

Yoshihara, A., Fujikawa, K., Seki, K., and Uehara, K. (2014). Predicting stock market trends by recurrent deep neural networks. In *Pacific rim international conference on artificial intelligence*, pages 759–769. Springer.

Yoshihara, A., Seki, K., and Uehara, K. (2015). Leveraging temporal properties of news events for stock market prediction. *Artificial Intelligence Research*, 5(1):103.

Zeileis, A., Hothorn, T., and Hornik, K. (2008). Model-based recursive partitioning. *Journal of Computational and Graphical Statistics*, 17(2):492–514.

Zhang, W. and Goh, A. T. (2016). Multivariate adaptive regression splines and neural network models for prediction of pile drivability. *Geoscience Frontiers*, 7(1):45–52.

Zhao, Y., Li, J., and Yu, L. (2017). A deep learning ensemble approach for crude oil price forecasting. *Energy Economics*, 66:9–16.

Zheng, Z. (2006). Boosting and bagging of neural networks with applications to financial time series. Technical report, Working paper, Department of Statistics, University of Chicago, Tech. Rep.

Zhou, J., Li, E., Wei, H., Li, C., Qiao, Q., and Armaghani, D. J. (2019). Random forests and cubist algorithms for predicting shear strengths of rockfill materials. *Applied Sciences*, 9(8):1621.

Zhou, L., Lu, D., and Fujita, H. (2015). The performance of corporate financial distress prediction models with features selection guided by domain knowledge and data mining approaches. *Knowledge-Based Systems*, 85:52–61.

Zhou, Z.-Q., Zhu, Q.-X., and Xu, Y. (2017). Time series extended finite-state machine-based relevance vector machine multi-fault prediction. *Chemical Engineering & Technology*, 40(4):639–647.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286.