Technical and Sentiment Analysis in Financial Forecasting with Genetic Programming

Eva Christodoulaki, Michael Kampouridis, Panagiotis Kanellopoulos
School of Computer Science and Electronic Engineering
University of Essex
Wivenhoe Park, United Kingdom
{ec19888, mkampo, panagiotis.kanellopoulos}@essex.ac.uk

Abstract— **Financial Forecasting is a popular and thriving research area that relies on indicators derived from technical and sentiment analysis. In this paper, we investigate the advantages that sentiment analysis indicators provide, by comparing their performance to that of technical indicators, when both are used individually as features into a genetic programming algorithm focusing on the maximization of the Sharpe ratio. Moreover, while previous sentiment analysis research has focused mostly on the titles of articles, in this paper we use the text of the articles and their summaries. Our goal is to explore further on all possible sentiment features and identify which features contribute the most. We perform experiments on 26 different datasets and show that sentiment analysis produces better, and statistically significant, average results than technical analysis in terms of Sharpe ratio and risk.**

Index Terms—**Technical Analysis, Sentiment Analysis, Genetic Programming, Financial Forecasting**

I. Introduction

Financial Forecasting is a very active research area that has attracted the interest of many research groups. We remark that the New York Stock Exchange has a $28.4 trillion market cap, the total value of all of the shares traded in its market, as of September 2021.

Research efforts are mainly based on indicators produced by technical analysis, while during the most recent decade we observe a rapid increase of studies focusing on sentiment analysis features and their results. Only in 2017 there were 28 published articles on sentiment analysis, and a further 27 articles in 2018, in contrast with one decade prior to these years, with 7 published articles in 2007, and even less articles published per year prior to 2007.

In this paper we investigate the financial benefits that sentiment analysis can offer when it is part of a trading strategy. We create trading strategies that use only sentiment analysis features and compare them with trading strategies that are only using technical analysis, which, so far, has been predominant among traders. Our goal is to demonstrate that sentiment analysis derived strategies can be as competitive as technical analysis strategies, and even outperform them.

To conduct the above comparison, we use a genetic programming (GP) algorithm to automatically generate trading strategies. We separately provide sentiment and technical analysis indicators as input, and allow the GP algorithm to evolve trading strategies. We then compare their performance in terms of three financial metrics, namely *Sharpe ratio*, *return*, and *risk*.

The rest of this paper is organized as follows: First, we discuss related work in the Literature Review (II) and we present the Background Information (III) regarding Financial Forecasting. Afterwards, we introduce the Methodology (IV) and the Experimental Setup (V). Lastly, we present the experimental results and their analysis (VI), along with the conclusion and further experiments we plan to conduct in the future (VII).

II. Literature Review

In this section we look into previous related work on Financial Forecasting, using features from technical and/or sentiment analysis, as well as previous studies that use genetic programming algorithms to perform Financial Forecasting.

A. Technical analysis

Technical analysts rely on historical prices, in the form of tendencies and charts, to estimate the future changes of the stock market. Since this is a very popular topic, with a large number of relevant research studies, in this section we limit our focus on previous work that combines technical analysis with machine learning; we follow a chronological order in this discussion.

Machine learning has provided a rich arsenal of approaches and techniques for tackling financial forecasting challenges with the help of technical analysis (TA). Such approaches include, among others, artificial neural network models, such as feedforward neural networks (FFNN), backpropagation models (BPNN) and recurrent neural networks (RNN). Artificial neural networks can be really useful in financial forecasting since these are data-driven, nonlinear and self-adaptive. Since the 1980s, a large body of relevant related work has studied these models, used them extensively and has compared them to more traditional forms of forecasting like linear models, with a more recent study being the one by Mostafa [1]. Furthermore, in 2014, Wang and Shang [2] explored the idea of least square support vector machine (LSSVM) and tested it in the estimation of the daily movement of China Security Index 300 (CSI 300). Their results demonstrated their model's superiority compared to probabilistic neural networks (PNN) and two discriminant analysis models, by outperforming them in the training and testing process. In another paper that combines machine learning with TA, Nelson et al. [3] used historical prices and technical indicators as inputs in a long short-term memory (LSTM) model to forecast the future trends of stock prices, comparing their model's performance to that of multi-layer perceptron, random forest, a pseudo-random model, as well as that of investment strategies.

Based on the conclusions in the work of Brabazon et al. [4], GP has the ability to co-evolve the solution form, along with the solution parameters. For this reason, GP can create new solutions and optimize over the solution parameters. A common use of GP in finance is to use technical indicators as inputs in the model, in order to identify the ones most correlated with the datasets and timelines they wish to predict.

One of the first papers to dive into the use of genetic programming for financial forecasting was by Li and Tsang [5], where their algorithm was able to outperform commonly used, non-adaptive, individual technical rules in prediction accuracy and average annualized rate of return. Similar findings and techniques can be found in more recent papers, (e.g., see [6, 7, 8]). Yang et al. [9] created sentiment feedback based strategies that were able to outperform those strategies based on technical indicators. An inspiring work is that by Berutich et al. [10], who introduced a robust GP approach to evolve trading strategies, resulting to solutions able to endure extreme market conditions.

B. *Sentiment analysis*

The stock market movement can be, also, influenced by macro-economic factors, global events, and human behavior. Hence, estimating the stock market can be quite challenging. Previous related work has studied the importance of events and news on predicting the stock market, by combining neural networks with news and events happening in a local and global spectrum. In the following, we discuss some notable publications regarding sentiment analysis (SA) in order of publication date.

One of the most influential and earliest papers to dive into the idea of sentiment analysis is by Kohara et al. [11] who investigated how to increase the predictive power of multivariate models for financial forecasting with prior knowledge from newspaper headlines and neural networks. Their experimental results demonstrated a decrease in the prediction error and, at the same time, increased profits. Another key publication is by Xie et al. [12] who explored the idea of using semantic frame parsers in order to generalize from sentences to scenarios. This way, they were able to detect the opinion of the people towards the company from the sentiment, either positive or negative. They used support vector machines (SVM) with tree kernels as predictive models and demonstrated that this approach provides better results than other techniques, such as the Bag of Words model. In addition, their method eases the human analysis for understanding the relation of a company's market value and its activities.

Furthermore, Ding et al. [13] produced a model from event-driven stock market prediction. They extracted events from news and used a deep convolutional neural network (CNN) to model the short-term and long-term influences on the prices' movements. Through their study, they were able to show that their model performs better than the baseline models they used, a feedforward neural network, as well as models studied in previous related work. More recently, Day and Lee [14] expanded their study by considering the source where the sentiment comes from; this forms an important aspect into understanding the quality of the news and their impact in the stock market. In their research, they used news articles, an approach we also follow in our work, and showed the difference between the news articles' source and their different characteristics. For example, the review's team knowledge and specialization on the topic, the different writing and wording style of their journalists, and the sensitivity to market trends.

From the above, we can see that both SA and TA have been applied with a number of different machine learning algorithms. In addition, genetic programming has been often used in the TA literature, producing quite successful results; however, GP has not yet been applied for financial forecasting with SA features. This type of study could generate financially profitable results and it seems to be worth searching into, since GP has many advantages in terms of effective global search, by combining good exploration and exploitation and producing white-box models. Thus, our aim is in this paper to use a GP to investigate the SA performance and also compare it to the TA performance.

III. Background Information

A. *Technical analysis*

Technical analysis is the art of analyzing statistical indicators especially created to estimate the stock market. These indicators are defined based of past prices, momentum, volume and volatility. The aim is to exploit such indicators and recognize various trends and patterns in the stock market, in order to estimate future prices. TA has been used extensively for many years, as researchers and technical analysts rely on price data to produce relevant indicators and charts to understand the state of a company and the market better.

In our research, we use 6 different indicators in 2 different time periods, lasting for 5 and 10 days, respectively. The set of indicators includes the *Moving Average*, the *Momentum*, the *Rate of Change (ROC)*, the *Williams %R*, the *Midprice* and the *Volatility*; a total of 12 features. The definition of these 6 indicators appears below. Let $n \in \{5, 10\}$ be the size of the lookup window. We use P_i to denote the adjusted closing price at the i day of this period, with the convention that the most recent adjusted closing price in the look back period is P_n, the first adjusted closing price in the same period is P_1, and the last date of the previous set of prices is P_0, which is needed to find the price change for the *Volatility*. We denote by Close the most recent closing price, and by H_h and L_l the highest high and the lowest low price over all days in the lookup window. Finally, we denote by Var the sample variance over a dataset.

$$\mathsf{MovingAverage} = \frac{\sum_{i=1}^{n} P_i}{n}$$

$$\mathsf{Momentum} = P_n - P_1$$

$$\mathsf{ROC} = \left(\frac{P_n}{P_1} - 1\right) \cdot 100$$

$$\mathsf{Williams\%R} = -100 \cdot \frac{H_h - \mathsf{Close}}{H_h - L_l}$$

$$\mathsf{Volatility} = \sqrt{\mathsf{Var}\left(\left\{\frac{P_i}{P_{i-1}} - 1\right\}_{i \in \{1,\ldots,n\}}\right)}$$

$$\mathsf{Midprice} = \frac{H_h - L_l}{2}$$

$\mathsf{MovingAverage}$ is used to smooth the data and helps eliminate noise and identify trends. $\mathsf{Momentum}$ and ROC are indicators showing the difference between the most recent adjusted closing price and the one n days ago; they differ in that ROC normalizes the price. William's $\%$R is an indicator that takes values between 0 and 100 and measures overbought and oversold levels. Historical $\mathsf{volatility}$ measures past performance and is a statistical measure of the dispersion of returns over a given period of time. The higher the historical volatility value, the riskier the security is. The last indicator, $\mathsf{Midprice}$, returns the midpoint value from two different input fields.

B. Sentiment analysis

Sentiment analysis is one of the most recent additions in the financial forecasting research arsenal. It is the process of understanding the meaning behind a sentence, an article or an online comment, and extracting useful information. It has become a common tool for financial forecasting as the ability of computers to analyze texts has increased. The motivation for combining sentiment analysis with financial forecasting stems from the logic that it is events that are the driving force behind stock movement, as the market will follow the sentiment of the external information and everything that can influence people's decisions. Thus, knowing what may influence investors, and to which extent, is a significant asset in financial forecasting. Such events are classified as positive, negative or neutral when applying sentiment analysis.

One of the most popular algorithms for text classification is CNNs, but their use requires a large amount of already classified data (split into positive and negative), in order to produce high accuracy classifications. In this research, lacking the large amount of classified data, we resorted to specialized sentiment analysis programs used in the relevant literature, i.e., TextBlob [15], SentiWordNet [16] and AFINN sentiment [17].

TextBlob is a Python library and offers a simple API to access its methods and perform basic Natural Language Processing tasks. It calculates the polarity and subjectivity by looking at the 'intensity' of the word. Intensity determines if a word modifies the next word, as in 'very good'. *WordNet* is a lexical taxonomy of the English language, in which *SentiWordNet* is based upon. SentiWordNet 3.0 is an enhanced lexical resource explicitly devised for supporting sentiment classification and opinion mining applications and it contains a list of words classified as positive, negative, or neutral. Then, we use the weighted average of the classified words in the text and assign an overall percentage of the sentiment. *AFINN* sentiment is a simple, yet popular lexicon used for sentiment analysis developed by Finn Årup Nielsen. It contains more than 3300 words with a polarity score associated with each word. We use the in-built function for this lexicon, which is available in Python.

We use all three programs with the full texts of the articles, their titles and their summaries in order to generate more sentiment analysis features and to be able to determine those that provide the best results. This gives rise to a total of 12 features for SA, which can be found in more detail in Table II in Section IV-B.

IV. Methodology

The genetic programming is being run with two different inputs. First, with 12 features from TA and later with 12 features from SA. In this section, we first introduce the two processes individually and then we present our GP algorithm.

A. Financial analysis processes

1) Technical analysis process: For TA, we downloaded the historical prices of the companies via Yahoo Finance!. The price columns we are using to calculate the TA features are Adjusted Close, Close, High, and Low. The features can be found in the Background Information for the TA, in Section III-A, where the usage of the columns is stated in detail. For this part, we normalized the values of the TA indicators to be between $[-1, 1]$.

2) Sentiment analysis process: As a first step, we downloaded the articles where we extract the sentiment from, using the exact same dates as the data as in IV-A1. The articles were downloaded via a made scrapper that was able to retrieve articles appearing from the first to the twentieth page of the results returned by the Google search engine. The scrapper was given in input the name of the company that we wanted to study and was searching the articles based on that. To achieve that we used the Google Search library as offered for use in Python.

Some of the data retrieved was not relevant to the companies, so we shortlisted them based on whether or not they include in the article the name of the company or its stock market name. Also, we kept only the articles with length of at least 500 characters, in order to avoid articles that were not downloaded properly or not all. Lastly, we sorted the articles in a chronological order from the oldest to the most recent one.

The sentiment of the articles was obtained using the TextBlob polarity and subjectivity tool, and the SentiWordNet and AFINN sentiment libraries available in Python.

After having the datasets with the articles and their sentiment ready, we noticed that occasionally more than one articles appeared at the same date. In these cases, we used the average features of the total amount of articles published at the same day. We then removed the duplicated dates and we normalized the sentiments, so all of the values are in $[-1, 1]$.

To conclude with the SA datasets, we matched the dates of the articles with the stock market dates, since this way we were able to link the sentiment directly with its date and stock price. That being said, we included the sentiment of the weekend days into the one for Fridays, as it is expected that these will influence the stock price of Monday, since the stock market is not open at the weekends. For those dates that belong to the stock price market, but do not have any articles, we set their sentiment as 0, as an indicator of neutrality and no movement.

B. *Genetic programming*

In this subsection, we present the specific details of the approach we follow regarding genetic programming, starting with the model representation, fitness function and the genetic programming operators.

1) Model representation: We used tree structures for the individuals, with the inner nodes of our trees composed of the logical functions AND, OR, Greater than (GT) and Less than (LT). Tables I-III present the function and terminal sets. The SA features include the text, title and summary of the polarity and subjectivity of TextBlob, the polarity of SentiWordNet, and the AFINN sentiment. The MovingAverage, Momentum, ROC, William's %R, Volatility and Midprice are included in the TA indicators, which we use for two different periods, i.e., for 5 and 10 days.

TABLE I
FUNCTION SET

Function set	
Function set	AND, OR, LT, GT

TABLE II
TERMINAL SET -SA

Terminal set	
SA-textBlob	TEXTpol, TEXTsub TITLEpol,TITLEsub SUMMpol,SUMMsub
SA-SentiWordNet	TEXTsenti, TITLEsenti, SUMMsenti
SA-AFINN	TEXTafinn, TITLEafinn, SUMMafinn ERC

TABLE III
TERMINAL SET -TA

Terminal set	
TA (for 5 and 10 days)	Moving Average Momentum ROC William's %R Volatility Midprice ERC

For both Table II and Table III, we added a variable named Ephemeral Random Constant (ERC). This variable takes random real values from -1 to 1, and performs the role of a threshold for the features, i.e., the algorithm checks whether the feature is greater than (or less than) this random value, as part of maximizing the *Sharpe ratio*. ERC is not a fixed universal variable, but it is being randomly generated for each feature. Figure 1 presents a sample image of a GP individual.

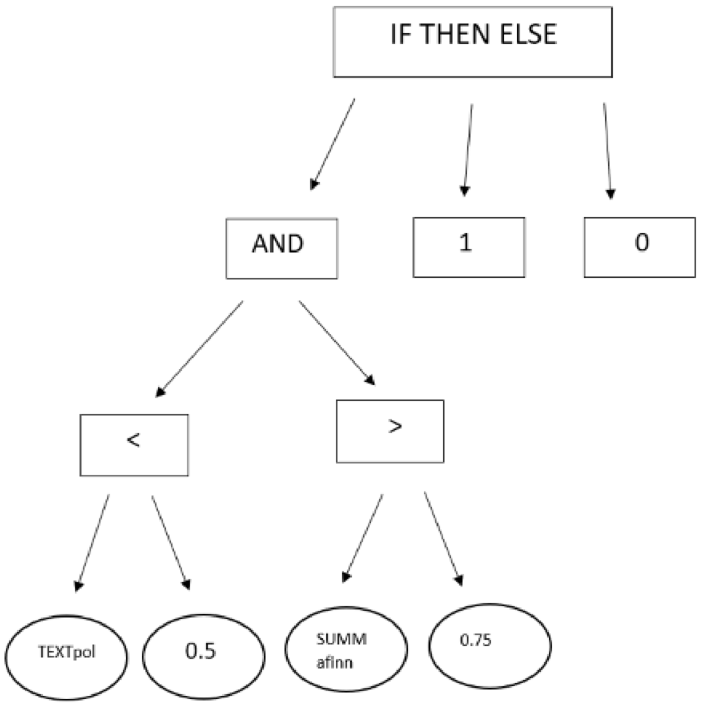

Fig. 1. This individual includes only SA features. In particular, the features represented are the TextBlob polarity for the article's text and the AFINN sentiment on the summary. 0.5 and 0.75 are the Ephemeral Random Constant (ERC).

If the tree structure for an individual evaluates to True (i.e., for a given data point Text Polarity is less than 0.5 and at the same time the AFINN sentiment on the articles' summary is greater than 0.75), then the tree will get the signal of 1 (i.e., buy), otherwise it will get a signal of 0 (i.e, hold). These signals are then embedded into a trading strategy, which is presented below in Section IV-C.

2) Fitness function: The fitness function used by genetic programming algorithm to train the models is the *Sharpe ratio*, which it tries to maximize. The Sharpe ratio is defined as

$$\mathsf{SharpeRatio} = \frac{\mathsf{E}(R) - R_f}{\sqrt{\mathsf{Var}(R)}}, \tag{1}$$

where E and Var stand for the sample mean value and the sample variance, R stands for the returns and R_f is the risk-free rate. The data used, i.e., the returns, for computing the Sharpe ratio were obtained by the trading algorithm outlined in Section IV-C, which indicates when the selling of the stocks will take place.

3) Genetic programming operators: The operators used for evolving the trees are point mutation and subtree crossover. The evolution of the trees is determined by a crossover ratio (p), while the mutation probability is $1-p$, which is a common scheme, as stated at [18]. Moreover, we use elitism to ensure that the best individual of each generation is copied to the next.

C. Trading algorithm

As discussed also in Section IV-B, based on the models' outcome, we will get a signal of 1, if the value of the feature is greater than (or less than) the ERC, otherwise we get a signal of 0. When the obtained signal is 1, the model buys a stock, which later sells based on other two parameters, n and r. The n denotes the number of days that the evaluation took place inside the datasets, while r is the increase rate of reference. For example, if $n = 20$ and $r = 0.03$, we would say "If within 20 days there is a price increase more than 3% compared to the buy price, the trade will take place at that point, otherwise the trade will take place at the end of those 20 days."

When the above trade occurs, the return from the stock is computed based on the price P_b we bought the stock at the beginning and the price P we sold the stock, as seen in Equation 2. These returns are being saved as a list and at the end of the dataset we find the sample mean of that list, which will give the overall return. The *risk*, as seen in Equation 3, is the standard deviation of the list of the returns.

$$R = \left\{ \frac{P - P_b}{P_b} \right\} \tag{2}$$

$$\mathsf{Risk} = \sqrt{\mathsf{Var}(R)} \tag{3}$$

Equation 2 presents the list of returns R, consisting of individual returns for each trade; the same R is fed as input to Equation 1 to determine the Sharpe ratio.

V. Experimental Setup

A. Data

We gathered 6 years worth of data from 26 companies across different sectors for technical (i.e., stock market data) and sentiment (i.e., news articles) analysis, aiming to identify the advantages of sentiment analysis in maximizing profit and minimizing the risk of stock investments.

For the technical indicators, recall that the price data were collected from Yahoo Finance!, while for sentiment analysis we downloaded articles, their titles and their summaries, with a code-made web scrapper that uses the Python library Google Search. The search included the company names and it was programmed to go into 20 web pages for each enterprise; we obtained 12 features for each analysis.

After downloading the prices and the articles, we proceeded with computing the six TA indicators for two periods in time each (5 and 10 days), generating 12 features. Furthermore, we used TextBlob, SentiWordNet and AFINN sentiment (as described in Section IV-B) to include the sentiment in the articles, their titles and summaries, giving us another 12 features and reaching the 24 in total.

The 6 years mentioned are from 1st of January 2015 until 31st of December 2020, that is our data, also, including the coronavirus crisis. The above dates were used to create the analyses' indicators, and the dates are the same for each company's datasets for these two analyses.

In both cases, we separated the datasets into 60% training, 20% validation and 20% for testing. The 12 features that have been created for the two analyses individually are used as datasets along with the adjusted closing prices of the companies' stock as an input to the genetic programming algorithm.

B. Parameter tuning

We performed a grid search for a set of possible GP parameters on the validation set for the population size, the crossover probability (p), the number of generations, the tournament size, and the maximum tree depth. As mutation probability is set to $1 - p$, we did not include this variable in the grid search. The best set of parameters that was returned in the validation set is presented in Table IV.

TABLE IV
GP Parameters

GP Parameters	
Population size	1000
Crossover probability	0.9
Mutation probability	0.1
Generations	30
Tournament size	4
Maximum depth	4

After, that, we had to select the trading parameters, i.e., the number of days n and the increase rate r for the trading algorithm described in the previous subsection; we allowed for different values of n and r for each dataset, to ensure that the trading strategy is tailored to each dataset. This part of tuning took, again, place in the validation set.

VI. Results and Analysis

In this section we present the results of the GP algorithm when run exclusively with TA features and, similarly, when run exclusively with SA features. In an effort to analyze these results in more depth, and in particular whether these are statistically significant, we perform Kolmogorov-Smirnov tests and discuss their outcome.

The results in Tables V - VII refer to the averages of 50 GP runs, when using only the runs that included at least four trades, in order to have a clear understanding of the risk. This is because when there are just two trades, i.e., a buy and a sell, we get one value for return in the list of returns, which we cannot perform standard deviation on and, in this situation, the gp is programmed to give *risk* the value of 0. Tables V - VII include all 26 companies involved in this experiment, starting with the average for the Sharpe ratio (Table V), followed with those for the return (Table VI) and the risk (Table VII).

A. *Results and analysis*

TABLE V
AVERAGES FOR SHARPE RATIO PER COMPANY

Company	***SA***	***TA***
AAPL	**5.127936736**	0.90454103
ADBE	**17.38794624**	0.564981425
ADIDAS	**42.09221369**	0.726818512
ALIBABA	-7.638711802	**1.105587684**
AMAZON	**3.425648293**	2.448287295
ASICS	1.430652867	**1.986743396**
ATVI	3.452821826	**4.158467463**
BMW	**5.783317978**	1.94667859
EBAY	7.367054479	**49.10791375**
FB	**9.626718322**	2.487577588
GOOGLE	0.345603234	**9.872997213**
HONDA	-20.5143675	**5.790378836**
IBM	**3.888354823**	2.486833884
INTEL	-0.441666991	**1.842496312**
JNJ	-	**0.604622416**
MSFT	**4.185270269**	2.213920551
NFLX	0.775344887	**2.421316557**
NESTLE	**5.155274875**	1.791303014
NIKE	**12.36417005**	0.523336983
NVDA	**4.283218119**	2.336279705
SUZUKI	**6.312935623**	4.231702068
TENCENT	**2.652200062**	1.632894286
TESLA	**7.238989225**	4.865312124
TOYOTA	**1.31306344**	-
WALMART	0.575568209	**2.051716383**
XEROX	0.539674435	**1.976882338**
Average	**4.489585822**	**4.233830362**

With a first glance at Table V, we see that the highest *Sharpe ratio* as an average of the 50 runs per company is produced by SA, although it is also SA that contains the lowest values. In particular, SA tops TA in 15 companies, while TA outperforms SA in the remaining 11; note also that TA always returns a non-negative Sharpe ratio.

As it is evident from Table VI, the highest *return* on average is achieved by TA. It leads to better performance in 16 companies (out of 26), leaving 10 companies that perform better with SA.

Furthermore, as seen in Table VII, we observe that Toyota has a risk of 0 when using solely TA features and, furthermore, a Sharpe ratio and return of 0 as well. Similarly, we observe the same behavior when focusing on Johnson & Johnson and SA

TABLE VI
AVERAGES FOR RETURN PER COMPANY

Company	***SA***	***TA***
AAPL	0.026978058	**0.027253178**
ADBE	**0.038907937**	0.032541961
ADIDAS	**0.038459072**	0.007118251
ALIBABA	0.006616963	**0.023657873**
AMAZON	**0.032106594**	0.023009331
ASICS	0.045025391	**0.048985513**
ATVI	0.013419181	**0.047379916**
BMW	0.023019944	**0.038472431**
EBAY	**0.043255031**	0.031272402
FB	**0.034540772**	0.030793037
GOOGLE	-0.067536941	**-0.000695799**
HONDA	0.009222055	**0.056859198**
IBM	**0.067741311**	0.038105615
INTEL	-0.041767506	**0.042890538**
JNJ	-	**0.013453758**
MSFT	0.023252809	**0.030466502**
NFLX	0.000252114	**0.024835323**
NESTLE	**0.040024147**	0.032802361
NIKE	**0.034573135**	0.019172256
NVDA	0.038292444	**0.046146135**
SUZUKI	0.012561748	**0.053641219**
TENCENT	0.039433789	**0.072139012**
TESLA	**0.044818951**	0.031603914
TOYOTA	**-0.001029652**	-
WALMART	0.020720561	**0.044433096**
XEROX	0.023005835	**0.035456367**
Average	**0.020995913**	**0.032761284**

TABLE VII
AVERAGES FOR RISK PER COMPANY

Company	***SA***	***TA***
AAPL	**0.024422119**	0.049337409
ADBE	**0.021307405**	0.059192656
ADIDAS	**0.007799016**	0.076697577
ALIBABA	**0.031438309**	0.050821269
AMAZON	**0.012356813**	0.024871077
ASICS	**0.045457951**	0.049424381
ATVI	**0.009826735**	0.018065891
BMW	**0.01501277**	0.027604447
EBAY	**0.02556621**	0.041181434
FB	**0.012950216**	0.045212079
GOOGLE	0.120961585	**0.070037629**
HONDA	**0.039232313**	0.03696648
IBM	**0.018169994**	0.05297858
INTEL	0.080627389	**0.067886322**
JNJ	-	**0.040375947**
MSFT	**0.01188538**	0.041427137
NFLX	0.027138512	**0.021586373**
NESTLE	**0.026280338**	0.026364602
NIKE	**0.0074631**	0.054787368
NVDA	**0.011027668**	0.03427774
SUZUKI	**0.030458522**	0.039599769
TENCENT	0.100001411	**0.062683065**
TESLA	**0.006160936**	0.039427163
TOYOTA	**0.023347804**	-
WALMART	0.041050218	**0.033825308**
XEROX	0.071700534	**0.067765565**
Average	**0.031601663**	**0.043553741**

features. The reason is that the first company had no tradings in all 50 runs for TA, and the second did not perform in any of the 50 runs more than 2 trades. In both cases, we chose to consider superior the strategy that produced trades, i.e., SA features for Toyota and TA features for Johnson & Johnson.

By considering the number of firms where each approach performs better, we remark that SA leads to better results with respect to *risk* for 19 companies, while TA is to be preferred in 7 companies. Note also that the lowest average risk in a company is produced by SA.

For further analysis we will now present the Kolmogorov-Smirnov tests for the *Sharpe ratio*, *return* and *risk*. Our objective is to understand better the results obtained by the two analyses and, in particular, examine the statistical significance of these results, under the assumption that the SA and TA datasets follow the same distribution.

The two-sample Kolmogorov-Smirnov test is a useful method when comparing two samples, as it is sensitive to differences in the location and the shape of the empirical cumulative distribution functions of the two samples. The test reports the maximum difference between the two cumulative distributions and then computes a p-value.

In our analysis, we will reject the null hypothesis, that the two samples come from the same distribution, at the significance level of 0.05.

When comparing the *Sharpe ratio* of the SA and TA strategies, we found a statistically significant higher Sharpe ratio in SA than TA, rejecting the null hypothesis that they follow the same distribution, with a p-value of **0.0308**. The same outcome can be, also, observed from Table V, where the average Sharpe ratio of SA is higher than that of TA.

For the *return* of the two strategies, we did not find a statistical significance for either SA and TA, thus we were not able to reject the null hypothesis, with the p-value being **0.2581**.

For the *risk* associated with the SA and TA features, we observed a statistically significant lower risk in SA than TA, rejecting the null hypothesis at a p-value of **0.0017**. This makes the SA a better choice for more risk averse investors.

From the experiments we conclude that, when it comes to the *Sharpe ratio* and *risk*, SA has been more financially advantageous than the use of technical indicators for these 26 companies in the past 6 years, when combined with the GP algorithm. Indeed, for both metrics the comparison is statistically significant, while for returns, even though Table VI indicates that the mean return is higher for TA than for SA, there is no statistical significance in this comparison.

B. Features' analysis

We now discuss the most common features used by the GP algorithm, based on the best results out of the 26 companies and the models that produced the highest *Sharpe ratio* for each company. In the tables below we present the best five features of each analysis.

Let us remark that Table VIII suggests that no specific sentiment analysis tool outperforms its competitors. Indeed, all three of TextBlob, SentiWordNet and AFINN appear in the top five features, although features regarding the text of the article seem to be preferred.

TABLE VIII
TOP SA FEATURES OUT OF 26 COMPANIES

Feature	***No of Times***
TextBlob summ Subjectivity	38
AFINN text	28
SentiWordNet summ	25
AFINN summ	24
TextBlob text polarity	22

TABLE IX
TOP TA FEATURES OUT OF 26 COMPANIES

Feature	***No of Times***
Moving Average 10days	24
Midprice 5days	22
Midprice 10days	22
Moving Average 5days	21
William's %R 10days	20

The features that can be observed the most in Table IX are MovingAverage, Midprice and William's %R, not including the other 3 TA features. Furthermore, we can notice that the indicators based on the longer period, i.e., the 10 days, seem to be slightly preferred compared to the indicators generated based on 5 days.

Moreover, observe that the top five SA features appear to be more dominant than the top five TA features, when considering all 26 companies. Indeed, Table VIII indicates that the most preferred SA features sum up to 137, while the corresponding TA features to 109, as seen in Table IX.

C. Computational results

In this subsection we will look at the computational costs for each analysis after 1 run and the 30 generations of the GP algorithm, keeping the same parameters as mentioned in Section V-B, at Table IV. The time for the two models appears in minutes.

TABLE X
COMPUTATIONAL TIMES PER ANALYSIS

Computational times per analysis	
SA	1.17
TA	1.16

As it can be observed in Table X, both models exhibit the same behavior with respect to computational efficiency. This time can be shortened even further, by parallelizing the execution of the models; since each candidate solution can be produced and evaluated individually from the others among the algorithm's populations. This has been shown in [19], where the algorithms were sped up by up to 21 times.

VII. Conclusion and Further Experiments

We investigated the performance of a genetic programming algorithm relying either on technical analysis or on sentiment analysis indicators. From the experiments, and as evidenced from the results from 26 different companies across various sectors, we concluded that using sentiment analysis features can produce competitive results to technical analysis. In fact, we found that SA statistically outperformed TA in both *Sharpe ratio* and *risk*.

Of course, these results could be dependent on the choice of indicators we used. So, we are not interested in making any claims that SA should be preferred to TA. Instead, we are more interested in demonstrating that SA can be a complimentary method to TA, and as a result enhance the predictive performance of a trading algorithm. Our next step is thus to build trading strategies that contain both sentiment and technical analysis indicators. We remark, however, that in preliminary experiments we have conducted, we observed that simply putting all indicators into a single file and feeding it to the genetic programming algorithm actually did not return as financially profitable results as the individual sentiment and technical analysis. This is potentially because of ineffective search. Our future plan is to create distinct branches in the genetic programming algorithm, one that will take only technical analysis indicators and one that will take only sentiment indicators. This will allow the search to focus on each indicator type, which will hopefully improve the quality of the search and return better performance results.

References

[1] M. M. Mostafa, "Forecasting stock exchange movements using neural networks: Empirical evidence from kuwait," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6302–6309, 2010.

[2] S. Wang and W. Shang, "Forecasting direction of china security index 300 movement with least squares support vector machine," *Procedia Computer Science*, vol. 31, pp. 869–874, 2014.

[3] D. M. Nelson, A. C. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with lstm neural networks," in *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 1419–1426.

[4] A. Brabazon, M. Kampouridis, and M. O'Neill, "Applications of genetic programming to finance and economics: past, present, future," *Genetic Programming and Evolvable Machines*, vol. 21, no. 1, pp. 33–53, 2020.

[5] J. Li and E. P. Tsang, "Improving technical analysis predictions: An application of genetic programming." pp. 108–112, 1999.

[6] M. Kampouridis and E. Tsang, "Investment opportunities forecasting: Extending the grammar of a gp-based tool," *International Journal of Computational Intelligence Systems*, vol. 5, no. 3, pp. 530–541, 2012.

[7] M. Kampouridis, A. Alsheddy, and E. Tsang, "On the investigation of hyper-heuristics on a financial forecasting problem," *Annals of Mathematics and Artificial Intelligence*, vol. 68, pp. 225–246, 2013.

[8] M. Kampouridis and F. Otero, "Heuristic procedures for improving the predictability of a genetic programming financial forecasting algorithm," *Soft Computing*, vol. 21, pp. 295–310, 2017.

[9] S. Y. Yang, S. Y. K. Mo, A. Liu, and A. A. Kirilenko, "Genetic programming optimization for a sentiment feedback strength based trading strategy," *Neurocomputing*, vol. 264, pp. 29–41, 2017.

[10] J. M. Berutich, F. López, F. Luna, and D. Quintana, "Robust technical trading strategies using gp for algorithmic portfolio selection," *Expert Systems with Applications*, vol. 46, pp. 307–315, 2016.

[11] K. Kohara, T. Ishikawa, Y. Fukuhara, and Y. Nakamura, "Stock price prediction using prior knowledge and neural networks," *Intelligent Systems in Accounting, Finance & Management*, vol. 6, no. 1, pp. 11–22, 1997.

[12] B. Xie, R. Passonneau, L. Wu, and G. G. Creamer, "Semantic frames to predict stock price movement," in *Proceedings of the 51st annual meeting of the association for computational linguistics*, 2013, pp. 873–883.

[13] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[14] M.-Y. Day and C.-C. Lee, "Deep learning for financial sentiment analysis on finance news providers," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2016, pp. 1127–1134.

[15] S. Loria, "textblob documentation," *Release 0.15*, vol. 2, 2018.

[16] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in *LREC*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias, Eds. European Language Resources Association, 2010. [Online]. Available: http://nmis.isti.cnr.it/sebastiani/Publications/LREC10.pdf

[17] F. Å. Nielsen, "A new ANEW: evaluation of a word list for sentiment analysis in microblogs," in *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, ser. CEUR Workshop Proceedings, M. Rowe, M. Stankovic, A.-S. Dadzie, and M. Hardey, Eds., vol. 718, May 2011, pp. 93–98. [Online]. Available: http://ceur-ws.org/Vol-718/paper_16.pdf

[18] R. Poli, W. Langdon, and N. McPhee, "A field guide to genetic programming," 2009.

[19] J. Brookhouse, F. E. Otero, and M. Kampouridis, "Working with opencl to speed up a genetic programming financial forecasting algorithm: Initial results," pp. 1117–1124, 2014.