4-23-2019 3:00 PM

Machine Learning for Stock Prediction Based on Fundamental Analysis

Yuxuan Huang
The University of Western Ontario

Supervisor
Capretz Luiz F.
The University of Western Ontario Co-Supervisor
Ho Danny
NFA Estimation Inc.

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Engineering Science

Abstract

Application of machine learning for stock prediction is attracting a lot of attention in recent years. A large amount of research has been conducted in this area and multiple existing results have shown that machine learning methods could be successfully used toward stock predicting using stocks' historical data. Most of these existing approaches have focused on short term prediction using stocks' historical price and technical indicators. In this thesis, we prepared 22 years' worth of stock quarterly financial data and investigated three machine learning algorithms: Feed-forward Neural Network (FNN), Random Forest (RF) and Adaptive Neural Fuzzy Inference System (ANFIS) for stock prediction based on fundamental analysis. In addition, we applied RF based feature selection and bootstrap aggregation in order to improve model performance and aggregate predictions from different models. Our results show that RF model achieves the best prediction results, and feature selection is able to improve test performance of FNN and ANFIS. Moreover, the aggregated model outperforms all baseline models as well as the benchmark DJIA index by an acceptable margin for the test period. Our findings demonstrate that machine learning models could be used to aid fundamental analysts with decision making regarding to stock investment.

Keywords: Stock prediction, fundamental analysis, machine learning, feed-forward neural network, random forest, adaptive neural fuzzy inference system.

Acknowledgments

I would like to thank my supervisor Dr. Luiz Fernando Capretz and co-supervisor Mr. Danny Ho, for supporting me during my research here at Western University. The patience, freedom and guidance they have provided me with are crucial to my personal development at Western and the completion of this thesis.

I would also like to thank Tamer and Mohammed for their encouragement and advices during this time.

Finally, I am also grateful to my parents and Brandy for their constant support throughout my research and thesis writing. The delicious food and comfortable bed on my trips home helped me cope with frustration and retain motivation.

Table of Contents

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1

1 Introduction

1.1 Purpose

The main motivation for predicting changes in stock price is the potential monetary returns. A large amount of research has been conducted in the field of stock performance prediction since the birth of this investment instrument, as investors naturally would like to invest in stocks which they have predicted will outperform the others in order to generate profit by selling them later. A large inventory of stock prediction techniques has been developed over the years, although the consistency of the actual prediction performance of most of these techniques is still debatable. The techniques for stock prediction can be classified into a small number of categories:

1. Fundamental analysis, where the predictions are made by studying the underlying companies through their published financial statements.
2. Technical analysis, where the predictions are made by analyzing only the historical prices and volumes.
3. Sentiment analysis, where the predictions are made by analyzing the published articles, reports and commentaries pertaining to certain stocks.

The last category is much newer than the other two, as it is only made possible by the invention of the Internet and the online databases of up-to-date news articles. Of the three general categories of stock prediction techniques, technical analysis and sentiment analysis are primarily used for short-term prediction on the scale of days or less. Fundamental analysis on the other hand, is used for mid-term and long-term prediction on the scale of quarters and years [1]. In recent years, the popularity of applying various machine learning and data mining techniques to stock prediction has been growing. The majority of the existing studies using machine learning and data mining focus on creating prediction models based on technical analysis and sentiment analysis [2] [3] [4]. Results from many of these studies have shown that prediction models trained with historical price and volume data can be successfully used towards short-term predicting [3] [4]. However, there is one major drawback for short-term prediction and high frequency trading, which is frictional cost or trading commission. For an investor trading stocks through a broker, there is typically a commission paid to the broker for each buy and sell. The rate of commission varies from broker to broker, but it can really eat up the potential profit as the trading frequency increases [5], even with discount brokers. Since the short-term prediction models from many of the studies do not incorporate frictional cost in evaluation [3] [4], the conclusiveness of the studies may be affected.

In this thesis, we aim to evaluate machine learning methods for long-term stock prediction based on fundamental analysis. We do so by comparing the prediction performance of three advanced machine learning methods based on fundamental analysis using fundamental features. To develop and test the machine learning models, we used data extracted from the quarterly financial reports of 70 stocks that appeared in the S&P 100 between 1996 and 2017. In order to evaluate the performance of different machine learning methods, we rank the 70 stocks based on their predicted relative return. Portfolios are constructed based on the ranking and the actual relative returns of the portfolios are used as the evaluating criteria.

1.2 Contributions

In this thesis, we evaluated advanced machine learning methods for long-term stock prediction based on fundamental analysis. We also proposed a portfolio selection method which selected stocks based on assembled results from multiple machine learning predictors. For experimentation, we worked with real financial data extracted from companies' quarterly financial reports.

The main contributions of this thesis can be summarized as follows:

1. We organized and cleaned financial data extracted from 70 companies' quarterly financial reports across a period of 22 years.
2. We examined three machine learning algorithms for stock prediction based on fundamental analysis: Adaptive Neural Fuzzy Inference System (ANFIS), Feed-Forward Neural Network (FNN) and Random Forest (RF). A ranking based portfolio selection method was used to construct portfolios based on predictions. The relative returns of constructed portfolios were evaluated with respect to benchmark index.
3. We used RF based feature selection method for identifying important features and improving model performance.
4. We used bootstrap aggregating to assemble the prediction results from three machine learning algorithms in order to improve the return of the constructed portfolios.

1.3 Thesis Outline

The reminder of this thesis is organized as follows: In Chapter 2, the background knowledge in stock prediction, financial analysis and related challenges is covered. The introduction to the three machine learning methods used in this research is also included in this chapter. In Chapter 3, researches in the area of the application of machine learning methods in stock prediction are reviewed. Chapter 4 discusses the data set used in the experiment as well as the details about the data preparation process. In Chapter 5, the methodology of the proposed experiment is discussed in detail. The results we obtained are presented and discussed in Chapter 6. Finally, the conclusion of the research and future work are presented in Chapter 7.

Chapter 2

2 Background

2.1 Stock Prediction Basics

The purpose of this thesis is to apply supervised learning methods to the financial data of particular stocks in order to predict future performance. A stock prediction problem like this involves employing knowledge from both machine learning and the stock market. Because of the cross-disciplinary nature of our research problem, it is necessary to cover the important general concepts of stock prediction and financial analysis before continuing.

2.1.1 Important Terms

As this thesis approaches the stock prediction problem from a software engineering perspective, some knowledge of the terminologies from the finance domain is essential. In order for clear understanding of the contents of this thesis, some important terms which would be used later are defined here.

Universe of Stocks: The goal of this research is to build stock prediction models to predict a set of stocks, and then select stocks from the set to construct portfolios. The set of stocks is formally called a universe [6]. In this research, the universe of stocks consists of the 70 stocks selected from the S&P 100 index.

Absolute Return: Absolute return is the return an asset achieves over a certain period of time, expressed as a percentage. The return is not compared to any other measure or benchmark [7]. For example, assume you invest $100 into a stock A at time t=0. You sell the investment at t=1 for $120. Then the absolute return of this investment over the period of time between t=0 and t=1 is (120-100)/100 = 20%.

Relative Return: In contrast to absolute return, relative return is the return an asset achieves over a period of time compared to the return of a benchmark over the same period of time. The relative return is the difference between the absolute return of the asset and the absolute return of the benchmark [8]. For example, assume you invest $100 into a stock A at time t=0. You sell the investment at t=1 for $120 to achieve an absolute return of 20% from this investment. Over the same period of time from t=0 to t=1, the benchmark S&P500 index achieves absolute return of 30%. Thus, the relative return of your investment with respect to benchmark S&P500 is 20%-30% = -10% between t=0 and t=1. Note here that the relative return of your investment with respect to benchmark is negative, even though it achieved positive absolute return. This means that you would be better off investing into the S&P500 index fund, which exposes your investment to much less risk than an individual stock does, for that period of time.

Portfolio: A portfolio is any combination of financial assets constructed and held with the intention of earning a profit.

Equal Weight: Equal weight is a strategy used for constructing a portfolio or index. In an equal-weight portfolio of stocks, each stock receives the equal weight, and thus the same amount of investment.

Backtesting: In order to evaluate the performance of a stock investment strategy, one can test the strategy in the real world by making a real or hypothetical investment according to the strategy. The future return of the investment could validate the strategy. This testing method is called forward testing. One obvious disadvantage of forward testing is that it requires an extended period of testing time. Alternatively, a stock investment strategy can be tested by simulating its usage on historical stock prices, where the investment returns are calculated as if the strategy were applied at the time. This testing method is called backtesting. Backtesting relies on the fundamental assumption that a strategy which worked well in the past is likely to work well in the future. In other words, history tends to repeat itself [9].

2.1.2 Types of Financial Analysis

There are various established techniques investors traditionally use for helping with evaluating stocks and predicting future price movement. These techniques can be classified into three major types: technical analysis, fundamental analysis and sentiment analysis.

Technical Analysis: technical analysts evaluate investments and identify buying or selling opportunities by analyzing statistical trends gathered from historical price and volume. Unlike fundamental analysts, who attempts to evaluate a stock's intrinsic value using publicly available information, technical analysts assume that a stock's price already reflects all publicly available information. There are three premises that technical analysis is based upon [10]:

1. Market action discounts everything
2. Prices move in trends
3. History repeats itself

Fundamental Analysis: fundamental analysis attempts to measure the intrinsic value of a stock by considering a broad number of factors from the overall economy in relation to industry performance and a company's financial factors such as earnings, profit margin, assets, liabilities and so on. Price movement history and volume are rather insignificant to fundamental analysts. World famous investor Warren Buffet is well recognized as a practitioner of fundamental analysis.

Sentiment Analysis: Sentiment analysis uses natural language processing and text analysis to systematically extract and identify subjective information. Sentiment analysis is widely applied to different areas. For stock market, sentiment analysis is used to identify the overall attitude of investors towards a particular stock or the overall market. Sentiment analysis is not within the scope of this research, thus it will not be discussed in detail.

In general, fundamental analysis and technical analysis have different importance when examined under the factor of the predicting horizon. Fundamental analysis is usually preferred when the predicting time horizon is a quarter, a year or longer. Technical analysis is preferred for short-term prediction such as for days or less.

2.1.3 Challenges of Stock Prediction

Stock trading is a process of buying and selling shares of publicly listed companies on a stock exchange platform, with millions of investors and traders from all over the world actively involved at any given time when the market is open. Stock market prediction is an extremely complex and difficult problem because there are simply too many factors and noises affecting the movement of the price. Many existing studies associated with stock market prediction support the well-known Efficient Market H(EMH) [11], according to which the price of a stock at any given time reflects all information available about it and is therefore impossible to predict [12]. In Figure 2.1, we can compare the real IBM historical price with randomly generated random walk results. One could easily be led to believe that both price trends are generated randomly. There are three forms of EMH, based on the degree of stock market efficiency:

Weak form EMH implies that the market efficiently reflects all past market information. The hypothesis assumes that past rates of return have no effect on future rates.

Semi-strong form EMH implies that the market efficiently reflects all publicly available information. This hypothesis assumes that the stock price adjusts quickly to absorb new information, e.g. a company’s financial reports. Semi-strong form incorporates weak form EMH.

Strong form EMH implies that the market efficiently reflects all information, both public and private. This hypothesis assumes that no investor would be able to achieve above average returns even if he/she was given new information that is not available publicly. Strong form EMH incorporates weak form and semi-strong form EMH.

Recent studies which have explored using machine learning and soft computing techniques for stock prediction, have achieved results that challenge the weak and semi-strong form EMH [17]-[23]. However, most of these studies use historical price, technical indicators or investor sentiments as independent variables for model training and prediction. The main motivation of our research is to develop machine learning models to simulate the decision-making process of investment experts based on a stock's fundamental financial ratios.

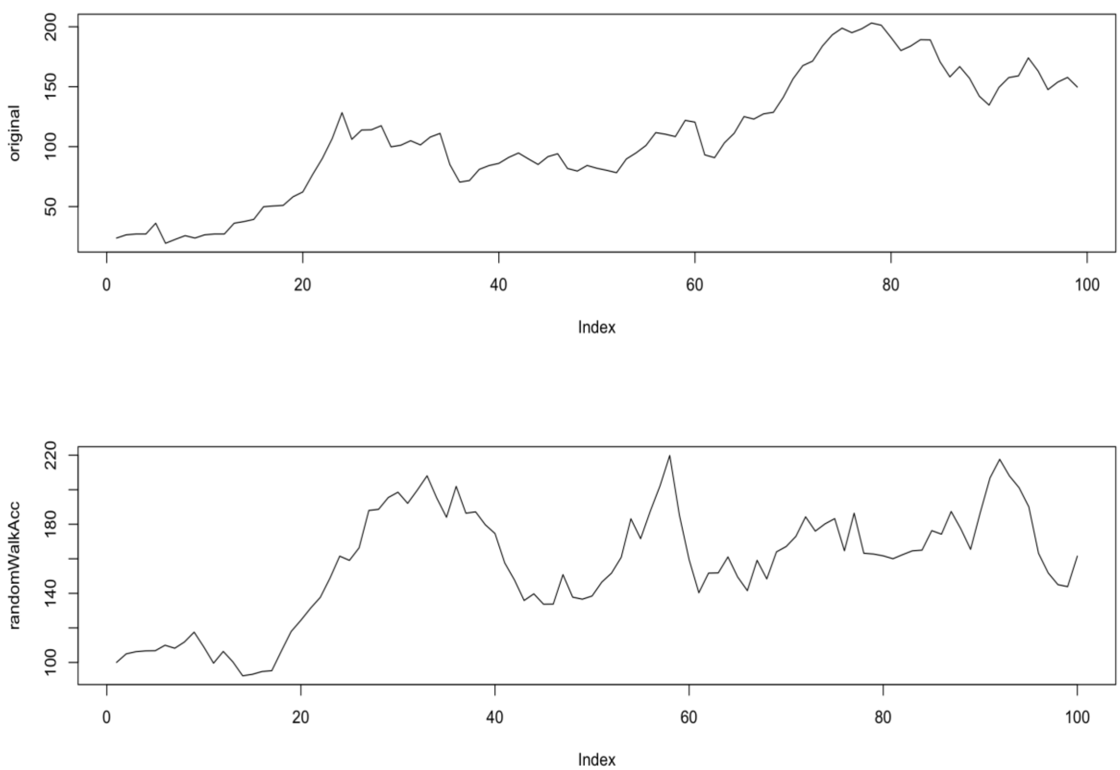

Figure 2.1: Real quarterly price of IBM stock(top) and random walk results of the same length(bottom)

2.2 Machine Learning Methods

This section presents background information on the machine learning methods used in this thesis: FNN, RF, ANFIS. Because of the nature of our problem and dataset, all three methods used are supervised learning methods.

2.2.1 Feed-forward Neural Network

Feed-forward Neural Network (FNN), or Multi-Layer Perceptrons (MLP), is the simplest and very versatile form of neural network architecture. An FNN consists of at least three layers: an input layer, a hidden layer and an output layer. The architecture of a typical FNN with one hidden layer is illustrated in Figure 2.2. The supervised learning technique of gradient descent is used for backpropagation. In the training process, the change of cost with respect to the weight between two nodes is calculated as [13]:

$$\Delta w_{ji}(n) = -\gamma \frac{\delta \varepsilon(n)}{\delta w_{ji}(n)} \tag{2.1}$$

where γ is the learning rate, ε is the error in the final output, w_{ji} is the weight between neuron j and neuron i.

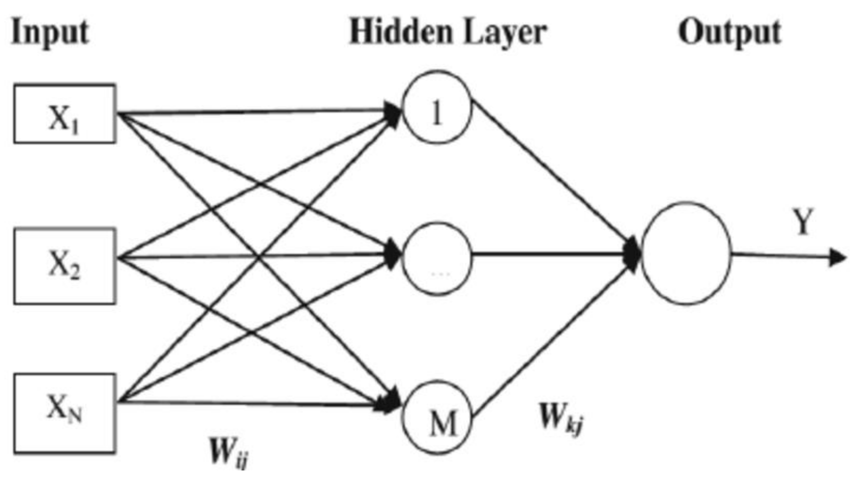

Figure 2.2: FNN architecture

There are many hyperparameters that can be tuned during the model validation of an FNN in order to achieve the optimal model generalization.

Weight Initialization Methods: The weights connecting neurons between different layers must be initialized before training the model. Good choice of initialization method could speed up the learning process of the network. Some of the popular weight initialization methods include initializing weights to random small values with normal distribution or uniform distribution.

Learning Rate: The learning rate controls the rate of adjustment to be made to the weights with respect to the loss gradient. Traditionally, constant learning rate or learning rate schedules are used. Common learning rate schedules include time-based decay, step decay and exponential decay. In recent years, many established adaptive learning rate methods such as Adagrad, Adadelta, RMSprop and Adam have become popular.

Number of Hidden Layers: The number of hidden layers needs to be determined during the initial design of any FNN. Generally, the number of hidden layers is based on the size and dimension of the dataset. Deep neural networks with many hidden layers are suitable for large datasets with high feature dimension.

Number of Hidden Units: The number of neurons in each hidden layer needs to be determined as well. Just like the number of hidden layers, the number of hidden units is also based on the size and dimension of the dataset.

Activation Functions: Each node in a neural network is a neuron that uses a nonlinear activation function, except for the input neurons. For a regression problem, the activation function of Rectified Linear Unit (ReLU) is typically used for hidden units.

2.2.2 Random Forest

Random Forest (RF) is a flexible supervised learning algorithm which can be used for both classification and regression tasks. It builds multiple decision trees during the data fitting process. A small decision tree is illustrated in Figure 2.3. For generating results, RF takes the mean value of the output of all decision trees for a regression problem. For classification problems, the majority voting from the decision trees is used as the result.

Many hyperparameters can be tuned to increase the performance of RF. A few of the most important hyperparameters for RF are listed below:

Number of Estimators: This is just the number of decision trees the algorithm builds before taking the maximum voting or the average of predictions. In general, a larger number of decision trees increases the performance of the algorithm at the cost of slower computation.

Minimum Sample Split: The minimum number of samples required to split an internal node. This should be based on the size of the dataset.

Maximum Features: The number of features to consider when looking for the best split. The dimension of the dataset needs to be taken into account when tuning this hyperparameter.

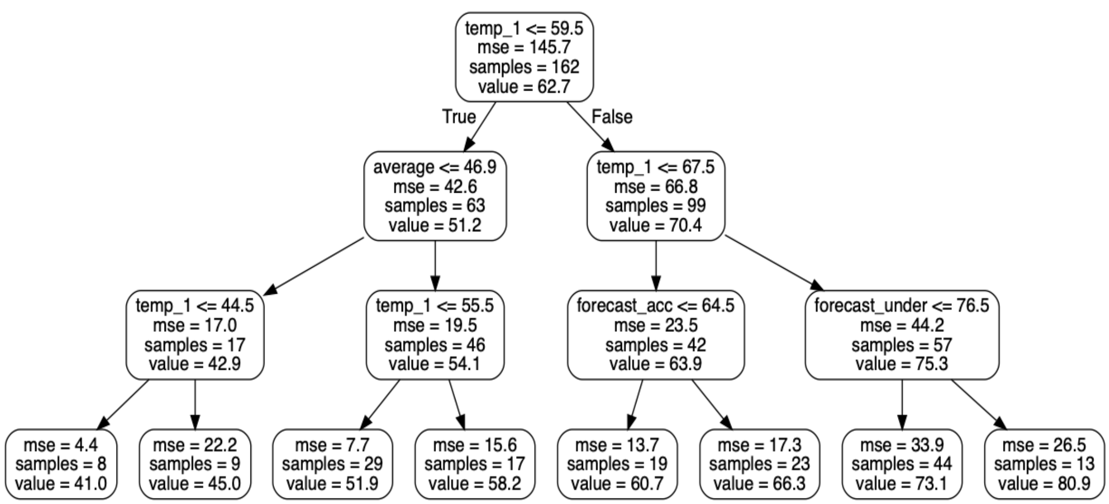

Figure 2.3: A small decision tree

2.2.3 Adaptive Neural Fuzzy Inference System

ANFIS is an instance of the more generic form of the Takagi-Sugeno-Kang (TSK) fuzzy inference system. It replaces the fuzzy sets in the implication with a first order polynomial equation of the input variables [14]. The ANFIS system consists of rules in IF-THEN form. In general, there are five different layers in an ANFIS system. Layer 1 converts each input value to the outputs of its membership functions:

$$O_i^i = \mu_{A_i}(x) \tag{2.2}$$

where x is the input to node i and $\mu_{A_i}(x)$ is the bell-shaped membership function with maximum equal to 1 and minimum equal to 0.

Layer 2 calculates the firing strength of a rule by simply multiplying the incoming signals. Layer 3 normalizes the firing strengths:

$$\overline{w_i} = \frac{w_i}{\sum_j w_j} \tag{2.3}$$

Layer 4 consists of adaptive nodes with function defined as [14]:

$$O_i^4 = \overline{w_i}(p_i x + q_i y + r_i) \tag{2.4}$$

where $\overline{w_i}$ is the normalized firing strength from the previous layer and $(p_i x + q_i y + r_i)$ is a first order polynomial with three consequent parameters $\{p_i, q_i, r_i\}$.

Layer 5 takes the weighted average of all incoming signals and delivers a final output:

$$O_1^5 = overall\ output = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{2.5}$$

Where f_i is the first order polynomial mentioned above. The structure of a typical ANFIS is shown in Figure 2.4.

Tuning an ANFIS involves determining the number of membership functions for each input and the type of input membership function. For the MATLAB Fuzzy Toolbox which we use for building ANFIS models, several clustering methods are available for defining membership functions and fuzzy rules automatically based on the input data, including grid partition, subtractive clustering and fuzzy C-Means clustering.

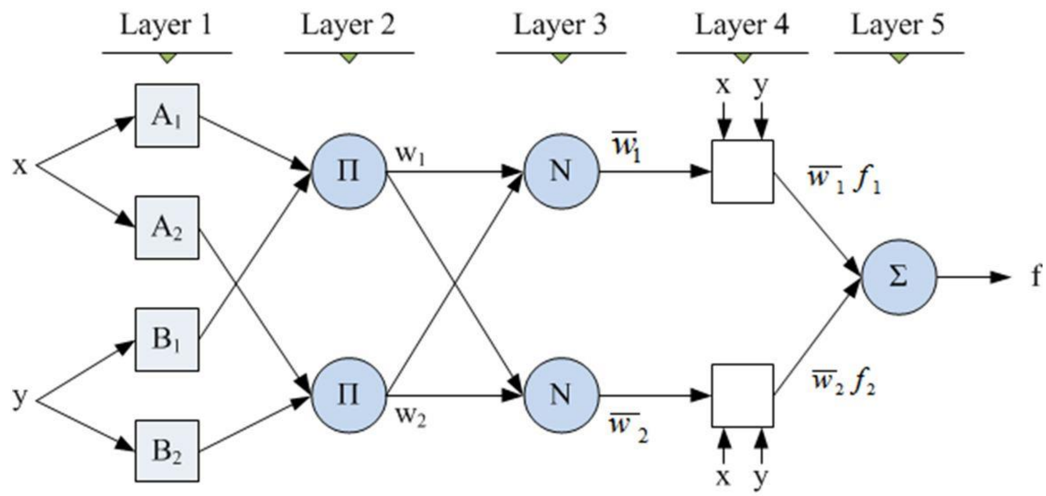

Figure 2.4: Structure of an ANFIS

Chapter 3

3 Related Research

This section focuses on highlighting related studies for the application of machine learning for stock prediction. We first look at researches that use classical approaches to predict stock performance, specifically using fundamental analysis. Next, studies which apply machine learning algorithms to stock's technical data are examined. Finally, we will cover approaches that use machine learning with fundamental analysis for stock prediction. Thus, the related research covered in this section are increasingly similar to ours.

3.1 Stock Prediction with Fundamental Analysis

As mentioned in the previous chapter, fundamental analysis techniques focus on evaluating a stock's intrinsic value based on publicly available financial ratios of the company. The history of fundamental analysis may be traced back to the book "Security Analysis" written by Benjamin Graham and David Dodd in 1934 [15]. This book laid the intellectual foundation of what was later called value investing. Many researches have been conducted in trying to formularize and extend the stock selection principles from the book.

Piotroski [16] proposed a logistic regression model called F-Score for assessing strength of a company's financial position. F-Score was calculated based on nine financial criteria extracted from a company's financial reports. These criteria were divided into 3 groups:

profitability, liquidity and operating efficiency. Some of the financial ratios are defined as follows:

$$Return\ on\ Assets(ROA) = \frac{Net\ Income}{Total\ Assets} \tag{3.1}$$

$$Debt\ Ratio = \frac{Total\ Debt}{Total\ Assets} \tag{3.2}$$

$$Current\ Ratio = \frac{Current\ Debt}{Current\ Assets} \tag{3.3}$$

$$Gross\ Margin = \frac{Revenue - Cost\ of\ Goods\ Sold}{Revenue} \tag{3.4}$$

$$Asset\ Turnover\ Ratio = \frac{Total\ Sales}{Average\ Total\ Assets} \tag{3.5}$$

Piotroski [16] backtested the F-Score model for stock selection with data from 1976 to 1996 and achieved positive results.

Similarly, Mohanram [17] developed a G-Score model for stock selection. The G-Score model was calculated based on a different set of financial criteria extracted from a company's financial reports. These criteria are divided into three categories: Profitability, Naïve Extrapolation and Accounting Conservatism. G-Score was backtested between 1978 and 2001, and a strong positive relationship was found between G-Score and realized returns.

3.2 Stock Prediction with Machine Learning

3.2.1 Based on Technical Analysis

The majority of the existing studies that apply machine learning to stock prediction are based on technical analysis [3] [4]. Machine learning models developed in these studies take historical prices or technical indicators derived from historical prices as inputs. The popularity of technical analysis based models is due to the popularity of technical analysis among the financial media and Wall Street financial advisors. In addition, stocks' technical data are available in much larger volume compared with financial fundamental data. This is because a stock's price and technical indicators are available with a daily sampling frequency, while its financial fundamental data is only published on a quarterly basis.

Kimoto et al. [18] studied the use of feed-forward neural network for stock prediction back in 1990. The inputs of their prediction model consisted of technical indicators as well as some macroeconomic indices such as interest rate and foreign exchange rate. They tested their model for generating buying and selling signals of the TOPIX index for a 33 months period, from January 1987 to September 1989. The results show that the neural network prediction model is able to achieve superior profit over the buy-and-hold strategy.

Patel et al. [19] explored four different machine learning algorithms for stock price predication, including Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF) and naïve-Bayes. For input data, ten technical indicators were used. They tested two approaches for model building. The first approach used the ten continuous-valued technical indicators as is. The ten features were normalized before being used for training. The second approach discretized the technical indicators to represent the deterministic trend. For experiment, 10 years of daily historical data of two stocks from S&P Bombay Stock Exchange (BSE) Sensex was used. The results indicate that RF model outperforms the other three prediction models on overall performance. The results also suggest that the prediction performance can be improved by converting inputs from continuous-values data into discrete trend deterministic data. Patel et al. [20] also proposed using a fusion of different machine learning techniques to improve prediction performance. In the proposed two stage model, Support Vector Regression (SVR) was first used to predict the value of technical indicators n days ahead. ANN, SVR and RF were used in the second stage for predicting closing price n days ahead using predicted technical indicators from the first stage. The results suggest that this two-stage fusion model is able to achieve superior performance over single stage models.

Chong et al. [21] examined deep neural network for the stock prediction problem. The assumption of this research was that properly tuned deep neural network is able to extract features from a large set of raw data without relying on prior knowledge of predictors to predict stock price movement with reasonable accuracy. For raw input data, 380 dimensional lagged stock price (38 stocks and 10 lagged prices) were used. Three unsupervised methods were tested for feature extraction: principal component analysis (PCA), autoencoder and restricted Boltzmann machine. As the research aimed to test deep neural network for high frequency trading, the time interval between each observation of stock price data was only 5 minutes apart. The deep neural network was trained to predict stock price movement 5 minutes ahead. Standard root mean square error (RMSE), mean absolute error (MAE) and normalized mean square error (NMSE) were used for performance evaluation. The results show that the deep neural network model achieves prediction performance similar to a simple linear autoregressive model. Chong et al. [21] further experimented applying the deep neural network to the residuals of the autoregressive model and achieved better results.

Bekiros et al. [22] compared ANFIS model and Recurrent neural network (RNN) model for predicting the next day trend of NASDAQ and NIKKEI indices. For both models, the previous closing price was used for predicting the next day's closing price. To avoid data snooping, they used data from 1971 to 1998 for model training and data from 1998 to 2002 for out-of-sample testing. The results suggest that the rate of return for ANFIS is superior to that of the RNN model as well as the buy-and-hold strategy for both indices.

Atsalakis et al. [23] proposed an ANFIS model for predicting the next day price trend. The proposed model took historical price and price moving average as inputs. Results of the model are evaluated in terms of hit rate, which is defined as:

$$Hit\ rate = \frac{h}{n} \tag{3.6}$$

where h denotes the number of correct predictions of the stock trend and n denotes the number of tests. Five stocks were chosen for backtesting. The ANFIS model achieved an average hit rate of 62.3%. Atsalakis further calculated the rate of return (ROR) of the proposed model over the test horizon and compared it with the buy-and-hold strategy. The ROR is defined as follows:

$$ROR = \frac{net\ gain\ in\ stock}{initial\ investment} \tag{3.7}$$

The results suggest that the ANFIS model is able to achieve significantly higher ROR than the buy-and-hold strategy for all five stocks. The ANFIS model was then compared horizontally with neural network models from previous studies. Atsalakis claimed that the proposed model is able to achieve a superior hit rate over previous models.

A k-NN based neuro-fuzzy system is proposed by Wei et al. [24]. In this study, the k-NN method was used to select k instances of historical data that are most similar to the testing input. These data were then used to create the predicting model. Thus, instead of using all training data to train a model, k-NN was utilized to dynamically select k instances for each prediction. The model was tested with TAIEX data from 1999 to 2004 and compared with univariate neural network and fuzzy time series models. The results suggest that the proposed model achieved a smaller RMSE than other baseline models.

3.2.2 Based on Fundamental Analysis

Next, we look at previous works on stock prediction and stock selection which combine machine learning with fundamental analysis.

Quah and Srinivasan [25] developed a feed-forward neural network model for stock selection using quarterly fundamental financial factors. Seven input features were chosen as listed below:

$$Historical\ PE\ ratio = \frac{Price}{Earning\ per\ share(EPS)} \tag{3.8}$$

$$Prospective\ PE\ ratio = \frac{Price}{consensus\ EPS} \tag{3.9}$$

$$Market\ Cap = Price \times number\ of\ shares\ outstanding \tag{3.10}$$

$$EPS\ uncertainty = \%\ deviation\ from\ median\ EPS\ estimates \quad (3.11)$$

$$Return\ on\ Equity(ROE) = \frac{Net\ Income}{Shareholder\ Equity} \quad (3.12)$$

$$Cashflow\ Yield = \frac{Price}{Operating\ cashflow} \quad (3.13)$$

The consensus EPS in Equation 3.9 is the average of financial analysts' published estimate EPS for next quarter. The last feature is a momentum factor derived by weighted average of historical price appreciation. For the experiment, quarterly data of 25 stocks from Q1 1993 to Q4 1996 were used. Note that there are only 16 observations for each stock. The first 10 observations are used for training and last 6 observations for testing. Due to the limited data size, another approach of moving window system is also tested. The moving window system uses three quarters as a training sample and the subsequent quarter as the testing sample. Stocks with the highest predicted returns were selected for a portfolio for each of the test points. The absolute returns of the portfolios were evaluated. The experimental results suggest that the proposed model is able to select portfolios that outperform the market 10 out of 13 testing quarters and achieve superior returns. However, as Quah and Srinivasan [25] noted in their conclusion, the experiment is largely constrained by the availability of data. Thus, the conclusiveness of their results is limited.

Lam [26] developed a similar feed-forward neural network model using more data. The model is trained and tested on 364 S&P companies for the period from 1985 to 1995. The inputs of the model included 16 financial statement variables and 11 macroeconomic variables. Lam set up 4 experiments to test different combinations of predictors. The first three experiments used one year's, two years' and three years' financial data as inputs, respectively. This setup helps to simulate the time series effect for analysis. The last experiment combined three years' financial data with macroeconomic data as inputs. Lam did not separate part of the data set for model validation but instead presented results from neural network models with different numbers of hidden layers. The top 33% of stocks with the highest predicted returns were chosen to build a portfolio, and the performance of the portfolio was evaluated. There are two key insights from the experimental results. Firstly, the rate of return increases gradually from experiment 1 through 3. Such results show that integrating the technical analysis technique of examining historical trend with fundamental analysis can improve the level of return. Secondly, the results from experiment 4 suggest that the addition of macroeconomic variables does not improve model performance.

Eakins and Stansell [27] examined whether using a neural network modelling procedure for stock forecasting based on a set of financial ratios could improve investment returns. They used the yearly financial data of all stocks listed on Compustat between 1975 and 1996. Stocks with small market capitalization or high volatility were filtered out. They trained one model for each year using stocks' financial ratios at the end of that year as predictors and next year's real returns as independent variables. The top 50 stocks with the highest predicted returns were selected into a portfolio for evaluation. Based on the experimental results, they argue that the neural network selected portfolio is able to consistently outperform the full sample as well as board market benchmarks, with an average annual return of 17.1% over the 20-year test period. The full sample achieves an average annual return of 7.93% while S&P 500 and Dow Jones Industrials achieve 11.4% and 11.3% respectively over the same period.

Quah [28] compared three different machine learning models for stock selection based on fundamental analysis. The machine learning methods tested in this research are FNN, ANFIS and general growing and pruning radial basis function (GGAP-RBF). A dataset of 1630 stocks which were extracted within a period of ten years from 1995 to 2004 was used. Out of the ten years' annual data, only the last year's data were used for test set. Quah picked 11 of the most commonly used financial ratios as predictors based on Graham's book [15]. Instead of training the supervised learning models to do regression, Quah converted the prediction problem into a classification problem by classifying target variable into two classes. "Class 1" was defined as any stock which appreciates in share price equal to or more than 80% within one year, otherwise was classified as "Class 2". Such classification naturally creates an imbalanced dataset, as very few stocks are able to appreciate over 80% within one year in any given year. Therefore, an over-sampling technique was used on the minority class in order to balance the dataset. Over-sampling was used on training set only for the purpose of avoiding data scooping. According to the experimental results, both FNN and ANFIS models were able to achieve above market average annual appreciation of selected stocks at 13% and 14.9% respectively. The average annual appreciation of the market for the test set is 11.2%. On the other hand, GGAP-RBF performed poorly. The author also mentioned in the conclusion that the availability of financial data is a major limitation of this study.

Shen and Tzeng [29] combined soft computing model using the dominance-based rough set approach (DRSA), formal concept analysis (FCA), and DEMATEL techniques for exploring the usefulness of fundamental analysis. 17 financial ratios of 112 IT stocks listed in Taiwan Stock Exchange from 2010 to 2013 were used for their research. Based on their value appreciation over each period, the stocks are divided evenly into three classes: “low”, “mid” and “high” holding-period-return (HPR). The DRSA was then trained to classify stocks’ HPR based on their financial ratios. The test results suggest that the DRSA model is able to separate winners and losers from the sample, as the average predicted “high”, “low” and market benchmark index HPR over the test period are 11.5%, -5.91% and 5.04% respectively. Moreover, FCA were conducted to extract the six most important features as well as twenty decision rules. The six most important features from FCA results are:

1. Revenue growth rate (ΔREV)
2. Gross profit growth rate (ΔGrossProfit)
3. ROA growth rate (ΔROA)
4. Debt ratio
5. Asset turnover rate
6. Average days for sales

Average days for sales ($DAYs$) was defined as:

$$DAYs = \frac{Average\ ending\ inventory}{Operational\ cost} \times\ 365\ days \qquad (3.14)$$

This study argues that the decision rules found could assist investors with investment decisions.

Hargreaves and Hao [30] applied two decision tree based methods (CHAID & C5.0) and FNN on stock trend prediction. Five financial ratios were used as predictors: Return on Equity, Return on Assets, Analyst Opinion, Annual Growth and Price. The classification models were used to predict positive of negative return in price for the next day. The evaluation criteria used in this research are sensitivity and specificity defined as follows:

$$Sensitivity = \frac{number\ of\ true\ positives}{number\ of\ true\ positives + number\ of\ false\ negatives} \qquad (3.15)$$

$$Specificity = \frac{number\ of\ true\ negatives}{number\ of\ true\ negatives + number\ of\ false\ positives} \qquad (3.16)$$

The results suggest that C5.0 decision tree achieved the best prediction sensitivity and specificity, at the rate of 98% and 88% respectively.

A recent study by Namdari and Li [31] also used FNN on stock trend prediction. They used 12 selected financial ratios of 578 technology companies on Nasdaq from 2012-06 to 2017-02 as their dataset. Instead of simply normalizing or standardizing these continuous features, they discretized all features by conducting topology optimization. For comparison, they also developed a different FNN model for predicting stock price trend based solely on historical price for the same companies and the same period of time. The results suggest that the FNN model based on fundamental analysis was able to outperform the alternative model based on technical analysis with overall directional accuracy of 64.38% and 62.84%, respectively.

Bohn [32] combined technical analysis, fundamental analysis and sentiment analysis and compared a set of machine learning models for long-term stock prediction. He used a universe of around 1500 stocks which appear in the S&P 500 between 2002 and 2016 for experiment. Regression models were built, and ranks were induced based on the model predictions for each validation and test week. He evaluated the model performance using the Spearman rank correlation coefficient between predicted rank and actual rank. The results suggest that the neural network model combined with iterative feature selection could match the performance of a model developed with human expertise from an investment firm.

Yu et al. [33] developed a novel sigmoid-based mixed discrete-continuous differential evolution algorithm for stock performance prediction and ranking using stock's technical and fundamental data. The evaluation metrics and feature selection process used in this study is the same as in [32]. 483 stocks listed in Shanghai A share market from Q1 2005 to Q4 2012 were used for model building and testing. The results suggest that proposed model can create portfolios that significantly outperform the benchmark.

Chapter 4

4 Data Preparation

One of the challenges we faced in this research was related to putting together a dataset of stocks' financial ratios for experimentation and testing. Building a dataset involves gathering data from various sources and putting them together. Data also needs to be prepared properly before being used for model training and testing. In the following sections of this chapter, we will discuss how data samples are selected, compiled and prepared.

4.1 Data Collection

Sample stocks used for this experiment were chosen from the S&P 100 Index components. The index includes 102 leading U.S. stocks which represent about 51% of the market capitalization of the entire U.S. equity market [34]. There are two major reasons behind choosing the S&P 100 components as sample stocks. First, financial fundamental ratios for the S&P 100 stocks are relatively complete and large in terms of data volume. This is because these stocks are large-cap, and most of them were publicly listed relatively early in history. Second, the S&P 100 components are well balanced across different sectors, and we decided that the number of its components was suitable for the size of our sample stock pool. Because the composition of the S&P 100 index is frequently revisited, we decided to use its components as of December 2018 [34].

Historical financial data for each of the S&P 100 components were retrieved online in csv format [35]. The original dataset contains 40 features as illustrated in Appendix A. These data were extracted from companies' SEC 10_Q filings, which are published quarterly.

4.2 Filling Missing Data Values

The raw fundamental data retrieved from online for stocks in our universe have a considerable fraction of data entries missing. According to Graham's study on missing data analysis [36], the existence of missing values in a dataset could create problems for data handling, and thus ultimately generate invalid conclusions. For machine learning problems in particular as most of machine learning methods are designed to have complete data for training and testing missing values in the dataset must be handled before being used for building machine learning models.

As covered in [36] [37], common approaches for dealing with missing data include:

1. **Listwise deletion:** Listwise deletion removes every record that has one or more missing values. For data that is missing completely at random (MCAR), listwise deletion would only lead to a decrease in statistical power. If the data is missing not at random (MNAR), this approach may yield biased parameter estimates.
2. **Pairwise deletion:** Pairwise deletion is usually used in conjunction with a correlation matrix. Each correlation is estimated based on the cases having data for both variables. Thus, pairwise deletion maximizes all data available on an analysis by analysis basis. As pairwise deletion also assumes that missing data are MCAR, it can still yield biased parameter estimates if data is MNAR.
3. **Mean substitution:** This approach replaces missing values with the average of the parameter values that are not missing. Use of the mean substitution may be based on the fact that the mean is a reasonable guess of a value for a randomly selected observation from a normal distribution. With missing values that are not MCAR, the mean substitution could be a poor guess [37]. This approach can be further extended as mean substitution for subgroups, which replaces missing

values with the average of values within defined subgroups in order to get better estimations.

4. **Maximum likelihood estimation (MLE):** MLE uses available data to compute maximum likelihood estimates using the maximum likelihood function. Again, MLE also assumes that the data is at least missing at random (MAR), if not MCAR.

Other methods not described above include dropping features and using expert knowledge to manually fill in missing values. Dropping entire features may be a good option if there are a large number of features and the density of missing values for a feature is high. Filling in missing values manually with expert knowledge is only viable if the number of missing values is small.

The original dataset has large blocks of missing values concentrated on a few features, while other missing values sparsely populated across the entire dataset. We eventually decided to use a combination of feature deletion and mean substitution. In cases where a fundamental factor had large blocks of missing values or over 50% values missing, it was removed. We also removed some non-fundamental features such as price high and price low. After the feature dropping, there were still some sparsely located missing values which account for less than 3% of total samples. These missing values were then substituted by the average of the two adjacent values. For example, if the revenue data for 2015Q3 is missing, it is substituted by the mean of the revenue values of the 2015Q2 and 2015Q4.

4.3 Trend Stationary

Our target variable in this research is quarterly relative returns, while many features from the raw dataset possess a clear global trend with respect to time. As we transfer this time series problem into a supervised learning problem, these features with global trends could hinder our machine learning models' ability to generalize and provide reliable predictions. We therefore took the percentage change between consecutive observations for all features, which is calculated as follows:

$$\Delta x_t = \frac{(x_t - x_{t-1})}{x_{t-1}} \times 100\% \tag{4.1}$$

An example of trend-stationarizing a feature is shown in Figure 4.1 and Figure 4.2.

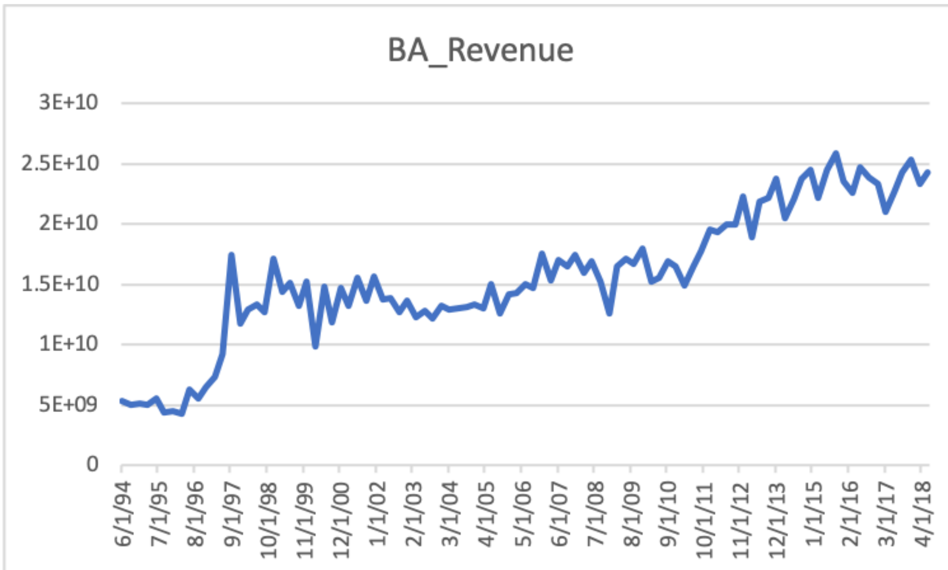

Figure 4.1: Historical quarterly revenue for BA – the original data

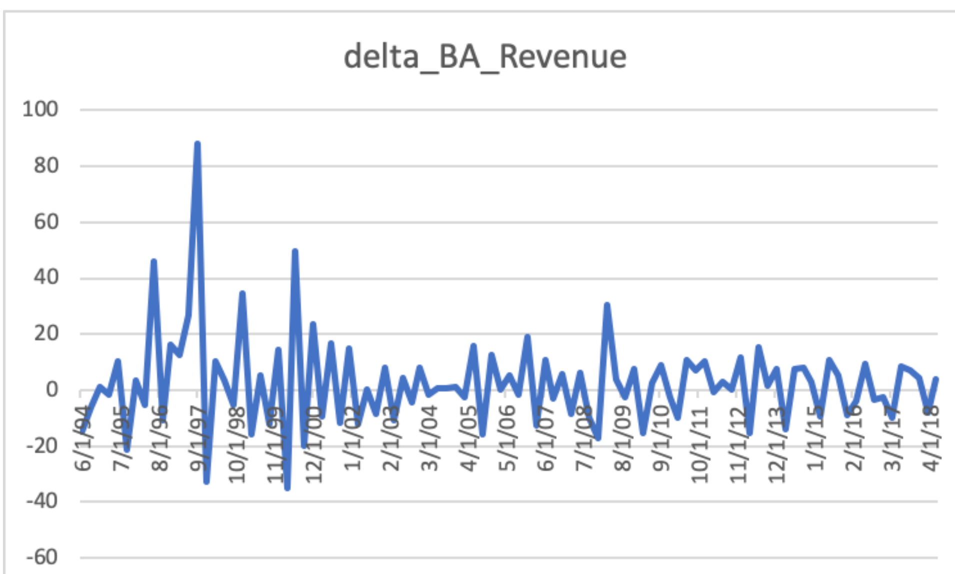

Figure 4.2: Historical revenue percentage change for BA

4.4 Standardization

As the scales of features can vary dramatically, standardization was applied to all features in order to improve the performance of our prediction models. Features are standardized using the following formula:

$$x' = \frac{x - \bar{x}}{\sigma} \tag{4.2}$$

Where x is the original feature vector, $\bar{x}$ is the mean of the feature vector, and σ is its standard deviation.

4.5 Relative Return

In the experiment, a stock's quarterly relative return with respect to the Dow Jones Industrial Average (DJIA) was used as the target variable instead of simple absolute return. The DJIA is one of the most widely used U.S. stock market benchmarks. The relative return of a stock is the difference between its absolute return and the return of some benchmark. There are two major benefits of using relative return instead of absolute return in our experiment. First, by subtracting overall market performance from the performance of each individual stock, we are able to filter out the factors affecting the broader market. In theory, using such a technique helps to reduce the complexity of the prediction problem and improve the prediction performance of our models. Second, it saves the step of comparing with the benchmark again in evaluation.

After the data preparation process was completed, we ended up with 21 features and 70 stocks. Each stock has 88 observations, ranging from Q1 1996 to Q4 2017, with an interval of one quarter between two consecutive observations. The 21 features are illustrated in Table 4.1:

No.	Feature Name	Feature Description
0	ΔPE	% change of price per earning
1	$\Delta Assets$	% change of total assets
2	$\Delta Current\_assets$	% change of current assets
3	$\Delta Liabilities$	% change of total liabilities
4	$\Delta Current\_liabilities$	% change of current liabilities
5	$\Delta Book\_value$	% change of book value
6	$\Delta Revenue$	% change of revenue
7	$\Delta Earning$	% change of earning
8	$\Delta Cash\_from\_Op$	% change of cash from operation
9	$\Delta Cash\_from\_Inv$	% change of cash from investment
10	$\Delta Cash\_from\_fin$	% change of cash from financing
11	$\Delta Cash$	% change of cash
12	$\Delta Capital\_Exp$	% change of capital expenditure
13	ΔPB	% change of price per book
14	$\Delta Cash\_per\_share$	% change of cash per share
15	$\Delta Current\_ratio$	% change of current ratio
16	ΔNet_margin	% change of net margin
17	ΔROA	% change of return on assets
18	$\Delta Asset\_turnover$	% change of asset turnover
19	ΔEPS	% change of earning per share
20	$Relative\_return$	Past quarter relative return on price

Table 4.1: Dataset features after data preparation

Chapter 5

5 Methodology

In this chapter, we focus on discussing the details in the experiment process. Dataset partition, evaluation metrics, feature selection and the final meta-algorithm for aggregating results from different models are covered.

5.1 Dataset Partition

It is easy to overfit when building machine learning models for financial prediction problems, especially with a limited amount of data. If all of the available data is used for model training, the ability of the generalizability of the model to further unseen data cannot be tested. Thus, it is crucial to hold-out part of the dataset as unseen data for testing throughout the model training process. However, there are also various hyperparameters which need to be tuned for optimal model performance. Different machine learning methods have different hyperparameters, as discussed in chapter 2. If we use the test dataset for model tuning, this would cause data snooping. As we already know the tuned model performs well on test data, the test data is not really unseen, and the generalizability of our machine learning model is weakened. Therefore, we partitioned our dataset into three sets: training, validation and test. The training set consists of 60% of the total data, while the validation set and test set consist of 20% each. From a time series perspective, data from Q1 1995 to Q1 2008 is used for training; data from Q2 2008 to Q2 2013 is used for validation, and data from Q3 2013 to Q4 2017 is used for testing. Moreover, we train the models with the training set and the validation set combined after model validation for generating the final test results on the test set. This helps us to maximize the usage of data for training the models without data snooping. Our strategy for data partition is illustrated in Figure 5.1.

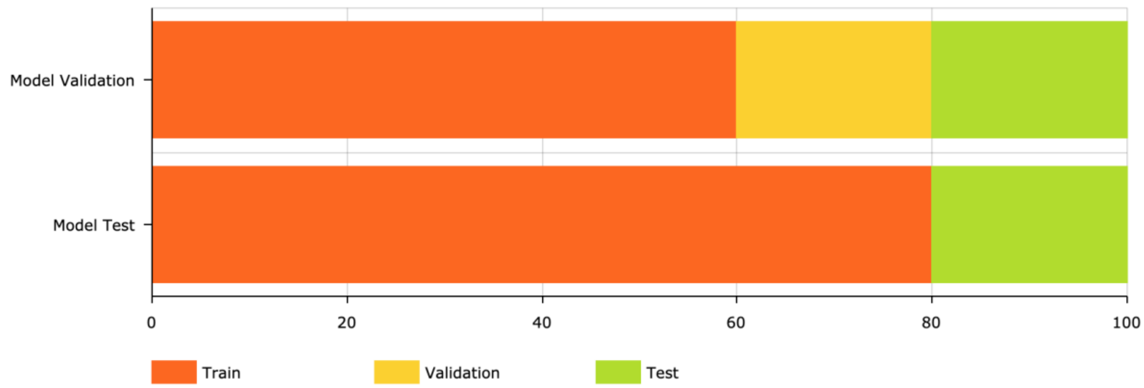

Figure 5.1: Data partition strategy

5.1.1 Standardization

As discussed in the previous chapter, the dataset needs to be standardized before being used for training. However, if standardization is applied before the data partition, we essentially use all data for calculating the mean and standard deviation in Equation 4.2. This means we use some information from the validation set and the test set even before the model validation phase. Such practice could be problematic as it leads to data snooping. On the other hand, standardizing data separately for training, validation and test set can avoid data snooping, but this practice could lead to poor validation and test results. This is because the mean and standard deviation of the same feature could be very different among different partitions, while the model is only trained to generalize the representations of the training set.

For our experiment, we first standardized the training set following Equation 4.2. Then we used the mean and standard deviation from the training set to standardize the validation set and the test set, following the formula:

$$x' = \frac{x - \overline{x_t}}{\sigma_t} \tag{5.1}$$

Where $\overline{x_t}$ and σ_t are the mean and standard deviation of the feature in the training set. This practice helps to avoid data snooping without harming model performance.

5.2 Local Learning

We tried both building a single model for all stocks and building one model for each stock. The two approaches can be classified as global learning and local learning. Models trained with global learning enjoy a larger set of training data, while models trained with local learning are more task specific and usually enjoy better performance [38]. Local learning approach was proven to have better performance in our early experiment, and thus we built one model for each stock for all three algorithms.

5.3 Evaluation Metrics

Previous studies on application of machine learning for stock prediction use different metrics for performance evaluation, as discussed in Chapter 3. The metrics are selected based on how the models are used for predicting stock performance:

Regression: For a regression model, the absolute or relative return of a stock at some time in the future is estimated. Metrics such as root mean square error (RMSE), mean square error (MSE), mean absolute error (MAE) and mean absolute percentage error (MAPE) are usually used for evaluating the accuracy of the regression model.

Classification: For a classification model, the possible return of a stock is divided into a small number of classes. For example, some studies [22] [23] which aim to predict a stock's price trend simply classify the future return into two classes: "up" and "down".

In this case, the hit rate defined as in Equation 3.6 is typically used. A stock's future return is classified more finely in some other studies [26] [28] [30]. In these cases, the correlation between actual class and predicted class, or sensitivity and specificity defined in Equation 3.15 and Equation 3.16 are usually used.

The goal of this project is to develop a system which can be used to guide stock portfolio design strategy for long term investment. Therefore, simple and general evaluation methods are preferred. We decided to build regression models to predict the price for each stock, and then induce a ranking of the stocks by sorting their predicted relative returns. The ranking can then be used for portfolio design, and the actual performance of the portfolios in terms of real relative return can be evaluated with ease. The performance evaluation methods of our model in training, validation and testing stages are discussed separately in the following sections.

5.3.1 Training Loss

When training a regression model, the metric or the loss function depends on the specific algorithm. Moreover, the loss function used in model training is also a hyperparameter which can be tuned. For the FNN and ANFIS models, we use RMSE as the training loss function. The RF algorithm, unlike FNN and ANFIS, does not involve training cycles and loss function.

5.3.2 Validation Performance

After fitting a model with the training data, it is then evaluated on the validation data. The stocks are ranked by their predicted relative returns for each of the quarters. The top one third stocks with the highest ranking are selected into a portfolio. The real relative return of the selected portfolio for each quarter is then calculated, assuming the portfolio is equal weight. The average real relative return of the equal weight portfolio is calculated with the formula:

$$\overline{R_p} = \frac{1}{\#quarters} \sum_{q=1}^{\#quarters} R_p\,(q) \tag{5.2}$$

Where $R_p(q)$ is the real relative return of the selected equal weight portfolio for quarter q calculated as follows:

$$R_p(q) = \frac{1}{\#stocks} \sum_{i=1}^{\#stocks} R_i(q) \tag{5.3}$$

Where $R_i(q)$ is the real relative return of a stock i in the selected portfolio for the single quarter q.

If the performance of the portfolio selected by our model is highly volatile from quarter to quarter, even if it can produce good relative return on average, it might still be undesirable. This is because high volatility leads to high risk, and high volatility can also diminish compounding return in long term. In the financial world, the Sharpe ratio is commonly used to help investors understand the return of an investment compared to its risk. The Sharpe ratio is a risk adjusted return ratio calculated as follows:

$$Sharpe\ Ratio\ = \frac{R_p - R_f}{\sigma_p} \tag{5.4}$$

Where R_p is the return of the portfolio, R_f is the risk-free rate (usually use the interest rate of saving account subtracted by inflation), and σ_p is the standard deviation of portfolio return over the considered duration. When constructing a portfolio, investors want to maximize the Sharpe ratio of the portfolio in order to get the maximum return with the minimum risk. For this project, we use a modified version of the Sharpe ratio as our risk-adjusted relative return metric:

$$Portfolio\ Score = \frac{\overline{R_p}}{\sigma_p} \tag{5.5}$$

Where $\overline{R_p}$ is calculated as in Equation 5.2. The risk-free rate is left out for simplicity.

5.3.3 Test Performance

After the hyperparameters are tuned on the validation data, the models are then tested on the test data. The final evaluation metrics are the same as those used in the validation stage.

5.4 Feature Selection

Feature selection is a process used to identify the features in a dataset that contribute most to the prediction of the target variable. Features proven irrelevant or redundant during this process could be dropped. There are two major benefits of feature selection. First, feature selection can identify and remove irrelevant and redundant features that do not contribute to the accuracy of a predictive model or may in fact decrease the accuracy of the model. Moreover, by removing irrelevant features, feature selection reduces the dimensionality of the dataset and therefore reduces the complexity of the model.

In this project, the RF algorithm is used for feature selection. The RF algorithm has demonstrated its efficiency in feature selection from previous studies [39] [40]. The algorithm is applied on the training data of all stocks in order to obtain estimates of feature importance of each feature. The most important features are then selected for model building.

5.5 Bootstrap Aggregation

After each individual algorithm is tested and evaluated, the bootstrap aggregating algorithm is applied in order to assemble the prediction results of different algorithms with the goal of improving stability and accuracy. Bootstrap aggregation is a simple and widely used meta-algorithm for aggregating predictive models. It is also the algorithm used in Random Forest for aggregating results from individual decision trees into a final output. Bootstrap aggregation can be used for both regression and classification tasks. For regression tasks, the average of the outputs from all models is taken as the aggregated output. For classification tasks, the class with the majority vote is taken.

In this project, bootstrap aggregation is used on the rankings of stocks produced by each algorithm. A stock is selected for inclusion in the portfolio if the majority of the algorithms predict that the stock's performance for the next quarter is in the top one third of all stocks.

Chapter 6

6 Results and Discussion

In this chapter, we present and discuss the results obtained from our experiments. This chapter is divided into three sections. First, the results from each of the three machine learning algorithms studied in this project: FNN, ANFIS and RF are presented, discussed and compared. Then, the RF algorithm is used to rank all features with respect to their feature importance. The most important features are then selected for a new round of model building and testing. The results of the three algorithms with the selected features are presented. Finally, bootstrap aggregation is applied to aggregate the predicting results of different models.

6.1 Baseline Models

For this phase of the experiment, the machine learning algorithms: FNN, ANFIS and RF are trained to predict the quarterly relative return of each of the 70 stocks. A rank of the stock is then induced from the predicted relative returns for each quarter. The ranking is then used for portfolio building. Before being used to produce predictions on the test set, each algorithm is first validated on the validation set for hyperparameter tuning. For evaluation, portfolios consisting of stocks from the top and bottom of the ranking are both evaluated. The reason for including the portfolios consisting of stocks with the worst predicted performance in our evaluation is that if our models can successfully identify the worst performing stocks, profit could potentially be generated by shorting these stocks.

The test results for each algorithm is discussed below. The overall results and cross model comparison are illustrated at the end.

6.1.1 Feed-forward Neural Network (FNN)

The FNN model is developed using Python and its TensorFlow interface: Keras. Hyperparameters of the FNN model, including number of hidden layers, number of neurons in each hidden layer, activation function, number of training epochs, initial learning rate and optimizer, are tuned through grid search. A set of hyperparameters is chosen based on the model with the best validation results. The hyperparameters are given in Table 6.1. Note that the model is quite small, with a single hidden layer of 21 nodes.

Layer sizes	[21]
# hidden layers	1
Activation	relu
loss	mse
# training epochs	100
Learning rate	0.01
Optimizer	adam

Table 6.1: FNN hyperparameters

The model is then used to generate predictions for each stock for the test period. The ranking for each quarter is then used to construct equal weight “Buy” and “Sell” portfolios with stock from the top and bottom of the rankings, respectively. For comparison purposes, five “Buy” portfolios and five “Sell” portfolios are constructed for each quarter. The five “Buy” portfolios consist of top 5, top 10, top 15, top 20 and top 30 stocks respectively. The five “Sell” portfolios consist of bottom 5, bottom 10, bottom 15, bottom 20 and bottom 30 stocks respectively.

When we look at the performance of the portfolios constructed based on the prediction of FNN for each testing quarter in Figure 6.1 and Figure 6.2, we can see that the "Buy" portfolios outperform the universe and the "Sell" portfolios underperform the universe most of the time. Moreover, portfolios consisting of more stocks are less volatile.

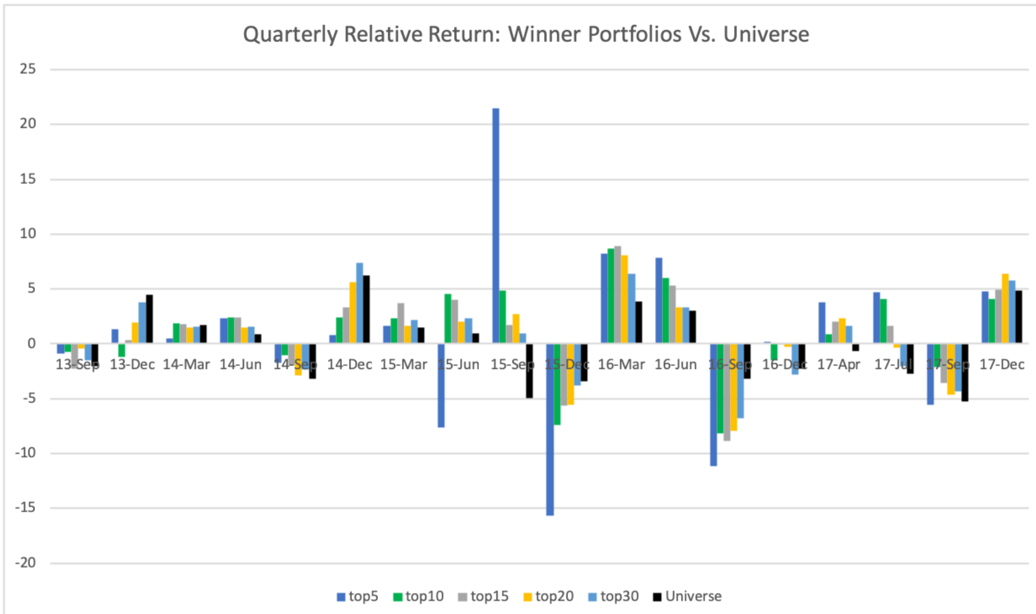

Figure 6.1: Relative return of FNN "Buy" portfolios for test quarters

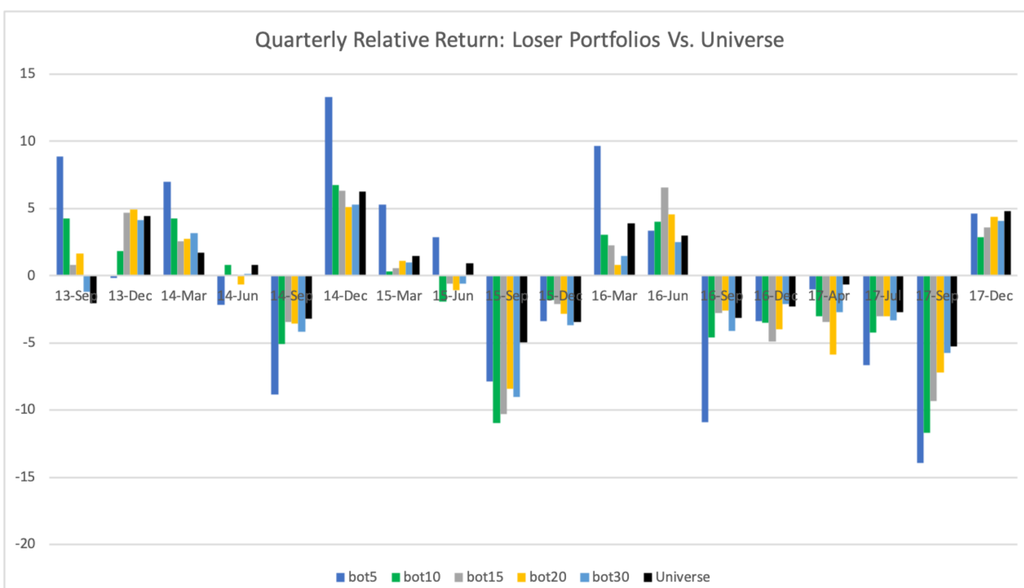

Figure 6.2: Relative return of FNN "Sell" portfolios for test quarters

	Mean	STD	Portfolio Score
Top5	0.824%	8.09	0.102
Top10	**1.10%**	4.33	**0.253**
Top15	0.984%	4.24	0.232
Top20	0.831%	4.11	0.202
Top30	0.721%	3.92	0.184
Universe	-0.0164%	**3.55**	-0.00460

Table 6.2: FNN results for “Buy” portfolios

	Mean	STD	Portfolio Score
Bottom5	-0.183%	7.63	-0.0240
Bottom10	**-1.03%**	5.14	-0.201
Bottom15	-0.694%	4.75	-0.146
Bottom20	-0.768%	4.22	-0.182
Bottom30	-0.825%	3.87	**-0.213**
Universe	-0.0164%	**3.55**	-0.00460

Table 6.3: FNN results for “Sell” portfolios

As we can see in Table 6.2 and Table 6.3, the constructed “Buy” portfolios consistently outperform the universe in terms of mean quarterly relative return and Portfolio Score, while the constructed “Sell” portfolios consistently underperform the universe. The standard deviation decreases as the number of stocks in a portfolio increases. For the “Buy” portfolios, the “Top10” portfolio has the best performance in terms of mean and Portfolio Score. For “Sell” portfolios, the “Bottom10” has the lowest mean while the “Bottom30” has the lowest Portfolio Score. Overall, we can see that FNN can successfully help to construct "Buy" and "Sell" portfolios that can outperform and underperform the universe, respectively.

6.1.2 Adaptive Neural Fuzzy Inference System (ANFIS)

The ANFIS model is developed using MATLAB and its Fuzzy Logic Toolbox. As with FNN, the ANFIS model is first validated on the validation set with grid search. The validated model uses subtractive clustering with a clustering influence range of 0.5 for defining membership functions and fuzzy rules. The model is trained for 20 epochs for optimal validation results.

The model is then tested on the test data. A number of portfolios are constructed for evaluation, same as for the FNN model. The relative return for each quarter in the test period for the "Buy" and "Sell" portfolios that were constructed based on ANFIS predictions are presented in Figure 6.3 and Figure 6.4. Similar to the results from FNN, we can see that the "Buy" portfolios can outperform the universe while the "Sell" portfolios underperform the universe for most of the test quarters.

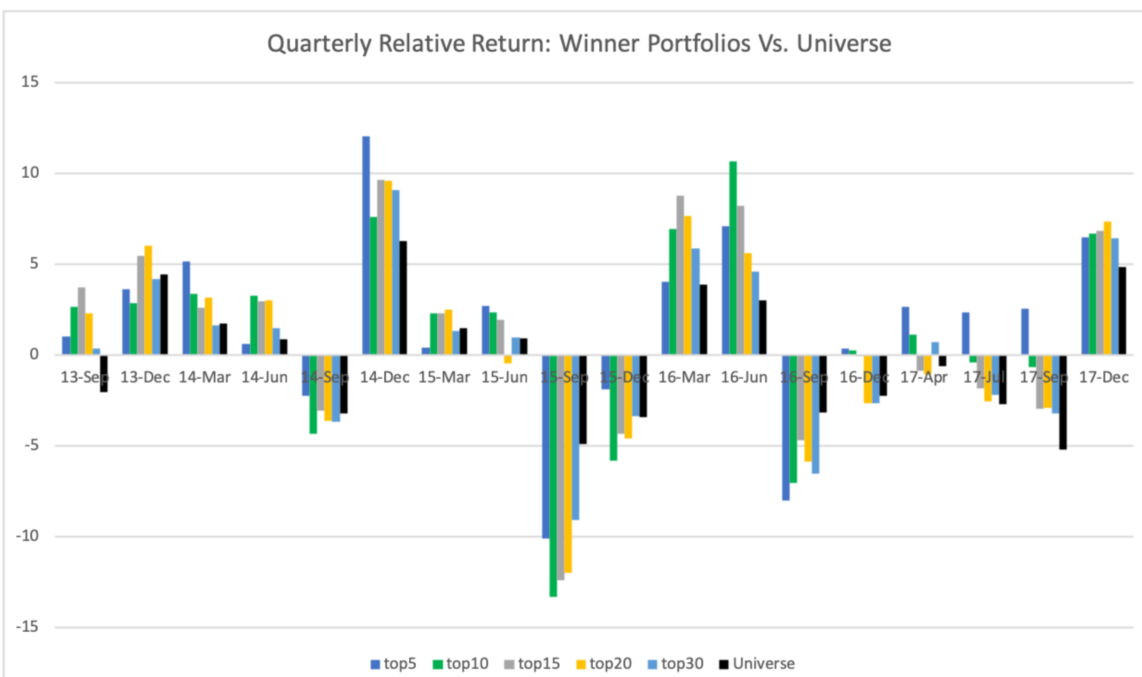

Figure 6.3: Relative return of ANFIS "Buy" portfolios for test quarters

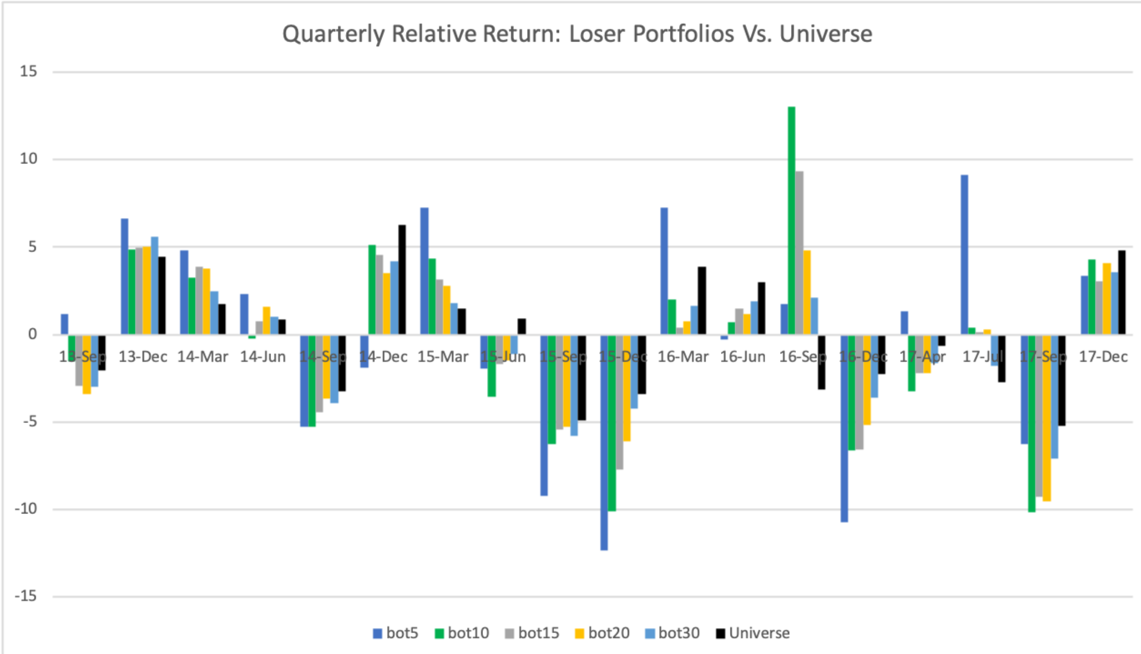

Figure 6.4: Relative return of ANFIS “Sell” portfolios for test quarters

As we can see from the results presented in Table 6.4 and Table 6.5, the “Buy” and “Sell” portfolios constructed based on the ANFIS predictions outperform and underperform the universe respectively. For the “Buy” portfolios, the “Top5” portfolio achieves the best mean quarterly relative return as well as Portfolio Score. For the “Sell” portfolios, the “Bottom5” portfolio has the lowest mean and Portfolio Score.

	Mean	STD	Portfolio Score
Top5	**1.60%**	5.15	**0.310**
Top10	1.01%	5.82	0.174
Top15	1.21%	5.66	0.215
Top20	0.621%	5.59	0.111
Top30	0.318%	4.70	0.0677
Universe	-0.0164%	**3.55**	-0.00460

Table 6.4: ANFIS results for “Buy” portfolios

	Mean	STD	Portfolio Score
Bottom5	**-0.168%**	6.45	**-0.0261**
Bottom10	-0.504%	5.99	-0.0840
Bottom15	-0.471%	4.98	-0.0947
Bottom20	-0.506%	4.29	-0.118
Bottom30	-0.440%	3.64	-0.121
Universe	-0.0164%	**3.55**	-0.00460

Table 6.5: ANFIS results for “Sell” portfolios

6.1.3 Random Forest (RF)

The RF model is developed using Python’s scikit-learn library. Again, the parameters for the RF model are tuned on the validation data with grid search. The chosen set of hyperparameters is presented in Table 6.4.

# estimaters	400
min_sample_split	5
min_sample_leaf	4
max_features	sqrt
max_depth	30

Table 6.6: RF hyperparameters

The RF model is then tested on the test data, and portfolios are constructed based on the predictions, just as with the other two models. The relative returns of the constructed portfolios for each quarter in the test period are illustrated in Figure 6.5 and Figure 6.6.

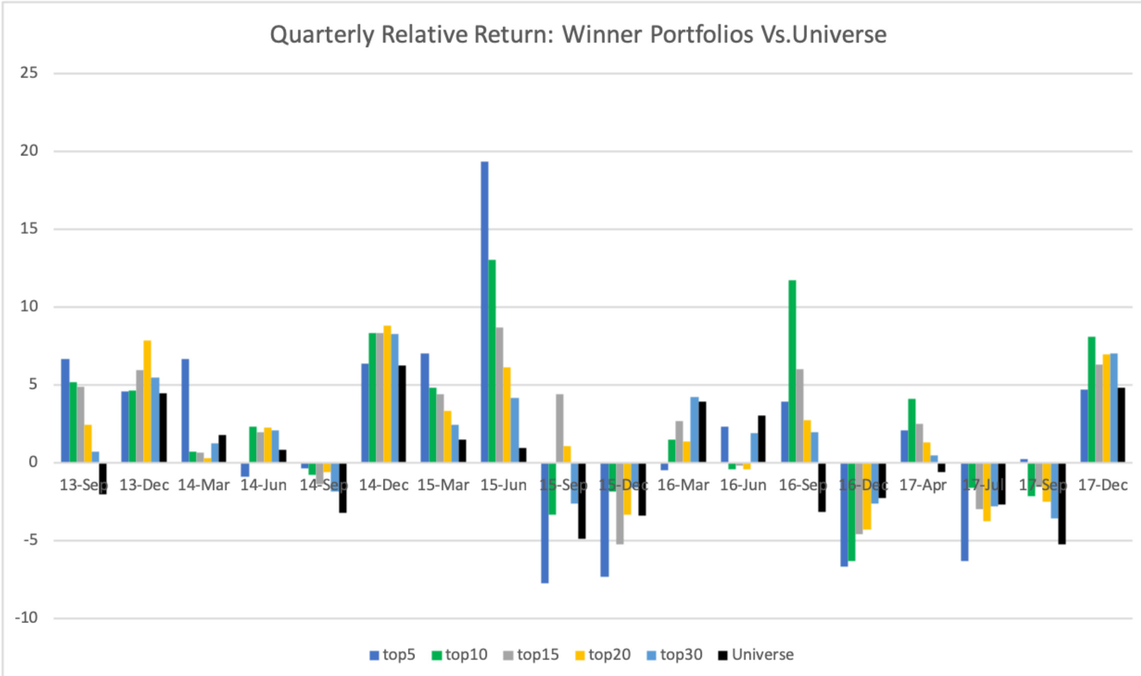

Figure 6.5: Relative return of RF “Buy” portfolios for test quarters

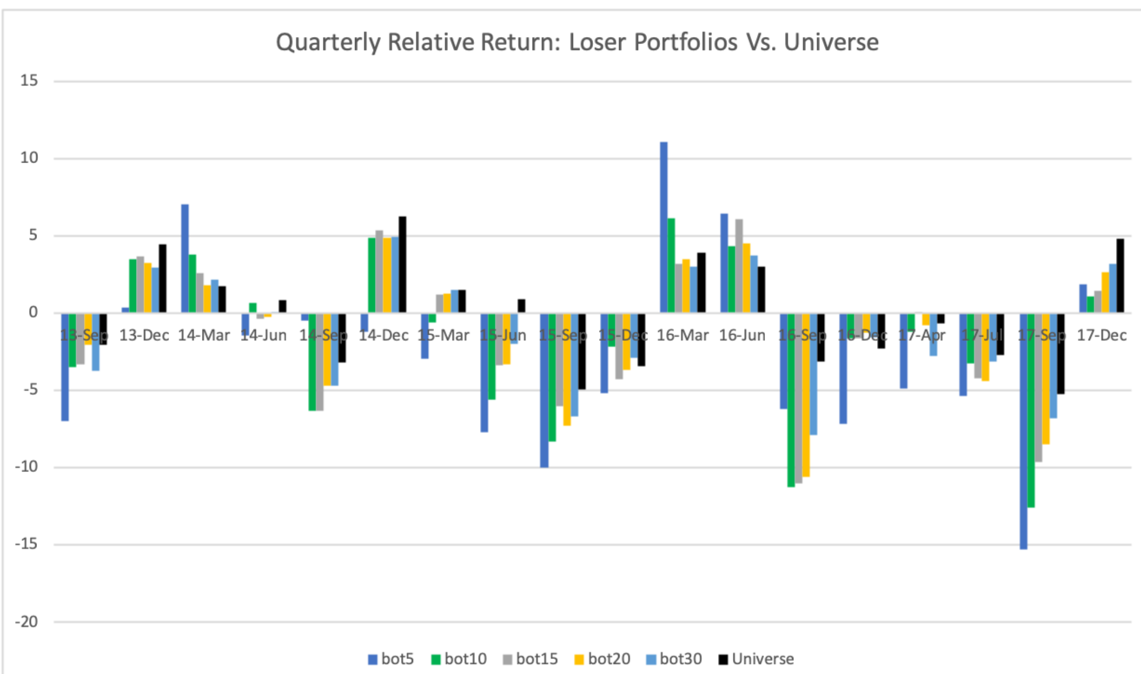

Figure 6.6: Relative return of RF “Sell” portfolios for test quarters

	Mean	STD	Portfolio Score
Top5	1.88%	6.67	0.282
Top10	**2.65%**	5.28	0.502
Top15	2.25%	4.24	**0.531**
Top20	1.63%	3.93	0.415
Top30	1.37%	**3.50**	0.390
Universe	-0.0164%	3.55	-0.00460

Table 6.7: RF results for "Buy" portfolios

	Mean	STD	Portfolio Score
Bottom5	**-2.68%**	6.47	**-0.414**
Bottom10	-1.78%	5.47	-0.325
Bottom15	-1.48%	4.92	-0.302
Bottom20	-1.39%	4.57	-0.305
Bottom30	-1.15%	4.00	-0.289
Universe	-0.0164%	**3.55**	-0.00460

Table 6.8: RF results for "Sell" portfolios

Based on the results in Table 6.7 and Table 6.8, the "Buy" and "Sell" portfolios by RF can significantly outperform and underperform the universe. The "Top10" portfolio achieved a mean quarterly relative return of 2.65%, which is the highest among all portfolios constructed based on any of the three machine learning methods. The "Top15" portfolio achieved the highest Portfolio Score among all portfolios. The "Bottom5" portfolio has the lowest mean and Portfolio Score among all portfolios across all models as well.

6.1.4 Overall Analysis

Finally, we compare the experimental results for different machine learning algorithms. The "Top20" and the "Bottom20" portfolios are used for cross-model comparison, because they represent roughly the top one third and the bottom one-third of the universe. Moreover, the compounded relative return over the test period of 18 quarters is calculated for each method as an additional metric. The results are presented in Table 6.9 and Table 6.10.

	Mean	STD	Portfolio Score	Compound
FNN	0.831%	4.11	0.202	14.4%
ANFIS	0.621%	5.59	0.111	8.85%
RF	**1.63%**	3.93	**0.414**	**32.1%**
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.9: Baseline model results for "Top20 Buy" portfolios

	Mean	STD	Portfolio Score	Compound
FNN	-0.768%	4.22	-0.182	-14.3%
ANFIS	-0.506%	4.29	-0.118	-10.2%
RF	**-1.39%**	4.57	**-0.305**	**-23.7%**
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.10: Baseline model results for "Bottom20 Sell" portfolios

The observations based on the experimental results are as follows:

1. All "Buy" portfolios outperform the universe in terms of average quarterly relative return, Portfolio Score and compound relative return by a significant margin. On the other hand, all "Sell" portfolios underperform the universe in terms of the same metrics. Therefore, we can posit a safe conclusion that all three

supervised learning models are able to predict, with a good degree of accuracy, the near-term winners and losers from a universe of stocks based on the stocks' most recent fundamental financial ratios. The results obtained challenge both the weak and the semi-strong form of EMH.

2. The RF outperforms other models in constructing both "Buy" and "Sell" portfolios in terms of all evaluation metrics by a significant margin, with the exception of standard deviation of its "Sell" portfolio. The "Buy" portfolio constructed by RF achieved a mean quarterly relative return of 1.63%, compared with the mean quarterly relative return of the universe at -0.0164%. The compound relative return outperforms the universe by 33.5% over the test period of 18 quarters or four and half years.
3. The ANFIS underperforms other models. The result could be because of the huge number of parameters to be tuned during the training process of the ANFIS model and the limited volume of training data. For instance, for a fuzzy inference system with 10 inputs, each with two membership functions, the ANFIS could generate 1024 (=2^10) rules. In our case, we have 21 inputs. More training data or fewer input features could potentially improve the prediction performance of ANFIS.
4. The standard deviations of quarterly relative returns for the selected portfolios are higher than that of the universe. This means the selected portfolios are more volatile than the benchmark. This is expected as the smaller number of stocks in a portfolio naturally leads to higher volatility.
5. All models seem to be better at identifying winner than identifying losers by a small margin. More investigation is required to find the reasons behind such a phenomenon.

6.2 Applying Feature Selection

The RF regressor is applied on the test data of all stocks for feature selection. The feature importance, as well as its standard deviation, are calculated for each feature. The features are then ranked according to their feature importance, as illustrated in Figure 6.7. The red bars represent feature importance and the black lines represent the standard deviation.

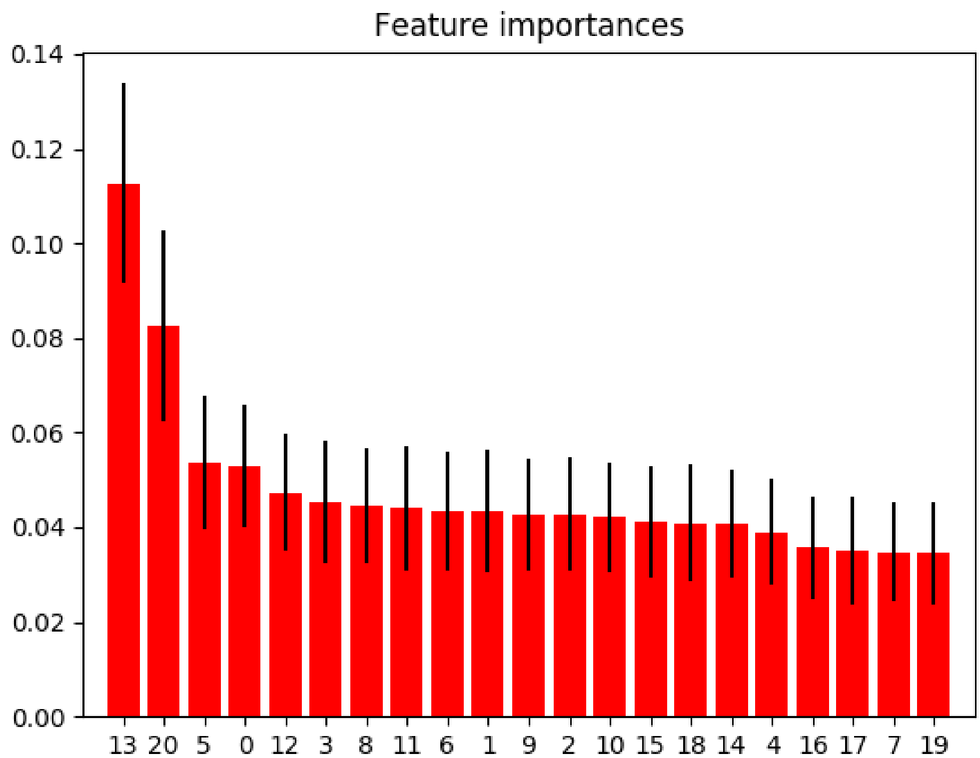

Figure 6.7: Feature importance based on RF

No.	Feature Name	Feature Description
13	ΔPB	% change of price per book
20	$Relative\_return$	Past quarter relative return on price
5	$\Delta Book\_value$	% change of book value
0	ΔPE	% change of price per earning
12	$\Delta Capital\_Exp$	% change of capital expenditure
3	$\Delta Liabilities$	% change of total liabilities

Table 6.11: Top six features selected by RF

As we can see from Figure 6.7, the top 2 features have significantly higher feature importance than the rest of the features, while rest of the features have similar feature importance. The primary reason for applying feature selection is to reduce model complicity of FNN and ANFIS models and mitigate potential overfitting. Using only the top 2 most important features could reduce model complicity significantly. However, we would also face the consequence of significant loss of information if we drop all other features. We decide to use the top 6 most important features for experiment as a balance between model complicity and information loss, although the best number of features to select could be further justified through experimentation in the future. The 6 selected features are illustrated in Table 6.11.

The FNN, ANFIS and RF models are validated and tested following the same procedure with the selected features. The test results are presented in Table 6.12 and Table 6.13.

	Mean	STD	Portfolio Score	Compound
FNN+FS	0.990%	3.62	0.273	18.1%
ANFIS+FS	0.647%	4.06	0.159	10.8%
RF+FS	**1.14%**	3.66	**0.310**	**21.2%**
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.12: Results for “Top20 Buy” portfolio with selected features

	Mean	STD	Portfolio Score	Compound
FNN+FS	**-1.45%**	4.80	**-0.302**	**-24.6%**
ANFIS+FS	-0.232%	3.99	-0.0582	-5.41%
RF+FS	-0.877%	4.32	-0.203	-16.0%
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.13: Results for “Bottom20 Sell” portfolio with selected features

The observations based on the experimental results are as follows:

1. Feature selection improves the prediction performance of FNN significantly. The Portfolio Score of the "Buy" portfolio improves from 0.202 to 0.274, and the Portfolio Score of the "Sell" portfolio improves from -0.182 to -0.302.
2. The Portfolio Score of the "Buy" portfolio produced by ANFIS improves from 0.111 to 0.159 with feature selection, while the "Sell" portfolio does not see an improvement.
3. Feature selection does not help to improve the performance of the RF model. In fact, the performance of RF is worsened with selected features.

6.3 Bootstrap Aggregation

In order to further improve prediction accuracy and stability, the predictions of the best performing models are aggregated using bootstrap aggregation. We decided to use the FNN and ANFIS models with feature selection and RF model without feature selection for final aggregation, because of the fact that feature selection does not improve the performance of RF. We tested two aggregation strategies: "agg2" and "agg3". In "agg2", for a stock to be selected into the "Buy" portfolio, there had to be at least 2 out of the 3 models that ranked the stock in the "Top20" for the quarter. In "agg3", all 3 models had to rank a stock in the "Top20" in order for the stock to be selected. Such aggregation naturally leads to fewer stocks being selected into the "Buy" portfolio for each quarter. For example, there could only be 10 stocks, which are in "Top20" portfolios for all three models, in the "Buy" portfolio for "agg3" in a test quarter. The smaller number of stocks in the aggregated portfolios could lead to higher volatility or standard deviation of performances over the test period. The relative returns of "agg2" and "agg3" for every test quarter are illustrated in Figure 6.8 and Figure 6.9. We can clearly see that "agg3" performs much better than the universe as well as "agg2" from the figures.

As we can see from the results presented in Table 6.14 and Table 6.15, "agg3" outperforms all individual models, as well as "agg2", for constructing both "Buy" and "Sell" portfolios. The "Buy" portfolio constructed by "agg3" achieves a mean quarterly relative return of 5.11%, a Portfolio Score of 0.759 and an impressive compounded relative return of 137% over the course of 18 test quarters.

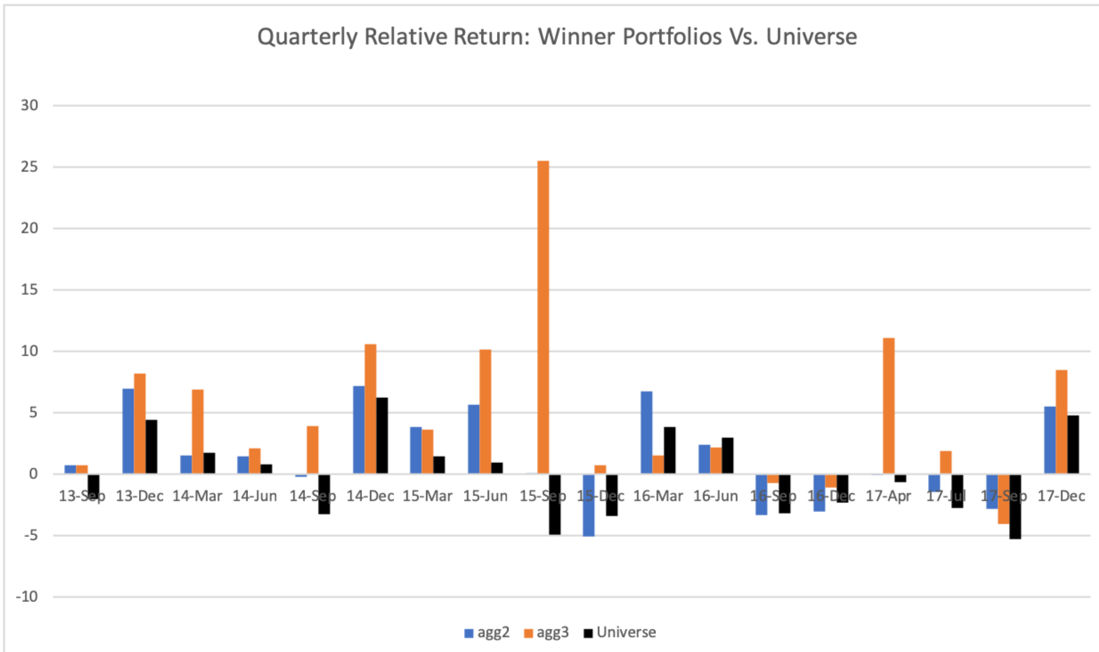

Figure 6.8: Relative return of aggregated “Buy” portfolios

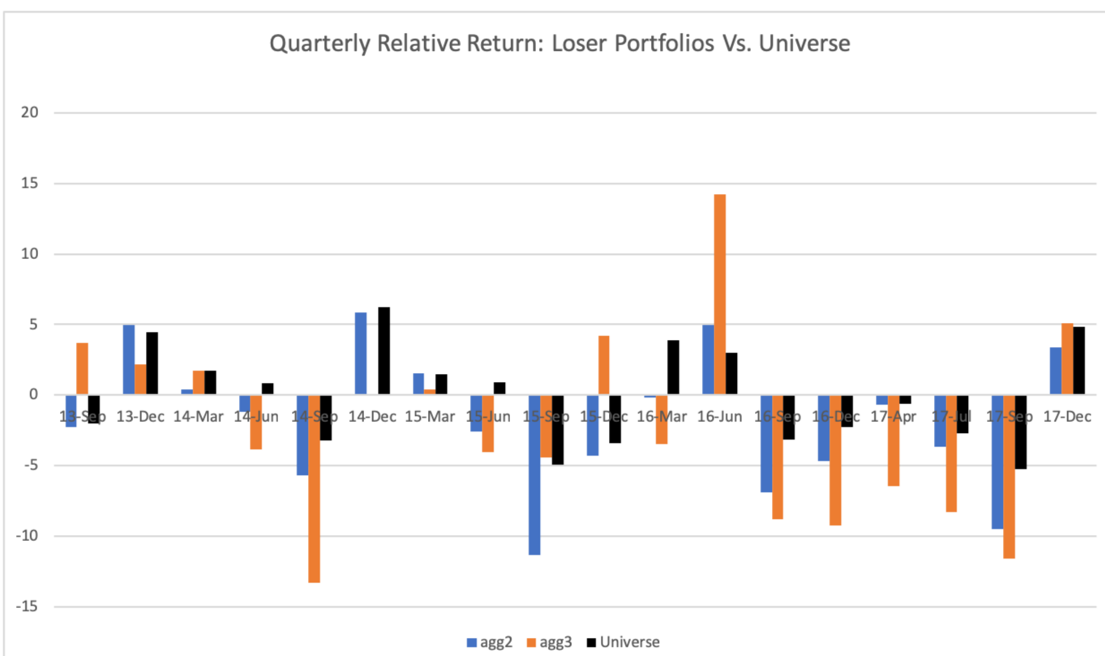

Figure 6.9: Relative return of aggregated “Sell” portfolios

	Mean	STD	Portfolio Score	Compound
FNN+FS	0.990%	3.62	0.274	18.1%
ANFIS+FS	0.647%	4.06	0.159	10.8%
RF	1.63%	3.93	0.414	32.1%
Agg2	1.45%	3.85	0.385	28.0%
Agg3	**5.11%**	6.73	**0.759**	**137%**
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.14: Results for “Buy” portfolios

	Mean	STD	Portfolio Score	Compound
FNN+FS	-1.45%	4.80	-0.301	-24.6%
ANFIS+FS	-0.232%	3.99	-0.0582	-5.41%
RF	-1.392%	4.57	-0.305	-23.7%
Agg2	-1.77%	4.89	**-0.362**	-29.0%
Agg3	**-2.32%**	6.93	-0.335	**-37.2%**
Universe	-0.0164%	**3.55**	-0.00460	-1.35%

Table 6.15: Results for “Sell” portfolios

6.4 Threats to Validity

The experiments in this research were designed only to test the performance of machine learning models based on selected fundamental data for stock appreciation prediction through backtesting. The underlying assumption is that a stock’s future price could be predicted with the company’s published financial data with some degree of accuracy. More broadly, we assume the stock market is not perfectly efficient and the semi-strong form of Efficient Market Hypothesis is false. In term of measurement of risk, we did not do a comprehensive covariance analysis within portfolios, which is a more rigorous approach in the finance industry.

All in all, although the experimental results suggest that machine learning models could be used towards stock prediction and portfolio management for better return based on experimentation with historical data, it does not imply that the models could be used to generate the same return in the future.

Chapter 7

7 Conclusion and Future Work

In this thesis, we looked at the problem of predicting stock performance with machine learning methods. Although a substantial amount of research exists on this topic, very few aims to predict stocks' long-term performance based on fundamental analysis. We prepared 22 years' worth of stock financial data and experimented with three different machine learning methods for long term stock performance prediction. In addition, we applied feature selection and bootstrap aggregation in order to improve the prediction performance and stability.

To produce effective and reliable models, we faced two major challenges. The first challenge was to put together a sizable dataset for experimenting. Due to the fact that publicly traded companies only publish their financial data on a quarterly basis and the relatively short history of digitally archiving these data, we did not have as much data as we wanted to work with. We extracted as much data as we could for 70 large-cap stocks which are S&P 100 components. The original dataset consisted of a large number of missing values, and we went through a series of data preprocessing steps in order to prepare the data for model training and testing, as discussed in Chapter 4. We experimented with building one model for all stock and building one model for each stock, and we decided on using the second approach for all algorithms based on early experimental results. The second challenge involves market efficiency, which places a theoretical limit on how historical patterns in the stock market could be used for predicting its future behavior. We took several measures to deal with this challenge. Firstly, we carefully split our data into training, validation and testing sets and made sure that we did not accidentally snoop the test data or overfit the models. Secondly, we used the Portfolio Score as our primary validation and evaluation metric. The Portfolio Score takes into account not only the performance of the constructed portfolio, but also its standard deviation over the validation period. Finally, we also employed the feature selection technique in order to remove unreliable features and reduce model complicity.

The experimental results we presented in Chapter 6 show that all three machine learning methods we experimented with are capable of constructing stock portfolios which outperform the market without any input of expert knowledge, if fed with enough data. Out of the three algorithms, RF achieves the best performance with a Portfolio Score of 0.414 and -0.305 for its “Buy” and “Sell” portfolios respectively. By applying feature selection and aggregating the different algorithms, our aggregated model achieves a Portfolio Score of 0.759 and -0.335 for the “Buy” and “Sell” portfolios respectively. This shows that our model could help to build portfolios which outperform the benchmark using historical financial data. However, as mentioned in section 6.4, discretion for applying the model in real stock market setting is advised.

7.1 Future Work

Due to the limited time and resources applied to this project, we cannot claim that we have built a perfect “crystal ball” in this thesis research, even assuming the thin possibility of the existence of such a model. There are many limitations to this research, and there are many ways this topic could be further explored. First of all, the prediction performances of the models used in this research are largely restricted by the limited volume of available data due to the fact that companies only publish their financial data on a quarterly basis. More data could potentially improve model performance as well as conclusiveness of our results. We used simple standard deviation of portfolio returns as a factor to measure risk. More rigorous stock covariance matrix analysis could be applied. More algorithms could be tested, such as different variations of neural network. Validation mechanisms such as cross validation, sliding window and expanding window could be applied to improve the model validation process and potentially improve model generalizability and robustness. Moreover, different feature selection methods could be explored, such as iterative feature selection. The number of features selected after feature selection is rather arbitrary. We could experiment with different number of most important features to further improve the effectiveness of our feature selection. We could also try to incorporate technical analysis and sentiment analysis in our model.

8 Bibliography

[1] J. Kuepper, "Technical Analysis: Fundamental Vs. Technical Analysis," Investopedia, [Online]. Available: https://www.investopedia.com/university/technical/techanalysis2.asp. [Accessed 25 April 2019].

[2] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques - Part I: Conventional methods," in *Computation Optimization in Economics and Finance Research Compendium*, New York, Nova Science Publishers, 2013, pp. 49-104.

[3] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques – Part II: Soft computing methods," *Expert Systems with Applications,* vol. 36, no. 3, pp. 5932-5941, 2009.

[4] Y. Hu, K. Liu, X. Zhang, L. Su, E. Ngai and M. Liu, "Application of evolutionary computation for rule discovery in stock algorithmic trading: A literature review," *Applied Soft Computing,* vol. 36, pp. 534-551, 2015.

[5] J. A. Brogaard, "High frequency trading and its impact on market quality," Northwestern University Kellogg School of Management, Northwestern University School of Law, 2010.

[6] J. Chen, "Investopedia-Universe of Securities," 31 May 2018. [Online]. Available: https://www.investopedia.com/terms/u/universeofsecurities.asp. [Accessed 10 Feb 2019].

[7] J. Chen, "Investopedia-Absolute Return," 16 Mar 2018. [Online]. Available: https://www.investopedia.com/terms/a/absolutereturn.asp. [Accessed 10 Feb 2019].

[8] J. Chen, "Investopedia-Relative Return," 14 Dec 2017. [Online]. Available: https://www.investopedia.com/terms/r/relativereturn.asp. [Accessed 10 Feb 2019].

[9] J. Chen, "Investopedia-Backtesting," 17 Jul 2018. [Online]. Available: https://www.investopedia.com/terms/b/backtesting.asp. [Accessed 10 Feb 2019].

[10] M. J. John, Technical analysis of the financial markets: A comprehensive guide to trading methods and applications, Penguin, 1999.

[11] P. Samuelson, "Proof that properly anticipated prices fluctuate randomly," *Industrial Management Review,* vol. 6, no. 2, 1965.

[12] E. F. Fama, "The behavior of stock-market prices," *The Journal of Business,* vol. 38, no. 1, pp. 34-105, 1965.

[13] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, 1998.

[14] J.-S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665-685, 1993.

[15] B. Graham and D. Dood, Security Analysis, McGraw-Hill Book Co., 1934.

[16] J. D. Piotroski, "Value investing: the use of historical financial statement information to separate winners from losers," *Journal of Accounting Research,* vol. 38, pp. 1-41, 2000.

[17] P. S. Mohanram, "Separating winners from losers among low book-to-market stocks using financial statement analysis," *Review of Accounting Studies*, vol. 10, no. 2-3, pp. 133-170, 2005.

[18] T. Kimoto, K. Asakawa, M. Yoda and M. Takeoka, "Stock market prediction system with modular neural networks," in *IEEE, IJCNN International Joint Conference on Neural Networks*, 1990.

[19] J. Patel, S. Shah, P. Thakkar and K. Kotecha, "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques," *Expert Systems with Applications,* vol. 42, no. 1, pp. 259-268, 2015.

[20] J. Patel, S. Shah, P. Thakkar and K. Kotecha, "Predicting stock market index using fusion of machine learning techniques," *Expert Systems with Applications,* vol. 42, no. 4, pp. 2162-2172, 2015.

[21] E. Chong, C. Han and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Systems with Applications,* vol. 83, pp. 187-205, 2017.

[22] S. D. Bekiros and D. A. Georgoutsos, "Evaluating direction-of-change forecasting: Neurofuzzy models vs. neural networks," *Mathematical and Computer Modelling,* vol. 46, no. 1-2, pp. 38-46, 2007.

[23] G. S. Atsalakis and K. P. Valavanis, "Forecasting stock market short-term trends using a neuro-fuzzy based methodology," *Expert Systems with Applications,* vol. 36, no. 7, pp. 10696-10707, 2009.

[24] C.-C. Wei, T.-T. Chen and S.-J. Lee, "A k-NN Based Neuro-Fuzzy System for Time Series Prediction," in *14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2013.

[25] T.-S. Quah and B. Srinivasan, "Improving returns on stock investment through neural network selection," *Expert Systems with Applications,* vol. 17, pp. 295-301, 1999.

[26] M. Lam, "Neural network techniques for financial performance prediction: integrating fundamental and technical analysis," *Decision Support Systems,* vol. 37, no. 4, pp. 567-581, 2004.

[27] S. G. Eakins and S. R. Stansell, "Can value-based stock selection criteria yield superior risk-adjusted returns: an application of neural networks," *International Review of Financial Analysis,* vol. 12, no. 1, pp. 83-97, 2003.

[28] T.-S. Quah, "DJIA stock selection assisted by neural network," *Expert Systems with Applications,* vol. 35, no. 1-2, pp. 50-58, 2008.

[29] K.-Y. Shen and G.-H. Tzeng, "Combined soft computing model for value stock

selection based on fundamental analysis," *Applied Soft Computing*, vol. 37, pp. 142-155, 2015.

[30] C. Hargreaves and Y. Hao, "Does the use of technical & fundamental analysis improve stock choice? : a data mining approach applied to the australian stock market," in *2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE)*, 2012.

[31] A. Namdari and Z. Li, "Integrating fundamental and technical analysis of stock market through multi-layer perceptron," in *2018 IEEE Technology and Engineering Management Conference (TEMSCON)*, 2018.

[32] T. Bohn, "Improving Long Term Stock Market Prediction with Text Analysis(Master Thesis)," 2017. [Online]. Available: https://ir.lib.uwo.ca/etd/4497.

[33] L. Yu, L. Hu and L. Tang, "Stock selection with a novel sigmoid-based mixed discrete-continuous differential evolution algorithm.," *IEEE Transactions on Knowledge and Data Engineering,* vol. 28, no. 7, pp. 1891-1904, 2016.

[34] "S&P 100 Fact Sheet," 30 Nov 2018. [Online]. Available: https://us.spindices.com/indices/equity/sp-100. [Accessed 10 Dec 2018].

[35] "Stockpup," [Online]. Available: http://www.stockpup.com . [Accessed 1 July 2018].

[36] J. W. Graham, "Missing data analysis: making it work in the real world," *Annual Review of Psychology,* vol. 60, pp. 549-576, 2009.

[37] A. C. Acock, "Working with missing values," *Journal of Marriage and Family,* vol. 4, no. 67, pp. 1012-1028, 2005.

[38] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Computation,* vol. 4, no. 6, pp. 888-900, 1992.

[39] W. Anani and S. Jagath, "Comparison of recurrent neural network algorithms for intrusion detection based on predicting packet sequences.," in *IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, 2018.

[40] Y.-W. Chen and C.-J. Lin, "Combining SVMs with various feature selection strategies.," in *Feature Extraction*, Springer, 2006, pp. 315-324.

Appendices

Appendix A: Original features

Quarter end	Shares	Shares split adjusted	Current ratio	Assets
Current Assets	Liabilities	Current Liabilities	Shareholders equity	Non-controlling interest
Preferred equity	Goodwill & intangibles	Long-term debt	Revenue	Earnings
Earnings available for common stockholders	EPS basic	EPS diluted	Dividend per share	Cash from operating activities
Cash from investing activities	Cash from financing activities	Cash change during period	Cash at end of period	Capital expenditures
Price	Price high	Price low	ROE	ROA
Book value per share	P/B ratio	P/E ratio	Cumulative dividends per share	Dividend payout ratio
Long-term debt to equity ratio	Equity to assets ratio	Net margin	Asset turnover	Free cash flow per share

Curriculum Vitae

Name:	Yuxuan Huang
Post-secondary Education and Degrees:	Bucknell University Lewisburg, PA, USA 2013-2017 B.Sc.
	University of Western Ontario London, Ontario, Canada 2017-2019 M.E.Sc.
Related Work Experience	Teaching Assistant and Research Assistant University of Western Ontario 2017-2019

Publications:

Yuxuan Huang, Luiz F. Capretz and Danny Ho, *"Neural Network Models for Stock Selection Based on Fundamental Analysis"*, IEEE 32nd Canadian Conference on Electrical and Computer Engineering (CCECE), accepted May, 2019.