COMP4801 Final Year Project
Financial Data Forecaster
Final Report

University of Hong Kong

Department of Computer Science

Student: Sumer Malhotra (3035347457)

Supervisor: Dr.Yip Beta

Submitted on: 3rd May 2020

Abstract

An investor's objective in the stock market, is to make profit based on calculated speculation. No investment strategy is the best, but some strategies have historically proven to be more effective than others. Over the last couple of years, Quantitative investing has become extremely prevalent in financial markets and institutions. This project aims at comparatively developing and testing various machine learning classification models on stock prices. The models will accept various fundamentals, return statistics and sentiment as input, offering predicted change in price direction over a 3 month time-frame as output.

Acknowledgments

I would like to express my utmost gratitude to my supervisor Dr. Yip Beta from the Department of Computer Science for his continued guidance through this project. I would also like to thank Julian Chase and Ken Ho from the Center for Applied English Studies for their suggestions on drafting this paper.

Abbreviations

ML: Machine Learning

ARIMA: Autoregressive Integrated Moving Average

SVM: Support Vector Machine

LR: Logistic Regression

LDA: Linear Discriminant Analysis

KNN: K-Nearest Neighbors

AUC: Area Under the ROC Curve

ROC: Receiver Operating Characteristics

GARCH: Generalized Autoregressive Conditional Heteroscedasticity

NLP: Natural Language Processing

MSE: Mean Squared Error

CART: Classification and Regression Tree

P/E: Price to Earnings

P/B: Price to Book

EPS: Earnings per Share

D/E: Net Debt to Equity Ratio

EV: Enterprise Value

EBIDTA: Earnings before Interest, Taxes, Depreciation, and Amortization

EBIT: Earnings before Interest and Taxes

ROE: Return on Equity

ROIC: Return on Invested Capital

NOPAT: Net Operating Profit after Tax

Tables & Figures

Tables

Figures

Contents

1. Introduction

1.1 Background

Over the last couple of years, Quantitative investing has become extremely prevalent in the world of financial markets and institutions, which can be seen by the mass adoption of the techniques involved with this type of investing by major hedge funds and investment banks. This type of technology has become staple in these institutions due to the ability of computers to analyze large amounts of data with ease. To give some perspective on this technique, we can look at some statistics related to it. Approximately 90% of public market trading volume in the United States is executed using quantitative analysis. Furthermore, this market has a capitalization of roughly $1 trillion dollars. Finally, around 6 of the top 10 performing hedge funds in the world majorly employ quantitative techniques in their investment process. Quantitative analysis is a process used to understand market behavior by analyzing a large number of variables including asset prices, trading volume, technical indicators, fundamental indicators, market sentiment, day and time, etc. and how these variables may be correlated with each other. Following

this, a mathematical model is developed which assumes these variables as input and outputs either a predicted direction in which the market may move or a signal (buy or sell) which serves as the building blocks of a trading strategy. A fundamental rule of economics is that when supply overwhelms demand, prices decline. This unfortunate rule serves as a potential drawback for quantitative analysis. According to a CNBC report from July, 2018, less than 40% of institutional managers have outperformed their benchmarks [1]. This figure may be overestimated due to the skewness from the inclusion of both qualitative and quantitative managers in the statistic. However the point is obvious, majority of quant managers no longer provide superior returns for their clients. In financial jargon, this term is called crowding, resulting from a rapid inflow of capital into these strategies in recent years which can be seen from an increasing adoption of these techniques by not only institutional managers but also individual investors. This phenomenon has significantly decreased prediction accuracy. Thus, this project aims at capitalizing on current inefficiencies.

1.2 Objective

This project aims at developing various machine learning classification algorithms to classify a stock as a good or bad investment opportunity. The algorithms will take ticker codes as input and use that information to generate the required market data, technical and fundamental indicators and market sentiment. To produce a satisfactory deliverable, a non-biased and consistent prediction evaluation technique has been devised. This project will bring to light, performance of the algorithms and how to go about fine tuning, optimizing and finally, improving said algorithms. A deeper investigation into the solution will reveal optimal conditions leading to such success. This insight can help with hyper-parameter tuning in future research, understanding advantages and limitations of various models and the impact of investment time-horizon on prediction accuracy. Apart from just quantitative objectives, another key take-away could be qualitative insights into investing. Market Psychology of investors, prediction of company growth, bankruptcy, market bubbles, inflated asset prices, impact of financing decisions and importance of accounting variables are factors prevalent within qualitative investing that can be inferred during the development and testing of this project. For the actual time period, we will consider data from the beginning of 2015 up until end March of 2020. This will lead to interesting insights about algorithm performance amidst the current coronavirus pandemic which has sent the market into a spiral.

2. Literature Review

2.1 Previous Works

2.1.1 ARIMA-SVM

The first approach to performing financial data forecasting is through time series forecasting. The most commonly used method within time series forecasting is ARIMA. The results from ARIMA can further be improved by applying a classification layer on top of it, so as to appropriately classify errors and fundamental price movements. This can be done through the application of various ML models. An example of which is the ARIMA-SVM model hypothesized by Tian Ye. First, wavelet analysis is used to decompose stock price into a reconstructed and error part. After this, ARIMA is used to forecast the reconstructed part, while SVM is used to forecast the error part. ARIMA is a time-series forecasting model which is trained with data in order to forecast future values. For example, the ARIMA model that Tian Ye used, forecasted price movements for a stock based on the past price movements. ARIMA model is built up from AR and MA models, namely

auto-regressive and moving-average models. The AR model is formed when regression is performed between a variable and a lagging version of itself. The MA model is obtained when a weighted combination of previous partials is taken. I is an acronym for "Integrated" and is a consequence of replacing current values with the difference of current and past values. This process can be performed multiple times in order to better fit the data to the model. ARIMA was chosen as it the most commonly used and most accurate forecasting model for stock prices [2]. The idea of SVM is to find an optimal hyper-plane such that, the data is separable into 2 classes. Support Vector Regression is a unification of the ideas of Linear regression and SVM. When the error falls within the acceptable range of errors, $f(x) + \varepsilon$ and $f(x) - \varepsilon$, it is classified as statistical error/noise and hence, ignored. Else, it is accounted for as a fundamental movement in the price and accordingly processed. ARIMA, when used in coalition with SVR is a highly efficient non-linear prediction model. As stock prices have strong non-linear characteristics, ARIMA or SVM alone will lead to greater error as demonstrated by Tian Ye's results [3]. MSE for a model is a statistic used to measure the average of the squares of the errors, which gives intuition about model performance. The higher the MSE, the lower the accuracy. From Table 2.1, it is evident that ARIMA-SVM has the lowest MSE and is hence, the most accurate. Figure 2.1 visually depicts the advantage of using ARIMA-SVM over each of it's individual constituents. From the figure, it is evident that the ARIMA-SVM predictive value most closely resembles the actual closing price.

Model	MSE
ARIMA	1.71
SVM	6.51
ARIMA-SVM	0.57

Table 2.1: MSE of Three Models

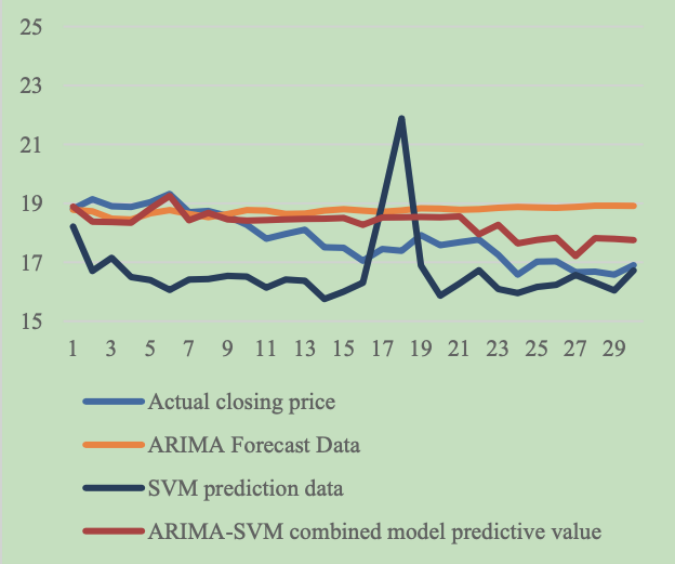


Figure 2.1: Comparison of the original and predicted data

Another author, Majumder [4] has used various models in forecasting Bangladeshi stock indices. Among the applied models, ARIMA model and Feed-Forward Neural Networks have shown the most promising results. However, Majumder's application of ARIMA(1, 0, 0) resulted in the best accuracy for Bangladeshi stock indices. On the contrary, application of ARIMA(5, 1, 5) by Tian Ye re. sulted in the most promising results. This implies that different financial products in different markets perform differently and therefore, have to be tested with not only various models, but also various parameters within each model.

2.1.2 LR, LDA and kNN

The second approach to performing financial data forecasting is through the application of machine learning or statistical based classification methods. These classification methods can be used on a number of dependent variables (The variable which is being predicted). For example, we can classify an investment opportunity as good or bad based on whether it outperforms a particular index or simply whether the return will be positive. The authors, Phongmekin and Jarumaneeroj have looked at both of these dependent variables during the testing, development and implementation of their model. They incorporated market sentiment from sources like Twitter into LR, LDA and kNN based ML models to classify directional changes in stock price. The results indicated that the incorporation of market sentiment into the prediction algorithm increased accuracy to above 70% [5]. When a dependant variable is binary (Good or Bad stock), LR is used as it makes no assumptions of the distribution or variance of input features. LDA however, assumes that variables are normally distributed and that each feature has the same or similar variance [6]. This idea is not fatal however and should still be tested. kNN is another ML algorithm that can be used for both classification and regression problems. To address the problem at hand, the author has chosen to proceed with a classification implementation of the kNN algorithm. The algorithm classifies data into 2 groups based on the minimum euclidean distance between a point and each of the 2 groups. This is intuitive as the algorithm assumes that similar points will exist in close proximity to each other [7]. However, since the algorithm is based on calculating absolute distance, regularization of the data is

required, as features are likely to have different scales. In their studies, rather than using MSE, they use AUC to test model performace. As MSE only takes error into account (false positives), it can be improved by considering true positives as well, which is what AUC does. Models with higher AUCs, have higher ROC, implying that on average they classify more true and false positives correctly. Accordingly, LR is more accurate than LDA, and kNN is the most accurate among the 3 models as seen from the AUC scores in Table 2.2.

Classifier	**AUC (Neutral)**	**AUC (Optimistic)**	**AUC (Pessimistic)**
kNN	0.759	0.816	0.702
LR	0.744	-	-
LDA	0.743	-	-

Table 2.2: AUC's of Models [5]

From Table 2.2, we observe AUC (Optimistic) and AUC (Pessimistic) values for kNN. This is a consequence of kNN not having an estimated probability function like LR and LDA. Optimistic scores are calculated when true positives are counted before false positive instances and the contrary is true for Pessimistic scores. The neutral score for kNN is calculated as an average of it's Optimistic and Pessimistic score, while the neutral AUC scores for LR and LDA are calculated from their estimated probability functions [8].

2.2 Feature Selection Review

In addition to the choice of models, features affect the accuracy of an ML classifier.

2.2.1 ARIMA-SVM

Tian Ye's choice of time-series forecasting on stock price compelled him to choose closing price as the only independent variable.

2.2.2 LR, LDA and kNN

Phongmekin and Jarumaneeroj use 43 variables in their classification model. These variables include a mixture of financial ratios and industry/sub-industry classification. The selection of the financial ratios by the authors have been heavily biased by past research. Consequently, popular ratios in fundamental analysis like Price-to-Book (PB), Return on Equity (ROE), Price-to-Earnings (PE), etc. have been used [5].

2.3 Limitations of Previous Research

2.3.1 ARIMA-SVM

The author suggests that using a classification model in combination with a time series forecasting model has greatly improved accuracy as presented in the Previous Works subsection. The inclusion of certain short term momentum indicators in the error forecasting model however, may increase accuracy.

2.3.2 LR, LDA and kNN

Phongmekin and Jarumaneeroj indicate that further research should be done on building similar models for other sectors, as their model focused on the finance sector on the Stock Exchange of Thailand (SET). Furthermore, they suggest that the incorporation of variables like sentiment analysis can improve accuracy. They also propose using particle swarm optimization to increase accuracy of the AUC of the models. Lastly, they recommend creating a generalized model that can incorporate all sectors of the market, as it will be more generally applicable [5].

3. Methodology

3.1 Overview

The project is carried out exclusively using Python 3.7.4 using Jupyter notebooks as an interface to allow continuous testing so as to spot and accordingly, fix bugs. The project aims at developing an ML classification based approach to quantify investment opportunities as good or bad. The project follows by first performing large scale data retrieval using traditional scraping and parsing methods. This data is then converted into a usable format on which classification algorithms can be applied. Finally, various classification algorithms are applied on the dataset and the results are observed. Accordingly, the most accurate algorithm is chosen. Some algorithms are particularly useful in understanding feature importance – particularly the graphs generated from CART.

3.2 Importing Libraries

For the purpose of carrying out the tasks required in completing the project, relevant libraries have been imported and used. Datetime library is used for converting string dates into dates of type datetime. This is useful as it allows the user to perform direct mathematical operations such as additions and subtractions on date. Re library is useful as it allows the user to use regular expression operations on strings. This is particularly useful during the filtering of tweets under certain conditions. Time library is used to compute how long a code cell takes to run. This was useful as it showed the performance gains available when using different webdrivers in scraping data. For example, the Chrome webdriver performs a task 3x faster on average than a Firefox webdriver. The string library is used for string manipulations, specifically in the tweet filtering portion of the program. The requests library is used for performing RESTful API calls to HTML pages and consequently, receiving HTML content, which in turn is parsed. Information is then received from these parsed pages. Warnings is used to filter out Deprecation and Future warnings to allow the code to be more presentable. Graphviz is used to generate diagrams/graphs of fit decision trees, which in turn can help us understand the independent features better. Pandas is a widely used library in machine learning. It is arguably the most used library in this project and allows the user to store data in DataFrames, an object on which data manipulation and analysis can easily be performed. Matplotlib is library which allows the user to plot data in a fashion which mimics MATLAB plotting. Numpy is a library which goes hand in hand

with pandas. It allows the user to easily perform scientific computing operations on data. GetOldTweets3 is an open source library used to scrape tweets. It is useful as it allows the user to collect thousands of tweets with a single query, allows the user to filter tweets from a start to end date – a feature not traditionally available with Tweepy, twitter's native Python library. Textblob and NLTK vader are libraries used to generate sentiment scores for a word or sentence. NLTK stopwords is a library used to filter tweets by removing stopwords from sentences. This is useful as these stopwords are traditionally sentiment-neutral and will have a negative impact on the accuracy of the true sentiment of a sentence. Beautiful Soup is a library which is used to parse HTML webpages. This is useful as it allows the user to extract meaningful and required information from the HTML page, which is often cluttered with unnecessary information like advertisements. Selenium is a library which is used to create a webdriver, an object which is key in performing webscraping. The sklearn library is extremely useful in performing tasks like data preprocessing, train-test splitting, normalization, algorithm fitting and result visualization.

3.3 Data Retrieval

3.3.1 Pricing Data and Financial Statements

The first step of the project is to retrieve data of interest. Multiple sources are available for scraping financial information, but for the purpose of this project and section, pricing data and financial statements (Income statement, Balance sheet, Cash Flow statement) are scraped from Yahoo Finance. This is carried out using Selenium webdriver and the requests library for scraping and the Beautiful Soup library for parsing. For pricing data, the data in practice should be retrievable through a simple call using requests, however since the pricing data loads dynamically on scrolling to the bottom of the page, this functionality has to be mimicked using a webdriver approach. The Chrome webdriver is used owing to its impressive speed when compared to Firefox. The time taken to scrape data for one stock is compared using a Chrome and Firefox webdriver and the results are given in Figure 3.1 and 3.2.

```
%%time

results = {}
options = ChromeOptions()
options.headless = True
driver = webdriver.Chrome(options=options)
df = pd.DataFrame()
for ticker in _tickers[:1]:
    print(f"Scraping information for {ticker[1]}...")
    results[ticker[0]] =  {
                           "Prices": getPrices(driver, ticker[0], start, end),
                           "Income": getIncomeStatement(driver, ticker[0]),
                           "Balance": getBalanceSheet(driver, ticker[0]),
                           "Cash": getCashFlow(driver, ticker[0])
                          }
    df = df.append(generateData(results[ticker[0]], ticker), ignore_index=True)
    print('Done...\n')
driver.quit()
```

```
Scraping information for PCCW...
Getting Prices...
Getting Income statement...
Getting Balance Sheet...
Getting Cash Flow...
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
```

```
/opt/anaconda3/lib/python3.7/site-packages/numpy/core/fromnumeric.py:3335: RuntimeWarning: Mean of empty slice.
  out=out, **kwargs)
/opt/anaconda3/lib/python3.7/site-packages/numpy/core/_methods.py:161: RuntimeWarning: invalid value encountered i
n double_scalars
  ret = ret.dtype.type(ret / rcount)
```

```
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Creating datapoints...
Done...

CPU times: user 2.86 s, sys: 123 ms, total: 2.98 s
Wall time: 1min 14s
```

Figure 3.1: Chrome Driver Performance

```
%%time

results = {}
options = Options()
options.headless = True
driver = webdriver.Firefox(options=options)
df = pd.DataFrame()
for ticker in _tickers[:1]:
    print(f"Scraping information for {ticker[1]}...")
    results[ticker[0]] =  {
                           "Prices": getPrices(driver, ticker[0], start, end),
                           "Income": getIncomeStatement(driver, ticker[0]),
                           "Balance": getBalanceSheet(driver, ticker[0]),
                           "Cash": getCashFlow(driver, ticker[0])
                          }
    df = df.append(generateData(results[ticker[0]], ticker), ignore_index=True)
    print('Done...\n')
driver.quit()
```

```
Scraping information for PCCW...
Getting Prices...
Getting Income statement...
Getting Balance Sheet...
Getting Cash Flow...
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
```

```
/opt/anaconda3/lib/python3.7/site-packages/numpy/core/fromnumeric.py:3335: RuntimeWarning: Mean of empty slice.
  out=out, **kwargs)
/opt/anaconda3/lib/python3.7/site-packages/numpy/core/_methods.py:161: RuntimeWarning: invalid value encountered i
n double_scalars
  ret = ret.dtype.type(ret / rcount)
```

```
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Getting Tweets...
Removing mentions, hashtags, URL links, punctuations, stopwords
Creating datapoints...
Done...

CPU times: user 2.94 s, sys: 150 ms, total: 3.09 s
Wall time: 4min 31s
```

Figure 3.2: Firefox Driver Performance

From Figure 3.1 and 3.2, it can be inferred that the Chrome driver leads to significant performance gains, performing at 3.66 times the speed of Firefox per stock.

3.3.2 Twitter Data

The second step of Data Retrieval proceeds with appropriate querying of tweets for a company. This is done by querying based of either the ticker or company name. The tweets are also filtered by an end date and max number of tweets to be retrieved for each time period. For example, PCCW would have 4 time periods - December 30 of 2016-17, 2017-18, 2018-19 and 2019-20. The tweets would appropriately be queried with December 30 of 2017, 18, 19 and 20 as the end periods. The December 30 date is inferred by processing the information presented within the financial statement for the company. Accordingly, this date can differ for different companies based on their fiscal periods.

3.4 Data Preprocessing

After scraping data related to pricing information, financial statements and twitter data, conversion of this raw information into usable data is essential. This is carried out using user-defined and pre-defined Preprocessing techniques. These techniques will be elaborated on in the following section.

3.4.1 Converting Raw to Usable Data

Pricing Data and Financial Statements

For pricing data and financial statements, data is available for each stock for a time period of 5.5 years where 1.25 years constitute a single time period. This allows a total of 4 time periods per stock, inclusive of leeway for discrepancies in fiscal year periods. Each data point consists of various metrics, but for the purpose of this section, we will examine the procedure by which we convert raw pricing data and financial statements into usable metrics. We store the data in pandas Data Frames, which are extremely convenient for performing mathematical, statistical or even filtering operations. Accordingly, we query pricing information for the concerned time period and generate return statistics for time periods of the last 1 month, 3 months, 6 months and 1 year. From the financial statements, we query appropriately from the various statements and generate the appropriate fundamental metrics, which will be explored in the variable selection section. We also generate a dependant variable using the pricing data, which is whether the return is positive over the last 3 months/0.25 years of the examined period. To summarize, the first

year of data is used to generate the independent variables while the last 0.25 years of data is used to generate the dependant variable.

Twitter Data

The twitter data received using the GetOldTweets3 library has an abundance of unusable tweets owing to varied languages present within returned tweets. Tweets are also filled with information such as retweets, usernames, hashtags, mentions, URLs, punctuation and stop words which need to be removed before generating a sentiment score. Accordingly, we define a function which helps clean each tweet. Once each tweet is cleaned, the number of words present in the tweet is checked. If it crosses the examination threshold of more than one word present in the tweet, it has passed the filtering conditions and can be further examined in constructing sentiment for the specified time period. Once, a filtered list of tweets is generated, sentiment analysis libraries like NLTK Vader and TextBlob can be applied to each tweet and accordingly generate a sentiment score. The procedure is fairly simple - apply both libraries to each tweet in the filtered list and generate 2 sentiment scores per tweet. Average the sentiment score per tweet and finally, take an average of the average sentiment score per tweet. This will lead to a single sentiment score per time period for each stock.

Data Consolidation

Once the appropriate data points for pricing and financial statements and twitter data are generated for each of the 4 time periods per stock, the columns are merged

by date using the pandas merge operation. This results in 4 complete rows, filled with both the independent and dependent variables.

3.4.2 Data Storing

The procedure mentioned in Section 3.3 and 3.4 is generalized and applied to a list of stocks, taken as input. The list is essentially a csv file, with the first column specifying the ticker code and the second column specifying the company name. The raw data is generated and converted into usable data. After which, it is appended to a pandas DataFrame which houses all the pertinent information. This procedure is iteratively repeated for each stock in the tickers list. We consider a ticker list of 25 stocks, where each stock generates 4 datapoints, leading to a complete dataframe of 100 rows. This dataframe is then converted into a csv file and stored locally. It is then reused as required, so as to avoid repeated construction of the dataset through intensive querying.

3.4.3 Splitting into Train and Test set

We use the train_test_split module within sklearn to split the data set into a training and testing set. For this project, an 80-20 train-test split has been used. The purpose of this is to allow the model to learn the optimal parameters required for predicting the dependent variable, given the independent variables. The model is then tested against the testing set, where accuracy of the model is inferred. We restrict the training of the model to just a training set as oppose to the whole dataset so as to prevent over fitting or under fitting of the model.

3.4.4 Normalization

The final step involved with preprocessing the data is normalization of columns. This is done using the MixMaxScaler module with sklearn.preprocessing. The purpose of normalization is to rescale features within the dataset to a common scale, without distorting differences in the ranges of values. Every machine learning problem will not require normalization. It should be applied on a case by case basis, where the main factor to consider is the possible ranges of the independent features. Normalization not only aids with accuracy of the algorithm but also learning speeds.

3.5 Machine Learning

This section explores the intricacies involved with the dataset, selection of variables and the models used in predicting the output.

3.5.1 Outliers

As previously stated, the dataset considers data for 25 stocks from a time period of 2015-01-01 to 2020-04-01. Consider the market situation during this time, where 2015-19 was primarily a bull market with markets across the globe performing well on average. The first 4 months of 2020 however, have shown a clear transition to a bear market. This was a consequence of the recent impact of the COVID-19 pandemic. During this period, we witnessed the S&P Index fall by around 27% [9]. This period for the 25 stocks is a considerable outlier owing to the sudden fall in performance, despite a bull market the previous year.

3.5.2 Variable Selection

The variable choice was heavily influenced on a combination of ideas adopted from an article [23] on Business Today, an Indian business newspaper and the studies done by Phongmekin and Jarumaneeroj.

This study considers 17 independent variables, from which 12 financial ratios are considered. 4 variables are concerned with return statistics and the last independent variable is the yearly averaged market sentiment of an asset. This section aims at further exploring each of the independent variables.

Asset Sentiment

The first variable being addressed is asset sentiment. This variable essentially quantifies what the overall attitude of the investors in the market is towards the chosen asset for the particular year. To establish notation, let us denote market sentiment as s. Naturally, when the current sentiment of an asset is positive (s in (0, 1]), we would expect a bull market. Conversely, when the current sentiment of an asset is negative (s in [-1, 0)), we would expect a bear market [10].

P/E ratio

P/E Ratio is the how much more expensive the share price is when compared to its EPS. It is also used in company valuation [11]. It is constructed using the average price for the year, which is computed as a mean of the closing price data for the year. EPS can be found in the income statement. The formula for P/E is:

$$\frac{Market\ value\ per\ share}{EPS}$$

P/B ratio

P/B ratio measures the valuation of a company relative to its book value [12]. Book value of a company is the total assets minus total liabilities. We calculate the price per share in a similar fashion to the methodology used for P/E ratio. Book value per share is computed using required information from the balance sheet. The

formula for P/B is:

$$\frac{Market\ value\ per\ share}{Book\ value\ per\ share}$$

D/E ratio

The D/E ratio is computed as the ratio of a company's total liabilities to it's total shareholder's equity [13]. These values are both inferred using data provided in the company's balance sheet. The formula for D/E is:

$$\frac{Total\ liabilites}{Total\ shareholders'\ equity}$$

Operating profit Margin

The operating profit margin is calculated as the relative operating profit by the net sales made by the firm [14]. It is computed using data available on a company's Income statement. The formula for D/E is:

$$\frac{Operating\ earnings}{Total\ revenue}$$

EV/EBIDTA

Similar to P/E ratio, this ratio is used to understand a company's valuation. This ratio is interesting as it considers a company's debt, resembling the way an investor would look at a company's value [15]. EV of a company is defined as the market capitalization plus total liabilities minus cash. This data can be found on the income

statement and balance sheet of a company. The formula for EV/EBIDTA is:

$$\frac{EV}{EBITDA}$$

ROE

ROE is a financial ratio used to measure financial performance and efficiency of a company [16]. It is computed as the net income made relative to shareholders' equity. These ratios can be found on the income statement and balance sheet respectively. The formula for ROE is:

$$\frac{Net\ Income}{Average\ shareholders'\ equity}$$

Interest Coverage ratio

The Interest coverage ratio is to understand how easily a company can pay off it's debt [17]. It is computed as a ratio of EBIT against the company's interest payments due for the considered period. Both of these values can be found on a company's income statement. The formula for Interest Coverage ratio is:

$$\frac{EBIT}{Interest\ expense}$$

Current ratio

The current ratio is used to measure a company's ability to pay shorter term obligations, around a 1 year period for example [18]. It is computed by dividing

a company's current assets by its current liabilities. Both of these metrics can be found on a firm's balance sheet. The formula for Current ratio is:

$$\frac{Current\ assets}{Current\ liabilities}$$

Asset Turnover

This ratio is used to measure a company's total revenues relative to its average total assets [19]. It is used to measure a company's resource efficiency. The metrics involved with this ratio can be found on the income statement and balance sheet respectively. The formula for Asset turnover is:

$$\frac{Total\ revenue}{Total\ assets}$$

ROIC

This ratio is again a measure of a company's resource efficiency, specifically in generating profits [20]. It is computed as a ratio of Net operating profit after tax to invested capital. NOPAT can be calculated by subtracting income tax paid from operational earnings. Invested capital can be calculated by adding debt to equity. These metrics can be found on a company's income and balance sheets. The formula for ROIC is:

$$\frac{NOPAT}{Invested\ capital}$$

Return on Asset

Return on assets is a ratio used to measure a company's resource management efficiency [21]. It gives an idea of how profitable a firm is relative to its assets. It is computed as a ratio of net income to total assets. These metrics can be found on a company's income statement and balance sheet respectively. The formula for Return on asset is:

$$\frac{Net\ income}{Total\ assets}$$

Profit Margin

This is a profitability ratio used to compute a company's money making capability [22]. It is computed as the ratio of gross profit to total revenue. Both of these metrics can be found on a company's income statement. The formula for Profit margin is:

$$\frac{Gross\ profit}{Total\ revenue}$$

Past returns

For the last 4 features, we consider return metrics for the past 1 month, 3 months, 6 months and 1 year.

3.5.3 Algorithm Selection

CART

CART is an ML based algorithm used in Classification and Regression problems. For the purpose of this project, we have used decision trees in a Classification problem. Decision trees are grown through recursive splitting of the data set features through a impurity based ranking system. This ranking system is implemented using either a Gini Index or Information Entropy [24].

Essentially, Gini Index calculates how often we incorrectly classify an element j from a particular feature f. The formula for the same is given below:

$$Gini_f = 1 - \sum_j p_{fj}^2$$

Information entropy is similar in task to Gini Index but approaches the computation problem using the additive property of the logarithmic function [24]. The formula for Entropy is given below:

$$Entropy_f = -\sum_j p_{fj} \log_2 p_{fj}$$

These algorithms are implemented in python using the sklearn.tree.DecisionTreeClassifier package by assigning either 'gini' or 'entropy' to the criterion parameter. To give further intuition of these 2 methods, fitted trees created through the application of the aforementioned ranking methods are shown in Figure 3.3 and 3.4.

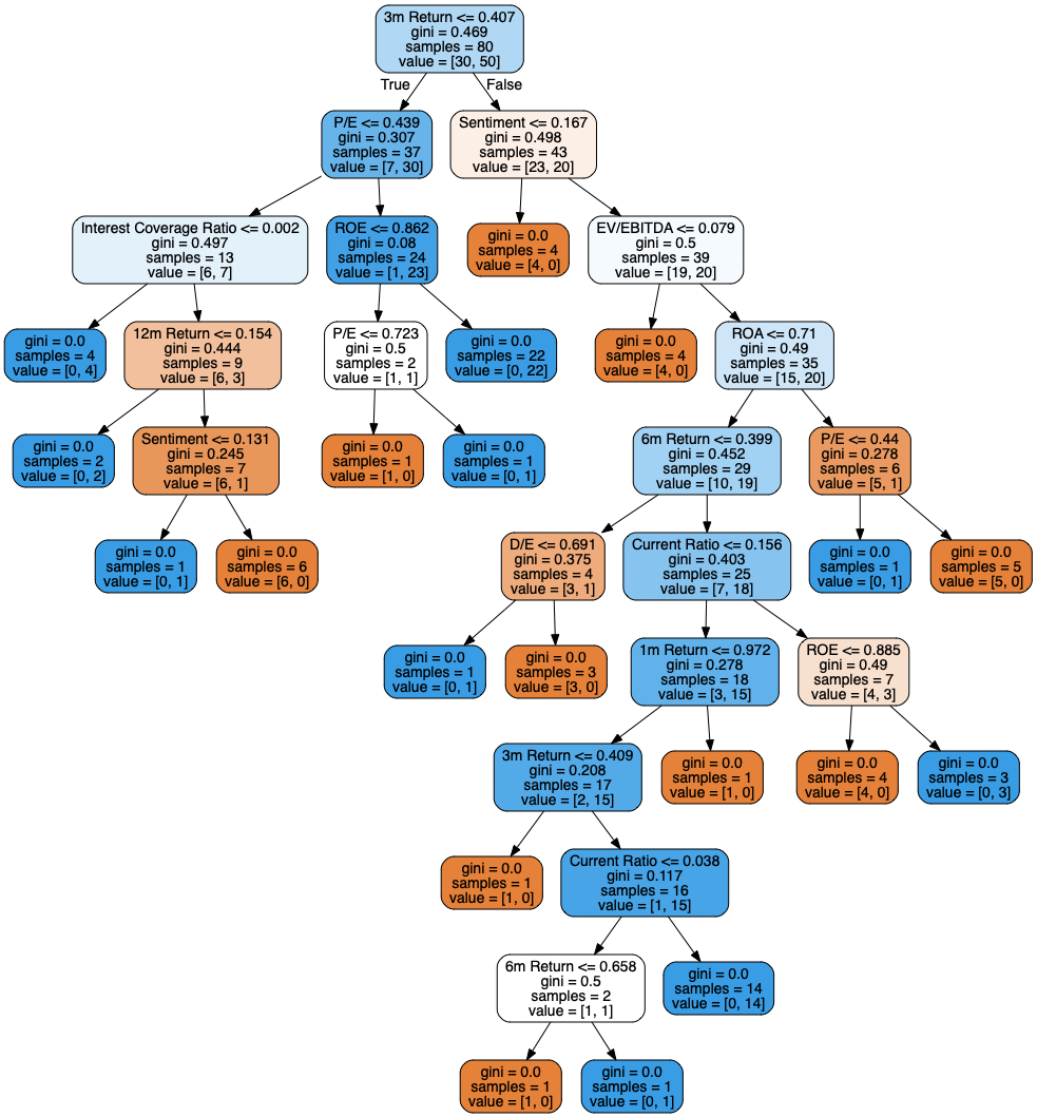


Figure 3.3: Gini Index

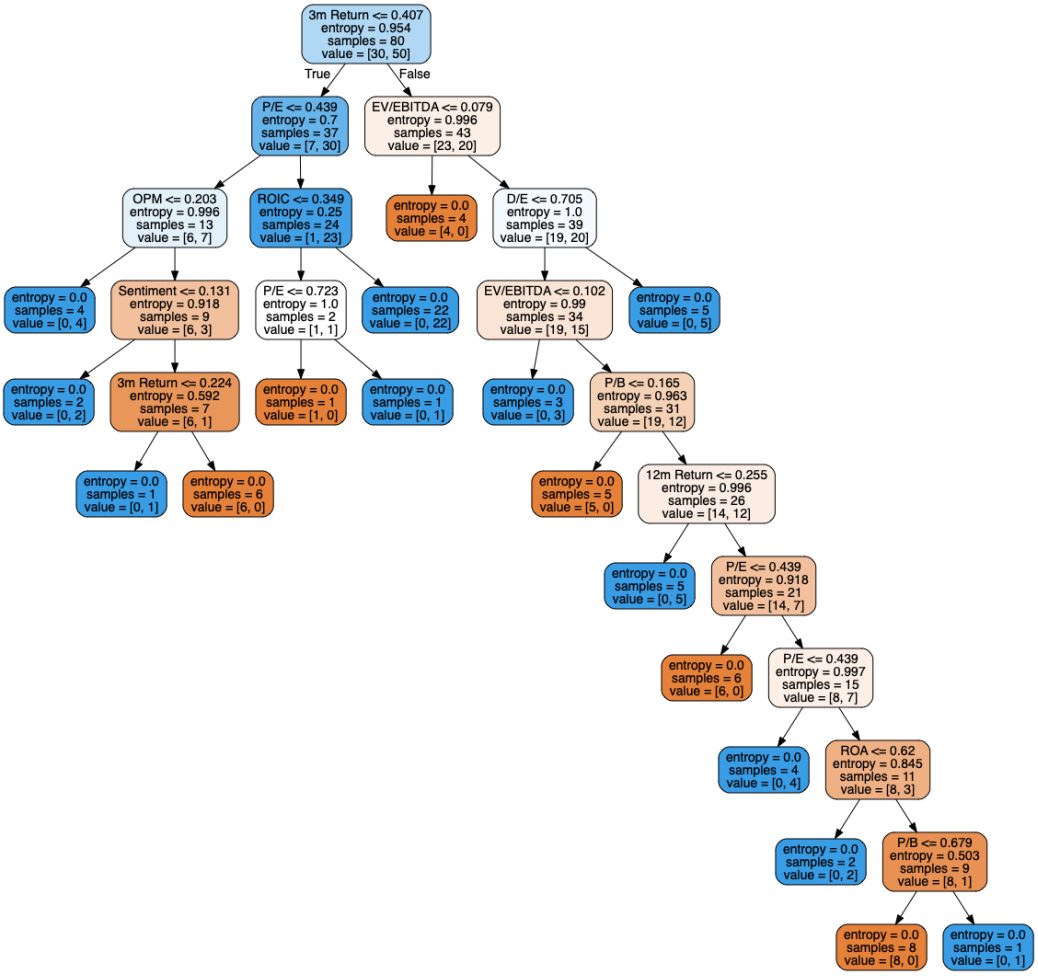


Figure 3.4: Information Entropy

Random Forest

Random forest is an ensemble based learning method which uses multiple uncorrelated decision trees to solve classification predictions. This is an extremely powerful ML based method, which uses a relatively simple principle - the Wisdom of crowds [25].

In python it is implemented using the sklearn.ensemble.RandomForestClassifier package.

To understand this concept better, consider a random forest with 9 uncorrelated decision trees predicting the same output. Assume there are 6 True and 3 False outputs. The majority votes are chosen, i.e: True. A graphic of this is shown in Figure 3.5.

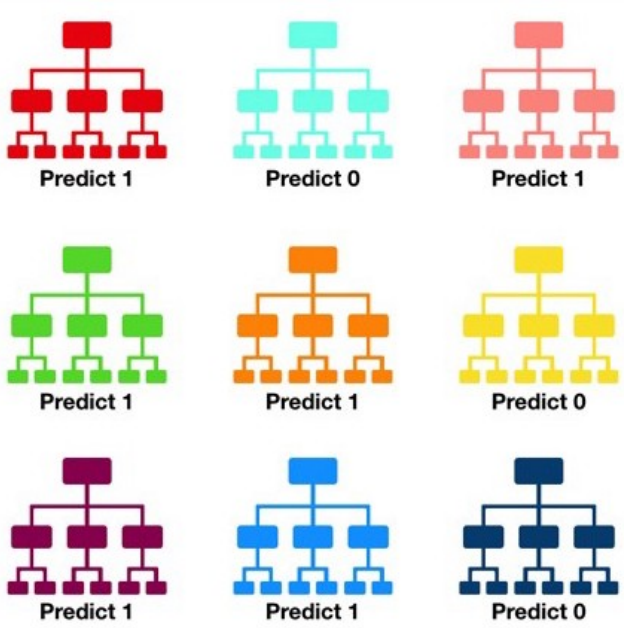


Figure 3.5: Random Forest

Support Vector Classifier

Support Vector classifier is a ML based algorithm that looks for a well-defined hyper plane in R^n dimensions, which is capable of classifying the datapoints. This hyperplane will also have the highest possible margin when separating the datapoints. This provides confidence when classifying out of sample datapoints. We minimize the hinge loss function to maximize this margin. The formula is given by [26]:

$$c(x, y, f(x)) = (1 - y * f(x))_+$$

In python, it is implemented using the sklearn.svm.SVC package by assigning either 'linear' or 'rbf' to the kernel parameter.

4. Results

The accuracy results for the previously mentioned ML models are given below in table 4.1. These results are obtained using the sklearn.metrics.accuracy_score package. Particular attention should be paid to the accuracy on the testing set.

Model	Accuracy on Training Set	Accuracy on Testing Set
Decision Tree - Gini	1.0	0.5
Decision Tree - Entropy	1.0	0.5
Random Forest	1.0	0.7
SVM - Linear	0.675	0.5
SVM - RBF	0.6625	0.6

Table 4.1: Model accuracies

From the given results, we can conclude that Random Forest is the most accurate model, with a 70% accuracy on the testing set. The confusion matrix for the same is given below.

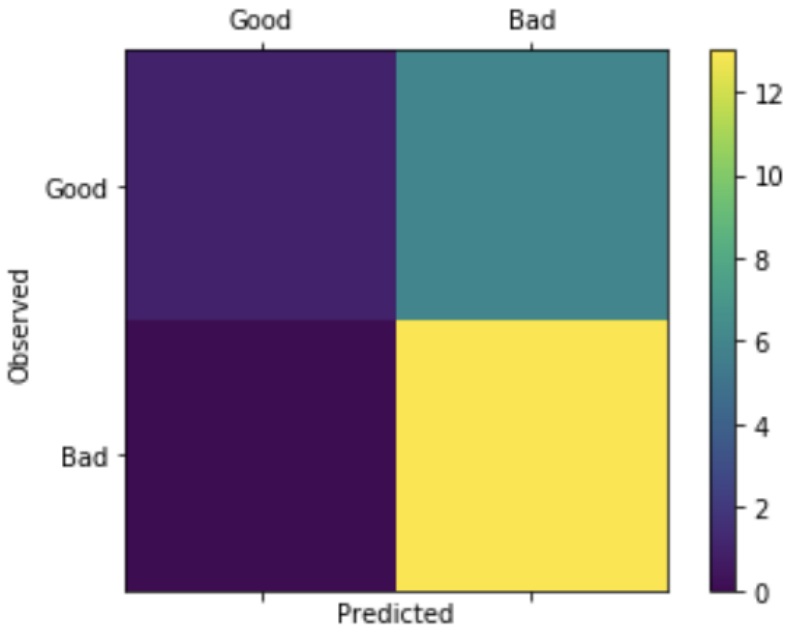
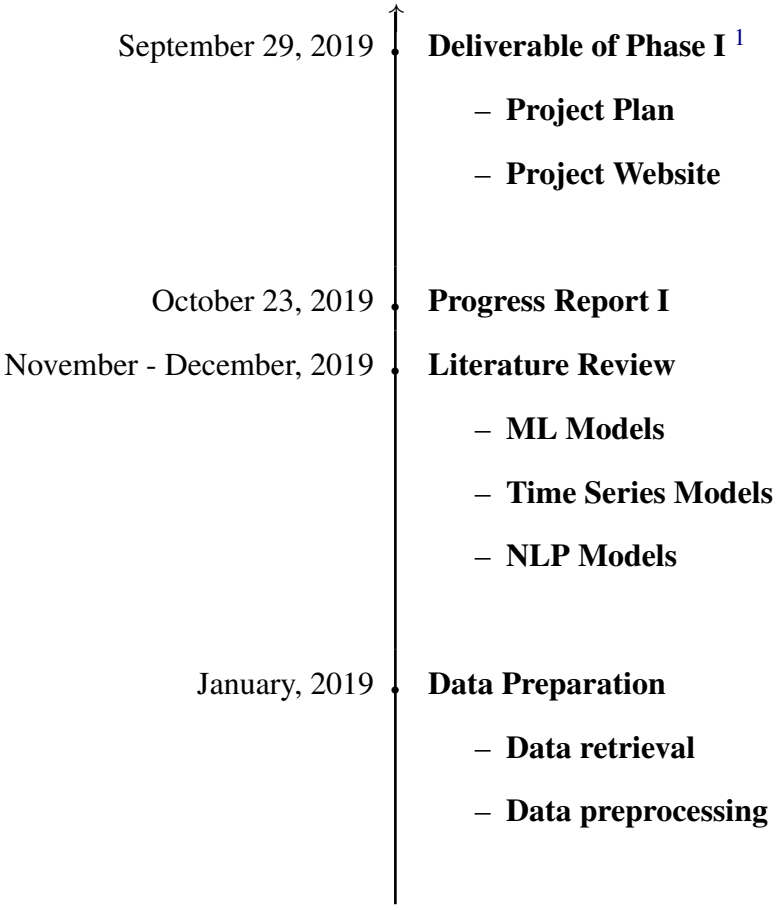


Figure 4.1: Confusion Matrix

5. Project Timeline



[1] Bold indicates completed

March, 2019
- **Variable Selection**
  - **Fundamental Ratios**
  - **Return Statistics**
  - **Asset Sentiment**

April, 2019
- **Algorithm Selection**
  - **NLP Model**
  - **Classification Model**

6. Conclusion

Several methods exist in tackling the problem of Financial Data Forecasting. Previous studies have explored time series forecasting, ML based classification techniques and various other probabilistic and statistical methods. This project explores classification based ML algorithms which haven't necessarily been explored in depth in previous studies. Namely techniques including Random Forest and kernel based SVM. Along with this project, came numerous challenges and limitations. First and foremost, this project only considers a universe of 25 stocks with 4 data points each. Ideally, this dataset should be in the range of hundred thousand to a million datapoints. This can be done by increasing the timeframe from 5.5 years to 100 years. This is extremely advantageous as we have not only have more data to train the model on, but also more outliers in the data which the model can understand. For example, the 2008 financial crisis will help the model understand what changes in metrics lead to such an event. Next, we should consider data of higher frequency. This project looks at datapoints from a yearly perspective. If the frequency of data can be improved to a monthly perspective, we should notice large accuracy gains. Further, for Sentiment analysis, more research

should be done on scraping appropriate articles from famous business newsletters, as oppose to just social media posts. Finally, the inclusion of additional relevant features may help in improving accuracy. This idea could be fatal however, as inclusion of additional features leads to the curse of dimensionality, where features start to develop co-relations with each other, leading to additional biases and higher variance.

Bibliography

[1] T. Kim, "Bad times for active fund managers again: Vast majority are underperforming this year", CNBC, 2019. [Online].

[2] Chen. J, “Autoregressive Integrated Moving Average (ARIMA)”, Investopedia, 2019. [online]

[3] Ye, T. Stock forecasting method based on wavelet analysis and ARIMA-SVR model. 2017 3rd International Conference on Information Management (ICIM), 2017.

[4] Majumder, M. M. R., Hossain, M. I., Hasan, M. K., Indices prediction of Bangladeshi stock by using time series forecasting and performance analysis. 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE).

[5] Phongmekin, A., Jarumaneeroj, P. Classification Models for Stock’s Performance Prediction: A Case Study of Finance Sector in the Stock Exchange of Thailand. 2018 International Conference on Engineering, Applied Sciences, and Technology (ICEAST), 2018.

[6] Gabriela Mircea, Marilen Pirtea, Mihaela Neamtu and S. Bazavan, "Discriminant analysis in a credit scoring model," presented at the Recent Advances in Applied and Biomedical Informatic and Computational Engineering in Systems Applications, Florence, 2011

[7] Harrison, O. (2019). Machine Learning Basics with the K-Nearest Neighbors Algorithm. [online] Medium. Available at: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761 [Accessed 4 Dec. 2019].

[8] GmbH, R. (2019). Performance (Binominal Classification) - RapidMiner Documentation. [online] Docs.rapidminer.com.

[9] Coy, P., 2020. Bloomberg - Are You A Robot?. [online] Bloomberg.com. Available at: <https://www.bloomberg.com/news/articles/2020-03-19/the-great-coronavirus-crash-of-2020-is-different> [Accessed 2 May 2020].

[10] Smith, T., 2020. Market Sentiment Definition. [online] Investopedia. Available at: <https://www.investopedia.com/terms/m/marketsentiment.asp> [Accessed 2 May 2020].

[11] Hayes, A., 2020. Price-To-Earnings Ratio – P/E Ratio. [online] Investopedia. Available at: <https://www.investopedia.com/terms/p/price-earningsratio.asp> [Accessed 2 May 2020].

[12] Hayes, A., 2020. What The Price-To-Book Ratio (P/B Ratio) Tells You?. [online] Investopedia. Available at: <https://www.investopedia.com/terms/p/price-to-bookratio.asp> [Accessed 2 May 2020].

[13] Hayes, A., 2020. Debt-To-Equity Ratio – D/E. [online] Investopedia. Available at: <https://www.investopedia.com/terms/d/debtequityratio.asp> [Accessed 2 May 2020].

[14] Kenton, W., 2020. Why Operating Margins Matter. [online] Investopedia. Available at: <https://www.investopedia.com/terms/o/operatingmargin.asp> [Accessed 2 May 2020].

[15] Hayes, A., 2020. Enterprise Multiple Definition. [online] Investopedia. Available at: <https://www.investopedia.com/terms/e/ev-ebitda.asp> [Accessed 2 May 2020].

[16] Hargrave, M., 2020. How Return On Equity Works. [online] Investopedia. Available at: <https://www.investopedia.com/terms/r/returnonequity.asp> [Accessed 2 May 2020].

[17] Hayes, A., 2020. Why The Interest Coverage Ratio Matters. [online] Investopedia. Available at: <https://www.investopedia.com/terms/i/interestcoverageratio.asp> [Accessed 2 May 2020].

[18] Kenton, W., 2020. Current Ratio. [online] Investopedia. Available at: <https://www.investopedia.com/terms/c/currentratio.asp> [Accessed 2 May 2020].

[19] Hayes, A., 2020. Asset Turnover Ratio. [online] Investopedia. Available at: <https://www.investopedia.com/terms/a/assetturnover.asp> [Accessed 2 May 2020].

[20] Kenton, W., 2020. Understanding Return On Invested Capital. [online] Investopedia. Available at: <https://www.investopedia.com/terms/r/returnoninvestmentcapital.asp> [Accessed 2 May 2020].

[21] Hargrave, M., 2020. How To Use Return On Assets When Analyzing A Company. [online] Investopedia. Available at: <https://www.investopedia.com/terms/r/returnonassets.asp> [Accessed 2 May 2020].

[22] Segal, T., 2020. Profit Margin. [online] Investopedia. Available at: <https://www.investopedia.com/terms/p/profitmargin.asp> [Accessed 2 May 2020].

[23] Oberoi, R., 2020. Key Financial Ratios You Must Look At Before Making Investment. [online] Businesstoday.in. Available at: <https://www.businesstoday.in/moneytoday/investment/key-financial-

ratios-analyze-company-stock-investment/story/209789.html> [Accessed 2 May 2020].

[24] Hershy, A., 2020. Gini Index Vs Information Entropy. [online] Medium. Available at: <https://towardsdatascience.com/gini-index-vs-information-entropy-7a7e4fed3fcb> [Accessed 3 May 2020].

[25] Yiu, T., 2020. Understanding Random Forest. [online] Medium. Available at: <https://towardsdatascience.com/understanding-random-forest-58381e0602d2> [Accessed 3 May 2020].

[26] Gandhi, R., 2020. Support Vector Machine — Introduction To Machine Learning Algorithms. [online] Medium. Available at: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47> [Accessed 3 May 2020].

A. Appendix

Targeted Companies include a universe of 25 stocks within the Technology sector, listed on the Hong Kong Exchange and NASDAQ.

1. PCCW (0008.HK)
2. SmarTone (0315.HK)
3. ASM Pacific Technology (0522.HK)
4. AAC Technologies Holdings (2018.HK)
5. Apple Inc (AAPL)
6. Microsoft (MSFT)
7. Facebook Inc (FB)
8. Intel Corporation (INTC)
9. International Business Machines Corporation (IBM)
10. Oracle Corporation (ORCL)
11. NVIDIA Corporation (NVDA)
12. Cisco Systems Inc (CSCO)
13. Micron Technology Inc (MU)

14. Adobe Inc (ADBE)
15. Tencent Holdings Limited (TCEHY)
16. QUALCOMM Incorporated (QCOM)
17. Xerox Holdings Corporation (XRX)
18. NetApp Inc (NTAP)
19. ServiceNow Inc (NOW)
20. Texas Instruments Incorporated (TXN)
21. Seagate Technology (STX)
22. Dell Technologies Inc (DELL)
23. Cognizant Technology Solutions Corporation (CTSH)
24. Accenture PLC (ACN)
25. Corning Incorporated (GLW)