Electronics and Computer Science
Faculty of Engineering and Physical Sciences
University of Southampton

Foivos Gaitantzis

12th May 2020

Machine Learning Approaches: Stock Price Prediction using Technical & News Sentiment Analysis

Project supervisor: Dr. Jize Yan
Second examiner: Prof. Chen George

A project report submitted for the award of
MEng Electrical & Electronic Engineering

Abstract

This project aims to predict the prices of Apple Stocks (NASDAQ: AAPL) by extracting relevant features from sentiment data derived from various news sources as well as technical indicators from time-series data of historic price and volume. The proposed algorithm is written in Python and makes use of two fundamental machine learning libraries; Scikit-Learn & TensorFlow.

The use of various sentiment analysers is proposed to aggregate the emotion portrayed by various headlines and normalize them to an average daily value. These daily values are then merged with historical prices and technical indicators such as the Relative Strength Index (RSI) and the Moving Average Convergence-Divergence (MACD) over a short and long-term period.

Three machine learning algorithms are experimented: Linear Regression, Support Vector Regression (SVR) & Long Short-Term Memory (LSTM) recurrent neural networks. Hyperparameters are tuned accordingly and the models are evaluated using the Root Mean Squared Error (RMSE) and the Mean Absolute Percentage Error (MAPE).

An algorithmic trading bot is built using the most promising algorithm indicating to what extent day trading predictions can be used to yield profit.

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.
We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

Several Python Libraries were used in this study: pandas, numpy, requests, bs4, vader, scikit-learn, tensorflow

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs apart from News Headlines which have been removed after feature processing to comply with the ERGO II Ethical Approval. Submission ID: 56095

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

Parts of this work have been adapted from the Semester 1 Progress Report.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

ERGO II Ethical Approval was obtained on 03/04/2020 for the collection of News Headlines from the New York Times, Financial Times & Business Standard. Submission ID: 56095

ECS Statement of Originality Template, updated August 2018, Alex Weddell aiofficer@ecs.soton.ac.uk

Acknowledgements

I would like to thank Dr. Jize Yan & Prof. Chen George for their continuous support and feedback throughout the entire year.

Additionally, I would like to thank my friends Stefan Lackanovic and Simas Stankus for proofreading my report.

Finally, a special thank you to my family and my friends for providing me with moral guidance throughout this project.

Table of Contents

1 Introduction

"Stock market prediction is the act of trying to determine the future value of a company's stock price traded on a financial exchange." [1] This has been a controversial subject in the field of finance and given that a market situation can instantaneously change, investors rely on predictions in order to make less risky trading decisions.

There are three main types of market analysis that influence trading decisions; Fundamental analysis is based on monitoring the current economic, social, and political welfare that can affect the aggregate supply and demand. Technical analysis makes use of indicators and various trading strategies to determine profitable entry and exit points by analysing trends (the general direction of an asset). [2] Sentiment analysis assesses investor moods and could be described as the "aggregated public mood or outlook that make up the market psychology at any point in time." [3]

The Efficient Market Hypothesis (EMH) theory states that stock prices are directly reflected by incoming streams of new information from the market entitling that no investor can outperform the market due to unpredictable shocks in market price. This assumption has been challenged by numerous studies arguing its validity by demonstrating high accuracy results through technical and fundamental market analysis techniques.

Time series predictions using machine learning have become very popular in the last few years. This arises due to the increased sophistication of machine learning algorithms enabling real-time analysis on large chunks of data. The most prominent prediction techniques make use of Support Vector Machines (SVM) or Artificial Neural Networks (ANN) in the form of Recurrent Neural Networks (RNN).

Recently, the process of text mining combined with machine learning has received a lot of attention since news content may alter investor sentiment, changing the direction of the opening stock value on major corporations. Corporate announcements will undoubtedly change the value of certain securities since the collective mood of news headlines has shown a strong correlation with stock market performance.

The aim of this report is to perform an investigation on the performance of various stock price prediction models using both technical and sentiment analysis. The revised project goals **(The Original Project Goals are shown in Appendix A)** are:

- To automate data collection (news headlines & historical stock data) and aggregate all news headlines over a certain date range in a single CSV-file.
- To perform data pre-processing on textual data utilizing sentiment analysis to determine the emotions portrayed by each headline.
- To aggregate the emotions portrayed by the headlines daily and to utilize Technical Analysis to extract technical indicator features.
- To build various machine learning models for stock price prediction and determine their accuracy (using RMSE & MAPE (%)).
- To build a final model using the most accurate algorithm and utilize it to build a virtual day-trading bot that may in turn yield profit.

2 Background

2.1 The Stock Market

The stock market acts as an intermediary bringing together buyers and sellers of certain financial securities held on various exchanges such as the NASDAQ or the New York Stock Exchange (NYSE). You can place stock trades through a broker who will then deal with the exchange on your behalf.

Companies list their shares on an exchange through a process called an Initial Public Offering (IPO). A company's stock price is initially determined by its valuation and the total number of shares it decides to issue; for example, if a company is valued at $500 million, it may issue 10 million shares at $50 per share or even 5 million shares at $100 per share. Buyers usually offer a bid price which is the highest amount they are willing to pay. This is usually lower than the amount sellers ask for. The difference between these two prices is called the bid-ask spread. For a trade to occur, a price needs to be agreed that will benefit both parties. [4] The stock price fluctuates over time due to supply and demand; that is if a company portrays good standing through continuous long-term earnings and growth, it is more likely to attract investors clamouring to purchase its stock and typically drive its price up. For example, Figure 1 below indicates the historical stock price fluctuations of Apple between the beginning of June and end of December in 2018 (used as the test set in the implementation stage).

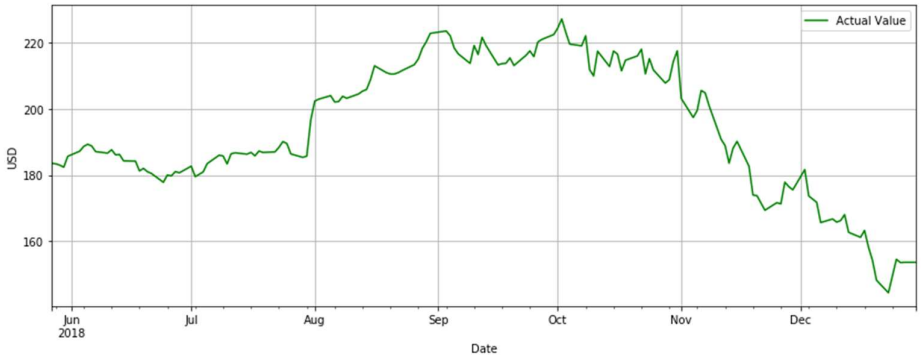


Figure 1: Apple Stock Price June 2018 – January 2019

As can be seen, Apple showed good market standing in the summer of 2018 when it became the first publicly traded U.S. company to hit $1 trillion in market value before taking a tumble in November and losing just about 30% of its value. This is due to the negative sentiment that carried over in December as a result of Apple issuing lower earning guidance (estimate of its future earnings) for the first fiscal quarter ending in November. On top of that, following a court order in China, Apple decided to halt sales of older iPhone models affecting short-term market performance. These are all factors that can affect the overall market value on certain financial securities.

Day trading involves the purchase and sale of a security within a single trading day, something that invokes high risk due to the short-term behaviour of markets that in turn approximate to very random results. Typically, financial institutions perform day trading

operations using high-speed computers that can execute thousands of high frequency trades using technical indicators to make profit over small fluctuations in price.

Technical analysts analyse historical data and make use of technical indicators to predict future market shifts. There are two basic types of Technical Indicators; Overlays use the same scale and are plotted over stock prices (such as Moving Averages). Oscillators are indicators that oscillate between a minimum and maximum value and are plotted using different scales. Two examples of oscillator indicators that are used for this experiment are shown below:

2.1.1 Relative Strength Index (RSI)

The Relative Strength Index (RSI) is amongst one of the most popular momentum indicators that measures the strength of a security by considering the magnitude of recent price changes (usually over a 15-day period) in order to determine whether a stock is overbought or oversold.

This is a value that ranges from 0 to 100 where typically a value of over 70 indicates that a stock is overbought and a value below 30 indicates that its oversold. A trader will most likely want to buy stock when its being oversold since an upward trend is anticipated. Similarly, they will want to sell stock when its overbought as its price trend is bound to reverse.

To calculate the Relative Strength Index (RSI), the intra-day changes between the closing prices are calculated. By taking a Simple Moving Average (SMA) of these changes over the 15-day period (current and previous 14 days), the average gain and loss are determined which are equivalent to the mean of the upward or downward price changes, respectively.

The Relative Strength Index (RSI) is then given by the following Equations (1) - (2):

$$RSI = 100 - \frac{100}{1 + RS} \quad (1)$$

$$where\ RS = \frac{Avg.Gain\ of\ Change\ During\ Time\ Period}{Avg.Loss\ of\ Change\ During\ Time\ Period} \quad (2)$$

2.1.2 Moving Average Convergence Divergence (MACD)

The Moving Average Convergence Divergence (MACD) is another popular momentum indicator that underpins the relationship between a long- and short-term moving average on a security's closing price. The short-term period tends to be around 12-13 days while the long period chosen is usually around 26 days.

In this case an Exponential Moving Average (EMA) is used as opposed to a simple moving average due to its property of reacting more significantly to recent price changes. This is calculated by following the steps shown by Equation (3).

1. $Find\ the\ First\ EMA\ Value\ by\ taking\ a\ Simple\ Moving\ Average\ (SMA).$
2. $Calculate\ the\ smoothing\ Constant\ as\ k = \frac{2}{Time\ Period+1}$
3. $Calculate\ the\ Exponential\ Moving\ Average\ as$
   $EMA = (Today's\ Closing\ Price - Previous\ Day's\ EMA) \times k\ (3)$

The formula for MACD is then given by Equation (4) below.

$$MACD = Short\ Period\ EMA - Long\ Period\ EMA\ (4)$$

A 9-day Exponential Moving Average (EMA) of the MACD is then plotted and defined as the 'signal line'. Traders will find the opportunity to buy when the MACD value crosses over the signal line indicating an oversold security. Similarly, an MACD value that falls below the signal line will indicate an overbought security signalling the trader to short their trades.

2.2 Machine Learning Algorithms

A simple definition of a machine learning algorithm entitles it as a formula that can "parse data, learn from that data and then apply what it has learned to make informed decisions." [5]

In simple terms machine learning is the science of programming computers so that they can learn from data. Machine learning systems can be classified based on whether they are trained using labelled data or whether they can learn incrementally from streams of incoming data. Supervised learning is possible when outputs are determined using labelled examples. On the other hand, unsupervised learning uses data sets with no labels attempting to find strong patterns between them. Batch learning ingests all data at once when building a model while in online learning the system can be trained incrementally using streams of data sequences. [6] A typical supervised learning task involves classification trained using examples of different classes with its main objective to classify new records into pre-determined categories. Regression on the other hand revolves around the prediction of a target numeric value given a set of features.

Take the spam filter of your mailbox as an example; it can learn to flag emails as spam or ham (non-spam) given a training set with examples of both labels. This is an example of an online supervised learning classification algorithm. [6]

2.2.1 Linear Regression

Linear Regression is the simplest model for statistical learning and a good starting point before moving into more complex approaches. It works by feeding in several training examples and allowing it to find the parameters that make the model best fit the data as shown by Figure 2 below.

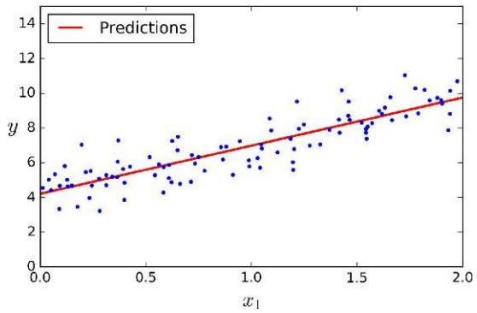


Figure 2: Linear Regression Model Predictions (sourced from [6])

It can be summarized as a linear function where predictions are made by simply computing a weighted sum of the input features plus a constant called the bias term as shown in Equation (5); where $\hat{Y}$ is the predicted value, θ is the model's parameters (bias terms and feature weights), x is the feature values and n indicates the total number of features. [7]

$$\hat{Y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n \quad (5)$$

In order to find the optimal parameters of a machine learning algorithm, iterative methods are used to minimize the cost function (sum of squared errors) which is the difference between the predicted $\hat{Y}$ value and true Y value as shown by Equation (6); where $\hat{Y}$ is the predicted value, Y is the actual value and m is the number of training instances.

$$MSE\ Cost\ Function = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{Y} - Y\right)^2 \quad (6)$$

The errors are squared to avoid any negative differences and to penalize large differences guaranteeing a better model. By making use of Gradient Descent, the values of θ are randomly initialized and improved gradually by measuring the local gradient of the cost function and going in the direction of descending gradient until a minimum is reached.

In some cases, data can project a more complex relationship than a simple straight line therefore polynomial regression can be utilized by appending the powers of each feature in the Data Frame. Polynomial Regression adds all combinations of features up to a given degree; with a degree of 2, the features a and b would be extended to include a^2, b^2 & ab. The regression model is then trained using this extended number of features.

Linear Regression, as mentioned by name is a regression algorithm which can be used for classification purposes as well. Logistic Regression (its classification variation) is used to estimate the probability of an instance belonging in a class by making use of decision boundaries.

2.2.2 Support Vector Machines (SVM)

A Support Vector Machine (SVM) is an example of a supervised machine learning algorithm that is more commonly used in classification problems however it can also be employed for regression purposes. [8]

Its main goal is to determine the optimal hyperplane that best separates a dataset into its classes as shown by Figure 3 below.

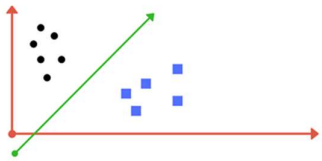


Figure 3: SVM Class Separation using Hyperplane (sourced from [9])

This is done by maximizing the distance between the hyperplane and the nearest data point from each class known as the margin. Since datasets are not clean and tend to show a non-linear correlation, kernels can be used that transform the plane into linear dimensions and are in turn mapped back to the original plane as irregular boundaries. This is shown by Figure 4.

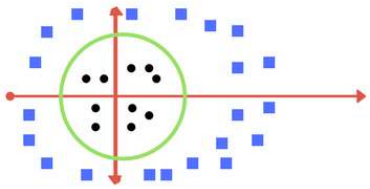


Figure 4: Demonstration of Kernel Transformation (sourced from [9])

Two more tuning parameters can be tweaked in the case of overlapping data points that achieve a non-linear classification line with higher accuracy.

The first one is called the regularization parameter C. A small value of C will cause the algorithm to find a large margin separating hyperplane misclassifying various points. A large value of C will conversely classify all training points correctly at the expense of overfitting the data.

The final parameter γ defines the extend that data points far away from the hyperplane may influence the boundary. A high value of γ indicates that solely the points close to the hyperplane are considered in training.

Support Vector Machines (SVM) can also be used as a regression model (Support Vector Regression (SVR)) using the same principles as for classification with a few minor differences. Just like all regressors, it attempts to fit a plane (known as the street) to the data by minimizing its cost function. Its objective is to try fit as many training instances on a street controlled by a hyperparameter ε as shown by Figure 5.

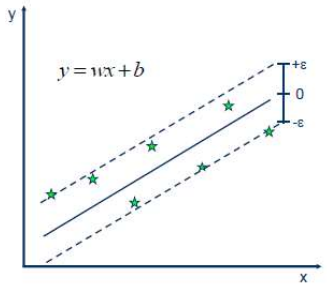


Figure 5: SVR ε Hyperparameter (sourced from [10])

The higher the value of ε, the larger the width of the street making it prone to an overfitted dataset.

2.2.3 Long Short-Term Memory (LSTM)

Deep learning is a subset of machine learning involving the workings of the human brain in processing data and is composed of several layers; the input layer which is fed in all the features of a specific Data Frame, the tens or hundreds of hidden layers which are able to distinguish different patterns in the data and the output layer which can then categorize the datapoints into one or multiple values. [11]

The neurons in this network are inter-connected by synapses that each carry a specific weight and bias allowing for valid connections between the input to hidden layer, the hidden to hidden layers and hidden to output layer. The weight represents the strength of correlation between a neuron and a preceding layer and the bias can be manually adjusted to vary the threshold of activation. The neuron's output will then equal to the sum of the biases and the weight multiplied by the input, a similar equation to Linear Regression. Unfortunately, the output can lie between any value on the number line, therefore an activation function is used in order to squeeze the values between 0 and 1. The output formula is shown below by Equations (7) - (8); where n is the number of data points and k is the number of features, making use of the sigmoid activation function σ. [12]

$$a^{(1)} = \sigma\left(\begin{bmatrix} w_{0,0} & \cdots & w_{0,n} \\ \cdots & \cdots & \cdots \\ w_{k,0} & \cdots & w_{k,n} \end{bmatrix}\begin{bmatrix} {a_0}^{(0)} \\ \cdots \\ {a_n}^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ \cdots \\ b_n \end{bmatrix}\right) \quad (7)$$

$$a^{(1)} = \sigma\left(\boldsymbol{W}a^{(0)} + \boldsymbol{b}\right) \quad (8)$$

Recurrent Neural Networks (RNNs) are powerful algorithms when it comes to processing sequential time-series data. Unfortunately, "RNN's suffer from short-term memory" therefore Long Short-Term Memory (LSTM) networks can be used instead that allow for information to flow from one step to the next by conserving contextual information at each stage. LSTMs can be conceptualized as multiple copies of the same network allowing a previous message to pass to its successor, mitigating memory. It is composed of internal mechanisms called gates that can filter out which data in a sequence is important to keep or throw away. This methodology is inspired from human brain: as you read this report you understand its meaning based on previous words instead of processing each word separately. [12]

There are plenty of deep learning frameworks available however Keras was utilized for this experiment due to its easy to learn framework, increased flexibility, and compatibility with other python libraries such as TensorFlow. There are several parameters that need to be tweaked in a Keras LSTM model to increase model accuracy. These are shown below:

- **units** indicate the number of neurons in the LSTM output layer. The higher the number of units, the higher the computational complexity of the model learning from smaller details and becoming more prone to overfitting.
- **timestep** indicates the number of samples prior to the current one that the LSTM model should consider before reaching an optimum solution.
- **input_shape** defines the 2-Dimensional size of the tensor that is processed by the first hidden layer. This is given as input_shape = (timestep, number of features).
- **batch_size** indicates the number of samples that will propagate through the network at each iteration. Using mini batches is computationally inexpensive and can train faster as it requires less memory at the expense of a less accurate gradient estimate.
- **epochs** indicates the number of passes through the entire training set before reaching an optimum solution.

Just like in previous machine learning algorithms, recurrent neural networks use a loss function measuring the mismatch between the predictions and actual output values. Optimizers tie together the model and its loss function by considering the gradient of the mismatch in order to reach a global minimum. A popular optimization algorithm includes Stochastic Gradient Descent (SGD) which deals with only a subset of the training set independently allowing it to handle very large data sets efficiently. AdaGrad, Adam and RMSProp are examples of different optimizers that extend the concept of Gradient Descent resulting in faster convergence rates by utilizing adaptive learning.

Regularization is a technique used to avoid overfitting by making the model more flexible to noise through finding patterns in the training set and generalizing it to predict corresponding target values for some new dataset. Dropout is an example of a regularization technique which introduces noise during the training phase by randomly dropping out different nodes at each layer upon every iteration. This introduces a new hyperparameter specifying the probability of the output layers getting 'dropped out'. This value ranges from 1.0, indicating no dropout, to 0.0 killing out every layer where an optimal value usually lies between 0.5 and 0.8.

2.3 Feature Scaling

Machine learning algorithms tend to underperform when each feature has a different numerical scale. The two most common methods to align all the features on the same scale involve normalization and standardization.

Normalization is the simplest method, scaling all values in the range of 0 to 1 through the following Equation (9).

$$X_{Normalized} = \frac{X - X_{Minimum}}{X_{Maximum} - X_{Minimum}} \quad (9)$$

Fortunately, scikit-learn provides a transformer that handles the entire normalization operation called the MinMaxScaler.

Standardization works differently, by rescaling the distribution to have a mean of zero and a standard deviation of 1 therefore not binding any values to a specific range. The main advantage of using standardization is the fact that it is much less affected by outliers with the drawback that several machine learning algorithms expect input values ranging from 0 to 1, such as neural networks. The equation for standardization is shown below (10).

$$X_{Standardized} = \frac{X - \mu}{\sigma} \quad (10)$$

Just like for normalization, scikit-learn provides a transformer for standardization called the StandardScaler.

For all transformations it is essential that only the training data is fitted as opposed to the entire dataset. This allows for the test set to only be transformed using scaling parameters inherited from the training data.

2.4 Algorithm Evaluation

A common performance metric for Regression Models makes use of the Mean Square Error (MSE). This function is used in two different stages: as an error function when training the model and as a performance metric when evaluating the accuracy on the test set.

An improved metric indicating the average deviation of the actual points from the predicted hyperplane is the Root Mean Square Error (RMSE) and is preferred as it has the same units as the output variable plotted on the vertical axis. It is calculated as the Square Root of the Mean Square Error (MSE) as shown by Equation (11).

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\hat{Y} - Y\right)^2} \quad (11)$$

The Mean Absolute Percentage Error (MAPE) is a performance metric indicating the accuracy of a system as a percentage average between the absolute differences in actual and predicted values. This is the base accuracy function underlining how far the model's predictions are off from their corresponding outputs. It is calculated as a percentage shown by Equation (12) since percentages are easier to conceptualize.

$$MAPE = \frac{100\%}{m}\sum_{i=1}^{m}\left|\frac{\hat{Y} - Y}{Y}\right| \quad (12)$$

2.5 Review of Previous Literature

As reported in most previous studies, the main sources of textual data collection revolve around major financial news sources that provide high factual integrity such as the New York Times, Reuters and Bloomberg. Others use News Aggregators such as Google News and Yahoo News. Even though there are studies that utilize full article bodies and snippets most purely make use of headlines since they are informative and faster to process as they do not contain noise from excessive text. As an alternative, some studies have collected textual data through Twitter Posts from popular Twitter accounts. As most of these sources provide API limitations on the number of articles that can be scraped over a certain time period most studies perform data gathering over elongated periods.

Most of these studies attempt to predict market movements (Up/Down) as opposed to predicting the future price value. This is known as a classification problem and tends to yield higher accuracy than the equivalent regression problem. Even though most studies use historical market data to supplement their models (especially the regression problems), some predictors solely make use of sentiment data to predict intra-day market shifts. The historical data are usually acquired through the Quandl or Yahoo Finance (less accurate as not correctly back adjusted) APIs.

By making use of sentiment analysis some studies were able to derive emotion features at different success rates. Even though most prediction targets are market indices such as the S&P500 or the Dow Jones, some studies perform prediction on company specific share prices by making use of topic modelling techniques that extract relevant sentiment information related to a specific corporation.

The number of features vary per study as some like to supplement their text-derived features with fundamental indicators and/or technical indicators such as the Relative Strength Index (RSI). Since most studies make use of classification algorithms, Support Vector Machines (SVM), Logistic Regression and Decision Trees are the most popular algorithms. Regression problems on the other hand tend to make use of algorithms like Linear Regression, Support Vector Regression (SVR) and Recurrent Neural Networks (RNN) such as Long Short-Term Memory (LSTM).

3 Implementation

3.1 Proposed Methodology

A high-level model overview has been designed displaying the basic functionality of the proposed methodology as shown below by Figure 6; where Green Boxes signify all the Data Gathering Steps, Pink Boxes show features that are entered into the Machine Learning Algorithm, Grey Boxes display Feature Engineering Steps, Blue Boxes show Implementations of different Machine Learning Algorithms & the Algorithmic Trading Bot and Yellow Boxes are dedicated for Evaluation Steps.

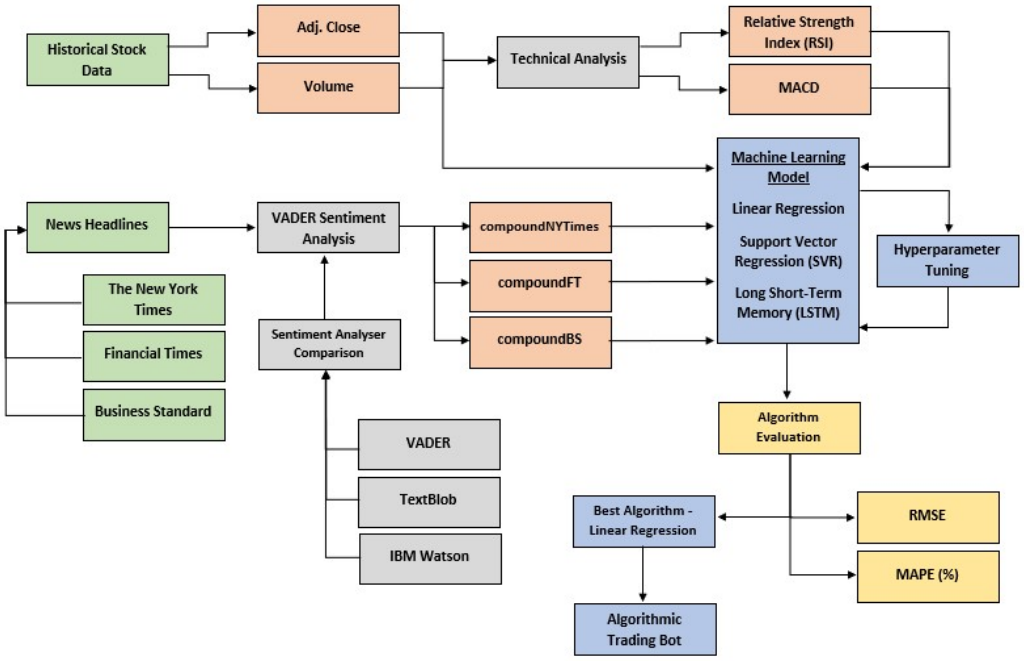


Figure 6: High-Level Model Overview

Primarily, Headlines from three different sources will be collected along with Historical Stock Data for an adequate date range. Using Technical Analysis, features such as the Relative Strength Index (RSI) and the Moving Average Convergence-Divergence (MACD) will be extracted from the historical stock data. A comparison between three different sentiment analysers will be performed and the best one will be chosen to process each headline. A sentiment value for each news source will be calculated by taking the mean sentiment of a range of headlines on a daily basis. After implementing and performing hyperparameter tuning on each of the three algorithms, they will be evaluated (using the RMSE and MAPE) allowing for an algorithmic trading bot to be built using the best predictor.

3.2 Data Collection

There are several libraries in Python that allow the collection of information from different webpages. BeautifulSoup is the most common one as it can transform an HTML documents into a complex tree of Python objects. Various elements can then be accessed from the tree by identifying them with their relevant HTML tags and attributes [13].

In this study four different news datasets were collected following an ERGO II Ethics Approval over a long-term period from January 2016 to December 2018. Two search queries were used ['aapl', 'apple inc'] to fetch the most accurate headlines. Finally, every headline was filtered to remove duplicates and those not containing the following keywords: ['aapl', 'iphone', 'ipad', 'apple', 'app', 'stock', 'aal', 'mac', 'steve'].

Primarily, daily headlines were collected from the Business Standard, "India's leading business daily which attracts over 10 million unique visitors every month", accounting for around 35% (10,023 headlines) of the total news. [14]. As most editors are credible when it comes to facts, Business Standard is rated high for factual reporting.

Overall, the New York Times is considered one of the most reliable sources when it comes to factual integrity due to its proper sourcing and well-respected journalists. They also offer an API for non-commercial use that returns several helpful metadata on years' worth of articles such as their headlines, abstracts and even term subjects. Unfortunately, "there are two rate limits per API: 4,000 requests per day and 10 requests per minute" therefore date ranges have been used in order ensure that the limit is not surpassed. [15] This dataset accounts for only around 32% (9,222 headlines) of the total news collected.

The final news source chosen was the Financial Times, as it is rated highly balanced and provides high factual integrity from its proper sourcing and clean fact checking. This dataset accounts for 32% (9,187 headlines) of the total news and is the smallest news data source used in the experiment.

API calls made for each day, query and pages were facilitated by the Requests Python library. For every news source discussed above, the total API calls were determined as shown by Equation (13).

$$Total\ API\ Calls = 1096\ days\ (3\ years) \times 2\ Search\ Queries \times N\ Pages \quad (13)$$

Due to 'GET' limitations, the total queries for certain news sources were cut down by introducing date ranges (September 2018 – October 2018 ex. New York Times). For other sources (ex. Business Standard) only a few hundred calls could be requested in a day. Therefore, data collection had to be spread over an elongated period of approximately two weeks.

The number of headlines per source is shown by Figure 7.

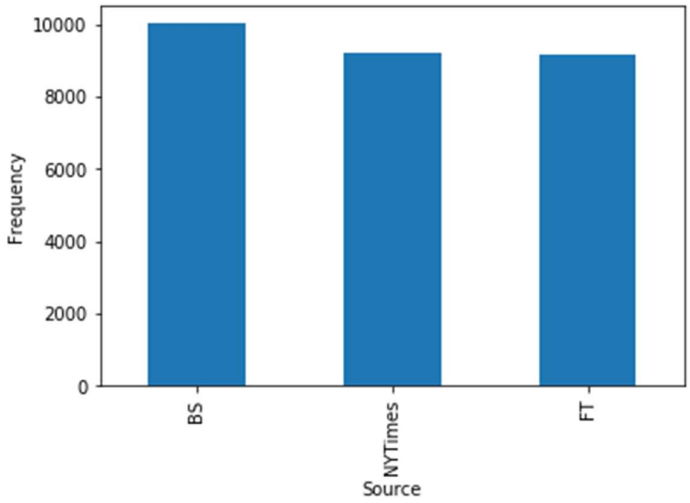


Figure 7: Frequency of Headlines per Source

3.3 Sentiment Analysis

Sentiment analysis involves the evaluation of a series of text in order to gain insight on the polarity of emotion that it portrays. In this case the series of words consists of headlines and by combining natural language processing (NLP) with machine learning techniques, a weighted sentiment score can be assigned to the entities within a sentence. A lexical-based sentiment library consists of very large collections of words and phrases that have been manually assigned a sentiment score by various programmers. When given a sentence, the sentiment values of the words and phrases are aggregated to give an average emotional polarity. By making use of machine learning, this process can be automated by feeding in pre-tagged examples of sentiment in order to improve accuracy and "shoulder the load of complex natural language processing tasks, such as understanding double-meanings." [16]

Three libraries were tested that follow either a lexical-based or a machine learning approach: NLTK-VADER, TextBlob and IBM Watson.

3.3.1 NLTK-VADER

Valence Aware Dictionary for Sentiment Analysis (VADER) is a pre-built model part of the Natural Language Toolkit (NLTK) library that can output the intensity and polarity of emotion of a certain piece of text. As it is 'pre-built', other people have already done the dirty work of building a sentiment lexicon based on human ratings via Amazon 9 Mechanical Turk, a platform that allows workers to perform minor tasks such as data cleaning, in exchange for a small amount of money. Each of these words are given a rating between -4.0 (Extremely Negative Sentiment) to +4.0 (Extremely Positive Sentiment). The final sentiment score of a passage is obtained by summing up the values of each individual word and then normalizing (-1.0 to +1.0) the final score to obtain an average. Other factors may also cause fluctuations in the final sentiment score. This includes the conventional use of punctuation (e.g., Good!!!), capitalization of certain words (e.g., GOOD), use of slang or emoticons, etc.

3.3.2 TextBlob

Just like NLTK-Vader, TextBlob is a sentiment lexicon which is leveraged to give polarity and subjectivity scores on a given sentence. Polarity is a float value between the range of -1.0 to 1.0 where 0 indicates neutral sentiment, +1 indicates a positive sentiment and -1 indicates a negative one. Subjectivity on the other hand is a float value between 0.0 and 1.0 where 0.0 indicates a sentence that is objective and 1.0 represents a subjective one (expresses opinions, personal views, speculations, etc.). Subjectivity is NOT relevant for the objective of this experiment.

3.3.3 IBM Watson

IBM Watson is an advanced AI solution that is powered by the latest innovations in machine learning providing an API for natural language processing such as sentiment analysis. It is trained using millions of tweets that are either given a thumbs up for positive sentiment or a thumbs down for negative sentiment along with their polarity. It is based on a recurrent network therefore previous words can affect the sentiment of proceeding ones and so on. [17] This technique is leveraged to output a polarity score between -1.0 to 1.0, just like TextBlob.

3.3.4 Sentiment Analysis Comparison & Headline Pre-Processing

By performing sentiment analysis on 5 headline samples that are most likely to show a mixture of emotion, the three natural language processing libraries were compared as shown by Table I.

Table I: Comparison of 3 Different Sentiment Analysers

Headline	Real Sentiment	VADER Prediction	TextBlob Prediction	IBM Watson Prediction
The new iPhone 7 will be waterproof	Positive	Positive: 0.5331	Positive 0.1363	Positive: 0.585501
Apple, Nike to unveil limited edition Apple Watch 2	Positive	Positive: 0.5576	Negative: 0.0714	Neutral: 0.0
Google Pixel may dent iPhone 7 hype	Negative	Negative: 0.2857	Neutral: 0.0	Neutral: 0.0
Apple paid Nokia $2 bn in cash to settle a long-running patent dispute	Negative	Negative: 0.1808	Neutral: 0.0	Neutral: 0.0

The above experiment yielded the most accurate results when using VADER and was therefore the chosen natural language processing library.

VADER has been found to be quite accurate when dealing with social media posts however it is not fully reliable when working with financial news. To our advantage, there is publicly available data from some of the textual-related publications with Tim Loughran, a professor of Finance who has created a lexicon of commonly used words in Finance. [18] These words are categorized into negative, positive, uncertainty, etc. where positive words are given a moderate magnitude of +2.0 and concisely negative ones are given a magnitude of -2.0. Finally, another lexicon was imported that was "automatically created using diverse statistical measures and a large set of labelled messages from StockTwits adapted to stock market conversations in microblogging services" [19] This lexicon provides commonly used words along with sentiment scores in both affirmative and negated contexts. The total sentiment value of each word is given as the average of these two values and is normalized to be in the range of -4.0 to +4.0. Both external lexicons were appended into the default VADER lexicon.

For accurate sentiment analysis, characteristics such as numbers, punctuation, emojis and stop words play an important role. Consider an example of a headline entitling Apple's customers' satisfaction as 1/5 projecting Negative sentiment. If the numbers in this sentence are removed, then this headline would be considered of Neutral emotion yielding incorrect results. Likewise, punctuation, capitalization and emojis are an essential part of the VADER lexicon and were maintained. The removal of stop words is usually essential to obtain an accurate representation of a headline without any noise. These consist of common words that do not add any meaning like 'I', 'we', 'myself'. Unfortunately, stopwords also consist of certain words that add emotion and is therefore recommended that they are not removed. An example of this problem can be seen in the following headline which portrays a negative sentiment: 'The most-loved phone in the US is not an iPhone'. If stopwords were to be removed, this would result as 'most-loved phone US iPhone' showing a positive sentiment. Therefore, all headlines were not formatted in any way.

3.4 Feature Engineering

Preceding sentiment analysis, a compound score was given between -1.0 (extremely negative) to +1.0 (extremely positive). This was the most useful metric as a normalized weighted score was given for each sentence. The pos, neu and neg scores show the ratio of the total number of words that fall in each category. These were proven to be useless metrics and were in turn removed. Finally, all sentences showing Neutral sentiment value ($0.05 \leq \text{compound} \leq 0.05$) were removed since they do not account for fluctuations in the stock market.

By plotting the frequency distribution against the compound score of the headlines as shown by Figure 8, it is seen that most Headlines yield a Positive-Neutral sentiment. The minority portray a Negative sentiment while the gap in the middle indicates the removed Neutral headlines.

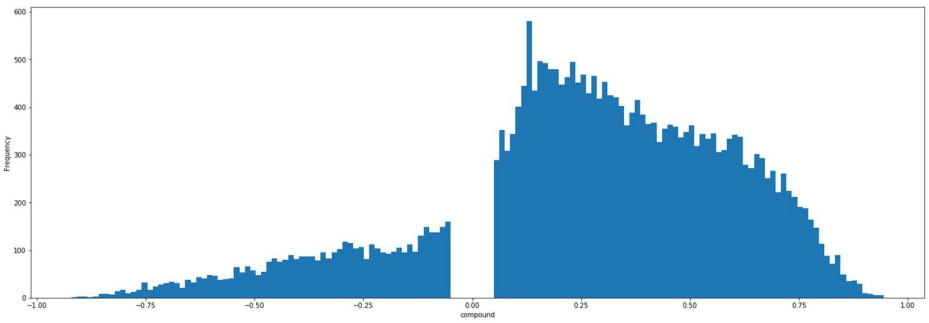


Figure 8: Headlines Compound Score Frequency Distribution

By taking each source and sentiment category and plotting them into a histogram as shown by Figure 9, it is confirmed that the majority yield a Positive Sentiment where the worst Positive-Negative Bias is given by Business Standard.

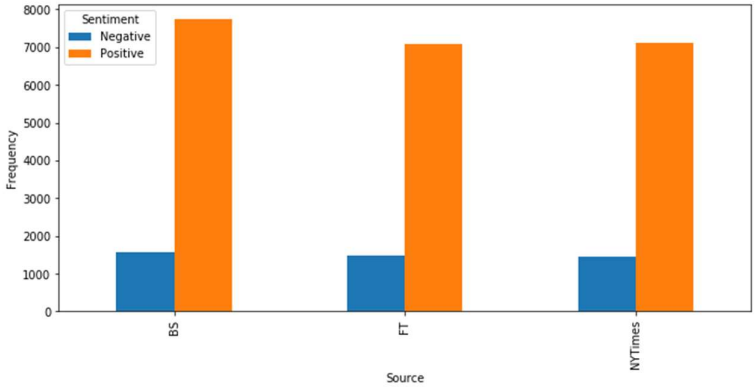


Figure 9: Proportion of Positive/Negative Headlines per Source

By making use of a WordCloud, the most common words for articles portraying Negative sentiment along with their frequency (the larger the word the higher the frequency) are shown by Figure 10.

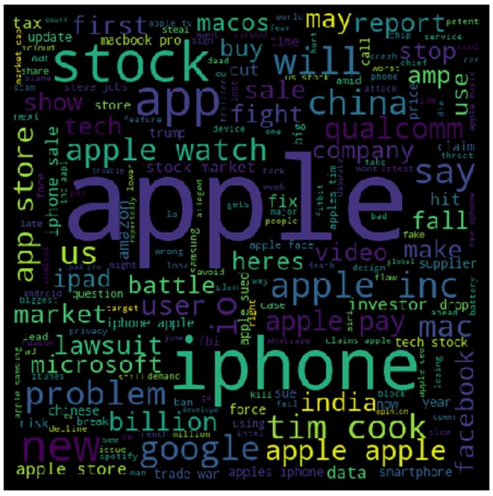


Figure 10: WordCloud between Headlines portraying Negative Sentiment

The words with highest frequency include the query and filter words that are common in all articles such as [‘apple’, ‘app’, ‘iphone’.] Some other words including [‘problem’, ‘fall’, ‘battle’, ‘tax’, ‘lawsuit’] are also included portraying Negative Sentiment.

Since the news sources vary in bias and objectivity, each headline’s compound score was separated to match its source. For example, if a headline was extracted from The Financial Times, its compound score would be set as the ‘compoundFT’ feature. The same principal applies for headlines from the New York Times ‘compoundNYTimes’ and Business Standard ‘compoundBS’. The default compound field was later removed, and the dataset was collapsed into its daily compound average as shown by Table II below.

Table II: Daily Mean of Compound Scores per Source

Date	compoundFT	compoundNYTimes	compoundBS
2016-01-01	0.188000	0.424900	0.361980
2016-01-02	-0.292075	0.301267	NaN
2016-01-03	0.434133	0.371400	NaN
2016-01-04	0.186520	0.267025	0.211217
2016-01-05	0.097792	0.116489	0.236300

All ‘NaN’ values in compound scores for days with missing headlines were replaced with the distribution’s average values using mean imputation. This was preferred over the K-Nearest Neighbours imputation as it does not use information over an observation to estimate a value generalizing the sentiment scores as much as possible. Even though this is not the most accurate approach since the variability in data is reduced, it is better than removing these days altogether.

The stock exchange operates on a five-day business week indicating that on Saturdays, Sundays and Federal Holidays the market is closed for business. This suggests that historical stock values are blank for certain days leading to ‘NaN’ rows. By making use of pandas interpolate in a forward manner, these blank values are filled in using a linear function that estimates the values between two points.

In order to turn this into a supervised learning problem, a label must be given in order to train the model and validate its accuracy. In this case the label is the upcoming day’s closing price as a forecast horizon of 1 day is used. This was achieved using the pandas shift operation which makes use of a scalar parameter representing the number of shifts to be made over a desired axis. The feature was named: ‘Predicted Adj Close’.

Finally, the RSI and MACD were calculated and appended into the Data Frame as indicated in the Background Section. For the MACD, pandas provide a function for the exponential moving average therefore the calculation just involved the subtraction of the 26-day span from the 12 day one. Unfortunately, RSI was more complicated as the change between closing prices had to be determined and placed in separate columns (Gain/Loss) depending on whether it was positive or negative. All changes comprising of zeros had to be replaced with ‘NaN’ values and a rolling mean using a window of 15 days was used to determine the Gain and Loss averages without accounting for blank values.

In machine learning, when two features change together linearly, they project high correlation. That is if one feature gets larger, the other one becomes either larger or smaller. By analysing the absolute correlation value between each feature in the Data

Frame and the Predicted Adj Closing label, a graph is plotted analysing the importance of each feature shown by Figure 11.

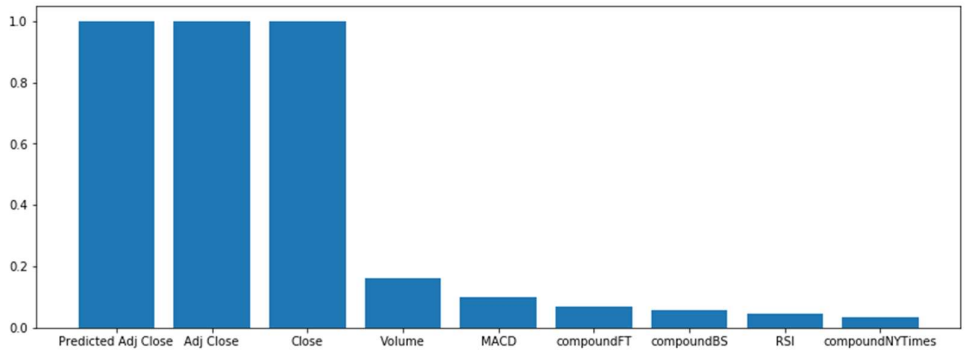


Figure 11: Feature Importance Extracted from Correlation

The final Data Frame was extracted into a CSV file and the following features were stored: ['Date', 'compoundFT', 'compoundNYTimes', 'compoundBS', 'Volume', 'Adj Close', 'RSI', 'MACD', 'Predicted Adj Close'].

3.5 Machine Learning

Implementation of the machine learning models made use of the scikit-learn library for Linear Regression & Support Vector Regression (SVR) while the Tensor-Flow library was utilized for Long Short-Term Memory (LSTM) by stacking layers.

For every algorithm, a comparison between several possible features sets was made to determine which type of analysis yields the lowest error. Four different feature sets were tested as summarized by Table III.

Table III: Type of Analysis According to Feature Selection

Analysis Type	Features
Just Historical Data	Adj. Close, Volume
Technical Analysis	Adj. Close, Volume, RSI, MACD
Sentiment Analysis	Adj. Close, Volume, compoundFT, compoundNYTimes, compoundBS
Technical & Sentiment Analysis	Adj. Close, Volume, compoundFT, compoundNYTimes, compoundBS, RSI, MACD

A general thumb rule was followed to split the data into a test and training set where the final 20% of all instances were set aside for testing purposes (Figure 1). Since the data is time dependent, shuffling was not utilized to maintain time dependency and avoid generating a different test set for every algorithm.

When trying to predict the stock price of a company, the previous day's price plays a very important role in making that prediction. In other words, the lag of a target variable $t-1$, $t-2$, $t-n$ can be utilized in feature engineering when trying to forecast a time series. Therefore, for each row in the dataset, N columns were added to display the previous N values of Adj Close.

This principle was used for Linear Regression & Support Vector Regression (SVR) by converting the time series data into extra features that are manipulated through these algorithms. Long Short-Term Networks (LSTM) works differently by manipulating a list of input sequences for each timestep rather than fixed-sized rows, as shown below by Figure 12.

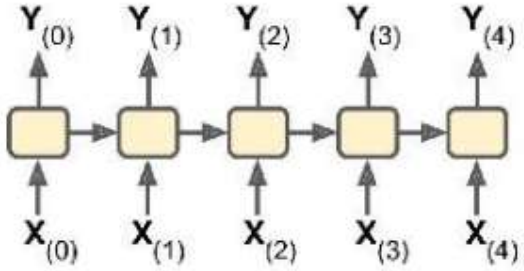


Figure 12: Sequence to Sequence Recurrent Network (sourced from [6])

3.5.1 Linear Regression

Initially, the Original Values of the hyperparameters were set following background reading of similar work as shown in Table IV. The data was separated to comprise of a training set with 870 records (8th of January 2016 - 26th of May 2018) and a test set with 218 records (27th of May 2018 – 30th of December 2018).

By performing feature scaling utilizing the StandardScaler transformer, all features were standardized to display a distribution with a mean of 0 and a standard deviation of 1. Following scaling, the PolynomialFeatures transformation was used to convert the original features into their higher order terms given a certain pre-defined polynomial degree.

The primary model was initialized, fitted with the training set, and using the test set predictions were made on the upcoming day's stock price. A plot indicating these predictions against their actual values is shown below by Figure 13, while the values of the Mean Absolute Percentage Error (MAPE) and the Root Mean-Squared Error (RMSE) are indicated in Table IV.

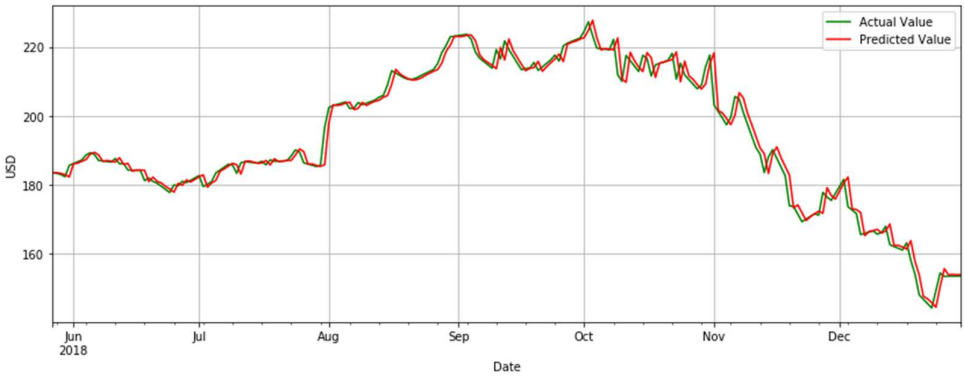


Figure 13: Primary Model Predictions using Initial Hyperparameters

By adjusting the features to experiment with all analysis types given in Table III above, making use of solely sentiment and historical data yielded the lowest error. This is summarized by Table V below.

Table V: Type of Analysis against RMSE & MAPE

Analysis Type	RMSE	MAPE
Just Historical Data	2.790	0.945
Technical Analysis	2.827	0.964
Sentiment Analysis	2.786	0.942
Technical & Sentiment Analysis	2.825	0.963

To calculate the optimal time step value, a range of timestep values between 3 and 34 were experimented with and a plot showing the error values against the timestep is shown by Figure 14(a) below. According to this, the optimal timestep parameter is 4 yielding a Root Mean Squared Error (RMSE) of 2.783.

Similarly, a range of polynomial degrees ranging from 1 to 4 were tested and a plot showing the error values against the polynomial degree is shown by Figure 14(b) below. The optimal parameter yielding the lowest Root Mean Squared Error (RMSE) is 1 which matches the original value therefore no changes were made.

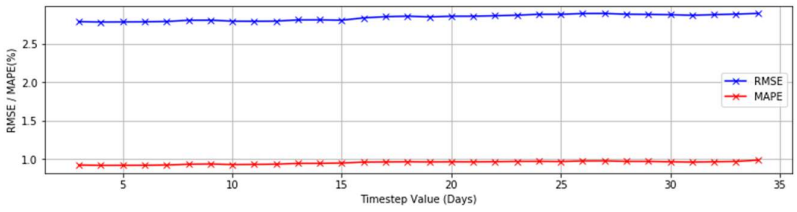


Figure 14(a): RMSE/MAPE(%) vs Timestep Value (Days)

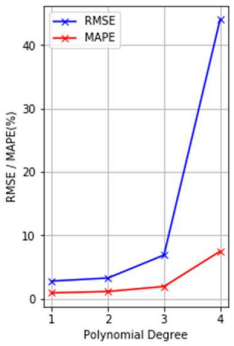


Figure 14(b): RMSE/MAPE(%) vs Polynomial Degree

By updating all the parameters to their tuned values, the final model stock price predictions are plotted as shown by Figure 15, yielding higher accuracy when compared to the initial model.

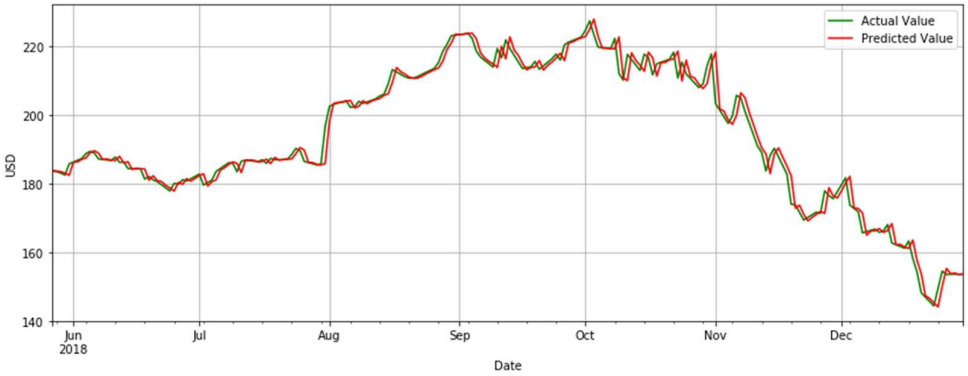


Figure 15: Final Model Predictions using Tuned Hyperparameters

Table IV below compares the error difference between the primary and final model following hyperparameter tuning.

Table IV: Primary vs Final Model Hyperparameter & Error Scores

Parameter	Primary Model	Final Model
Analysis Type	Technical & Sentiment Analysis	Sentiment Analysis
Timestep Value (Days)	3	4
Polynomial Degree	1	1
MAPE (%)	0.938	0.916
RMSE	2.819	2.783

3.5.2 Support Vector Regression (SVR)

Just like in Linear Regression, the Original Hyperparameter values were set following background reading of similar work as shown in Table VI. The data was separated in the same way as for Linear Regression with 870 training and 218 test instances.

A StandardScaler was used that normalizes all features to display a distribution with a mean of 0 and a standard deviation of 1. Polynomial Features were not utilized since Support Vector Regression (SVR) provides an integrated polynomial kernel, experimented with when tuning the hyperparameters.

The primary model was initialized, fitted with the training set, and using the test set predictions were made on the upcoming day's stock price. A plot indicating these predictions against their actual values is shown below by Figure 16, while the values of the Mean Absolute Percentage Error (MAPE) and the Root Mean-Squared Error (RMSE) are indicated in Table VI.

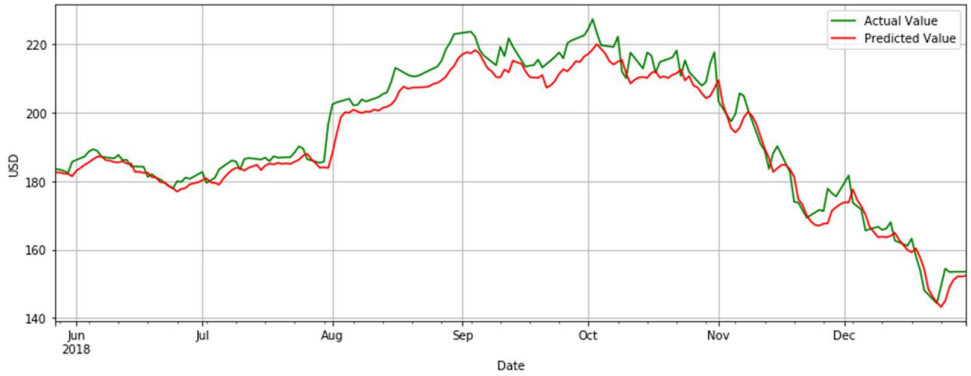


Figure 16: Primary Model Predictions using Initial Hyperparameters

By adjusting the features to experiment with all analysis types given in Table III above, making use of solely historical data yielded the lowest error. This is summarized by Table VII below.

Table VII: Type of Analysis against RMSE & MAPE

Analysis Type	RMSE	MAPE
Just Historical Data	32.420	15.523
Technical Analysis	33.063	16.109
Sentiment Analysis	33.387	16.028
Technical & Sentiment Analysis	34.441	16.850

To calculate the optimal time step value a range of timestep values between 3 and 34 were experimented with and a plot showing the error values against the timestep is shown by Figure 17 below. According to this, the optimal timestep parameter is 3 yielding a Root Mean Squared Error (RMSE) of 4.222.

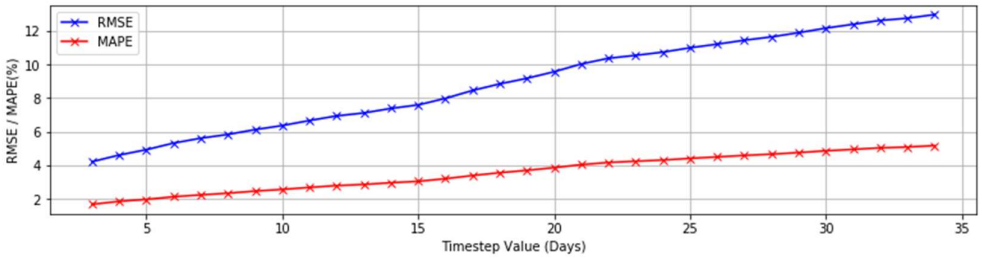


Figure 17: RMSE/MAPE(%) vs Timestep Value (Days)

The optimal γ and C parameters were experimented with together because their values must match one another. The values of C tested include [0.1, 1, 10, 50, 100] while the values of γ include [0.0001, 0.001, 0.005, 0.1, 1, 3, 5]. A plot showing the Root Mean Squared Error (RMSE) against γ for every C value tested is shown by Figure 18 below. The optimal parameters for γ and C yielding the lowest Root Mean Square Error (RMSE) of 5.022 are C: 200 and γ: 0.0001 as shown in Table VI.

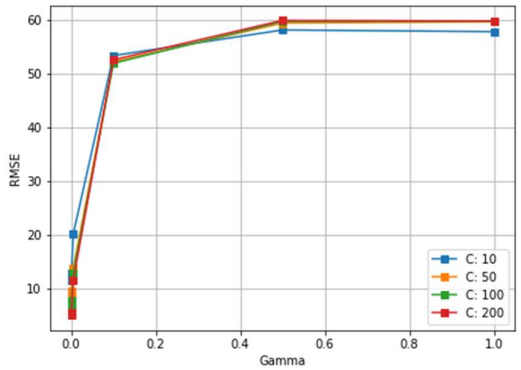


Figure 18: RMSE vs Gamma (γ) for different values of C

Several ε values were also tested in hope to determine a value that yields the lowest error. The values experimented with include [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05] and their errors are shown below by Figure 19(a). The ε value yielding the lowest Root Mean Square Error (RMSE) of 3.238 is 0.01. Finally, the optimal kernel was discovered as linear by experimenting with the following four types [‘linear’, ‘rbf’, ‘poly’, ‘sigmoid’]. A plot indicating each kernel in comparison with its error is shown by Figure 19(b) below where a linear kernel yields the lowest error.

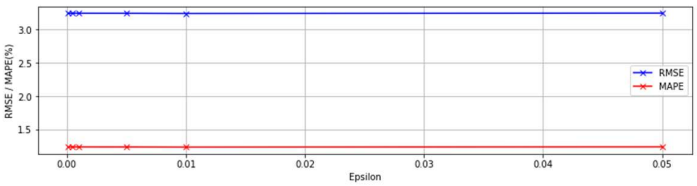


Figure 19(a): RMSE/MAPE(%) vs Epsilon (ε)

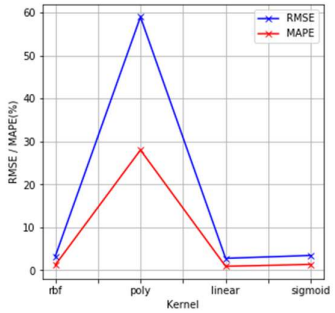


Figure 19(b): RMSE/MAPE(%) vs Kernel Type

By updating all the parameters to their tuned values, the final model stock price predictions are plotted as shown by Figure 20, yielding higher accuracy when compared to the initial model.

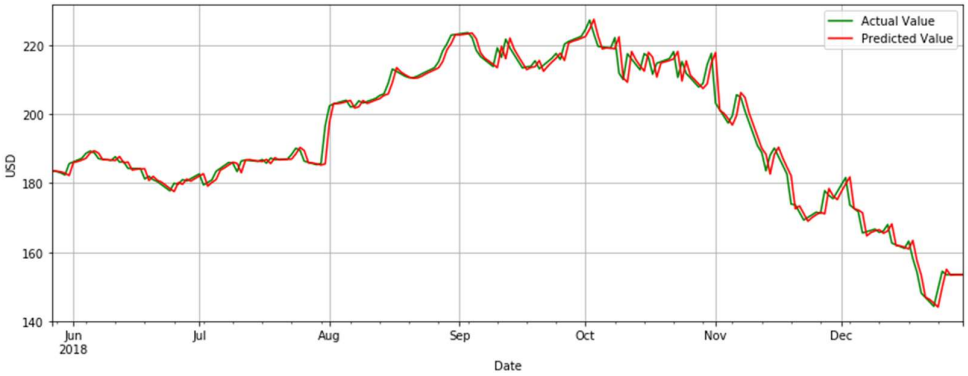


Figure 20: Final Model Predictions using Tuned Hyperparameters

Table VI below compares the error difference between the primary and final model following hyperparameter tuning.

Table VI: Primary vs Final Model Hyperparameter & Error Scores

Parameter	Primary Model	Final Model
Analysis Type	Technical & Sentiment Analysis	Just Historical Data
Timestep Value (Days)	3	3
C	10	200
Gamma (γ)	0.001	0.0001
Epsilon (ε)	0.01	0.01
Kernel Type	rbf	linear
MAPE (%)	1.795	0.923
RMSE	4.451	2.784

3.5.3 Long Short-Term Memory (LSTM)

Just like in the previous examples, original hyperparameter values were set following background reading of similar work as shown in Table VIII. The data was separated in the same way with 870 training and 218 test instances.

A custom scaling method was utilized for this algorithm that scales each feature per timestep sample to have a mean of 0 and a variance of 1 resulting in the best performance. For example, for predictions on day N, all features of the last timestep days (days N – timestep to N – 1) are set to a mean of 0 and a variance of 1.

A model was built that composed of two Long Short-Term Memory Layers followed by two dropout layers to avoid overfitting. Finally, a hidden layer was placed at the end to calculate next day's closing price. This is summarized by Figure 21.

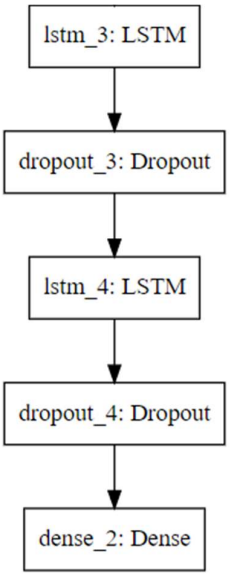


Figure 21: Recurrent Neural Network Model Summary

The primary model was initialized, fitted with the training set, and using the test set predictions were made on the upcoming day's stock price. A plot indicating these predictions against their actual values is shown below by Figure 22, while the values of the Mean Absolute Percentage Error (MAPE) and the Root Mean-Squared Error (RMSE) are indicated in Table VIII.

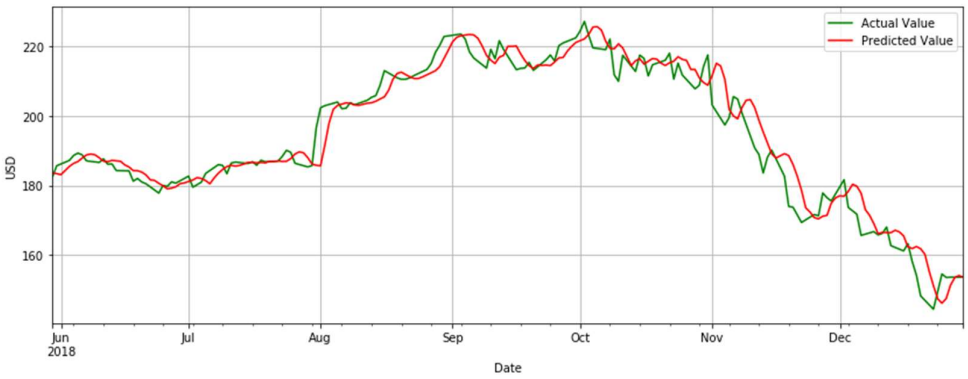


Figure 22: Primary Model Predictions using Initial Hyperparameters

By adjusting the features to experiment with all analysis types given in Table III above, making use of Technical & Sentiment Analysis yielded the lowest error. This is summarized by Table IX below.

Table IX: Type of Analysis against RMSE & MAPE

Analysis Type	RMSE	MAPE
Just Historical Data	4.813	11.991
Technical Analysis	4.603	12.013
Sentiment Analysis	4.559	12.007
Technical & Sentiment Analysis	4.479	11.990

To calculate the optimal time step value a range of timestep values between 3 and 34 were experimented with and a plot showing the error values against the timestep is shown by Figure 23 below. According to this, the optimal timestep parameter is 3 yielding a Root Mean Squared Error (RMSE) of 4.479.

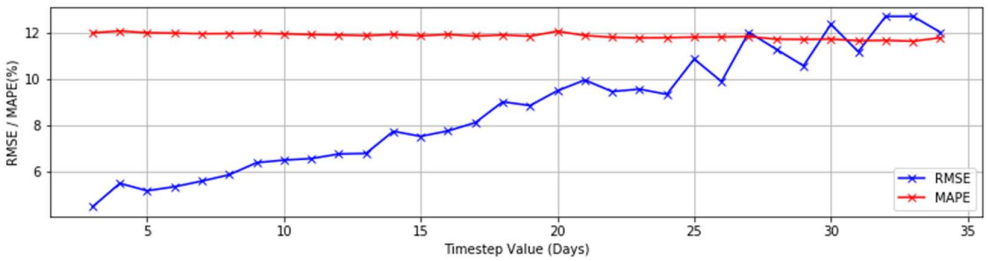


Figure 23: RMSE/MAPE(%) vs Timestep Value (Days)

The Epochs and Batch Size parameters were experimented with together because their values must affect one another. The values of epochs tested include [5, 10, 20, 50, 80, 100] while the values of Batch Size include [4, 8, 16, 32, 64, 128]. A plot showing the Root Mean Squared Error (RMSE) against Epochs for every Batch Size value tested is shown

by Figure 24 below. The optimal parameters for Epochs and Batch Size yielding the lowest Root Mean Square Error (RMSE) of 4.484 are Batch Size: 200 and Epochs: 50 as shown in Table VIII.

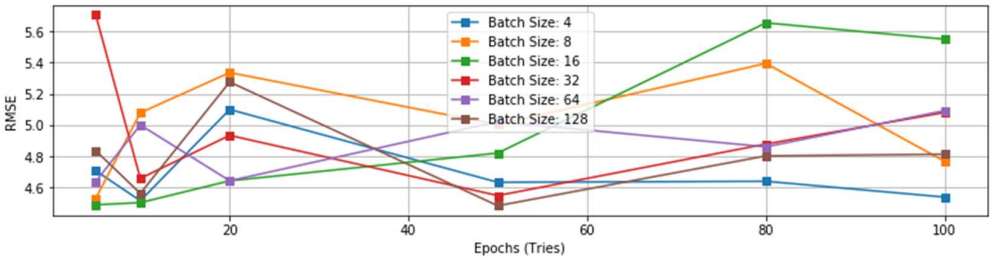


Figure 24: RMSE vs Epochs (Tries) for different values of Batch Size

Likewise, the total neurons per layer and the dropout probability values were tested together. The values of neurons tested include [5, 10, 15, 20, 40] while the values of dropout probability include [0.5, 0.6, 0.7, 0.8, 0.9, 1]. A plot showing the Root Mean Square Error (RMSE) against neurons for every dropout probability value tested is shown by Figure 25 below. The optimal parameters for neurons and dropout yielding the lowest Root Mean Square Error (RMSE) of 4.478 are Neurons / LSTM Units: 200 and Dropout Probability: 0.5 as shown in Table VIII.

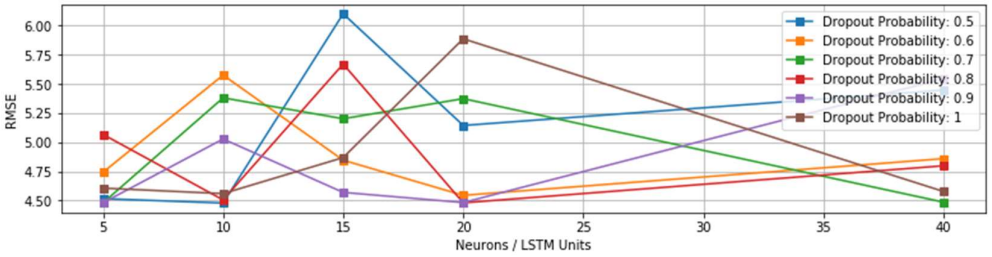


Figure 25: RMSE vs Neurons / LSTM Units for different Dropout Probability Values

Several optimizers were also tested [‘adam’, ‘sgd’, ‘rmsprop’, ‘adagrad’, ‘adadelta’, ‘adamax’, ‘nadam’] and their RMSE values are shown by Figure 26 below. The optimal optimizer is adamax as indicated below by Table VIII.

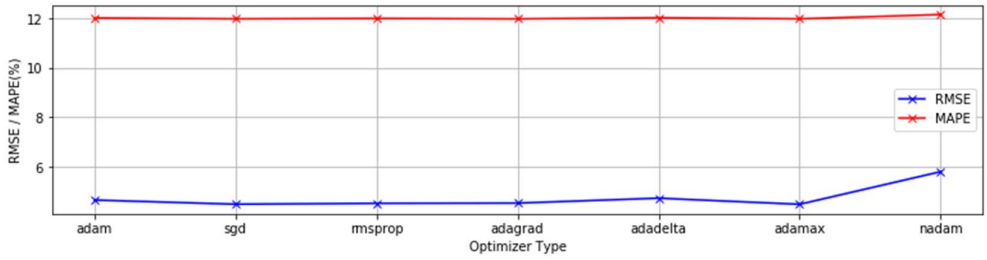


Figure 26: RMSE/MAPE(%) vs Optimizer Type

By updating all the parameters to their tuned values, the final model stock price predictions are plotted as shown by Figure 27, yielding higher accuracy when compared to the initial model.

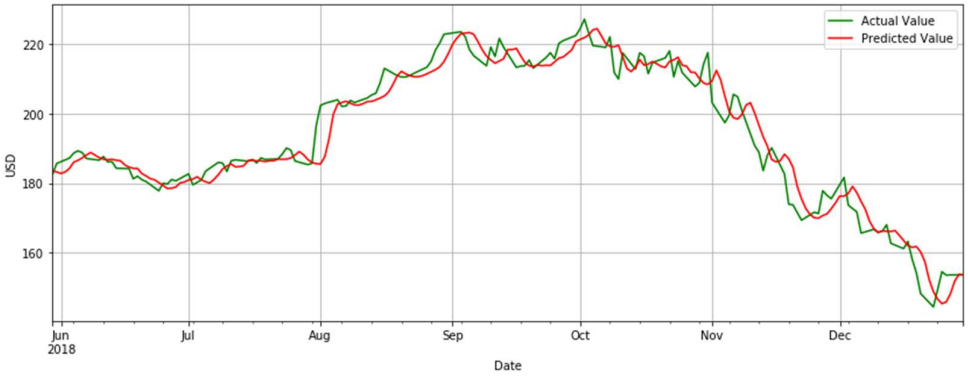


Figure 27: Final Model Predictions using Tuned Hyperparameters

Table VIII below compares the error difference between the primary and final model following hyperparameter tuning.

Table VIII: Primary vs Final Model Hyperparameter & Error Scores

Parameter	Primary Model	Final Model
Analysis Type	Technical & Sentiment Analysis	Technical & Sentiment Analysis
Timestep Value (Days)	3	3
Epochs	1	50
Batch Size	8	128
Neurons / LSTM Units	128	10
Dropout Probability	1	0.5
Optimizer Type	nadam	adamax
MAPE	12.062	12.001
RMSE	5.001	4.523

3.6 Evaluation & Automated Day-Trading Bot

A comparison of all three algorithms tested is shown below by Table X indicating that Linear Regression is the algorithm that yields the lowest error followed by Support Vector Regression (SVR). Long Short-Term Memory (LSTM) was expected to indicate the highest accuracy however ended up showing the worst overall performance.

Table X: Overall Algorithm Comparison

Algorithm	Linear Regression	Support Vector Regression (SVR)	Long Short-Term Memory (LSTM)
RMSE	2.783	2.784	4.523

Linear Regression, being the most promising algorithm was used to build a virtual day-trading bot that uses predictions made during the test set period (June 2018 – January 2019) to signal stock trades.

The way it works is simple; primarily the user must input how much money they are willing to invest (in our case: $10,000). Using the training set, the biggest change in stock price is determined as a base point. Daily, the predicted stock price for the upcoming day is determined. If the stock price is predicted to rise the total money is divided by the current

stock price to determine how many shares can be bought that day. This is compared to the maximum possible change seen in the training data to determine if the predicted rise is significant. The more significant the rise, the more shares that are bought. If on the contrary the stock price is predicted to fall, all stocks in the user's portfolio are monitored individually to examine whether the fall will lead to a loss forcing the algorithm to short sell the shares. Depending on the prediction, money is either withdrawn or inserted to the user's wallet. On the final day, all stocks are sold to display the total Profit yielded.

By using the entire test set for Day-Trading bot, a loss of $341.7 is yielded. This is to do with the negative trend in stock price that Apple has showed between the beginning of November to the end of December. By making use of the entire test set excluding the last 60 days (November & December) a profit of $98.24 is made.

The process of the Day-Trading bot for both the entire test set B1 and the test set excluding November and December B2 is shown in Appendix B.

4 Project Planning & Management

4.1 Risk Assessment

Table XI shows the initial risk mitigation plan clearly indicating the Probability of the risk occurring (on a scale of 1 least likely to 5 most likely), the Severity (on a scale of 1 low to 5 high) and the Risk Exposure which is calculated as shown below:

$$Risk\ Exposure = Probability \times Severity$$

Table XI: Initial Risk Mitigation Plan

Risk	Probability	Severity	Risk Exposure	Mitigation Strategy
Rejection of Ethics and API approval request for Headline Scraping	3	4	12	Submit application early to allow for time to change data sources (or strategy) in the case of rejection. If all applications get rejected, textual data will not be used.
Loss of Data (Code and Report Writing)	2	5	10	Regularly backup ALL data on the ECS file store and Google Drive.
Hardware not capable of handling large datasets.	3	3	9	Make use of the ECS computers due to their high performance. Decrease the dataset size if necessary or use alternate parameters (Stochastic Gradient Descent)
No access to ECS Computer Lab Facilities due to the Covid-19 outbreak.	5	2	10	Make use of the remote desktop service provided by the university. Home computer should be able to handle.
Unexpected Additional Tasks / Tasks Taking Longer than Anticipated	4	2	8	Constantly monitor the Gantt Chart, rescheduling initial tasks if needed. Make sure that there is extra time in the schedule between the finished implementation and project deadline.
Severe Illness	2	5	10	Make sure that there is extra time in the schedule between the finished implementation and project deadline. Depending on the severity, request Special Considerations.

4.2 Schedule Management

Even though most risks were anticipated early on and extra time was allocated between the final implementation and the project deadline to allow for unexpected tasks, the initial project schedule was not fully followed. Additionally, due to the Covid-19 virus outbreak, deadlines were pushed back reflecting on the schedule.

This project has been split into four different stages: Background Research (Green), Implementation (Orange), Testing & Evaluation (Orange) and Report Writing (Blue). The final Gantt Chart, as shown by Figure 28, displays all tasks carried out during this project.

The project implementation was completed over a month earlier than the submission deadline to allow for a flexible time schedule in the case of any risk weighted problems. Additionally, certain tasks were parallelized to balance the workload from technical implementation and report writing yielding better quality results.

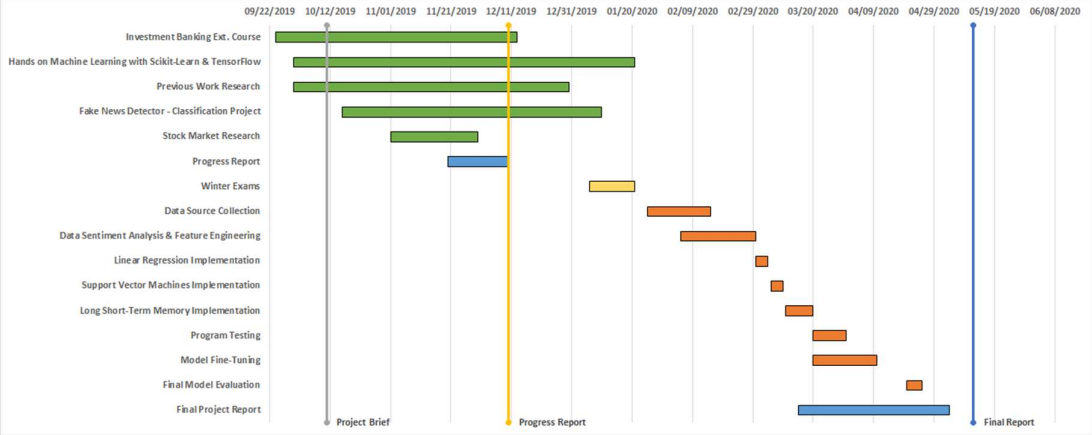


Figure 28: Final Gantt Chart

Finally, background reading was not included in the Gantt Chart as it was carried throughout the entire year as an iterative process.

Original (Progress Report) Gantt Chart is shown in Appendix A

5 Conclusions

This project has successfully demonstrated the application of three different machine learning algorithms in forecasting the upcoming stock price of a financial security. By making use of different feature engineering techniques with sentiment analysis, this project has developed models with accuracies that are comparable to industry standards.

In machine learning, it all comes down to having "sufficient data, computing power and patience". [6] On top of that, the performance of a machine learning algorithm will largely depend on the quality of data and pre-processing procedures. Even though most sources showed credible factual integrity, most published headlines are neutral or relatively positive and do not cause an effect on the stock price therefore sentiment values portrayed low feature importance.

VADER played a very important (probably the most important) role in this investigation and even though its considered to be the leading sentiment analyser, it was one of the feature engineering techniques that lead to the highest error rate due to the complexity in emotion portrayed by financial news.

On a general note, it is seen by the results that all algorithms project a high Root Mean Square Error (RMSE). This is largely to do with the complexity of the test set chosen as Apple saw random shocks in stock price for that specific period. Even though Long Short-Term Memory (LSTM) was expected to yield the best results, Linear Regression had the lowest error rate followed by Support Vector Regression (SVR).

By showing plots of predicted and actual values, a common problem is lag. Lag is what makes predictions obsolete and is the biggest reason day trading does not work out for a lot of people. On top of that people need to account for the effects of commission and slippage that take a larger percentage out of profits than when carrying out long term trades.

Finally, the stock market is a very complex system where machine learning algorithms see historical stock data as random noise. Factors such as Technical Indicators and Sentiment Analysis values have the potential to improve predictions but only slightly for the time being. Future data availability and algorithm innovation will allow for better models.

5.1 Future Work

There are possible improvements in both the data and the algorithm.

On the dataset, future work will extend this project to yield better results by using data from more news sources. Additionally, editors offer better quality data to people that are willing to pay for it therefore, premium data can be integrated into the project leading to higher feature importance. Additionally, a topic modelling feature engineering step can be implemented to detect headlines that are only relevant to that specific security using a machine learning algorithm rather than by purely using keywords.

Even though regression models allowed for predictions plots that were useful throughout this report, most people tend to use classification models that trigger a Buy/Sell signal leading to better results. Future works will experiment with classification models as well as statistical models that leverage last values and moving averages in closing price.

Word Count: 9551 words

References

[1] Osman Hegazy, Omar S. Soliman and Mustafa Abdul Salam - Machine Learning Model for Stock Market Prediction, 2013: https://www.researchgate.net/publication/259240183_A_Machine_Learning_Model_for_Stock_Market_Prediction

[2] Technical Vs. Fundamental Analysis - Which Is Best For You, My Trading Skills, 2020: https://www.mytradingskills.com/forex-for-beginners/technical-vs-fundamental-analysis

[3] Richard Bowman, Sentiment Analysis - Market Sentiment And How It Affects The Stock Market, Catana Capital, 2019: https://catanacapital.com/blog/sentiment-analysis-stock-market-sentiment/#market-sentiment

[4] A. Jackson and A. O'Shea, Stock Market Guide: What Is the Stock Market and How Does it Work?, NerdWallet, 2019: https://www.nerdwallet.com/blog/investing/what-is-the-stock-market/

[5] Brett Grossfeld, A simple way to understand machine learning vs deep learning - Zendesk, 2020, https://www.zendesk.com/blog/machine-learning-and-deep-learning/

[6] A. Géron, Hands-on machine learning with Scikit-Learn and TensorFlow

[7] Marco Peixeiro, Linear Regression - Understanding The Theory, Towards Data Science, 2018, https://towardsdatascience.com/linear-regression-understanding-the-theory-7e53ac2831b5

[8] Support Vector Machines: A Simple Explanation, KDnuggets, 2016, https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html

[9] Savan Patel, SVM (Support Vector Machine) – Theory, Towards Data Science, 2017, https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72

[10] Support Vector Machine – Regression (SVR), http://saedsayad.com/support_vector_machine_reg.htm

[11] 3Blue1Brown, But what is a Neural Network? | Deep learning, chapter 1, YouTube, 2019 https://www.youtube.com/watch?v=aircAruvnKk

[12] Michael Nguyen, Illustrated Guide to LSTM's and GRU's: A step by step explanation, Towards Data Science, 2019, https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-bystep-explanation-44e9eb85bf21

[13] Miguel Fernández Zafra, Web Scraping News Articles In Python, Towards Data Science, 2019, https://towardsdatascience.com/web-scraping-news-articles-in-python-9dd605799558

[14] About Us | Brand, Business Standard, 2020, https://www.business-standard.com/about-us

[15] Developer FAQ, NYTimes, 2020. https://developer.nytimes.com/faq

[16] Sentiment Analysis Explained, Lexalytics, 2020, https://www.lexalytics.com/resources/Lexalytics_Sentiment_Analysis_Whitepaper.pdf

[17] Cognitive APIs with Watson: Sentiment Analysis, YouTube, 2016, https://www.youtube.com/watch?v=xU6nkznjr34

[18] Software Repository for Accounting And Finance, University of Notre Dame, 2018, https://sraf.nd.edu/textual-analysis/resources/#LM%20Sentiment%20Word%20Lists

[19] Oliveira, Nuno, Paulo Cortez, and Nelson Areal, Stock market sentiment lexicon acquisition using microblogging data and statistical measures, Decision Support Systems 85, 62-73, 2016

Appendix A – Original Project Brief & Gantt Chart

Problem
In financial markets, stock prices are heavily affected by various external factors ranging from minor company announcements or shifts between the interest rate, to political restructures and natural disasters. This poses trouble for investment bankers and traders, as they are not able to keep up with all these rapidly changing factors.

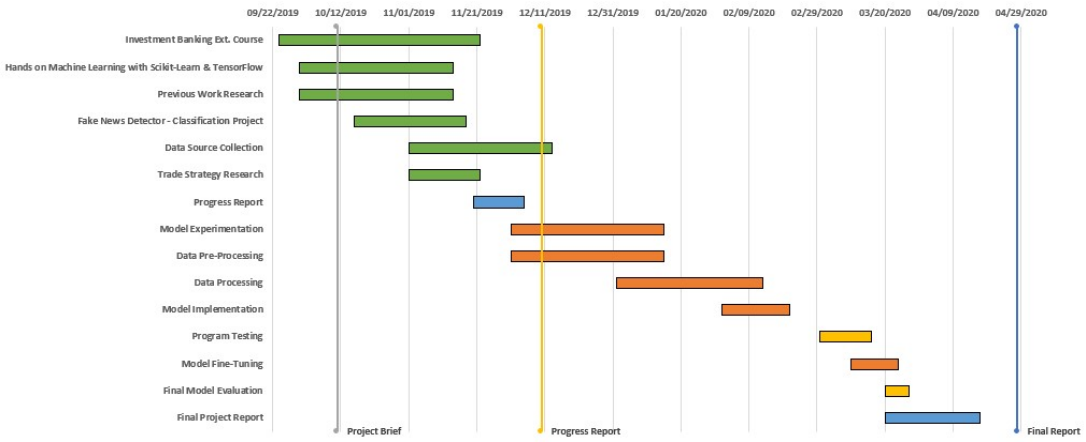
A strong topic of interest is the relationship between the past and future predictions of market prices using technical analysis on previous time-series data. Considering the increase in the availability of financial information, stock price prediction using various machine learning algorithms is a trending subject amongst investors and researchers.

Goals
The main project goal is to perform an investigation on the performance of various stock price prediction models using both technical and sentiment analysis. This will be achieved through the following objectives:

- Automate data gathering to collect separate datasets such as news articles/tweets/etc., timeseries data of stock prices and trading volume.
- Perform data cleaning on the data utilizing sentiment analysis to determine the positivity and relevance of an article.
- Train various machine learning algorithms on the dataset to predict the future trend of the market price.
- Compare the performance of the implemented algorithms and constantly re-evaluate them by alternating hyperparameters and performing alternate data manipulations to find different trading strategies.

Scope
Depending on the complexity of the algorithm, this project will be scoped towards investment bankers or regular experienced traders aiding them to decide how the market will perform. Data collection APIs impose limits restricting access to large training date periods.

Therefore, this project will not be able to predict with complete accuracy and will need human intel as well. It is more focused on evaluating the efficiency of various data manipulations and machine learning algorithms.



Original Gantt Chart (Progress Report)

Appendix B – Algorithmic Trading Bot Outputs

B1 Test Run (27-05-2018 – 30-12-2018) – Loss Example

1. 2018-05-27: 1 Shares Bought for: $183.56 Each, Total: $183.56
2. 2018-05-27: 1 Shares Bought for: $183.56 Each, Total: $183.56
3. 2018-05-28: 1 Shares Sold for: $183.39 Each. Profit Made: $-0.17
4. 2018-05-28: 1 Shares Sold for: $183.39 Each. Profit Made: $-0.17
5. 2018-05-31: 2 Shares Bought for: $185.68 Each, Total: $371.36
6. 2018-06-01: 1 Shares Bought for: $186.2 Each, Total: $186.2
7. 2018-06-02: 2 Shares Bought for: $186.71 Each, Total: $373.43
8. 2018-06-03: 1 Shares Bought for: $187.23 Each, Total: $187.23
9. 2018-06-04: 1 Shares Bought for: $188.68 Each, Total: $188.68
10. 2018-06-05: 1 Shares Bought for: $189.33 Each, Total: $189.33
11. 2018-06-06: 1 Shares Sold for: $188.82 Each. Profit Made: $-0.51
12. 2018-06-07: 1 Shares Sold for: $187.1 Each. Profit Made: $-1.57
13. 2018-06-07: 1 Shares Sold for: $187.1 Each. Profit Made: $-0.13
14. 2018-06-08: 1 Shares Bought for: $186.95 Each, Total: $186.95
15. 2018-06-09: 1 Shares Bought for: $186.8 Each, Total: $186.8
16. 2018-06-10: 1 Shares Sold for: $186.65 Each. Profit Made: $-0.15
17. 2018-06-10: 1 Shares Sold for: $186.65 Each. Profit Made: $-0.31
18. 2018-06-10: 2 Shares Sold for: $186.65 Each. Profit Made: $-0.14
19. 2018-06-11: 1 Shares Bought for: $187.67 Each, Total: $187.67
20. 2018-06-12: 1 Shares Sold for: $186.13 Each. Profit Made: $-1.54
21. 2018-06-12: 1 Shares Sold for: $186.13 Each. Profit Made: $-0.07
22. 2018-06-13: 1 Shares Bought for: $186.23 Each, Total: $186.23
23. 2018-06-14: 1 Shares Sold for: $184.31 Each. Profit Made: $-1.91
24. 2018-06-14: 2 Shares Sold for: $184.31 Each. Profit Made: $-2.73
25. 2018-06-15: 1 Shares Bought for: $184.28 Each, Total: $184.28
26. 2018-06-16: 1 Shares Sold for: $184.25 Each. Profit Made: $-0.03
27. 2018-06-19: 1 Shares Bought for: $182.03 Each, Total: $182.03
28. 2018-06-20: 1 Shares Sold for: $181.01 Each. Profit Made: $-1.02
29. 2018-06-21: 1 Shares Bought for: $180.49 Each, Total: $180.49
30. 2018-06-22: 1 Shares Sold for: $179.59 Each. Profit Made: $-0.89
31. 2018-06-25: 1 Shares Bought for: $180.01 Each, Total: $180.01
32. 2018-06-26: 1 Shares Sold for: $179.74 Each. Profit Made: $-0.26
33. 2018-06-27: 2 Shares Bought for: $181.05 Each, Total: $362.11
34. 2018-06-28: 2 Shares Sold for: $180.67 Each. Profit Made: $-0.76
35. 2018-06-29: 1 Shares Bought for: $181.35 Each, Total: $181.35
36. 2018-06-30: 1 Shares Bought for: $182.02 Each, Total: $182.02
37. 2018-07-01: 1 Shares Bought for: $182.69 Each, Total: $182.69
38. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-3.18
39. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-2.51
40. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-1.83
41. 2018-07-03: 1 Shares Bought for: $180.23 Each, Total: $180.23
42. 2018-07-04: 1 Shares Bought for: $180.95 Each, Total: $180.95
43. 2018-07-05: 2 Shares Bought for: $183.46 Each, Total: $366.93
44. 2018-07-06: 1 Shares Bought for: $184.31 Each, Total: $184.31
45. 2018-07-07: 2 Shares Bought for: $185.16 Each, Total: $370.32
46. 2018-07-08: 1 Shares Bought for: $186.01 Each, Total: $186.01
47. 2018-07-09: 1 Shares Bought for: $185.79 Each, Total: $185.79
48. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-2.41
49. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-2.64
50. 2018-07-10: 2 Shares Sold for: $183.38 Each. Profit Made: $-3.57
51. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-0.94
52. 2018-07-10: 2 Shares Sold for: $183.38 Each. Profit Made: $-0.18
53. 2018-07-11: 2 Shares Bought for: $186.45 Each, Total: $372.9
54. 2018-07-12: 1 Shares Bought for: $186.74 Each, Total: $186.74
55. 2018-07-13: 1 Shares Bought for: $186.61 Each, Total: $186.61
56. 2018-07-14: 1 Shares Sold for: $186.47 Each. Profit Made: $-0.14
57. 2018-07-14: 1 Shares Sold for: $186.47 Each. Profit Made: $-0.27
58. 2018-07-15: 1 Shares Bought for: $186.33 Each, Total: $186.33
59. 2018-07-16: 1 Shares Bought for: $186.86 Each, Total: $186.86
60. 2018-07-17: 1 Shares Sold for: $185.84 Each. Profit Made: $-1.02
61. 2018-07-17: 1 Shares Sold for: $185.84 Each. Profit Made: $-0.5
62. 2018-07-17: 2 Shares Sold for: $185.84 Each. Profit Made: $-1.23
63. 2018-07-18: 2 Shares Bought for: $187.28 Each, Total: $374.56
64. 2018-07-19: 2 Shares Sold for: $186.85 Each. Profit Made: $-0.86
65. 2018-07-21: 1 Shares Bought for: $186.96 Each, Total: $186.96
66. 2018-07-22: 1 Shares Bought for: $187.02 Each, Total: $187.02
67. 2018-07-23: 1 Shares Bought for: $188.37 Each, Total: $188.37
68. 2018-07-24: 1 Shares Bought for: $190.15 Each, Total: $190.15
69. 2018-07-25: 1 Shares Bought for: $189.55 Each, Total: $189.55
70. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-3.15
71. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-3.75
72. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-1.97
73. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-0.61
74. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-0.56
75. 2018-07-30: 1 Shares Bought for: $185.73 Each, Total: $185.73
76. 2018-07-31: 5 Shares Bought for: $196.67 Each, Total: $983.34
77. 2018-08-01: 3 Shares Bought for: $202.42 Each, Total: $607.25
78. 2018-08-02: 2 Shares Bought for: $203.0 Each, Total: $406.01
79. 2018-08-03: 1 Shares Bought for: $203.35 Each, Total: $203.35
80. 2018-08-04: 1 Shares Bought for: $203.71 Each, Total: $203.71
81. 2018-08-05: 1 Shares Bought for: $204.06 Each, Total: $204.06
82. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.91
83. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.56
84. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.21
85. 2018-08-06: 2 Shares Sold for: $202.14 Each. Profit Made: $-1.72
86. 2018-08-06: 3 Shares Sold for: $202.14 Each. Profit Made: $-0.82
87. 2018-08-07: 1 Shares Bought for: $202.28 Each, Total: $202.28
88. 2018-08-08: 1 Shares Bought for: $203.87 Each, Total: $203.87
89. 2018-08-09: 1 Shares Sold for: $203.26 Each. Profit Made: $-0.61
90. 2018-08-10: 1 Shares Bought for: $203.7 Each, Total: $203.7
91. 2018-08-11: 1 Shares Bought for: $204.14 Each, Total: $204.14
92. 2018-08-12: 1 Shares Bought for: $204.58 Each, Total: $204.58
93. 2018-08-13: 1 Shares Bought for: $205.44 Each, Total: $205.44
94. 2018-08-14: 1 Shares Bought for: $205.92 Each, Total: $205.92
95. 2018-08-15: 1 Shares Bought for: $208.94 Each, Total: $208.94
96. 2018-08-16: 2 Shares Bought for: $213.11 Each, Total: $426.22
97. 2018-08-17: 1 Shares Bought for: $212.42 Each, Total: $212.42
98. 2018-08-18: 1 Shares Bought for: $211.72 Each, Total: $211.72
99. 2018-08-19: 1 Shares Sold for: $211.03 Each. Profit Made: $-0.69
100. 2018-08-19: 1 Shares Sold for: $211.03 Each. Profit Made: $-1.38
101. 2018-08-19: 2 Shares Sold for: $211.03 Each. Profit Made: $-4.15
102. 2018-08-21: 1 Shares Bought for: $210.63 Each, Total: $210.63
103. 2018-08-22: 1 Shares Bought for: $211.06 Each, Total: $211.06
104. 2018-08-23: 1 Shares Bought for: $211.72 Each, Total: $211.72
105. 2018-08-24: 1 Shares Bought for: $212.3 Each, Total: $212.3
106. 2018-08-25: 1 Shares Bought for: $212.88 Each, Total: $212.88
107. 2018-08-26: 1 Shares Bought for: $213.46 Each, Total: $213.46
108. 2018-08-27: 1 Shares Bought for: $215.18 Each, Total: $215.18
109. 2018-08-28: 1 Shares Bought for: $218.4 Each, Total: $218.4
110. 2018-08-29: 1 Shares Bought for: $220.41 Each, Total: $220.41
111. 2018-08-30: 1 Shares Bought for: $222.95 Each, Total: $222.95
112. 2018-08-31: 1 Shares Bought for: $223.13 Each, Total: $223.13
113. 2018-09-01: 1 Shares Bought for: $223.31 Each, Total: $223.31
114. 2018-09-02: 1 Shares Bought for: $223.49 Each, Total: $223.49
115. 2018-09-03: 1 Shares Bought for: $223.67 Each, Total: $223.67
116. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.46
117. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.28
118. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.1

119. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-0.92
120. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-0.74
121. 2018-09-05: 1 Shares Sold for: $218.51 Each. Profit Made: $-1.89
122. 2018-09-06: 1 Shares Sold for: $216.75 Each. Profit Made: $-1.65
123. 2018-09-08: 1 Shares Sold for: $214.81 Each. Profit Made: $-0.37
124. 2018-09-10: 2 Shares Bought for: $219.25 Each, Total: $438.5
125. 2018-09-11: 2 Shares Sold for: $216.53 Each. Profit Made: $-5.45
126. 2018-09-12: 2 Shares Bought for: $221.76 Each, Total: $443.51
127. 2018-09-13: 2 Shares Sold for: $219.24 Each. Profit Made: $-5.03
128. 2018-09-16: 1 Shares Sold for: $213.4 Each. Profit Made: $-0.06
129. 2018-09-19: 1 Shares Bought for: $215.51 Each, Total: $215.51
130. 2018-09-20: 1 Shares Sold for: $213.19 Each. Profit Made: $-2.32
131. 2018-09-21: 1 Shares Bought for: $214.21 Each, Total: $214.21
132. 2018-09-22: 1 Shares Bought for: $215.23 Each, Total: $215.23
133. 2018-09-23: 1 Shares Bought for: $216.25 Each, Total: $216.25
134. 2018-09-24: 1 Shares Bought for: $217.62 Each, Total: $217.62
135. 2018-09-25: 1 Shares Sold for: $215.89 Each. Profit Made: $-1.73
136. 2018-09-25: 1 Shares Sold for: $215.89 Each. Profit Made: $-0.36
137. 2018-09-26: 1 Shares Bought for: $220.33 Each, Total: $220.33
138. 2018-09-27: 1 Shares Bought for: $221.1 Each, Total: $221.1
139. 2018-09-28: 1 Shares Bought for: $221.6 Each, Total: $221.6
140. 2018-09-29: 1 Shares Bought for: $222.09 Each, Total: $222.09
141. 2018-09-30: 1 Shares Bought for: $222.59 Each, Total: $222.59
142. 2018-10-01: 1 Shares Bought for: $224.57 Each, Total: $224.57
143. 2018-10-02: 1 Shares Bought for: $227.3 Each, Total: $227.3
144. 2018-10-03: 1 Shares Sold for: $223.3 Each. Profit Made: $-4.0
145. 2018-10-03: 1 Shares Sold for: $223.3 Each. Profit Made: $-1.26
146. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-2.91
147. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-2.41
148. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-1.92
149. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-1.42
150. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-0.65
151. 2018-10-08: 1 Shares Bought for: $222.21 Each, Total: $222.21
152. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-10.29
153. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-3.32
154. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-2.3
155. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-0.97
156. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-0.39
157. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-1.67
158. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-1.02
159. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-0.59
160. 2018-10-11: 1 Shares Bought for: $217.55 Each, Total: $217.55
161. 2018-10-12: 1 Shares Sold for: $215.99 Each. Profit Made: $-1.55
162. 2018-10-15: 2 Shares Bought for: $217.58 Each, Total: $435.17
163. 2018-10-16: 2 Shares Sold for: $216.64 Each. Profit Made: $-1.88
164. 2018-10-18: 1 Shares Bought for: $214.8 Each, Total: $214.8
165. 2018-10-20: 1 Shares Bought for: $215.68 Each, Total: $215.68
166. 2018-10-21: 1 Shares Bought for: $216.12 Each, Total: $216.12
167. 2018-10-22: 1 Shares Bought for: $218.15 Each, Total: $218.15
168. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-7.48
169. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-5.45
170. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-5.01
171. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-4.13
172. 2018-10-24: 2 Shares Bought for: $215.28 Each, Total: $430.57
173. 2018-10-25: 2 Shares Sold for: $211.85 Each. Profit Made: $-6.86
174. 2018-10-26: 1 Shares Bought for: $210.53 Each, Total: $210.53
175. 2018-10-27: 1 Shares Sold for: $209.2 Each. Profit Made: $-1.33
176. 2018-10-28: 1 Shares Sold for: $207.88 Each. Profit Made: $-1.06
177. 2018-10-29: 1 Shares Bought for: $208.92 Each, Total: $208.92
178. 2018-10-30: 2 Shares Bought for: $214.36 Each, Total: $428.72
179. 2018-10-31: 2 Shares Bought for: $217.65 Each, Total: $435.31
180. 2018-11-01: 2 Shares Sold for: $203.22 Each. Profit Made: $-28.87
181. 2018-11-01: 2 Shares Sold for: $203.22 Each. Profit Made: $-22.29
182. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-5.7
183. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-2.7
184. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-2.22
185. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-1.36
186. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-0.92
187. 2018-11-01: 1 Shares Sold for: $203.22 Each. Profit Made: $-0.49
188. 2018-11-02: 1 Shares Sold for: $201.29 Each. Profit Made: $-0.99
189. 2018-11-05: 1 Shares Bought for: $199.58 Each, Total: $199.58
190. 2018-11-06: 2 Shares Bought for: $205.63 Each, Total: $411.27
191. 2018-11-07: 1 Shares Bought for: $204.92 Each, Total: $204.92
192. 2018-11-08: 1 Shares Sold for: $200.97 Each. Profit Made: $-3.95
193. 2018-11-08: 2 Shares Sold for: $200.97 Each. Profit Made: $-9.34
194. 2018-11-09: 1 Shares Sold for: $197.59 Each. Profit Made: $-1.99
195. 2018-11-10: 5 Shares Sold for: $194.22 Each. Profit Made: $-12.26
196. 2018-11-13: 1 Shares Sold for: $183.6 Each. Profit Made: $-2.13
197. 2018-11-14: 2 Shares Bought for: $188.13 Each, Total: $376.26
198. 2018-11-15: 1 Shares Bought for: $190.21 Each, Total: $190.21
199. 2018-11-16: 1 Shares Sold for: $187.7 Each. Profit Made: $-2.51
200. 2018-11-16: 2 Shares Sold for: $187.7 Each. Profit Made: $-0.86
201. 2018-11-19: 1 Shares Sold for: $173.95 Each. Profit Made: $-7.01
202. 2018-11-19: 1 Shares Sold for: $173.95 Each. Profit Made: $-6.29
203. 2018-11-23: 1 Shares Bought for: $170.1 Each, Total: $170.1
204. 2018-11-24: 1 Shares Bought for: $170.86 Each, Total: $170.86
205. 2018-11-25: 1 Shares Bought for: $171.63 Each, Total: $171.63
206. 2018-11-26: 1 Shares Bought for: $171.25 Each, Total: $171.25
207. 2018-11-27: 4 Shares Bought for: $177.84 Each, Total: $711.36
208. 2018-11-28: 4 Shares Sold for: $176.47 Each. Profit Made: $-5.46
209. 2018-11-29: 1 Shares Bought for: $175.52 Each, Total: $175.52
210. 2018-11-30: 1 Shares Bought for: $177.56 Each, Total: $177.56
211. 2018-12-01: 2 Shares Bought for: $179.61 Each, Total: $359.22
212. 2018-12-02: 2 Shares Bought for: $181.65 Each, Total: $363.31
213. 2018-12-03: 2 Shares Sold for: $173.66 Each. Profit Made: $-15.98
214. 2018-12-03: 2 Shares Sold for: $173.66 Each. Profit Made: $-11.89
215. 2018-12-03: 1 Shares Sold for: $173.66 Each. Profit Made: $-3.9
216. 2018-12-03: 1 Shares Sold for: $173.66 Each. Profit Made: $-1.86
217. 2018-12-06: 1 Shares Sold for: $165.6 Each. Profit Made: $-5.65
218. 2018-12-06: 1 Shares Sold for: $165.6 Each. Profit Made: $-6.02
219. 2018-12-06: 1 Shares Sold for: $165.6 Each. Profit Made: $-5.26
220. 2018-12-06: 1 Shares Sold for: $165.6 Each. Profit Made: $-4.5
221. 2018-12-07: 1 Shares Bought for: $165.97 Each, Total: $165.97
222. 2018-12-09: 1 Shares Bought for: $166.69 Each, Total: $166.69
223. 2018-12-10: 1 Shares Sold for: $165.74 Each. Profit Made: $-0.95
224. 2018-12-10: 1 Shares Sold for: $165.74 Each. Profit Made: $-0.23
225. 2018-12-11: 1 Shares Bought for: $166.2 Each, Total: $166.2
226. 2018-12-12: 2 Shares Bought for: $168.02 Each, Total: $336.04
227. 2018-12-13: 2 Shares Sold for: $162.64 Each. Profit Made: $-10.75
228. 2018-12-13: 1 Shares Sold for: $162.64 Each. Profit Made: $-3.56
229. 2018-12-14: 1 Shares Bought for: $162.14 Each, Total: $162.14
230. 2018-12-15: 1 Shares Sold for: $161.64 Each. Profit Made: $-0.5
231. 2018-12-17: 2 Shares Bought for: $163.22 Each, Total: $326.45
232. 2018-12-18: 2 Shares Sold for: $158.13 Each. Profit Made: $-10.18
233. 2018-12-24: 3 Shares Bought for: $149.4 Each, Total: $448.19
234. 2018-12-25: 4 Shares Bought for: $154.48 Each, Total: $617.91
235. 2018-12-26: 1 Shares Bought for: $153.47 Each, Total: $153.47
236. 2018-12-27: 2 Shares Bought for: $153.55 Each, Total: $307.11
237. 2018-12-28: 4 Shares Sold for: $153.55 Each. Profit Made: $-3.7
238. 2018-12-29: 2 Shares Sold for: $153.55 Each. Profit Made: $0.0
239. 2018-12-29: 1 Shares Sold for: $153.55 Each. Profit Made: $0.08
240. 2018-12-29: 3 Shares Sold for: $153.55 Each. Profit Made: $12.47
241. Total Money at the End: $9658.3 Profit Made: $-341.7

B2 Test Run (27-05-2018 – 30-10-2018) – Profit Example

1. 2018-05-27: 1 Shares Bought for: $183.56 Each, Total: $183.56
2. 2018-05-27: 1 Shares Bought for: $183.56 Each, Total: $183.56
3. 2018-05-28: 1 Shares Sold for: $183.39 Each. Profit Made: $-0.17
4. 2018-05-28: 1 Shares Sold for: $183.39 Each. Profit Made: $-0.17
5. 2018-05-31: 2 Shares Bought for: $185.68 Each, Total: $371.36
6. 2018-06-01: 1 Shares Bought for: $186.2 Each, Total: $186.2
7. 2018-06-02: 2 Shares Bought for: $186.71 Each, Total: $373.43
8. 2018-06-03: 1 Shares Bought for: $187.23 Each, Total: $187.23
9. 2018-06-04: 1 Shares Bought for: $188.68 Each, Total: $188.68
10. 2018-06-05: 1 Shares Bought for: $189.33 Each, Total: $189.33
11. 2018-06-06: 1 Shares Sold for: $188.82 Each. Profit Made: $-0.51
12. 2018-06-07: 1 Shares Sold for: $187.1 Each. Profit Made: $-1.57
13. 2018-06-07: 1 Shares Sold for: $187.1 Each. Profit Made: $-0.13
14. 2018-06-08: 1 Shares Bought for: $186.95 Each, Total: $186.95
15. 2018-06-09: 1 Shares Bought for: $186.8 Each, Total: $186.8
16. 2018-06-10: 1 Shares Sold for: $186.65 Each. Profit Made: $-0.15
17. 2018-06-10: 1 Shares Sold for: $186.65 Each. Profit Made: $-0.31
18. 2018-06-10: 2 Shares Sold for: $186.65 Each. Profit Made: $-0.14
19. 2018-06-11: 1 Shares Bought for: $187.67 Each, Total: $187.67
20. 2018-06-12: 1 Shares Sold for: $186.13 Each. Profit Made: $-1.54
21. 2018-06-12: 1 Shares Sold for: $186.13 Each. Profit Made: $-0.07
22. 2018-06-13: 1 Shares Bought for: $186.23 Each, Total: $186.23
23. 2018-06-14: 1 Shares Sold for: $184.31 Each. Profit Made: $-1.91
24. 2018-06-14: 2 Shares Sold for: $184.31 Each. Profit Made: $-2.73
25. 2018-06-15: 1 Shares Bought for: $184.28 Each, Total: $184.28
26. 2018-06-16: 1 Shares Sold for: $184.25 Each. Profit Made: $-0.03
27. 2018-06-19: 1 Shares Bought for: $182.03 Each, Total: $182.03
28. 2018-06-20: 1 Shares Sold for: $181.01 Each. Profit Made: $-1.02
29. 2018-06-21: 1 Shares Bought for: $180.49 Each, Total: $180.49
30. 2018-06-22: 1 Shares Sold for: $179.59 Each. Profit Made: $-0.89
31. 2018-06-25: 1 Shares Bought for: $180.01 Each, Total: $180.01
32. 2018-06-26: 1 Shares Sold for: $179.74 Each. Profit Made: $-0.26
33. 2018-06-27: 2 Shares Bought for: $181.05 Each, Total: $362.11
34. 2018-06-28: 2 Shares Sold for: $180.67 Each. Profit Made: $-0.76
35. 2018-06-29: 1 Shares Bought for: $181.35 Each, Total: $181.35
36. 2018-06-30: 1 Shares Bought for: $182.02 Each, Total: $182.02
37. 2018-07-01: 1 Shares Bought for: $182.69 Each, Total: $182.69
38. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-3.18
39. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-2.51
40. 2018-07-02: 1 Shares Sold for: $179.51 Each. Profit Made: $-1.83
41. 2018-07-03: 1 Shares Bought for: $180.23 Each, Total: $180.23
42. 2018-07-04: 1 Shares Bought for: $180.95 Each, Total: $180.95
43. 2018-07-05: 2 Shares Bought for: $183.46 Each, Total: $366.93
44. 2018-07-06: 1 Shares Bought for: $184.31 Each, Total: $184.31
45. 2018-07-07: 2 Shares Bought for: $185.16 Each, Total: $370.32
46. 2018-07-08: 1 Shares Bought for: $186.01 Each, Total: $186.01
47. 2018-07-09: 1 Shares Bought for: $185.79 Each, Total: $185.79
48. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-2.41
49. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-2.64
50. 2018-07-10: 2 Shares Sold for: $183.38 Each. Profit Made: $-3.57
51. 2018-07-10: 1 Shares Sold for: $183.38 Each. Profit Made: $-0.94
52. 2018-07-10: 2 Shares Sold for: $183.38 Each. Profit Made: $-0.18
53. 2018-07-11: 2 Shares Bought for: $186.45 Each, Total: $372.9
54. 2018-07-12: 1 Shares Bought for: $186.74 Each, Total: $186.74
55. 2018-07-13: 1 Shares Bought for: $186.61 Each, Total: $186.61
56. 2018-07-14: 1 Shares Sold for: $186.47 Each. Profit Made: $-0.14
57. 2018-07-14: 1 Shares Sold for: $186.47 Each. Profit Made: $-0.27
58. 2018-07-15: 1 Shares Bought for: $186.33 Each, Total: $186.33
59. 2018-07-16: 1 Shares Bought for: $186.86 Each, Total: $186.86
60. 2018-07-17: 1 Shares Sold for: $185.84 Each. Profit Made: $-1.02
61. 2018-07-17: 1 Shares Sold for: $185.84 Each. Profit Made: $-0.5
62. 2018-07-17: 2 Shares Sold for: $185.84 Each. Profit Made: $-1.23
63. 2018-07-18: 2 Shares Bought for: $187.28 Each, Total: $374.56
64. 2018-07-19: 2 Shares Sold for: $186.85 Each. Profit Made: $-0.86
65. 2018-07-21: 1 Shares Bought for: $186.96 Each, Total: $186.96
66. 2018-07-22: 1 Shares Bought for: $187.02 Each, Total: $187.02
67. 2018-07-23: 1 Shares Bought for: $188.37 Each, Total: $188.37
68. 2018-07-24: 1 Shares Bought for: $190.15 Each, Total: $190.15
69. 2018-07-25: 1 Shares Bought for: $189.55 Each, Total: $189.55
70. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-3.15
71. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-3.75
72. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-1.97
73. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-0.61
74. 2018-07-26: 1 Shares Sold for: $186.4 Each. Profit Made: $-0.56
75. 2018-07-30: 1 Shares Bought for: $185.73 Each, Total: $185.73
76. 2018-07-31: 5 Shares Bought for: $196.67 Each, Total: $983.34
77. 2018-08-01: 3 Shares Bought for: $202.42 Each, Total: $607.25
78. 2018-08-02: 2 Shares Bought for: $203.0 Each, Total: $406.01
79. 2018-08-03: 1 Shares Bought for: $203.35 Each, Total: $203.35
80. 2018-08-04: 1 Shares Bought for: $203.71 Each, Total: $203.71
81. 2018-08-05: 1 Shares Bought for: $204.06 Each, Total: $204.06
82. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.91
83. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.56
84. 2018-08-06: 1 Shares Sold for: $202.14 Each. Profit Made: $-1.21
85. 2018-08-06: 2 Shares Sold for: $202.14 Each. Profit Made: $-1.72
86. 2018-08-06: 3 Shares Sold for: $202.14 Each. Profit Made: $-0.82
87. 2018-08-07: 1 Shares Bought for: $202.28 Each, Total: $202.28
88. 2018-08-08: 1 Shares Bought for: $203.87 Each, Total: $203.87
89. 2018-08-09: 1 Shares Sold for: $203.26 Each. Profit Made: $-0.61
90. 2018-08-10: 1 Shares Bought for: $203.7 Each, Total: $203.7
91. 2018-08-11: 1 Shares Bought for: $204.14 Each, Total: $204.14
92. 2018-08-12: 1 Shares Bought for: $204.58 Each, Total: $204.58
93. 2018-08-13: 1 Shares Bought for: $205.44 Each, Total: $205.44
94. 2018-08-14: 1 Shares Bought for: $205.92 Each, Total: $205.92
95. 2018-08-15: 1 Shares Bought for: $208.94 Each, Total: $208.94
96. 2018-08-16: 2 Shares Bought for: $213.11 Each, Total: $426.22
97. 2018-08-17: 1 Shares Bought for: $212.42 Each, Total: $212.42
98. 2018-08-18: 1 Shares Bought for: $211.72 Each, Total: $211.72
99. 2018-08-19: 1 Shares Sold for: $211.03 Each. Profit Made: $-0.69
100. 2018-08-19: 1 Shares Sold for: $211.03 Each. Profit Made: $-1.38
101. 2018-08-19: 2 Shares Sold for: $211.03 Each. Profit Made: $-4.15
102. 2018-08-21: 1 Shares Bought for: $210.63 Each, Total: $210.63
103. 2018-08-22: 1 Shares Bought for: $211.06 Each, Total: $211.06
104. 2018-08-23: 1 Shares Bought for: $211.72 Each, Total: $211.72
105. 2018-08-24: 1 Shares Bought for: $212.3 Each, Total: $212.3
106. 2018-08-25: 1 Shares Bought for: $212.88 Each, Total: $212.88
107. 2018-08-26: 1 Shares Bought for: $213.46 Each, Total: $213.46
108. 2018-08-27: 1 Shares Bought for: $215.18 Each, Total: $215.18
109. 2018-08-28: 1 Shares Bought for: $218.4 Each, Total: $218.4
110. 2018-08-29: 1 Shares Bought for: $220.41 Each, Total: $220.41
111. 2018-08-30: 1 Shares Bought for: $222.95 Each, Total: $222.95
112. 2018-08-31: 1 Shares Bought for: $223.13 Each, Total: $223.13
113. 2018-09-01: 1 Shares Bought for: $223.31 Each, Total: $223.31
114. 2018-09-02: 1 Shares Bought for: $223.49 Each, Total: $223.49
115. 2018-09-03: 1 Shares Bought for: $223.67 Each, Total: $223.67
116. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.46
117. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.28
118. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-1.1
119. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-0.92
120. 2018-09-04: 1 Shares Sold for: $222.21 Each. Profit Made: $-0.74
121. 2018-09-05: 1 Shares Sold for: $218.51 Each. Profit Made: $-1.89
122. 2018-09-06: 1 Shares Sold for: $216.75 Each. Profit Made: $-1.65
123. 2018-09-08: 1 Shares Sold for: $214.81 Each. Profit Made: $-0.37
124. 2018-09-10: 2 Shares Bought for: $219.25 Each, Total: $438.5
125. 2018-09-11: 2 Shares Sold for: $216.53 Each. Profit Made: $-5.45
126. 2018-09-12: 2 Shares Bought for: $221.76 Each, Total: $443.51
127. 2018-09-13: 2 Shares Sold for: $219.24 Each. Profit Made: $-5.03
128. 2018-09-16: 1 Shares Sold for: $213.4 Each. Profit Made: $-0.06

129. 2018-09-19: 1 Shares Bought for: $215.51 Each, Total: $215.51
130. 2018-09-20: 1 Shares Sold for: $213.19 Each. Profit Made: $-2.32
131. 2018-09-21: 1 Shares Bought for: $214.21 Each, Total: $214.21
132. 2018-09-22: 1 Shares Bought for: $215.23 Each, Total: $215.23
133. 2018-09-23: 1 Shares Bought for: $216.25 Each, Total: $216.25
134. 2018-09-24: 1 Shares Bought for: $217.62 Each, Total: $217.62
135. 2018-09-25: 1 Shares Sold for: $215.89 Each. Profit Made: $-1.73
136. 2018-09-25: 1 Shares Sold for: $215.89 Each. Profit Made: $-0.36
137. 2018-09-26: 1 Shares Bought for: $220.33 Each, Total: $220.33
138. 2018-09-27: 1 Shares Bought for: $221.1 Each, Total: $221.1
139. 2018-09-28: 1 Shares Bought for: $221.6 Each, Total: $221.6
140. 2018-09-29: 1 Shares Bought for: $222.09 Each, Total: $222.09
141. 2018-09-30: 1 Shares Bought for: $222.59 Each, Total: $222.59
142. 2018-10-01: 1 Shares Bought for: $224.57 Each, Total: $224.57
143. 2018-10-02: 1 Shares Bought for: $227.3 Each, Total: $227.3
144. 2018-10-03: 1 Shares Sold for: $223.3 Each. Profit Made: $-4.0
145. 2018-10-03: 1 Shares Sold for: $223.3 Each. Profit Made: $-1.26
146. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-2.91
147. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-2.41
148. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-1.92
149. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-1.42
150. 2018-10-04: 1 Shares Sold for: $219.68 Each. Profit Made: $-0.65
151. 2018-10-08: 1 Shares Bought for: $222.21 Each, Total: $222.21
152. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-10.29
153. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-3.32
154. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-2.3
155. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-0.97
156. 2018-10-09: 1 Shares Sold for: $211.91 Each. Profit Made: $-0.39
157. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-1.67
158. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-1.02
159. 2018-10-10: 1 Shares Sold for: $210.04 Each. Profit Made: $-0.59
160. 2018-10-11: 1 Shares Bought for: $217.55 Each, Total: $217.55
161. 2018-10-12: 1 Shares Sold for: $215.99 Each. Profit Made: $-1.55
162. 2018-10-15: 2 Shares Bought for: $217.58 Each, Total: $435.17
163. 2018-10-16: 2 Shares Sold for: $216.64 Each. Profit Made: $-1.88
164. 2018-10-18: 1 Shares Bought for: $214.8 Each, Total: $214.8
165. 2018-10-20: 1 Shares Bought for: $215.68 Each, Total: $215.68
166. 2018-10-21: 1 Shares Bought for: $216.12 Each, Total: $216.12
167. 2018-10-22: 1 Shares Bought for: $218.15 Each, Total: $218.15
168. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-7.48
169. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-5.45
170. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-5.01
171. 2018-10-23: 1 Shares Sold for: $210.67 Each. Profit Made: $-4.13
172. 2018-10-24: 2 Shares Bought for: $215.28 Each, Total: $430.57
173. 2018-10-25: 2 Shares Sold for: $211.85 Each. Profit Made: $-6.86
174. 2018-10-26: 1 Shares Bought for: $210.53 Each, Total: $210.53
175. 2018-10-27: 1 Shares Sold for: $209.2 Each. Profit Made: $-1.33
176. 2018-10-28: 1 Shares Sold for: $207.88 Each. Profit Made: $-1.06
177. 2018-10-29: 1 Shares Bought for: $208.92 Each, Total: $208.92
178. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $5.45
179. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $8.44
180. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $8.92
181. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $9.78
182. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $10.22
183. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $10.66
184. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $12.08
185. 2018-10-30: 5 Shares Sold for: $214.36 Each. Profit Made: $88.46
186. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $28.63
187. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $33.41
188. 2018-10-30: 1 Shares Sold for: $214.36 Each. Profit Made: $34.13
189. Total Money at the End: $10098.24 Profit Made: $98.24