|  |
| --- |
| Solent University |
| Faculty of Business, Law, and Digital Technologies |
|  |

|  |
| --- |
| Supervisor: Darren Cunningham  Date of Submission: May 2022 |

A picture containing text

Description automatically generatedIcon

Description automatically generated with medium confidence

Software Engineering: 2021/2022

COM616: Dissertation

‘Intelligent Income and Expenditure System’

Rahul Shrestha

Q13671294

## Acknowledgement

Write a paragraph here acknowledging everyone who has helped you while you have been preparing the content of your dissertation.

This may be you supervisor or other academic staff that have provided guidance and support, other students or colleagues with whom you have collaborated on any research or project work, interviewees, librarians, or perhaps any external bodies that have given you assistance, such as access to data or the opportunity for hands on experience.

## Abstract

This should clarify to the reader why they should read your report, Abstracts are a short summary, one paragraph 300 words max. giving a snapshot of your entire project; why, how, results and conclusions/ recommendations. The Abstract needs to work as a “standalone” so avoid using any citations. Write your abstract last.

Table of Contents

[Acknowledgement 1](#_Toc102646000)

[Abstract 1](#_Toc102646001)

[1. Introduction 3](#_Toc102646002)

[2. Literature Review 4](#_Toc102646003)

[3. Design Implementation 5](#_Toc102646004)

[4. Methodology 6](#_Toc102646005)

[4.1 Regression 15](#_Toc102646006)

[4.1.1 Regression: Exploratory Data Analysis & Data Cleaning 15](#_Toc102646007)

[4.1.2 Regression: Data Pre-Processing 25](#_Toc102646008)

[4.1.3 Regression Model: Modelling and Results 26](#_Toc102646009)

[4. 2 Classification 29](#_Toc102646010)

[4.2.1 Classification: Exploratory Data Analysis & Data Cleaning 30](#_Toc102646011)

[4.2.2 Classification: Data Pre-Processing 32](#_Toc102646012)

[4.2.3 Classification Model: Modelling & Results 34](#_Toc102646013)

[5. Results 45](#_Toc102646014)

[6. Limitations & Recommendations 45](#_Toc102646015)

[References 45](#_Toc102646016)

[Appendix- A (Mobile Wireframes) 47](#_Toc102646017)

[Appendix B-(Web Wireframe) 50](#_Toc102646018)

[Appendix-C (Mobile-Mock-up) 51](#_Toc102646019)

## 1. Introduction

Debt is a term that no one wants to hear when it comes to their personal finance. Debt can be defined as owing something, usually money, to someone (Chen, 2022). Research from Close Brothers (2019) has found that almost 94% of UK employees are suffering from money worries and almost 77% of employees have said that it has affected them at work. Additionally, The Money Charity has stated that as of November 2021, the average total household debt in the UK is £63,122.

There are many factors as to why someone can get into debt; Zhen (2022) has stated one of the most common causes was due to poor money management. This can take form in several different ways such as impulsive buying, using overdraft and simply spending more than you are earning. Whistl (2017) have found that 91% of the nation have admitted to making impulsive purchases every month and on top of that, Hall (2018) stated in a news article that an average UK adult will spend over £144,000 on impulsive buying during their lifetime. This could be due to the advancement of technology over the years which has allowed the rapid growth of e-commerce and in turn have amplified impulsive buying behaviours. Additionally, due to COVID-19 pandemic, the UK’s top retailers have stated that their online traffic has increased by 52% (Jobling, 2021).

Furthermore, the rise of contactless payments and mobile wallets have also been seen to contribute overspending. A study conducted by Xu et al. (2019) found that using mobile wallets can lead to people spending more money, more frequently. Other common causes of debt include spending future money and having no savings, such as an emergency fund.

One of the best ways to manage your expenses and control your spending is to utilise a budgeting app. Using one can help you keep track of your spending, bills, and generally allows you to become more aware of your finances (Lake & Foreman 2021).

One of the approaches taken to save money is to store it away in a savings account, however, Barclays (2021) have stated in terms of accumulating more wealth for the future, it is better to invest into the stock markets rather than leave it in a savings account. Likewise, Money Helper (2022) have stated that leaving your money in a ­­savings account is not the best option as the interest rate in the saving account is nearly always lower than the rate of inflation. Furthermore, Inman (2022) has reported that inflation in the UK has risen to its highest levels in 30 years, currently around 5.4%. The rise of inflation refers to an increase in prices and the decrease of purchasing power. This means that consumers can purchase less goods and services compared to before (Davies 2022). Moreover, Clark (2020) has reported that the average saving account interest rates have fallen to their lowest levels on record at 0.64% in 2020, meaning that the return on your money will be non-existent, especially when factoring in the increase of inflation rates. In contrast, the S&P 500, which is defined as a stock market index that tracks the US 500 large-cap companies (Amadeo, 2022), has reported an average annual return of around 10.5% since its inception in 1957, beating the inflation rate and any other savings account (Maverick, 2022).

On the other hand, investing is a lot riskier than storing your money in a savings account, therefore it is not advised for short-term goals, such as anything less than 5 years (Barclays, 2021). Stock markets are volatile, meaning the values of stocks can fluctuate and even drop in value drastically. For this reason, it is advised to aim to invest for at least 5 years as a longer time frame will allow your investments to recover over time (HSBC, n.d). Additionally, HSBC (n.d) have advised that before participating in saving or investing any money, it is important to have an emergency fund in case of any unexpected expense.

## 2. Literature Review

A scoping review conducted by Harper et al. (2021) found that people involved in the criminal justice system are disproportionally indebt compared to the average person. They suggested that reducing debt in this population can improve re-entry outcomes and quality of life. Furthermore, Van Beek et al. (2021) systematic review found debt to be a risk factor for criminal behaviour. Thus, utilisation of a financial app could be particularly beneficial for those who have a criminal background, manage their debt. This could result in fewer crimes.

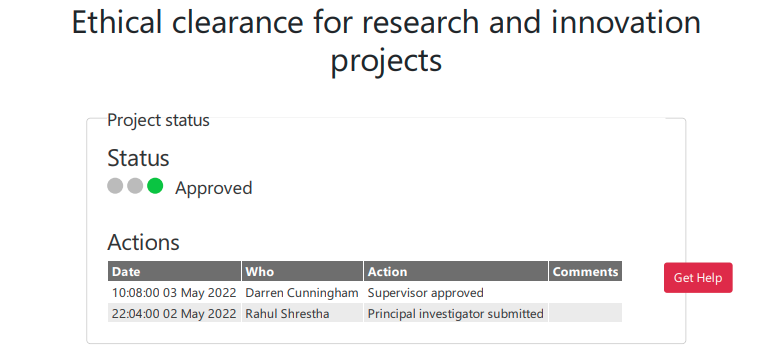
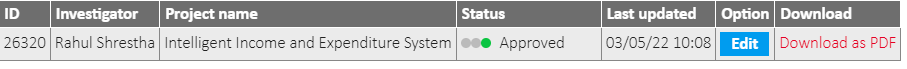
Additionally, A review conducted Swanton and Gainsbury (2020) found that debt problems led people to take part in gambling addiction which in turn resulted in bigger mental health problems. It was also stated gambling-related debt problem increased the likelihood of psychological distress, substance use, crime, and suicidality. Plus, findings from research conducted by Franzen and Bradaric (2018) showed that there is a gap of knowledge when it comes to managing money and being financially aware, especially in college students. It was also stated that due to the poor money management skills, students had increased stress levels and were not performing well in their academics. It also led to some students dropping out of school. Additionally, they suggested that utilisation of budgeting apps could lead to student maintaining and attaining financial wellness.

In a study by Ong et al. (2019), it was founded that when those in debt where given debt relief and they had experienced significant improvements in their cognitive function and reports of less anxiety. Furthermore, French et al. (2020) founded that utilisation of a finance app significantly improved financial knowledge which translated to improved financial behaviours.

## 3. Design Implementation

In an attempt to address the outlined issues of debt, I propose to build a prototype that allows you to manage your income and expenses, with an additional feature that makes investing in stocks less freighting. This feature will consist of utilising machine learning model to forecast stock prices and forecast trading signals (buy/sell or hold) for a particular stock.

As the main intention of the project is to explore and experiment how viable machine learning models can perform on the stock market, there are no legal or ethical issues concerned with this project, such that I have received ethical clearance from my supervisor.



The main project requirements and objectives include:

* Designing and creating a clean, aesthetic front end user interface (UI) for a web version and a mobile phone app version.
* A web application prototype such that, users can:
  + Sign up and log in to access the app.
  + View, add and manage their budgets.
  + View, add and manage their income and expenditures.
  + View, add and manage any upcoming bills.
* A machine learning application such that users can:
  + Enter and receive stock data information.
  + View technical analysis on the stock.
  + Forecast stock prices using machine learning models.
  + View forecasted trading signal from machine learning model

The project followed a waterfall methodology such that it started off with design phase which consisted of creating wireframes.

Wireframes are considered as blueprints for applications that allows programmers to visualise the structure of application. Hannah (2021) describes that wireframe allows for a clear overview of the page structure, user flow and functionality. Additionally, mobile-first approach was taken when creating the wireframes as Morales (2021) stated that designing smaller screens first allows designers to focus more on the core functions of the application.

Designing for the smallest screens first, and then working your way up empowers designers to focus on the core functions of their product

Mobile-first design is a design philosophy that aims to create better user experiences by beginning the design process with mobile devices in mind first, often prioritizing the smallest of screens. This mobile-first strategy stems from the concept of progressive advancement, which means that when a designer sits down to prototype or sketch a website layout, they begin with how it will look on a mobile device screen and then move to bigger screens later.

After completing our initial wireframe design, we moved on to creating a mock-up of the application using Adobe XD. The mock up consisted of changes in the layout, implementation of colour scheme, images, and styles. The purpose of mock up acted as a visual draft of our application. Additionally, it allowed us to visualise how different elements and functionality would work. The mock-ups can be found on [Appendix C](#_APPENDIX_C).

Following the design phase, we as a team decided to utilise the MERN technology stack to build the application. The MERN stack includes MongoDB, Express, React and Node.js; is known to make the development process smoother and easier, which allows for a fast deployment of web applications (Jasraj, 2021). Tanjona (2020) has stated that the MERN stack is one of most requested criteria from their customers when it comes to developing web, mobile or software applications.

Followed a waterfall methodology which consisted of designing the wire frame, creating a mock-up and building on the application itself based on the mock-ups.

Wireframes can be found in Appendix A.

Mock-up designs can be found in Appendix B.

## 4. Methodology

The stock market can be explored using two methods known as technical analysis and fundamental analysis. Fundamental analysis is defined as a method to determine the real (intrinsic) value of a stock by examining economic and financial factors of the company (Segal, 2021). Investors and traders that use fundamental analysis believe that the market does not accurately estimate the value of stocks and therefore they try and find a true worth of a company (The Street, 2022). They find and invest in stocks; they believe are undervalued by the market and hope the stock’s value increases over time.

On the other hand, technical analysis is defined as using historical market data to evaluate the price trends and patterns, to predict future markets behaviour (Chen, 2021). Saravanan (2019) has stated that fundamental analysis is more theoretical and that using technical analysis is seen to be more practical as it uses more factual, concrete data. Additionally, The Street (2022) has claimed that trading decisions are best made from technical analysis using trend evaluation and pattern recognition as they believe that stocks are accurately valued, thus fundamental analysis is unnecessary.

Technical Indicators fall into the realm of technical analysis, and Chen (2021) defined it as mathematical calculations and patterns derived from historical data. There are many technical indicators available out there and they can be classed into five categories: trend, momentum, relative strength, mean reversion, and volume (Barone, 2022). Folger (2022) has advised that when developing a trading strategy, it is recommended not to use different indicators from the same category as this can result in multicollinearity but as this project is aimed towards beginners, I have chosen easy to understand and beginner-friendly indicators which goes against Folgers’ advice.

The dataset used to train and test the models will be historical stock data, in this case Apple’s (AAPL) stock data was used; scarped from Yahoo Finance using the python library ‘yfinance’. Additionally, extra columns will be added to the dataset which will consist of four technical indicators data and for the classification models, an extra column will be added which will act as a target variable, this column will be an overall recommendation for a trading signal derived from the four technical indicators. This project takes on a supervised learning approach; Petersson (2021) defines supervised learning as models that are trained on input data labelled to specific output. This allows the model to learn and detect underlying patterns and relationship between the input and output data so that it can accurately predict on unseen input data. The aim of this project is to utilise technical analysis to predict future stock prices and trading signals of a stock such as buy, hold, or sell based on its historical data and the technical indicators data.

The technical indicators that are used and added to the dataset are:

|  |  |
| --- | --- |
| COLUMN | EXPLANATION |
| Stochastic Oscillator | Stochastic Oscillator (SO), which was developed by George Lane in the 1950’s, is a popular technical indicator when it comes to generating oversold and overbought signals (Hayes, 2021). Anderson (2022) defines SO to describe the relationship between the stock price, relative to its high and low prices over a predetermined period (14 days being the popular period). Additionally, Anderson (2022) has stated that SO has a good history of being accurate when it comes generating buy and sell signals.  SO has two components that work together in building a trading signal, the fast line denoted as ‘%K’ and the slow line denoted as ‘%D’ (West, n.d). Both signals produce a value that ranges between 0 to 100, typically values below 20 are seen as oversold which infers a buy signal and values over 80 are seen as overbought which infers a sell signal (West, n.d).  K% is calculated by = 100 \* ((14 Day Closing Price – 14 Day Lowest Price ) – (14 Highest Price – 14 Day Lowest Price))  D% is calculated by = moving average of %K over 3 days.  (For Clasifcation Model Only)  For this project, the SO indicator will follow the traditional rules when producing a trading signal such that :  A ‘buy’ signal will be created when:   * The %K value/line is below 20 * The %D value/line is below 20   A sell signal will be created when:   * The %K value/line is above 80 * The %D value/line is above 80   Chart, histogram  Description automatically generatedHere is a graph displaying the SO indicator based on the past 6 months of the Apple (AAPL) stock: |
| Relative Strength Index (RSI) | The Relative Strength Index (RSI), which was developed by J. Welles Winder in 1970, is also a momentum indicator like the stochastic oscillator that is used by traders to identify whether the market is an overbought or oversold state. Gumparthi (2017) describes RSI to measure the speed and change of price movements over a previous trading period.  The RSI also produces a value ranging from 0 to 100 but unlike the SO, values over 70 are seen as overbought and values under 30 are seen as oversold, according to Fernando (2022).  Even though, the RSI and SO are both momentum indicators, they both have different underlying methods and theories. Ross (2021) has stated the RSI is more useful in trending markets whereas SO is more useful when the market is trading in consistent ranges.  A study conducted by Gumparthi (2017) to the test validity of RSI signals in trading strategies found that the RSI to be an effective indicator, that was able to produce an accurate buy and sell signals for both short-term and long-term investments. It was also discovered that it successfully predicted future trends in the market.  Fernando (2022) described the RSI to be calculated using the following formulas:   1. Avg Loss = Sum of Losses over the past 14 periods / 14 2. Avg Gain = Sum of Gains over the past 14 periods / 14 3. RS = Average Gain / Average Loss 4. RSI = 100 – 100 ( 1 + RS).   (For Clasifcation Model Only)  For this project, the traditional boundaries will be used to create a trading signal for the RSI; such that values under 30 will be seen as buy signals and values over 70 will be seen as sell signals.  Graphical user interface, chart  Description automatically generated |
| Moving Average Convergence Divergence (MACD) | The Moving Average Convergence Divergence (MACD) was developed by Gerald Appel in 1979 and it used as trend-following momentum indicator (Schlossberg, 2022). Silberstein (2022) defined MACD to describe the relationship between two moving averages of a stock and it is calculated by subtracting the 26-period exponential moving average (EMA) from the 12-period EMA, this is referred to as the MACD line. Additionally, there is another component referred to as the signal line, that works with the MACD line to come up with a trading signal. The signal line is calculated by finding out the 9-period EMA of the MACD. Mathematically written as:   * MACD = 12D EMA – 26DEMA * Signa = 9D EMA of MACD   Here is a diagram displaying the MACD line and the Signal line for the past 6 months of the Apple (AAPL) stock:  Chart, line chart, histogram  Description automatically generated  (For Clasifcation Model Only)  For this project, MACD indicator will produce a buy signal when the MACD line crosses **above** the signal line thus the sell signal will be created when the MACD line crosses **below** the signal line. |
| Bollinger Bands | Bollinger Bands (BB) was created by John Bollinger in the 1980’s and it has been described to offer numerous insights into price and volatility, such as monitoring breakouts, following trends and determining overbought and oversold levels (Mitchell, 2022).  BB consist of three components that work together to highlight how prices are distributed around an average value. Binance Academy (2018) described the components to be calculated using the following formulas:   * Middle Band= 20-day simple moving average (SMA) * Upper Band = Middle Band + (2 x 20-day stand deviation) * Lower Band = Middle Band – (2 x 20-day stand deviation)   Here is a diagram displaying the BB for the past 6 months of the Apple (AAPL) stock:  A picture containing diagram  Description automatically generated  (For Clasifcation Model Only)  For this project, the BB indicator will be used to determine overbought and oversold level to create buy and sell signals. Buy signals will be created when the price crosses below the lower band and alternatively, sell signals when the price cross above the upper band. |
| Recommender  (Target Variable for Classification Model) | (For Clasifcation Model Only)  The Recommender column (dependant variable) contains an overall recommendation in whether to buy, sell, or hold the stock based on the signals from the other indicators.  Upon further inspection, the function I created to derive trading signals the MACD indicators were producing inaccurate signals so therefore they have not taken in consideration when creating the overall signals, however the MACD line and the signal line will still be used when training the models.  To ensure signals were as accurate as possible I followed the following steps:  A simple if-else function, where if all three of the indicators stated the same signal, the value would be declared as that signal or if at least two out of three indicators stated the same signal, it was declared as that signal. Everything else that did not fit into the above statements were labelled as ‘Unclassed’ and they were manually given a signal. This table outlines the above function:   |  |  |  |  | | --- | --- | --- | --- | | RSI | SO | BB | Recommender | | Buy | Buy | Buy | Buy | | Sell | Sell | Sell | Sell | | Hold | Hold | Hold | Hold | | Buy | Buy | ? | Buy | | ? | Buy | Buy | Buy | | Buy | ? | Buy | Buy | | Sell | Sell | ? | Sell | | ? | Sell | Sell | Sell | | Sell | ? | Sell | Sell | | Hold | Hold | ? | Hold | | ? | Hold | Hold | Hold | | Hold | ? | Hold | Hold | |

## 4.1 Regression

Predicting future stock prices can be also classed as a time series problem in which time series forecasting methods can be applied. Tableau (2022) defined time series forecasting as making scientific predictions based on historical timed stamped data. In relation to stock prices, Christie (2020) stated that stock prices should be treated as discrete time series data as stock prices are taken sequentially in time. As mentioned above, AAPL’s historical stock data will be used as the dataset to train and test the regression models, unlike with the classification models, this dataset will contain all AAPL’s stock data available, dating back to when the company first went public on December 12, 1980.

### 4.1.1 Regression: Exploratory Data Analysis & Data Cleaning

Conducting an exploratory data analysis (EDA) is an important step in which preliminary investigation are conducted to provide an insight into the data and their interactions (Sonal, 2021). EDA normally consists of using graphical representations and summary of statistics.

Table containing the EDA that was conducted:

|  |
| --- |
| 1. Gathered historical data for the AAPL stock using ‘yfinance’ and stored in a panda’s data frame. Applied the function ‘.shape’ which returns the dimensions of the dataset. From the result, you can interpret the base dataset to contains 10427 rows and 7 columns. |
| 2. Explored the dataset by applying the function ‘.columns’ which returned the columns headers in the dataset. From the result, you can see it to contain:   * Open   + Open represents the stock’s initial price at the start of the trading day. * Close   + Close represents the stock’s final price at the end of the trading day. * High   + High represents the stock’s highest trading price for the day. * Low   + Low represents the stock’s lowest trading price of the day. * Volume   + Volume represents the number of shares that was traded in the day. * Dividends   + Dividends represent the number of shares that was paid to the shareholders instead of cash. * Stock Splits   + Stock Splits represents the ratio in which the stocks are split, this occurs when a company wants to boot its stock liquidity by increasing the number of it outstanding shares (Hayes, 2022). |
| 3. Explored the values of the columns “Dividends” and “Stock Splits”. From the results, you can see the majority of the value contained in these columns were ‘0’. As these two columns are not of any use, I decided to drop them before moving on. |
| 4. When I came to dropping these two columns, I encountered a bug where that even after dropping the columns successfully, calling the data frame again would result in the columns being reinstated. To resolve this issue, I utilised the ‘.copy’ function to copy the relevant columns on to new data frame. |
| 5. Calculated and inserted the technical indicators data on to the dataset. |
| 6. Explored the dataset using the ‘.info’ function which provides a brief overview of the dataset. From the result, you can see the date range, the column headers, the number of values per each column and its data types. Being able to view the data types of the columns is especially valuable as if there any non-numerical values, you would need to encode these values before moving onto modelling. In this case all the values are numerical. |
| 7. Explored the dataset columns using the function ‘.describe’ which provides a statistical summary of the numerical columns in the dataset. |
| 8. Explored the dataset rows by applying the ‘.tail’ function which returns the last 5 rows in the data frame. From the results, you can see we have the latest stock data available. |
| 9. Explored the dataset rows by applying the ‘.head’ function which returns the first 5 rows in the data frame. From the result, you can see, we have stock data from far back as ‘1980-12-12’ in the dataset. |
| 10. Performed checks on the dataset by applying the ‘.isna’ function which returns if there are any null values present in the dataset. It is very important to handle any missing values as most machine learning models do not support missing values and therefore this can result in building a biased model which can lawed to inaccurate results (Tamboli, 2021). From the result, you can see there are some null values: |
| 11. Dropped null values using the function ‘.dropna’. There are multiple ways in which you can handle missing data but, in this case, I decided to drop the rows entirely as there were only a small percentage of null values and dropping these rows would not impact anything later. |
| 12. Performed more checks on the dataset using the ‘.duplicated’ function which returns if there are any duplicated values present in the dataset. From the result, you can see that for this instance, there was no duplicates. |
| 13. Plotting Heatmap: Heatmap displays the correlation between the different variables on scale from -1 to 1. From the results, you can interpret how the different technical indicators, closing price and volume are correlated. |
| 14: Plotting Line Graph: This first graph represents the stocks opening and closing price throughout the years. To get a better insight, the second chart looks back over the past 1 year. |
| 15. Plotting Line Graph: This graph represents the stocks high and low prices over the past 1 year. |
| 16. Plotting Bar Graph: These graphs display the number of shares being traded throughout the years. The second graph displays the result from the past 1 year. |
| 17. Plotting Scatter Graph: This graph displays the returns of the stock over the 1 past year. |
| 18. Plotting Line Graph: This graph displays the closing price against the moving averages of 50, 100 and 200 days. Moving Averages (MA) are popular tool when to comes to technical analysis, Potters (2022) defines MA to smooth out the price trend from short-flucations by filtering out the noise. |
| 19. Plotting Technical Indicator: RSI (10 Years Duration) |
| 20. Plotting Technical Indicator: BB (10 Years Duration) |
| 21. Plotting Technical Indictor: MACD (10 Years Duration) |
| 22. Plotting Indicator: SO (10 Years Duration) |

### 4.1.2 Regression: Data Pre-Processing

Preparing and pre-processing the dataset prior to modelling is a critical step that must be taken to ensure that the machine learning model is accurate and efficient. Baheti (2022) defined data pre-processing as a series of step that must be followed to transform and encode the data so that it can be easily parsed by the machine learning model. As we have already cleaned the data by removing null values and duplicates, the next steps include feature scaling and splitting the dataset into train and test sets.

Feature Scaling can be defined as a method to transform the numeric features in the dataset, to a standard range (Munagala, 2021). Roy (2020) has stated that implementing feature scaling is a crucial step that can determine the difference between a weak and a strong model. Additionally, Brownlee (2020b) and Sharma (2021) has reported that many machine learning algorithm perform better when the numerical features are scaled to a standard range, such that scaling increases precision and reduces memory consumption. The main reasoning behind for this is because most machine learning algorithms cannot comprehend the true meaning behind numbers such that they think features with higher range values are more important and tend to ignore features with smaller range values which can lead inaccurate predictions (Munagala, 2021). Additionally, Roy (2020) explained that machine learning algorithms cannot differentiate between 10g of weight and £10 in price, this leads to hierarchy and bias between the variables. However, some models can perform well without feature scaling as its accuracy won’t be dependent on the range, for example: tree-based models.

Normalisation and Standardisation are the two most popular technique that are used to scale numerical data. Brownlee (2020b) describes normalisation as rescaling the data so that all the values fit into a range of 0 and 1, where 1 represents the highest feature value and the ‘0’ the lowest. Secondly, Liu (2020) describes standardisation as transforming the data so that the features are rescaled to have a standard deviation of 1 and a mean of 0. To determine which technique to use, Brownlee (2020b) expressed that there is no correct answer such that it depends on many different factors like the specifics of the problem, the choice of models and the state of the variables.

For this project, ‘MinMaxScaler’ (MMX) from scikit-learn was used to scale the dataset between the default range of 0 to 1. MMX has been described to preserve the original shape of the distribution without changing the meaning of the values from the original data, such that the importance of outliers isn’t reduced (Hale, 2019).

### 4.1.3 Regression Model: Modelling and Results

Furthermore, the following metrics will be used to evaluate the models, however the RMSE score will be considered as the main metric. Models scoring less than 5 RMSE will be classed as successful, thus be tuned, and implemented on the app.

|  |  |
| --- | --- |
| Mean Absolute Error (MAE) | The MAE refers to the average in the absolute difference between the actual value and the predicted value (Bajaj, 2022). Brownlee (2021b) has stated the MAE increases the scores linearly with the increases in errors and that it does not give any sort of weight to the different types of errors.  Formula used to calculate the MAE:  Image by Ghosh (2022) |
| Mean Squared Error (MSE) | The MSE refers to the average of the squared differences between the predicted and the actual values (Brownlee, 2021b). In other words, the MSE provides an absolute number on how much the predicted values deviate from the actual values. Additionally, Ghosh (2022) has stated that the MSE is highly sensitive to outliers and small errors which can give a high error score and can lead to a misinterpretation on how poorly the model performed.  A picture containing text, clock, watch  Description automatically generatedFormula used to calculate the MSE:  Image by Ghosh (2022). |
| Root Mean Squared Error (RMSE) | The RMSE is the squared root of the MSE and the purpose of it is to have the error score in the same scale as the original units. Additionally, it has been stated to handle the penalisation of small errors from the MSE (Ghosh, 2022).  A picture containing text, clock  Description automatically generatedFormula used to calculate the RMSE:  Image by Ghosh (2022). |
| R-Squared (R2) | R2 is also known as the Coefficient of determination, refers to the variance between the original values(independent variable) in the dataset and the predictions (dependant variable) made by the model (Kharwal, 2021). In other words, Wu (2020) described the R2 to measure how much variability in the dependant variable can be explained by the model.  Formula used to calculate the R2    Image by Wu (2020) |

For this project, I have explored the two most popular time series models used to predict stock prices.

|  |
| --- |
| Auto-Regressive Integrated Moving Average (ARIMA) |
| The ARIMA model has been defined as a statistical analysis model that is used to forecast time series problem such that it examines the difference between the values against the time series (Hayes, 2021). A glaring drawback of this model is that it assumes that future prices and trends will resemble the past and consequently make false predictions.  The ARIMA model can broken down to three components, which are also important parameters to the model, Hayes (2021) described them as:   * AR (Auto Regression) is denoted as p, representing the amount of lag observations in the model, also known as the lag order. * I (Integrated) is denoted as d, representing the number of differences with the raw observations. * MA (Moving Average) is denoted as q, represents the size of the moving average window.   To find these parameters, Partial Autocorrelation (PACF) and the Auto Correlation (ACF) were inspected.    Additionally, “Auto-ARIMA” model was implemented to find best p,d,q parameters. The results were:  Models perf  As you can see from the results, the model very poorly scoring 59.9 on the RMSE.  From the graph you can see that the ARIMA model, failed to predict the growth and future trend of the Apples’ stock, proving the limitation of this model. |
| Long Short-Term Memory (LSTM) |
| The Long Short-Term Memory (LSTM)-model is described by Brownlee (2021a) as an advanced recurrent neural network (RNN) that is capable of learning and remembering selective patterns over a long period of time. RNN are an extension of artificial neural network (ANN) which consist of a set of algorithms that try to imitate how a human brain would function. Additionally, Saxena (2021) has stated the LSTM has been explicitly designed to avoid the shortcomings of RNN, such that RNN were not able to remember long term dependencies due to the vanishing gradient.  The model was composed of a sequential input layer, followed by four extra layers consisting of 50, 60, 80 and 120 neuros. Here are the results from the model:    The LSTM model also did not perform that well, although performed significantly better than the ARIMA, such that it was predicting well until 2020. |
| To conclude, neither of these models were able to accurately predict the growth in the Apples’ stock and neither scored a RMSE less than 5. Therefore, the regressor side of this project has been deemed as unsuccessful and no further research were concluded. |

## 4. 2 Classification

Classification is defined as a process of predicting which class label or category, a given observation belongs to (Nabi, 2018). The aim of our classifier is seen as a multi-class classification problem where we are trying to predict stock signals into 3 classes: buy, hold, and sell. Like the regression model, AAPL’s historical stock data will used to train and test the models but in this case the dataset will only contains 10 years’ worth of data. As defined above, this dataset will contain an extra target variable column.

### 4.2.1 Classification: Exploratory Data Analysis & Data Cleaning

The EDA for this dataset will mostly follow the same process as the regression dataset therefore only the differences are listed below:

|  |
| --- |
| 1. The state of the dataset after data cleaning. From the results, you can see that this dataset contains records from ‘2012-05-23’ to the ‘2022-04-26’ and has the extra trading signals columns that was used to calculate the target variable column, their datatypes are denoted as objects. The extra columns containing the trading signal from each indicator (highlighted in red) will be dropped prior to modelling.  2. Plotting Pie Chart: Distribution of trading signals from SO. |
| 3. . Plotting Pie Chart: Distribution of trading signals from SO. |
| 4. Plotting Pie Chart: Distribution of trading signals from BB. |
| 5. Plotting Pie Chart: Distribution of trading signals from MACD. |
| 6. Plotting Pie Chart: Distribution of trading signals in the target variable. |
| 7. From the above charts, you can see collectively, there are significantly low ‘buy’ and ‘sell’ signals produced compared to ‘hold’ signals. This makes the dataset imbalanced which will be rectified in the pre-processing stage. |

### 4.2.2 Classification: Data Pre-Processing

In addition to feature scaling, data encoding, and handling of imbalanced dataset must be completed prior to modelling.

Data Encoding is defined as the process of converting categorical data into integer format (Verma, 2021). Dhandare (2020) described that it is necessary to encode categorical variables as many machines learning models can only work with numerical variables as they are required to perform mathematical operations. Categorical variables can be divided into 2 parts: nominal and ordinal; Singh (2020) defined nominal variables to have no intrinsic ordering whereas ordinal to have a clear ranked ordering. In this project, python dictionary was used to map every category to a numerical value such that 0 = Hold, 1 = Sell and 2 = Buy. This may have not been the best method to use as it is important to choose the right technique depending on the type of categorical data as forcing an ordinal relationship between nominal variables can be misleading to the model and result in poor performance.

Next, handling the imbalanced dataset, as mentioned above the ‘hold’ signals make up majority of the dataset accounting for 80% of the records whereas ‘sell’ signals account for 16% and ‘buy’ signals for only 4%. It is essential to resolve this to produce good, accurate results as models trained on imbalance dataset will cause a bias and falsely predict on the majority class. The main techniques used to manage imbalance datasets are oversampling and under sampling. Oversampling is described as duplicating or creating new synthetic examples in the minority class whereas under sampling includes merging or deleting examples in the majority class (Brownlee, 2020a). For this project, I have chosen to oversample my dataset using Synthetic Minority Oversampling Technique (SMOTE) as stock data are classed discrete time series data therefore deleting record from the majority class could lead to removable of valuable information which in turn lead to inaccurate predictions. SMOTE has been described to generate synthetic samples from interpolating between the positive instances that lie together (Satpathy, 2020).

Contrastingly, the models’ performance will be analysed using both the original dataset and the SMOTE dataset.

|  |  |
| --- | --- |
| Trading Signals prior SMOTE. (Original Dataset) | Chart, bar chart  Description automatically generated |
| Trading Signals after SMOTE. There are now equal number of hold, buy and sell signals. |  |

### 4.2.3 Classification Model: Modelling & Results

Furthermore, Cross validation have been implemented to evaluate if the models have been generalised to the dataset. Seldon (2021) has described cross validation as a technique to assess a machine learning model’s accuracy on new and unseen data, beyond the training dataset. It also said that cross validation is a useful tool to highlight if the model has been overfitting to the training data. There are numerous cross validation techniques out there, for this project I have chosen a technique called Stratified K-Fold (SKF) cross validation. SKF is a variation of the K-Fold cross validation technique such that the dataset is split up into ‘k’ folds, the ‘k-1’ fold will be assigned as the training data - the rest will be testing data. At each fold, the model is trained with the assigned training/test data and evaluated. This process iterates through until all the folds have been used as testing data (Krishni, 2018). Unlike K-Fold where the data are split up randomly, SKF splits up the data in a stratified manner to account for the class imbalance in the dataset. Lendave (2021) has described SKF to maintain the same class ratio through the K-folds as the ratio in the original dataset.

Moreover, hyperparameter tunning will also be implemented to the best performing model. Rouse (2021) defined hyperparameters as machine learning parameters that manages and controls the behaviour of the model, such that hyperparameter tuning is the process of finding the optimal combination hyperparameters to maximise the model’s performance and minimise the loss function. Additionally, Lee (2019) stated that failing to utilise hyperparameter tuning can lead to sub-optimal results, but on the other hand, Lee also expressed that hyperparameter tuning may not be worth its time-consuming and computationally expensive process, just to achieve minor improvements.

Nonetheless, I have utilised two hyperparameter tuning methods called Grid Search and Randomised Search in this project. Grid search works in a similar way to a brute force algorithm such that it iterates through the all the possible of combination of hyperparameters to find the best performing one (Badr, 2019). The glaring drawback of this is method is it is very time consuming and requires lots of computational space. Alternatively, random search iterates through randomly picked sets of hyperparameters which is less time consuming and less computationally demanding. However, a drawback of this method is that it may not return the best possible combinations. Additionally, random search does not remember its past iterations which makes its inefficient (Badr, 2019). To mitigate the drawbacks of these methods, I ran random search, first to narrow down the range of values of each hyperparameter and afterwards grid search is ran focusing on the most promising hyperparameter ranges found in the random search.

Moving on, the following metrics will be used to evaluate the models, however the F1 score will be considered as the main metric thus models scoring over 85 will be deemed as successful.

|  |  |
| --- | --- |
| Accuracy | Accuracy score represents the percentage of correct predictions out of the total predictions. Accuracy has been stated to only be a useful metrics when there is a balanced dataset (Sunasra, 2027).  Formula used to calculate accuracy:  Image by Korstanje (2022) |
| Precision | Precision represents the number of true positives predicted out of all the positive predictions. It is calculated by:    Image by Korstanje (2022) |
| Recall | Recall represents the number of true positives correctly identified out of the total number of positive predictions. It is calculated by:    Image by Korstanje (2022) |
| F1 | F1 is a combination of the precision and recall metric, where it has been described as harmonic mean of precision and recall (Korstanje, 2021). Additionally, Korstanje (2021) has described the F1 score to work well on imbalanced dataset. It is calculated by: |

For this project, I have explored the seven most classification models and as I have mentioned above, the best performing model will get further tuned and be implemented on to the application as the main model, granted it scored over 85 on the F1 metric.

|  |  |  |
| --- | --- | --- |
| MODELS | RESULTS | |
| No SMOTE | SMOTE |
| Logistic Regression (LR)-uses mathematical probability to predict the target variable into categories (Agrawal, 2021). Normally, LR is used for binary classification but as we have 3 classes for our target variable, multinomial LR was used. | *Avg Accuracy : 0.897*  *Avg Recall : 0.545*  *Avg Precision: 0.620*  *Avg F1 : 0.569*  Although accuracy seemed to be good, the recall, precision and F1 have terrible score. LR did not perform well against an imbalanced dataset. | *Avg Accuracy : 0.785*  *Avg Recall : 0.8974*  *Avg Precision: 0.6279*  *Avg F1 : 0.6804*  A slight improvement when using SMOTE, Recall significantly improved however the precision and F1 are still low.  To conclude, the F1 score both scenarios were low, therefore LR model did not perform well in this case. |
| Decision Tree (DT)-has been described to predict classes by learning simple decision rules from the training data (Chauhan, 2022). Additionally, Roy (2020) has defined DT as a graphical representation that maps all possible solutions to a decision based on certain conditions. | *Avg Accuracy : 0.928*  *Avg Recall : 0.851*  *Avg Precision: 0.847*  *Avg F1 : 0.846*  DT model performed well scoring over 80 in all the metrics. | *Avg Accuracy : 0.9303*  *Avg Recall : 0.8891*  *Avg Precision: 0.8349*  *Avg F1 : 0.8556*  Considering DT performed well on the imbalanced dataset, there was only a small improvement in the scores. |
| Random Forest (RF)-consists of building multiple decision trees that work together as an ensemble. Yiu (2019) explained that each individual tree in RF make a class prediction and the class with most votes becomes the models’ predictions. | *Avg Accuracy : 0.949*  *Avg Recall : 0.830*  *Avg Precision: 0.926*  *Avg F1 : 0.869*  RF also performed well on the imbalanced data, scoring 92% on precision. | *Avg Accuracy : 0.9415*  *Avg Recall : 0.902*  *Avg Precision: 0.8558*  *Avg F1 : 0.8746*  Like DT, RF also behaved in the same way such that recall improved, and precision decreased when trained on the SMOTE dataset. |
| Support Vector Machine (SVM)-is made up of finding a decision boundary line, known as the hyperplane, to separate data into different classes (Pupale, 2018). | *Avg Accuracy : 0.927*  *Avg Recall : 0.779*  *Avg Precision: 0.888*  *Avg F1 : 0.821*  SVM, although performed satisfactorily, recall was significantly lower than the other metrics. | *Avg Accuracy : 0.867*  *Avg Recall : 0.9163*  *Avg Precision: 0.7115*  *Avg F1 : 0.7756*  Unlike other models, SVM performed significantly worse on the SMOTE dataset. Although it improved recall , all the other metrics were worse. |
| K-Nearest Neighbours (kNN)-also known as the lazy learning algorithm, uses feature similarity to predict the correct classes. Dwivedi (2020) explained kNN to classify new data points based on similarity such that data points falling near to each other will fall in the same category. | *Avg Accuracy : 0.925*  *Avg Recall : 0.799*  *Avg Precision: 0.883*  *Avg F1 : 0.833*  KNN, mimicking the results of SVM, performed better when trained on the original dataset, although recall scored higher in the SMOTE dataset. | *Avg Accuracy : 0.8907*  *Avg Recall : 0.9057*  *Avg Precision: 0.7445*  *Avg F1 : 0.8032* |
| Naive Bayes (GNB)-is a statistical model that uses conditional probability to make predictions derived from Bayes theorem. Additionally, Navlani (2018) explained that GNB assumes independence between every pair of features in the data. | *Avg Accuracy : 0.818*  *Avg Recall : 0.864*  *Avg Precision: 0.651*  *Avg F1 : 0.708*  GNB, the first model to score significantly lower on precision than recall. | *Avg Accuracy : 0.7989*  *Avg Recall : 0.8619*  *Avg Precision: 0.6366*  *Avg F1 : 0.6925*  Additionally, the results did not seem to improve when using SMOTE dataset, making it the first model to score better on the original dataset in every single metric. |
| Multi-Layer Perceptron (MLP)-is a deep learning method that relies on it underlying neural networks to make predictions (Nair, 2019). | *Avg Accuracy : 0.928*  *Avg Recall : 0.826*  *Avg Precision: 0.869*  *Avg F1 : 0.843*  MLP also performed decently, scoring over 80 in every single metric. Additionally, the model seemed to perform better on the original dataset. | *Avg Accuracy : 0.888*  *Avg Recall : 0.917*  *Avg Precision: 0.7455*  *Avg F1 : 0.8051*  On the other hand, whilst the recall improved scoring over 90 using the SMOTE dataset, all other metrics score lowered such that precision dropped quite significantly. |
| Overview of Reults  To conclude, more models performed better on the original dataset than the  SMOTE dataset in regards to the F1 score. Additionally, the ranking of models performance stayed consistent and followed the same order, between both datasets. On the other hand, the general trend was that training the models on the SMOTE dataset greatly improved the recall score but decreased precision slightly. RF managed to perfom the best all-around between both datasets, scoring the highest in accuracy, precision and F1 score. Thus RF will be tuned and implemented on to the app. | | |
| Chart, bar chart  Description automatically generatedFurthermore, I decided to experiment on one more scenario which consisted of balancing the dataset in the following ratio: 50% Hold, 25% Buy, 25% Sell. This was done to explore if the models performed better on a dataset that mimicked real world data.  From the results, you can see the models peformed quite well such that 6/7 models scored over 80 in the F1 metric. Additionally, 3 models highlighted in green performed better in this datasest compared to before. However, RF still managed to hold the lead, scoring the highest F1 again. | | |
| Hyperparameter Tuning Main Model: Random Forest | | |
| 1. Retrieveing the current paremeters that our default model is using by appling the function ‘get\_params()’. | | |
| 2. From the results above, you can see there are numerous parameters. For this project, I only investigated the following five parameters:   * n\_estimators = represents the number of trees in the model. * max\_features = reqresents the maximum number of features required to split a leaf node. * max\_depth = represesnts the maxium of depth of each decision tree. * min\_samples\_split = represents the mininum number of samples placed in a leaf node before the node splits. * min\_samples\_leaf = represents the mininum number of data points reuired in a leaf node.   (Meinert, 2019). | | |
| 3. Created a grid containing the above hyperparameters so that it can be passed on to random search for samlping. | | |
| 4. Carried our random search with the settings of ‘n\_iter’ set to 100; this cotnrols the number of different combination to try and ‘cv’ set to 5; this represents the number of folds for cross validation (Koehrsen, 2018). After its lengthy search, it provided the best parameters of: | | |
| 5. Evaluated the model using the hyperparamters from random search. From the results you can see the results have improved, even if it is by a very small amount. | | |
| 6. As utilising random search narrowed the range of each hyperparameters, I utilised grid search to exhaustively search for every combinations. Grid serach provided the best parameters of: | | |
| 7. Evaluated the model using the hyperparameteres produced from grid search. As you can see from the result, the improvements were microsopic. However, the time computation required for grid search was far greater than random search. | | |
| 8. Additionally, expiremented with another hyperparamter turning method named ‘HalvingGridSearch’. Gilde (2021) has described this method to evaluate all the canddiates with a small number of resource at first and the iteratively selects the back candidates, using more and more resources. This method also produced good results considering the computation time was dramatically lower than gird search. | | |
| 8. From the comparison of resutls, the grid search model perfromed the best therefore exported the model to a file to be tested against other stock data and be implemented on to the app. | | |
| 9. Tested the exported model on Google’s stock data. As you can see from the result, the model performed quite well considering it was trained on a completely different stocks’ data. | | |

## 5. Results

## 6. Limitations & Recommendations

## References

## Graphical user interface, application Description automatically generatedGraphical user interface, application Description automatically generatedAppendix- A (Mobile Wireframes)

Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

## Appendix B-(Web Wireframe)

Graphical user interface, application, Word

Description automatically generatedA screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

## Graphical user interface, application Description automatically generatedAppendix-C (Mobile-Mock-up)

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated