Brunel University of London

Brunel University of London

Department of Mathematics

2024-2025

ANALYZING AND FORECASTING THE S&P 500 DIRECTION

AUTHOR: KRISHNA MANVITHA AKULA

Master of Science in Statistics with Data Analytics

ID: 2346457

RESEARCH GUIDE: Dr. CORMAC LUCAS

September 2025

Table of Contents

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to my supervisor, **Dr. Cormac Lucas,** for their continuous guidance, encouragement, and fruitful suggestions throughout the duration of my dissertation "Analyzing and Forecasting the S&P 500 direction". The path and quality of this research have been greatly influenced by his advice and support.

Additionally, I want to thank my professors from department of mathematics at Brunel University of London for their encouragement, stimulating discussions, and insightful suggestions.

This work would not have been possible without the unwavering support and patience of my Mom, loved one, family and friends, for which I am especially grateful.

I extend my heartfelt gratitude to everyone listed and to the several others who assisted with this work in different ways. Without all your help and advice, this thesis would not have been possible.

ABSTRACT

This dissertation explores the use of machine learning models to forecast the directional movement of the S&P 500, using the data from 2020 to 2025. Financial markets are noisy, non-linear, and volatile, making it difficult to predict the direction of the S&P 500 index. The accurate forecasts help traders, investors and policymakers in making decisions and effective strategies.

The involved features in this study are trading volume, prior day closing prices, and short-term volatility and macroeconomic variables such as oil, gold, sectoral indices, and volatility indices. These features are compared using the models such as logistic regression, naïve bayes, decision trees, random forest and multi-layer perceptron.

The study's overall findings show that, even if more complex relationships can be captured by advanced models, easier methods, such as logistic regression, are still effective and easy to understand. By showing the importance of machine learning in financial forecasting and the significance of combining technical and macroeconomic variables, this study adds to both academic research and real-world applications.

INTRODUCTION

1.1 Background

The financial market plays a crucial role in the global economy, facilitating capital, investments, and wealth accumulation. In the global economy, the S&P 500 index is one of the most recognized indicators of market performance. Among many stocks, the S&P 500 is the standard benchmark for investors and mutual funds. The Standard and Poor's 500, or the S&P500, is a stock market index tracking the performance of 500 listed leading companies on the stock exchange in the United States. It accounts for around 80% of the total US stock market value, making it a good measure of overall performance and analysis. This is also used to record the daily changes of the American stock market. Many large companies, such as Microsoft, Apple, Amazon, Coca-Cola, and others, are also listed in the S&P 500. The S&P 500 role in the United States is equivalent to that of the FTSE 100 in the United Kingdom.

Financial markets are interlinked with the movement of the S&P 500 which has an effect on all the global stock market. Because of that, understanding and forecasting its direction is of interest not only to investors and global businesses, as it influences the decisions across global wealth, risk management, and economic decision-making. Predicting its direction is complex in both practical and theoretical ways. Even modest fluctuations in its value lead to major economic consequences. In the past, analysts have used basic data, technical analysis, and market instincts to predict its movement. According to **Fama's (1970)** Efficient Market Hypothesis, predicting the stock market is impossible. However, with the use of advanced technology, machine learning can be employed as a powerful tool to forecast its direction. Unlike standard statistical models, machine learning algorithms help us to find patterns and relationships within the data.

1.2 Problem Statement

The primary challenge in forecasting the S&P 500 lies in the non-stationary character of financial time series data. In general, the markets are influenced by a vast number of indicators like economic, geopolitical events, and financial dynamics, many of which are difficult to forecast. So, to reduce uncertainty and randomness in this study, we apply data-driven models using machine learning models. The implemented models include logistic regression, naïve bayes, decision trees, random forest and multi-layer perceptrons (MLP). These models improve prediction accuracy by using indicators and market movement. In this study, we use statistical approaches to analyze previous price movements and technical indicators to identify the numerous elements that influence the market direction.

1.3 Overview of Machine learning:

Machine learning is another form of artificial intelligence that allows systems to analyze data and perform tasks more efficiently without the need for specific programming. Due to its ability to analyze large datasets, it has become a powerful tool across all industries. It is classified into three types of learnings: supervised learning, unsupervised learning and reinforcement learning. In this study, we are using a supervised learning

approach, where the algorithm learns from labelled data. The models in this study are trained to predict future trends and directions.

The models are as follows:

- **Logistic Regression:**

  Logistic regression is easy to use and highly efficient for analyzing market dynamics. It is a supervised learning technique utilized for binary classification tasks by predicting the probability of an outcome is called a logistic regression. Logistic regression is not a regression algorithm; as its name suggests, it is a classification algorithm that is commonly used for binary classifications. A logistic function is used to convert predicted values into probabilities ranging from 0 to 1. Such as predicting whether the S&P 500 index direction is up or down. The mathematical formula is as follows:

$$P(Y = 1|X) = \frac{1}{1 + (e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots\ldots\ldots + \beta_n x_n)}}$$

  Where,

  P(Y=1|X) denotes the probability of a positive outcome

  β_0 is the intercept

  $\beta_1, \beta_2, \ldots\ldots\ldots\ldots\ldots \beta_n$ are the coefficients with variables $x_1, x_2, \ldots\ldots\ldots x_n$

  e is the base of the logarithm

  This model can also be expressed in terms of log-odds, also known as logit. It shows the linear combination of input features to the probability. The formula is as follows,

$$logit(P) = \ln\left(\frac{P}{1 - P}\right)$$

  The output is always 0 and 1. If the predicted value is greater than or equal to 0.5, then the value is said to be up; otherwise it's down.

$$Predicted\ direction = \begin{cases} DOWN, & if\ x < 0.5 \\ UP, & if\ x \geq 0.5 \end{cases}$$

  This model predicts the direction and allows us to understand how features influence the market direction.

- **Naïve Bayes:**

  This is a Bayesian-based classification approach that assumes that the characteristics are independent of each other and is known as Naïve Bayes. Naïve Bayes is a simple probabilistic model which is useful for high- dimensional data and also less accurate due to its unrealistic independence assumptions in features.

  This approach is widely used in spam detection, sentiment analysis and financial forecasting and is especially beneficial for high dimensional data. It is primarily based on Bayes theorem, which describes the probability of class C with a feature vector X,

$$P(C|X) = \frac{(P(X|C)P(C))}{P(X)}$$

Where,

P(C|X) is the probability of event C given that X occurs called the posterior probability.

P(C) is the probability of event C occurring called prior probability.

P(X) is the probability of event X occurring called marginal probability.

P(X|C) is the probability of X given that C occurs called the likelihood of vector X.

The naïve independence assumption can be expressed mathematically as:

$P(X \mid C) = P(x_1 \mid C) \times P(x_2 \mid C) \times \ldots\ldots\ldots\ldots \times P(x_{n-1} \mid C)$

The feature vector, in this case is $X = x_1, x_2, \ldots\ldots\ldots x_n$

This simplification considerably reduces the computational complexity and makes the algorithm more efficient. For forecasting the S&P 500 direction, it estimates the probability of the market direction up or down based on the given features.

- **Decision trees:**

  Decision trees are nonparametric supervised learning methods for classification and regression tasks. With its root node, internal node, and leaf node, it resembles a tree structure. The root node represents the whole data. The predictions for the classification tasks are based on the majority class. The leaf node is the prediction or an outcome, and the internal node is the decision of a feature value. In regression tasks, the average of the target values within a leaf node is used to make predictions. The diagram below shows the decision tree structure.

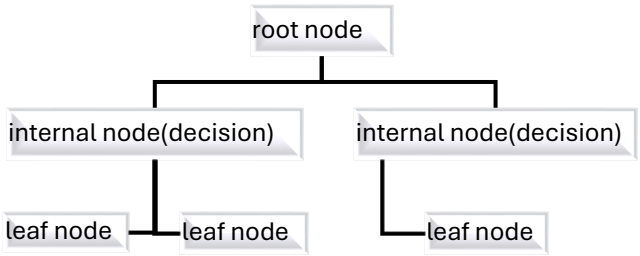

Fig:1.1 decision trees

The decision tree selects the split that optimizes the data acquisition or reduces the impurity. The splitting of the nodes is performed either by Gini impurity or Entropy.

Gini impurity is a measure of decision trees to assess the homogeneity of a sample data. It is computed as

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Where p_i is the probability of class i.

Entropy measures the uncertainty in the dataset.

$$Entropy = -\sum_{i=1}^{n} p_i \, log_2(p_i)$$

In analysing the direction, decision trees continuously split the features into smaller parts until they are capable of predicting movement. These are helpful for understanding and visualizing the forecasts.

- **Random forest:**

  Random forest is a powerful method of an ensemble learning algorithm that can be utilized in both regression and classification tasks. Several decision trees are generated by random forest and each of them has been trained with a random subset of elements and data. Using bootstrap aggregation (bagging), it builds multiple decision trees on trained data. For classification tasks, the random forest results in the class selected by the large number of trees, whereas in the regression, the output is the average of the predictions of the trees.

  In regression,

$$random\ forest = \frac{1}{A}\sum_{a=1}^{A} R_a(x)$$

  Where,

  A is the total number of decision trees.

  $R_a(x)$ is the a^{th}decision tree for input x.

  The following diagram shows its working:

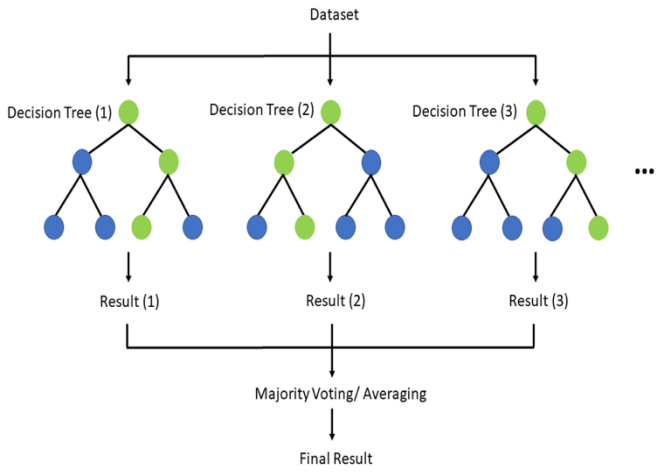

Fig: 1.2 Random forest

The dataset is split into multiple decision trees. Each tree makes an individual prediction (result 1, result 2, result 3……..). These results together form a majority voting for classification and averaging for regression.

Compared to the single decision tree, this method improves predictive performance by lowering the risk of overfitting and boosting model stability. Random forest is widely used in marketing trends and financial forecasts because of its accuracy and adaptability.

- **Multi-layer perceptron:**

  A multi-layer perceptron (MLP) is a type of supervised learning and also an Artificial neural network, commonly used in backpropagation methods to manipulate weights according to error rate. A perceptron is a linear classifier that maps input features to output. Whereas MLP has a collection of interconnected perceptrons called neurons. It consists of three layers:

  The layer that receives the information is called the input layer.

  The layers that handle the input are known as the hidden layer.

  The layer that provides the final result is called the output layer.

  Where every layer is connected to each layer, this also has a non-linear characteristic, where the usage of the activation function shows the non-linear interaction between input and output.

  The mathematical formula for MLP is that each neuron in the hidden layer processes the given input to the hidden layer.

$$z = \sum(w.x) + bias$$

  w is the weights

  x is the input

  In the hidden layer, to handle complex patterns we use the activation function. The activation function is the key element in the multilayer perceptron without this it become like a linear model.

$$a = f(z)$$

  f(z) is the activation function

  a is the output

  The following are common activation functions: sigmoid, tanh, and RELU.

  Sigmoid function- it is useful for probabilities with an output range (0,1)

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

  Hyperbolic tangent function (tanh)- similar to sigmoid but centred around zero with output range (-1,1)

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

  Rectified linear unit (RELU)- it is a simple function, that returns the input directly positive, otherwise 0. The output range is [0,∞)

$$f(x) = max(0, x)$$

  The below diagram explains the structure of MLP:

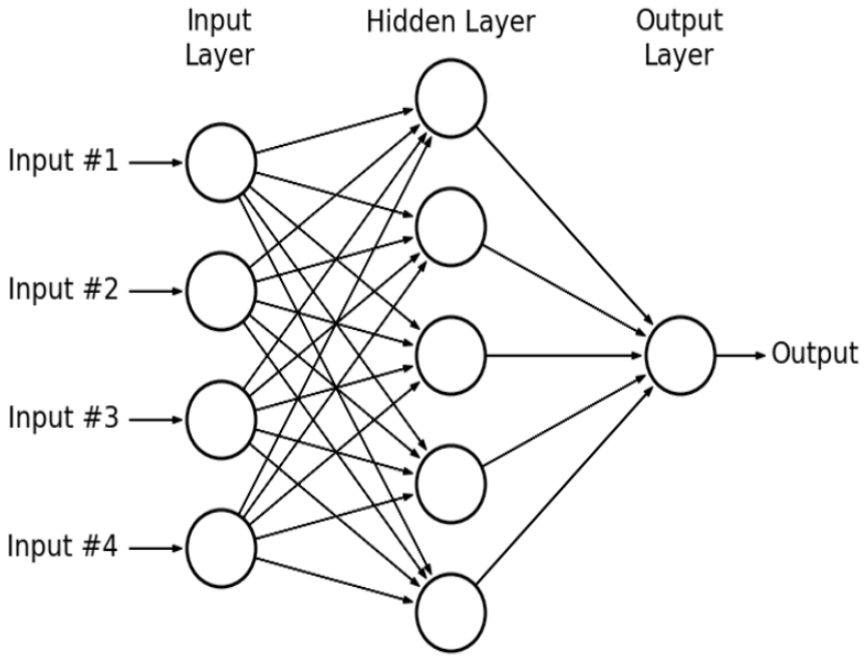

Fig: 1.3 multi-layer perceptron

From figure 1.3 we can start with the input layer, where it has four inputs (#1, #2, #3, #4) each input consists of a feature from the dataset. The diagram shows that it has one hidden layer with five neurons, where it receives input from all the input neurons. Each neuron in the hidden layer has a weight and bias. To provide nonlinearity and to understand complex patterns, the neurons apply an activation function. All the neurons in the hidden layer are connected to one output. The output can be regression, classification and multi-class classification.

1.4 Research Aim and Objectives:

The main aim of this research is to examine the historical trends and create a forecasting model to predict the future direction of the S&P500 index using machine learning analysis. The findings aim to assist market practitioners in making data-driven decisions while exploring the limitations of prediction technologies.

The objectives of the study are as follows:

- To analyze the historical trends of the S&P500 by examining the past price movements and the effects of economic events.
- To build a forecasting model that predicts the direction of the S&P500 (up/down).
- To implement and train the supervised machine models like logistic regression, naïve bayes, random forest, decision trees and multi-layer perceptron.
- To compare performance, by analyzing the confusion matrix, the accuracy score and moving averages.
- To assess the influential factors like the volatility index (VIX), energy, financial, health, oil and gold on the S&P500 index.

- To provide information to stakeholders and investors to improve their risk management and decision-making.

1.5 Significance of the study:

The significance of this study highlights the challenges involved in predicting the direction of the S&P 500. To provide valuable insights, this study analyses the data using ensemble models like decision trees and random forest, neural networks like MLP, and classical models like logistic regression and naïve Bayes. This study also captures the complex nature of the financial market in the real world by combining technical indicators with economic indicators like gold, oil, energy, health, financial and volatility.

In the context of financial forecasting, it also shows the strengths and limitations of logistic regression, naïve bayes, decision trees, random forest and MLP. By determining the efficient models, this study provides a foundation for future research and applications in trading strategies.

1.6 Organization of thesis:

The format of this report is as follows:

- In Chapter 2, we provide a literature review on the analysis of S&P 500 direction using machine learning models.
- In Chapter 3, we describe the methodology, which includes describing the dataset, preprocessing the data, detailing the models used in the study and applying the models to the data.
- In Chapter 4, we outline the analysis and interpret the results by describing the model's limitations, performance, and comparisons.
- In Chapter 5, we summarize the findings and recommendations for further research.
- In the Appendices, we have included all the relevant and reviewed articles connected to our work, as well as the research code.

LITERATURE REVIEW

Forecasting the S&P 500 is extremely complicated because of its non-stationary and noisy nature due to the influence of economic indicators and geopolitical events. This literature review focuses on methods for forecasting financial markets, including traditional methodologies, machine learning techniques, deep learning techniques, and the impact of feature engineering and technical indicators. By examining the data from other studies, we can identify research gaps and limitations that will help our study.

According to Ahmed et al. (2022), there is a growing trend in the literature on the use of artificial intelligence (AI) and machine learning in finance, especially in the fields of big data analytics, blockchain, anti-money laundering, oil price prediction, stock price prediction, bankruptcy prediction, and portfolio management. It also shows that machine learning can be used for high-dimensional noisy data, whereas traditional statistical methods have limitations.

2.1 Traditional models:

In stock market prediction, most of the basic forecasting methods used are traditional statistical methods such as autoregressive moving averages (ARMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH).

ARMA was introduced by Box and Jenkins in 1976. It is considered robust and is used widely for time series forecasting. ARMA (p, q) is defined as:

$$X_t = \varepsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}$$

Where X_t is the forecast value at time t, ε_t is white noise, ϕ_i is the autoregressive coefficients, and θ_j is the moving average parameters. Using the ARMA model to forecast the stock market, Lv et al (2022) showed that despite the ARMA model being good at identifying short-term autocorrelation, they are unable to cope with non-linearity in financial data.

Financial time series exhibits volatility clustering, which cannot be identified by ARMA. So, in 1986 Bollerslev introduced the GARCH model, which is widely used to forecast volatility. A GARCH (1,1) process is defined as

$$\sigma_t^2 = \alpha_0 + \alpha_1 \varepsilon_{t-1}^2 + \beta_1 \sigma_{t-1}^2$$

Where σ_t^2 is the conditional variance.

The GARCH model is into risk and clustering methods, but it remains limited for directional classification tasks due to its linearity.

2.2 Early Study:

One of the earliest empirical studies relating accounting earnings to stock returns was conducted by Ball and Brown (1968), showing that investor decisions are influenced by financial information releases. In 1970, Fama stated through the efficient market hypothesis that the stock market direction is unpredictable. By showing that risk perception and loss aversion have a major impact on investment decisions, Kahneman and Tversky's (1979) Prospect Theory also had an impact on finance. However, in later studies it is suggested that in some conditions models can still be predictive. From behavioral finance studies by De Bondt & Thaler (1985) emphasized sentiment effects and investor overreaction, indicating that there might be inefficiencies. In parallel with the behavioral study, researchers investigated the impact of external variables. While Beaulieu et al. (2005) studied political risk as a cause of volatility, Andersen et al. (2007) explored real-time price discovery across global financial markets.

These theoretical discussions set the framework for machine learning, which can detect hidden non-linear patterns.

2.3 Machine learning approaches:

Logistic regression:

These limitations are overcome by machine learning techniques. The popular model for financial forecasting is logistic regression, which is particularly useful for classification problems where binary outcomes must be predicted. Zhang and Zhou (2017) used logistic regression to forecast the daily direction of the market, showing that when features are chosen and scaled properly, logistic regression was said to function effectively despite being linear. Due to its robustness, logistic regression is one of the best models for forecasting financial markets. A comparative study for stock prediction by Patel et al (2015) using logistic regression, decision trees, random forests and support vector machine (SVM) explained that its performance depends on feature structure and market conditions.

Naïve Bayes:

Naïve Bayes is also applied to forecasting stock; it is based on Bayes theorem with the assumption of conditional independence. Rishi et al (2001) suggested that it underperformed for correlated features, whereas in a recent study Alkubaisi (2019) showed that it can achieve high accuracy in sentiment classification. These studies find that naïve bayes is a simple and useful model which can perform efficiently with feature engineering.

Ensemble methods:

For stock price prediction, three well-liked machine learning techniques are decision trees, XGBoost, and random forests. The decision tree technique is a divide and conquer approach for identifying characteristics and generalizing data regularities, which is helpful and important for modeling and forecasting, according to Myles et al. (2004). They also tend to overfit noisy data. They also use recursive data partition by choosing the features which maximize information gain, as follows:

$$IG(S,A) = Entropy(S) - \sum_{v \in A} \frac{|S_v|}{|S|} Entropy|S_v|$$

Where S is the dataset of the node, A is the feature chosen for the split, $|S_v|$ is the number of samples in the subset.

In 2001, Breiman proposed the random forest approach to address overfitting by averaging multiple decision trees with bagging and feature selection in the random forest has better stability and accuracy. According to Patel et al (2015) and Ghosh et al (2022), random forests are effective classifiers for stock prediction, however their performance is based on data characteristics. Compared to logistic regression, random forest is less interpretable.

XGBoost is an advanced ensemble learning achieving results in various stock market predictions.

Multi-layer perceptron:

Among neural networks, a multi-layer perceptron performs well in financial forecasting. Kamalov et al. (2021) conducted a comparative analysis on deep learning architectures, including MLPs, for predicting the daily direction of the S&P 500 index. Despite the dominance of convolutional models, MLPs maintained their competitiveness in terms of computing efficiency and performance, particularly when trained using well-chosen financial indicators like past price return, volatility, technical indicators and features (VIX, gold, oil….). Bieganowski and Slepaczuk (2024), extended this by stating a supervised autoencoder MLP model for financial time series forecasting, showed that improvements such as noise augmentation and triple barrier labeling can enhance performance in predicting the direction of the S&P 500 index. In financial forecasting, MLPs are highly adaptable, but comprehending them is difficult.

2.4 Advanced approaches

Traditional statistical models and fundamental machine learning techniques have provided valuable insights, but increasingly recent research has concentrated on sophisticated strategies that use deep learning architectures or combine the advantages of several approaches. These methods seek to overcome the drawbacks of past models, especially when it comes to handling dynamic market structures, high dimensionality, and non-linearity.

Recently, the role of hybrid models is growing in which they combine preprocessing, feature engineering and machine learning. Alhanity (2015) concluded that the results are enhanced by combining techniques such as EMD (Empirical Mode Decomposition) with neural networks and regression. In the same way, Alsubaie et al. (2019) showed that in financial forecasting, deep learning performs better than traditional statistical methods.

2.5 Feature Engineering

Feature engineering is important for prediction as it captures complex patterns rather than price movements. Researchers use past price movements, volatility, technical indicators and external factors. The S&P 500 is

heavily impacted by the price of gold and oil, according to Gokmenoglu and Fazlollahi (2015), whereas Ahas et al. (2022) emphasized the wider application of AI in commodity prediction and portfolio optimization.

Instead of depending only on past price movement, it includes elements that ensure that models capture macroeconomic patterns. To support this proposal Baker, Bloom and Davis (2016) proposed the Economic Policy Uncertainty (EPU) index and explained how uncertainty increases market volatility.

2.6 Research Gaps and Contributions

Despite improvements, several gaps still exist.

- Since performance frequently varies depending on dataset features, feature selection, and time horizons, there is no superior model for financial forecasting.
- In a machine learning context, only a small number of studies effectively combine behavioral and macroeconomic elements with technical indicators, despite their growing recognition as key influencers of stock market dynamics.
- Additionally, there is still little research on the trade-off between predictive strength (as in Multi-Layer Perceptrons and ensemble approaches) and model interpretability (as in Logistic Regression), especially when it comes to short-term forecasting of equity indices like the S&P 500.

This literature review shows us an understanding of statistical models like ARMA and GARCH, as well as machine learning models, and deep learning models. While advanced models frequently provide better performance, simpler methods are still useful for benchmarking and understanding. This project attempts to apply and compare the performance of logistic regression, naïve bayes, decision trees, random forest and multi-layer perceptron on the S&P 500. By discovering their performance through accuracy, F1 score and feature importance which provides the insights for the best fit and identifying the influencing indicators

METHODOLOGY

This implements machine learning models to forecast the direction of the S&P 500. In this methodology, we provide a detailed flow that involves data collection, data preprocessing, feature engineering, model training and evaluation of metrics. The steps are as follows:

3.1 Data Collection

In this study, a dataset is collected from the historical data of the S&P 500 from the period of 2020 to 2025, and it includes everyday trading value of the features. The data is taken from the publicly available financial database or source called Bloomberg. The variables included in the study are:

features	description
SP_VOLUME	SP 500 trading volume
SP_LOW	SP 500 low price
SP_HIGH	SP 500 high price
SP_PX_VOLUME	SP 500 price weighted volume
SP_VOLATILITY_30D	S&P 500 30-Day Historical Volatility
SP_VOLATILITY_90D	S&P 500 90-Day Historical Volatility
SP_PX_OPEN	S&P 500 Index Opening Price
SP_PX_CLOSE_1D	SP 500 close price (change compared to the previous day)
SP_PX_LOW	S&P 500 Index Daily Low Price
SP_PX_HIGH	S&P 500 Index Daily high Price
VIX_OPEN	VIX index opening value
VIX_PX_CLOSE_1D	VIX Index Closing Price (change compared to the previous day)
VIX_HIGH	VIX high value
VIX_LOW	VIX low value
VIX_VOLATILITY_30D	VIX 30-Day Historical Volatility
VIX_VOLATILITY_90D	VIX 90-Day Historical Volatility
VIX_PX_OPEN	VIX Index Opening Price
VIX_PX_HIGH	VIX Index daily high Price
VIX_PX_LOW	VIX Index daily low Price

ENERGY_PX_LAST	Energy Sector Index Last Traded Price
FIN_PX_LAST	Financial Sector Index Last Traded Price
GOLD_PX_LAST	Gold Price – Last Traded Value
HEALTH_PX_LAST	Healthcare Sector Index Last Traded Price
OIL_PX_LAST	Oil Price – Last Traded Value

Table-3.1 features

From these features volatility indices (VIX), energy, financial, health, gold and oil are macroeconomic indicators whereas close, open, low, high, short/long term volatility indices of 30D and 90D are technical indicators.

This combination of technical and economic elements ensures a comprehensive understanding of the causes impacting stock market movements.

3.2 Data Processing:

Data processing is an important step in preparing raw financial data for machine learning models. Because stock market data is noisy, unreliable, and affected by multiple external factors, preprocessing is necessary to achieve consistency and clear patterns. This step usually emphasizes cleaning up data and preparing the data for model training.

The following steps were carried out:

Handling missing values:

The data that we obtain from open sources often contain missing values in the dataset, which occasionally occur due to non-trading days, technical issues or incomplete reporting. To maintain data integrity, these values were removed from the dataset rather than imputed. Because the dataset included several years of daily observations, deleting rows with missing values did not considerably reduce the sample size and ensured that only complete and reliable records were used for model training and testing. To handle the missing value in the model, the dropna ( ) function was used.

Feature Scaling:

In the dataset, the features contains variables which are measured on different scales. For instance, S&P 500 index values are in thousands whereas commodity prices (gold, oil, health, energy and financials) are much smaller. These characteristics can have a negative impact on some machine learning models performance. For models like logistic regression and multi-layer perceptron are sensitive to the input values, as they rely on optimization approaches that develop quickly when all variables are on the same scale.

To ensure model comparability, we apply standardization (Z-score normalization) which gives a mean of 0 and a standard deviation of 1 for each feature. It is given by:

$$Z = \frac{X - \mu}{\sigma}$$

Z is the standardised value, X is the original value of a feature, μ is the mean and σ is the standard deviation. In this, the values can take both positive, negative and around zero; it doesn't have a fixed range like min-max scaling.

Label Creation:

Target variable- In our study, the prediction is a binary classification task. The target variable represents the daily directional movement of the S&P 500. It is defined as:

$$Y_{t+1} = \begin{cases} 1, if\ P_{t+1} - P_t\ > 0 \\ 0, otherwise \end{cases}$$

P_t represents the closing price value of the S&P 500 at time t.

A positive daily return, or upward movement, is indicated by a number of 1, while a downward or neutral movement is indicated by a value of 0. This approach makes it possible for the models to focus on the direction of price change rather than its value, which is more relevant for tasks involving decision-making like risk assessment and trading strategy development.

Train/test split:

After handling missing values, scaling the data and labelling the target value, the dataset is used to train machine learning models. The dataset was divided into two subsets: training and testing to evaluate the performance. In this study, we considered an 80-20 split where 80% of the dataset was used for training to fit models and 20% of the dataset was used for testing, which provides performance on unseen observations. This ratio is a good balance to provide a reliable evaluation.

The dataset contains the past five years of data, where the previous observations are used for training and the most recent data was retrieved for testing the dataset.

The models (Logistic Regression, Naïve Bayes, Decision Trees, and Random Forests) were trained with default hyperparameters using their standard implementations in scikit-learn because this study is more concerned with comparative performance rather than optimizing parameters.

Moving Averages:

Moving averages (MA) are used to identify the direction of the stock market. It is commonly used in time series to smooth out short term cycles and highlight long term trends. We identified moving averages from the closing prices. The two types of moving averages are:

Simple Moving Average (SMA):

A simple moving average is the arithmetic mean of the previous n prices. It relies on the past price movements. The formula is

$$SMA = \frac{P_1 + P_2 + P_3 + \ \ldots\ldots + P_k}{k}$$

Where P= closing prices and k =number of days

Exponential Moving Average (EMA):

It is also known as exponentially weighted moving average (EWMA) where it weights effects of the most recent prices. The formula is

$$EMA_t = \alpha . P_t + (1 - \alpha) . EMA_{t-1}$$

Where,

$$\alpha = \frac{smoothing}{K + 1}$$

In the above formula smoothing factor has many possible choices, but the study choice is 2 where the study choice is 5 and 20. More recent observations have a greater impact on the EMA if the smoothing factor is increased.

The EMA's behaviour is frequently equivalent to that of a Simple Moving Average (SMA) of the same length because to this formula, which ensures that it gives more weight to recent prices while still taking previous observations into consideration.

P_t = price at time t, EMA_{t-1}= price from previous day of t and k= number of days.

In our study, we perform both SMAs and EMAs for 5-day and 20-day cycles.

3.3 Software and tools:

- The extracted raw data from the open source was gathered and handled in **Excel**.
- **Python (Jupyter notebook)** was used as a programming language where model training and evaluation were applied.
- **Word** was used for the write up and report of the study.
- **Pandas** are used for loading the data, data manipulation, data preprocessing and feature engineering.
- For mathematical computations, **NumPy** was implemented.
- **Seaborn and matplotlib** are used for data visualization and feature importance ranking.
- **Scikit-learn** was chosen to implement our models logistic regression, naïve bayes, decision trees, random forest and multi-layer perceptron and to evaluate the models.

3.4 Evaluation of the model:

The prediction performance is evaluated using machine learning models with several kinds of evaluation metrics. To provide a comprehensive evaluation for the forecasting task the chosen metrics are classification metrics i.e., confusion matrix, accuracy, F-1 score, precision and recall.

Confusion matrix:

A confusion matrix is a table which is used to identify the performance of a classification model. In our study, forecasting is in binary classification, which predicts upward and downward movement. It represents models predicted values with actual values. Allowing for a deep understanding of precise predictions and errors. The confusion matrix is shown below.

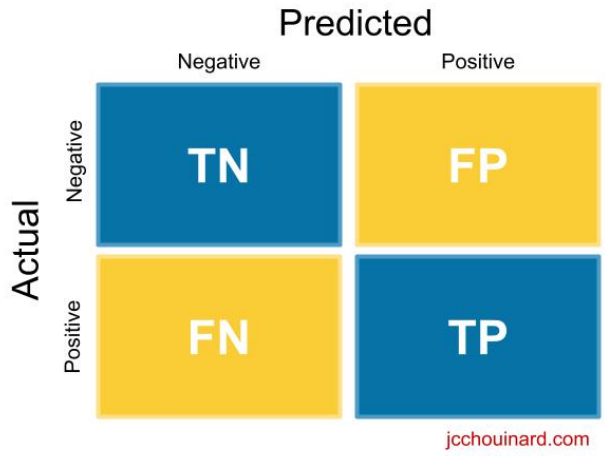

Fig- 3.2 confusion matrix

From the figure, it has four outcomes:

- TP (True Positive) – correctly predicted positive (upward direction)
- TN (True Negative) – correctly predicted negative (downward direction)
- FP (False Positive) – incorrectly predicted positive (when the market is in a downward direction, it predicts an upward direction)
- FN (False Negative) – incorrectly predicted negative (when the market is in an upward direction, it predicts as downward direction)

From this confusion matrix all other metrics (accuracy, precision, recall and F-1 score) are calculated.

Accuracy:

The widely used metric for evaluating classification models is accuracy. It evaluates the amount of positive and negative events that are correctly classified out of all the assumptions.

The mathematical formula for accuracy is:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

It provides a clear understanding of the overall performance. Sometimes, if the dataset is imbalanced, it can give misleading predictions. For example if there are more frequent upward market movements than downward market movements, it predicts upward and achieves good accuracy score without understanding the market behaviour.

To avoid this, accuracy is used as a base metric and always evaluated with the other metrics (precision, recall and F-1 score).

Precision:

Precision is the ratio between true positives and all the positives. It mainly focuses on the models positive prediction.

The formula for precision is:

$$Precision = \frac{TP}{TP + FP}$$

In financial forecasting, this reduces the possibility of acting on incorrect upward signals by ensuring that a model's predictions of a market rise are reliable.

Whereas it doesn't consider the false negatives. A model with good precision can still fail to capture false negatives, which are addressed by the recall.

Recall:

It is also known as true positives or sensitivity. Recall is the measure of correctly predicted positive outcomes of the model.

The formula for recall is:

$$Recall = \frac{TP}{TP + FN}$$

A model with a high recall is extremely good at not missing anything important. It identifies nearly all of the positive observations of the data. Focusing only on recall can often lead to a high number of false positives as it takes all the positives, even though sometimes there is an incorrect labelling of negatives as positives.

F-1 score:

The F-1 score is a combination of precision and recall into a single metric. For an imbalanced dataset, it provides a better overall models performance. It performs well when both false positives and false negatives

are significant. It makes the assumption that recall and precision are equally relevant, but in some circumstances, one may be more significant than the other.

The formula for the F-1 score is

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

When Precision and Recall compete, as they often do in financial forecasting, the F1-Score provides a balanced evaluation. It is useful when the dataset is uneven, as it keeps models from unfairly choosing the majority class.

Specificity:

Specificity is also called the true negative rate. It shows the negative outcomes that are identified correctly by the model.

The formula is:

$$Specificity = \frac{TN}{TN + FP}$$

Specificity only considers that are negative. A model with high specificity avoids too many incorrect upward predictions, which could mislead us.

Support:

Support is the number of actual occurrences of each class in the dataset that were used for evaluation.

$$support = no.of\ classes\ in\ a\ matrix$$

Macro Average:

Macro average calculates the metric individually and then it uses the simple arithmetic mean.

$$macro\ average = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Where n is the number of classes and x_i is the metric class value (precision, recall or F1 score) of i.

Each class is treated equally no matter of its value.

Weighted Average:

Weighted Average can be applied to assess model performance by computing the average of metrics across all classes; unlike the macro average it takes values of each class.

$$weighted\ average = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

Where w_i is the weight of the metric class.

$$w_i = \alpha(1-\alpha)^{i-1}$$

i= 1,2,3,4……n

Here weights decreases exponentially for previous prices

This applied when the dataset is imbalanced.

Summary:

In financial forecasting, where datasets can appear unbalanced and some types of errors carry more weight than others, accuracy is a common measure of overall correctness, but it is insufficient on its own. So the additional metrics are also considered. Precision assures that upward predictions are accurate, eliminating false market predictions, whereas recall evaluates the model's capacity to record all actual upward moves, minimizing missed possibilities. These two metrics are balanced by the F-1 score and provide a comprehensive evaluation of performance.

We derived all these metrics from a confusion matrix, to compare classical, ensemble, and neural models in predicting the direction of the S&P 500 direction using a thorough evaluation methodology.

MODEL IMPLEMENTATION AND RESULTS

The process of implementation started with Python programming in Jupyter Notebook as the development environment. The implementation followed a step-by-step method, beginning with data input and proceeding through preprocessing, splitting, model training, and evaluation. The procedure followed a standardized workflow, ensuring results that were transparent and reliable. The detailed primary coding stages were:

4.1 Importing libraries:

In coding, the first step always start with importing all the necessary libraries required for analysis, visualization, and machine learning. Every library has its unique role in coding.

The following are the libraries involved in this study.

```
[132]: import pandas as pd
       import numpy as np
       import matplotlib.pyplot as plt
       import seaborn as sns
       from sklearn.preprocessing import StandardScaler
       from sklearn.model_selection import train_test_split
       from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classification_report
```

Fig: 4.1

- **Pandas (pd)** : It was used to handle the structured data and prepared for the machine learning models which helps in analysis and manipulation.
- NumPy (np) : This library carries out effectively for mathematical computation specifically while using the matrices and arrays.
- Matplotlib.pyplot (plt) : It is widely used for visualizing and plotting the patterns in the dataset like trends, graphs, and heatmaps.
- Seaborn (sns) : This library is also based on matplotlib for visualization which provides advanced plotting for statistical graphs (boxplot, pairwise plot, histogram……..).
- Sckit-learn : This library is applied to implement the machine learning models where,
    i. Standard scaler - is used to standardised the dataset.
    ii. Train test split - ii. Split the dataset into two sets for training and testing.
    iii. Accuracy score, f1 score, confusion matrix, classification report – are the evaluation metrics used to access the model performance.

4.2 Loading the dataset:

The dataset included the last five year daily data of the S&P 500 from 2020 to 2025, together with technical and economic indicators such as oil, gold, energy, volatility and financials indices. These characteristics were chosen because they are known to affect market behaviour.

Using the pandas library we imported the dataset into python.

```
[138]: df=pd.read_csv("data.csv")
       print(df.info())
       print(df.head())
```

Fig: 4.2

The file (data.csv) was loaded into a data frame for further analysis using the read_csv() function. The loaded dataset was analysed using the functions:

info() function, which gives a thorough overview of the dataset by displaying information about its rows, columns, data types, and missing values.

head() function, which displayed the first few rows of the dataset and allow a quick view of variables. This ensured that the data was loaded accurately and showed that it followed the format of the expected financial time series.

Summary of the dataset:

The below figure shows us the overview of the dataset.

	SP_PX_VOLUME	SP_PX_LOW	SP_PX_HIGH	SP_PX_OPEN	SP_PX_CLOSE_1D	SP_VOLUME	SP_HIGH	SP_LOW	SP_VOLATILITY_30D	S
count	1.255000e+03	1255.000000	1255.000000	1255.000000	1255.000000	1.255000e+03	1255.000000	1255.000000	1255.000000	
mean	6.978523e+08	4507.936765	4561.797968	4535.480143	4534.185594	6.978523e+08	4561.797968	4507.936765	16.597673	
std	2.790198e+08	764.574260	767.298988	766.237531	765.984906	2.790198e+08	767.298988	764.574260	6.976514	
min	2.065437e+08	3101.170000	3128.440000	3105.920000	3100.290000	2.065437e+08	3128.440000	3101.170000	6.510000	
25%	5.717279e+08	3938.955000	4000.945000	3971.275000	3971.040000	5.717279e+08	4000.945000	3938.955000	11.685000	
50%	6.421719e+08	4355.410000	4407.550000	4380.580000	4378.380000	6.421719e+08	4407.550000	4355.410000	14.570000	
75%	7.356206e+08	5089.895000	5145.000000	5118.455000	5110.465000	7.356206e+08	5145.000000	5089.895000	19.820000	
max	3.305469e+09	6174.970000	6215.080000	6193.360000	6173.070000	3.305469e+09	6215.080000	6174.970000	43.550000	

8 rows × 24 columns

: .ITY_30D	VIX_VOLATILITY_90D	VIX_PX_OPEN	VIX_PX_HIGH	VIX_PX_LOW	ENERGY_PX_LAST	FIN_PX_LAST	GOLD_PX_LAST	HEALTH_PX_LAST	OIL_PX_LAST
5.000000	1255.000000	1255.000000	1255.000000	1255.000000	1255.000000	1255.000000	1255.000000	1255.000000	1255.000000
5.015657	116.942414	20.465450	21.581426	19.385068	554.672677	618.896916	178.382789	1532.625116	515.617825
6.719960	32.446598	5.749617	6.470999	5.129963	150.851576	113.642886	33.112815	140.438072	151.145279
5.950000	62.820000	11.530000	12.230000	10.620000	209.690000	376.870000	108.270000	1173.970000	162.140000
1.195000	86.280000	16.060000	16.815000	15.315000	414.305000	550.710000	151.405000	1474.295000	394.510000
3.520000	119.410000	19.540000	20.600000	18.570000	621.920000	605.190000	178.800000	1541.870000	578.120000
3.995000	147.920000	23.910000	25.145000	22.450000	673.080000	678.065000	201.530000	1621.335000	624.870000
5.780000	204.600000	60.130000	65.730000	38.580000	749.390000	871.950000	288.200000	1829.710000	734.300000

The S&P 500 has mean values around 4500 which indicates the index level for the observed time period. However, the trade volume (SP_VOLUME) fluctuates significantly, from approximately 206 million to more than 3.3 billion shares. The 30-day and 90-day volatility indices, shows how the market is dynamic and non-stationary, with significant variations across the sample. Macroeconomic indicators, such as oil, gold, and sector indices, also show considerable changes, indicating larger economic influences on the stock market.

In general, the dataset includes both short-term technical movements and external macroeconomic variables, making it ideal for forecasting the S&P 500's direction.

4.3 Data preprocessing:

The steps involved in this process are as follows:

Handling missing values:

Missing values must be handled carefully to ensure the dataset remains reliable. To handle the missing values we applied two approaches:

- Row deletion- The dropna() function was used to remove rows with missing entries, ensuring that only complete observations remained.
- Filling with zero - The fillna(0) function was used to replace missing values with zeros while keeping the dataset size constant. However, in financial time series data, assigning a value of zero has no economic value and might bias the study.

As a result, the row deletion method was ultimately chosen since it produced cleaner and more accurate data for model training and evaluation.

Scaling:

It was required to standardize the features in order to avoid any one variable from influencing the learning process because the dataset included variables that had various scales.

```
[84]: from sklearn.preprocessing import StandardScaler
      scaler = StandardScaler()
      numeric = df.select_dtypes(include='number').columns
      scaled_data = scaler.fit_transform(df[numeric])

[85]: scaled_df = pd.DataFrame(scaled_data, columns=numeric, index=df.index)

[86]: print(scaled_df.head())

   SP_PX_VOLUME  SP_PX_LOW  SP_PX_HIGH  SP_PX_OPEN  SP_PX_CLOSE_1D  SP_VOLUME  \
0     -0.306412  -1.838019   -1.866309   -1.863993       -1.870141  -0.306412
1     -0.391037  -1.807365   -1.817424   -1.814585       -1.849741  -0.391037
2     -0.274438  -1.766970   -1.795473   -1.799324       -1.831202  -0.274438
3     -0.438597  -1.783196   -1.793433   -1.784719       -1.766073  -0.438597
4     -0.374013  -1.791598   -1.809588   -1.802232       -1.811143  -0.374013

    SP_HIGH    SP_LOW  SP_VOLATILITY_30D  SP_VOLATILITY_90D  ...  \
0 -1.866309 -1.838019           1.484253           5.746257  ...
1 -1.817424 -1.807365           1.462737           5.746257  ...
2 -1.795473 -1.766970           1.507204           5.746257  ...
3 -1.793433 -1.783196           1.522983           5.672186  ...
4 -1.809588 -1.791598           1.491425           5.673698  ...

   VIX_VOLATILITY_30D  VIX_VOLATILITY_90D  VIX_PX_OPEN  VIX_PX_HIGH  \
0            0.705586            2.083983     1.819829     1.564058
1            0.692929            2.086451     1.361657     1.050554
2            0.679199            1.770226     1.262358     1.033540
3            0.696576            1.776396     1.438310     1.234611
4            0.710735            1.696800     1.469667     1.322773

   VIX_PX_LOW  ENERGY_PX_LAST  FIN_PX_LAST  GOLD_PX_LAST  HEALTH_PX_LAST  \
0    1.715886       -1.799169    -2.085030      0.529168       -2.523892
1    1.266506       -1.778309    -2.082827      0.490493       -2.464620
2    1.075031       -1.771378    -2.017248      0.490493       -2.379975
3    1.530272       -1.831119    -2.088380      0.511644       -2.452964
4    1.528319       -1.833034    -2.052946      0.636432       -2.451543

   OIL_PX_LAST
0    -1.928034
1    -1.892480
2    -1.895246
3    -1.944165
4    -1.941203

[5 rows x 24 columns]
```

Fig: 4.3

Before performing the transformation, the fit_transform() method computed each feature's mean and standard deviation. After scaling, the converted dataset's values were spread about zero, with the majority falling between -1 and +1.

To ensure that every feature contributes fairly to the learning process, this phase was especially important for models that are sensitive to feature magnitudes, such as Logistic Regression, Naïve Bayes, and MLP.

Target:

A target variable based on the daily closing values of the S&P 500 index was constructed in order to convert the forecasting problem into a binary classification task. The aim is to forecast the market movement whether it will move upward or downward.

```
[87]: # Create the target: 1 if next day is up, 0 if down
      df['Target'] = (df['SP_PX_CLOSE_1D'].shift(-1) > df['SP_PX_CLOSE_1D']).astype(int)
      X = scaled_df
      Y = df['Target']
```

Fig: 4.4

A target variable Y is created if its
1, then next day the market will go upward.
0, then next day market will go downward.
Scaled data is considered as X variable.

(df['SP_PX_CLOSE_1D'].shift(-1) > df['SP_PX_CLOSE_1D']) with this expression the movement between tomorrows closing price is compared with today. Using the function shift (-1), the prices moved up one day as a result, bringing them into line with the movement of tomorrow. The astype(int) function shows true for 1 as upward movement and false for 0 as downward movement.

Train/test split:

To evaluate the model performance, the dataset was split into two subgroups training and testing where machine learning models were fit into training group and testing evaluates the performance of unseen data. Using the function train_test_split () we implemented the split.

```
[88]: from sklearn.model_selection import train_test_split
      X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

[89]: print(f"training set size:{X_train.shape[0]} samples")
      print(f"test set size:{Y_test.shape[0]} samples")

      training set size:985 samples
      test set size:247 samples
```

Fig:4.5

Test_size =0.2, which means we applied 80-20 split where the testing size 20% and training size is 80%. In financial forecasting we commonly apply this split as it provides enough data for model learning and reliable performance of the unseen data.

random_state = 42 is a fixed random seed was chosen to assure consistency. This ensures reliable outcomes by checking that the same train-test split takes place each time the code is executed.

The target variable for training is X_train and Y_train.

The target variable for testing is X_test and Y_test.

From the dataset it contains 985 samples for training set and 247 samples for testing set.

Moving Averages:

In financial forecasting, moving averages are used as technical indicators as they capture short and long term fluctuations in the market movement. Simple Moving Averages (SMA) and Exponential Moving Averages (EMA) were computed using various window lengths in order to provide the dataset with trend-based features.

```
[101]: import pandas as pd

# Assuming your DataFrame is called df and contains 'SP_PX_CLOSE_1D'

# Simple Moving Averages (SMA)
df['SMA_5'] = df['SP_PX_CLOSE_1D'].rolling(window=5).mean()
df['SMA_10'] = df['SP_PX_CLOSE_1D'].rolling(window=10).mean()
df['SMA_20'] = df['SP_PX_CLOSE_1D'].rolling(window=20).mean()
df['SMA_50'] = df['SP_PX_CLOSE_1D'].rolling(window=50).mean()

# Exponential Moving Averages (EMA)
df['EMA_5'] = df['SP_PX_CLOSE_1D'].ewm(span=5, adjust=False).mean()
df['EMA_10'] = df['SP_PX_CLOSE_1D'].ewm(span=10, adjust=False).mean()
df['EMA_20'] = df['SP_PX_CLOSE_1D'].ewm(span=20, adjust=False).mean()
df['EMA_50'] = df['SP_PX_CLOSE_1D'].ewm(span=50, adjust=False).mean()

#Drop rows with NaN values from moving averages (first few rows)
df = df.dropna()

print(df[['SP_PX_CLOSE_1D', 'SMA_5', 'SMA_20', 'EMA_5', 'EMA_20']].head(15))
```

Fig: 4.6

- 5-day, 10-day are short term average which identifies the immediate fluctuations in the market movement.
- 20-day, 50-day are long term average which highlights market trends and removes the daily noise.
- Rolling window () – the average of the given period.

The numerical output of moving averages is;

	SP_PX_CLOSE_1D	SMA_5	SMA_20	EMA_5	EMA_20
49	3398.96	3438.732	3431.1365	3418.487920	3418.810419
50	3339.19	3390.402	3429.0785	3392.055280	3411.227522
51	3340.97	3367.584	3427.4555	3375.026853	3404.536330
52	3383.54	3358.900	3427.9900	3377.864569	3402.536679
53	3401.20	3372.772	3428.9505	3385.643046	3402.409376
54	3385.49	3370.078	3428.7360	3385.592031	3400.798007
55	3357.01	3373.642	3427.8440	3376.064687	3396.627721
56	3319.47	3369.342	3424.5420	3357.199791	3389.279366
57	3281.06	3348.846	3418.7370	3331.819861	3378.972760
58	3315.57	3331.720	3412.9515	3326.403241	3372.934402
59	3236.92	3302.006	3402.6165	3296.575494	3359.980649
60	3246.59	3279.922	3391.0095	3279.913662	3349.181540
61	3298.46	3275.720	3381.7050	3286.095775	3344.350917
62	3351.60	3289.828	3373.8845	3307.930517	3345.041306
63	3335.47	3293.808	3365.6425	3317.110344	3344.129753

Fig- 4.7

From this output we can observe how differently short term moving averages and long term moving averages behave compared to the S&P 500 closing price. The long-term averages (SMA-20 and EMA-20) are smoother and less sensitive, reflecting the market's overall trend, whereas the short-term averages (SMA-5 and EMA-5) track the closing prices more closely and respond quickly to sudden changes. For example on day 59, the closing value is 3236.92, where the short-term averages SMA-5 (3302.006) and EMA-5 (3296.57) follow the decline in which EMA-5 responded quickly. The long-term averages SMA-20 (3402.61) and EMA-20 (3359.98) remains higher than the closing day price on day 59 showing upward trend.

Among the two trends, the EMA adjusts faster than the SMA, keeping it closer to recent price movements. This interpretation highlights that EMA's capture short term movement and SMA's identify long term movement.

From the graphical representation,

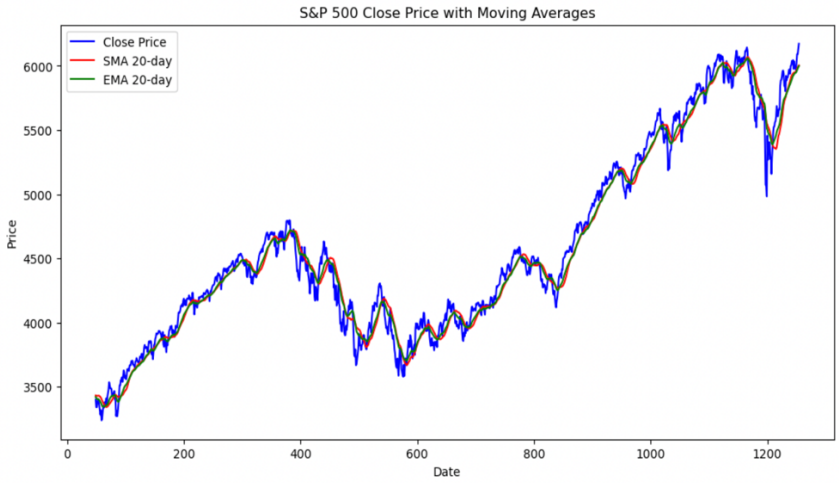

Fig- 4.8

The blue line shows closing price value, the orange line shows SMA-20 day and the green line shows EMA-20 day. The both the line shows the overall upward trend. The SMA line responds slowly to fluctuations in the price movement compared to EMA line, where it captures the market short term stability and volatility.

4.4 Implementing models:

All the models were implemented using the library scikit-learn.

Logistic Regression:

Logistic regression is a good and easy fit for binary classification problems which provides clear decision boundaries. It was implemented to predict the direction of the S&P 500 (up or down). To ensure consistency, the logistic regression classifier was used with fixed random state is used in the python implementation. The model was trained on the training dataset X_train and Y_train and evaluate on the unseen testing dataset X_test and Y-test. The predicted outcomes were carried out in the Y_pred.

The accuracy of the model was computed to evaluate the overall correctness of predictions. However, a classification report was also produced because accuracy might not be enough to display performance by class. For each target variable, up and down it consists of precision, recall, F-1 score and support. These metrics provide a detailed overview of model performance.

The confusion matrix is also generated to visualize and understand the predictions of each target variable clearly.

Result:

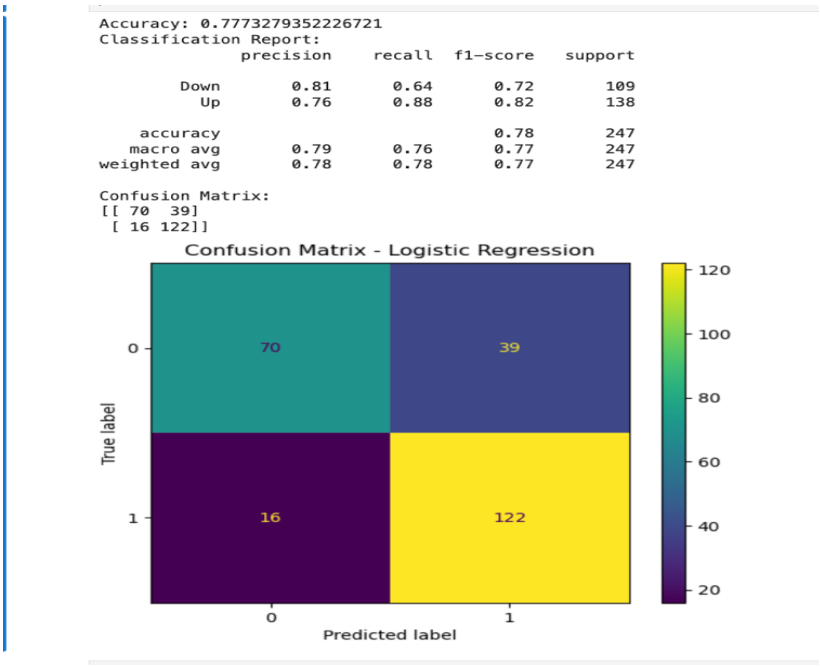

Fig: 4.9

The accuracy of the Logistic Regression model on the test dataset was 77.7%.
From the classification report,

The downward movement has precision 0.81, where 81% of the time it showed downward movement. It has lower recall 0.64 with 64% which shows it missed some actual down movements. The upward movement has precision is slightly lower with 0.76, where it observed 76% upward movement. It has a better recall 0.88 with 88% capturing upward movements showing it has less false negatives. The F-1 score for downward movement is 0.72 and upward movement is 0.82 which shows the balanced performance of precision and recall.

From the confusion matrix,

True negative is 70 that correctly predicts downward movement.
True positive is 122 which correctly predicts upward movement.
False negative is 16 that predicts downward movements as upward.
False positive is 39 that predicts upward movements as downward.

This results showed that logistic regression performed effectively at predicting upward movement than downward movement. However with overall accuracy 77.7% the model performed well in predicting the movement of the S&P 500.

Naïve Bayes:

A naïve bayes is a simple probabilistic model which beneficial for high dimensional data. In this study, direction of the S&P 500 were predicted using the GaussianNB implementation from the Scikit-learn library. The nb_model was trained on the training dataset X_train and Y_train and evaluate on the testing dataset X_test where the predicted outcomes are stored in the Y_pred_nb.

The correct predictions and overall model performance were measured using the accuracy. The precision, recall, and F1-score for each class were included in the classification report that was produced, demonstrating the model's reliability and efficiency in differentiating between "Up" and "Down" movements.

A clear representation of models performance can be visualized in the confusion matrix.

Result:

```
Accuracy: 0.5789473684210527

Classification Report:
              precision    recall  f1-score   support

        Down       0.54      0.34      0.42       109
          Up       0.60      0.77      0.67       138

    accuracy                           0.58       247
   macro avg       0.57      0.55      0.54       247
weighted avg       0.57      0.58      0.56       247

Confusion Matrix:
 [[ 37  72]
 [ 32 106]]
```

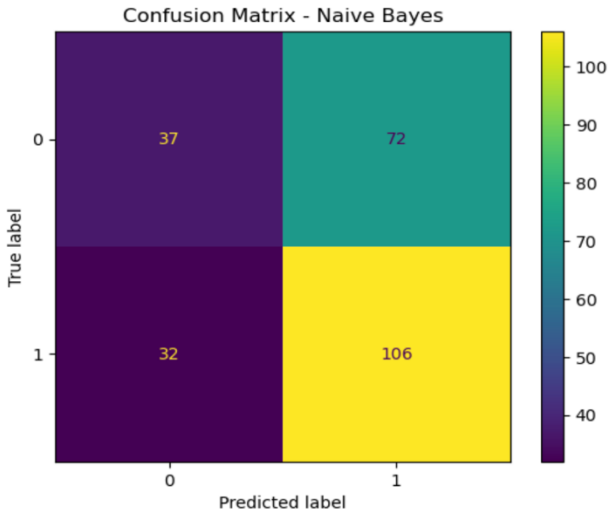

Fig:4.10

The accuracy of naïve bayes model is 57.8% on the test dataset which is lower compared to logistic regression due to its strong independent assumptions it failed to align with financial market trends.

From the classification report,

The downward precision is 0.54, with just 54% of data showing downward movement, and the recall is weaker at 0.34, indicating that the model missed a significant amount of downward movement. It has a lower F-1 score in this movement, at 0.42. The upward movement has a precision of 0.60 and a recall of 0.77, indicating that it is more effective at identifying upward movements. Its F-1 score of 0.56 suggests it performs moderately well with imbalance classes.

From the confusion matrix

True negative is 37 that correctly predicts downward movement.
True positive is 106 which correctly predicts upward movement.
False negative is 32 that predicts downward movements but actually it was upward.
False positive is 72 that predicts upward movements but actually it was downward.

Naïve Bayes was skewed towards predicting "Up" movements, as shown by this distribution. The accuracy is also low with 57.8% as a result, the model was less dependable overall than other classifiers, even though it showed significant prediction possibility of rising trends.

Decision trees:

Decision trees are best to understanding and visualizing the forecasts. The model was implemented using the DecisionTreeClassifier with the fixed random state to ensure its consistency. The dt_model was trained on the training dataset X_train and Y_train and evaluated on the testing dataset. In Y_pred_dt predicted outcomes are stored.

The model's accuracy was evaluated on the overall performance to calculate the number of correct predictions. A classification report for the target variable which includes precision, recall, F-1 score and support.

ConfusionMatrixDisplay was used for showing the confusion matrix, with a label that showed it corresponded to the Decision Tree model for easier comprehension.

Result:

```
Accuracy: 0.5708502024291497

Classification Report:
              precision    recall  f1-score   support

        Down       0.51      0.61      0.55       109
          Up       0.64      0.54      0.59       138

    accuracy                           0.57       247
   macro avg       0.57      0.57      0.57       247
weighted avg       0.58      0.57      0.57       247

Confusion Matrix:
 [[66 43]
 [63 75]]
```

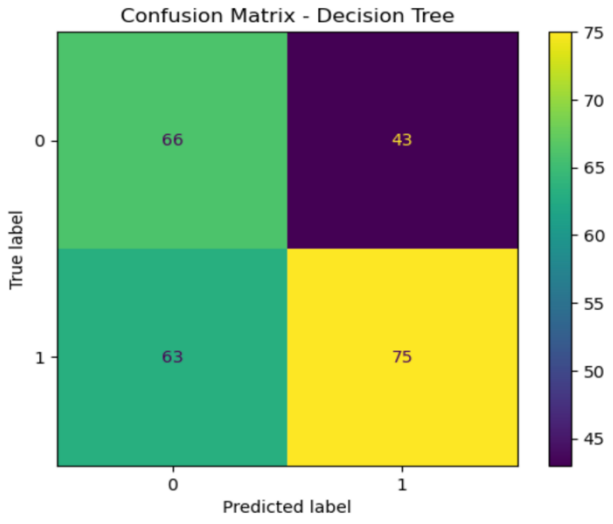

Fig: 4.11

The decision tree obtained an overall accuracy at 57.08% on test dataset. Whereas these are good at capturing non-linear correlations, in this situation, the model had limited ability to forecast for the S&P 500 movements.

From the classification report,

The precision and recall for the downward movement were 0.51 and 0.51, respectively. This indicates that 61% of actual downward movements were identified by the model, even though only slightly more than half of the predicted downward movements were accurate. A weak performance was indicated by the F1-score of 0.55. The upward movement has good precision at 0.64 and weak recall at 0.54 indicating it was missing half of actual upward movement. The F-1 score is 0.59.

From the confusion matrix,
True negative is 66 that correctly predicts downward movement.
True positive is 75 which correctly predicts upward movement.
False negative is 63 that predicts downward movements but actually it was upward.
False positive is 43 that predicts upward movements but actually it was downward.

This result suggests that it performs evenly in both the upward and downward movement, where its overall efficiency limited between precision and recall. The low accuracy indicates the overfitting or it failed to capture the complex patterns in forecasting financial data.

Random forest:

The random forest is applied to deal the limitations of decision trees. To ensure reliability, the model was developed using 100 trees (n_estimators=100) and with a fixed random state. The rf_model was trained on the training dataset X_train and Y_train. The test dataset's predictions (Y_pred_rf) were generated after training.

Accuracy, Classification Report (Precision, Recall, F1-Score), and the Confusion Matrix are classification metrics that were used to assess the model's performance. A confusion matrix visualization was developed as well to help with the interpretation of the classification results.

A confusion matrix visualization was developed as well to help with the interpretation of the classification results. The true positives, true negatives, false positives, and false negatives that the Random Forest model predicted were then represented graphically.

Result:

```
Accuracy: 0.5465587044534413

Classification Report:
              precision    recall  f1-score   support

        Down       0.49      0.58      0.53       109
          Up       0.61      0.52      0.56       138

    accuracy                           0.55       247
   macro avg       0.55      0.55      0.55       247
weighted avg       0.56      0.55      0.55       247

Confusion Matrix:
 [[63 46]
 [66 72]]
```

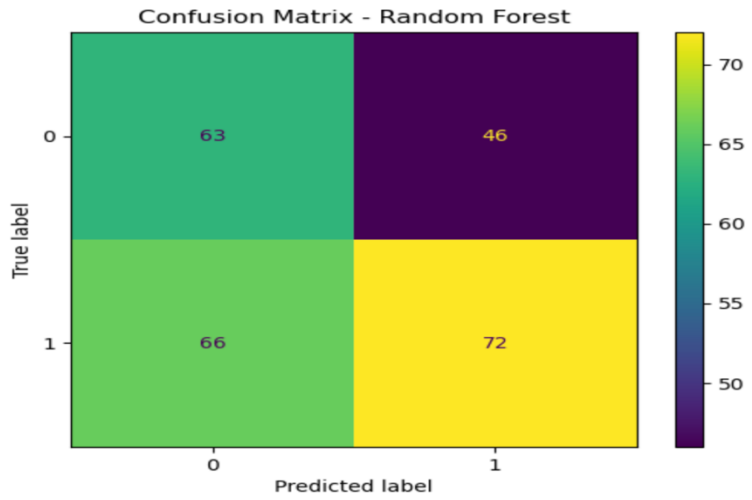

Fig: 4.12

The random forest has an overall accuracy of 0.546 which 54.46% on the test dataset, showing a poor performance on predicting the direction of the S&P 500.

From the classification report,

The downward movement precision is 0.49 and recall is 0.58 which tells us that it identified 58% of actual downward movement with the F-1 score of 0.53. The upward movement has good precision with 0.61 suggesting reliability in its movement. It has a recall with 0.52 which shows that it nearly miss 50% of the actual upward movements. It has a F-1 score of 0.56

From the confusion matrix,
True negative is 63 that correctly predicts downward movement.
True positive is 72 which correctly predicts upward movement.
False negative is 66 that predicts downward movements but actually it was upward.
False positive is 46 that predicts upward movements but actually it was downward.

The random forest results didn't show the significance performance in forecasting the direction of the S&P 500 as the model struggled to distinguish between the upward and downward movements. The noisy and non-linear character of financial time series data may have influenced Random Forest's overall lack of effectiveness with this dataset.

Multi-layer Perceptron:

Multi-layer perceptron is effective for datasets where relationships cannot be effectively captured by simpler models. This model was implemented using MLP Classifier where it trained on X_train and Y_train and evaluate on testing dataset X_test.

In this model the hidden network used 100 neurons, that balances computing efficiency with the ability to model non-linear patterns. The maximum number of iteration are 500 to ensure the convergence of learning process and random state is fixed.

Accuracy is the percentage of accurately predicted outcomes. The classification report offers detailed precision, recall, and F1-score for both the "Up" and "Down" classes, allowing a more fair assessment of the model's performance. The Confusion Matrix Calculates the total number of correct and wrong predictions across both classes.

Result:

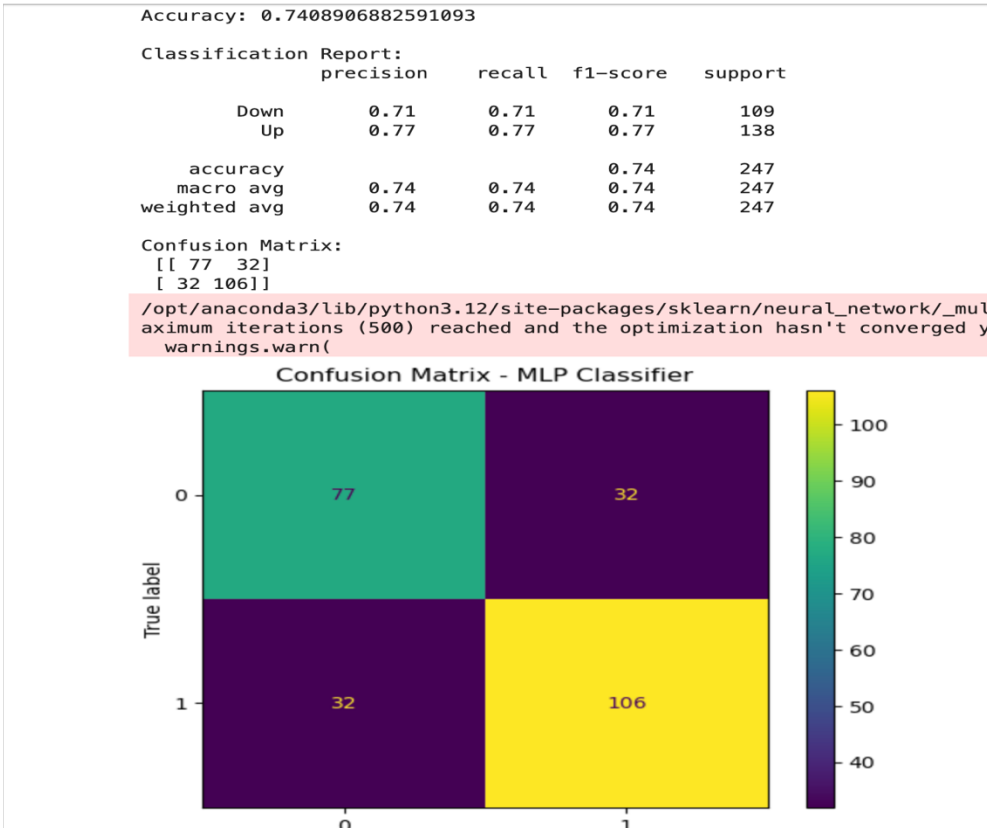

Fig: 4.13

The multi-layer perceptron suggested an overall accuracy with 74.08% which shows as the another strongest models in this study. It understands the complex patterns than the ensemble models.

From the classification report,

The downward motions of both precision and recall are 0.71, and the F-1 score is 0.71, indicating that it understands model reliability in downward movement. The upward movement precision and recall are 0.77 and the F-1 score is 0.77 showing that it was strong in identifying the upward movement.

From the confusion matrix,
True negative is 77 that correctly predicts downward movement.
True positive is 106 which correctly predicts upward movement.
False negative is 32 that predicts downward movements when it was upward.
False positive is 32 that predicts upward movements when it was downward.

Overall multi-layer perceptron suggests that maintaining balanced forecasting performance it minimized all the false negative and false positive predictions.

The MLP reliably detected both upward and downward movements with high accuracy, in comparison to Random Forest, which had difficulty capturing either class, or Naïve Bayes, which was biased towards forecasting upward movements.

FINDINGS

5.1 Model Comparison:

The comparison of the five models shows significant differences in their ability for forecasting the direction of the S&P 500.

	Model	Accuracy	Precision	Recall	F1-score
0	Logistic Regression	0.777328	0.757764	0.884058	0.816054
1	Naive Bayes	0.578947	0.595506	0.768116	0.670886
2	Decision Tree	0.570850	0.635593	0.543478	0.585938
3	Random Forest	0.546559	0.610169	0.521739	0.562500
4	MLP Classifier	0.740891	0.768116	0.768116	0.768116

Fig-5.1

The overall logistic regression performed well with the accuracy of 0.77 and F-1 score with 0.816. It also has a high recall (0.884) shows that it has accurately recorded upward movements, making it a reliable benchmark model for financial forecasting.

The second multi-layer perceptron performed better with accuracy of 0.74 and attained balance performance with precision, recall and F-1 score at 0.77. This shows how well it can handle non-linear relationships, even though its overall performance was not better than that of Logistic Regression.

Naïve Bayes, Decision Tree, and Random Forest had much lower accuracies (55-58%). Although Naïve Bayes had a good recall (0.768), it lacked precision and resulting in more false positives. The poor performance of ensemble tree-based approaches showed their drawbacks in collecting complex financial time-series data.

In conclusion, the results show that, even though more advanced models like MLP perform well, standard statistical methods like logistic regression can still be very effective at predicting market direction.

5.2 Feature Influenced:

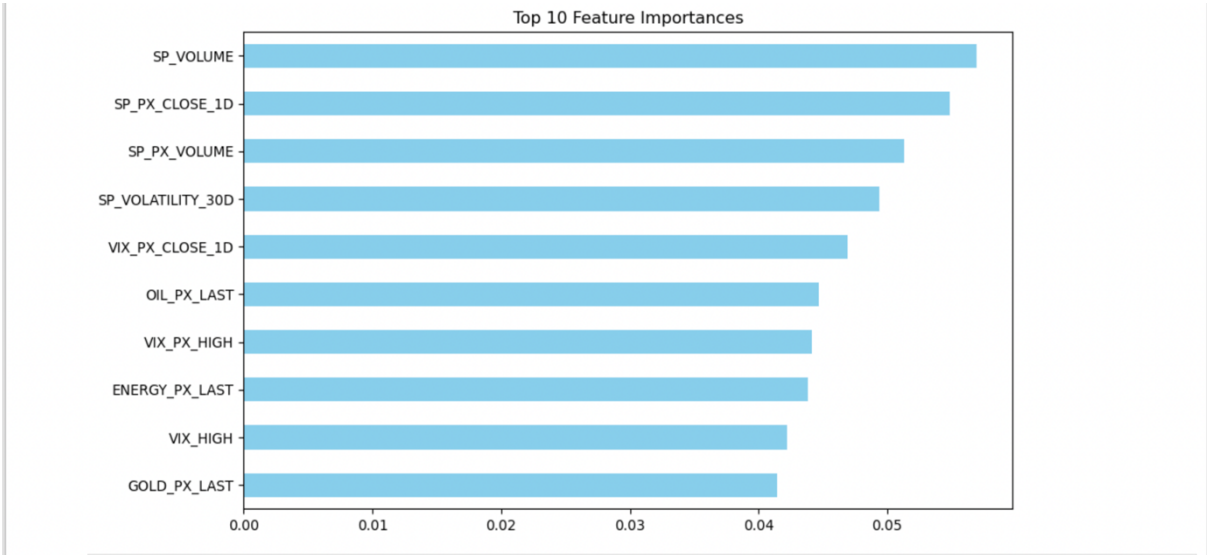

Fig: 5.2

From the above figure it shows the top 10 most important features influencing direction of the S&P 500. The ranking explains the both technical and macroeconomic indicators of the S&P 500.

The most significant factors were SP_VOLUME, 30-day volatility (SP_VOLATILITY_30D), and the closing price of the previous day (SP_PX_CLOSE_1D). These results show the vital role of short-term movement and volatility dynamics for forecasting daily index fluctuations.

Also, volatility indices (VIX) made an important impact, especially indicators that reflect changes in investor sentiment and uncertainty, including VIX_PX_CLOSE_1D and VIX_PX_HIGH. The price of gold and oil

showed up among external factors as important predictors, highlighting how global commodity markets affect risk perceptions and equity performance.

Whereas long term volatility measures SP_VOLATILITY_90D, VIX_VOLATILITY_90D and daily price movements like SP_HIGH, SP_LOW, SP_PX_HIGH, SP_PX_LOW have less impact compared to short term measures and macroeconomic indicators.

Overall, the ranking shows that short-term technical indicators are the most effective in forecasting the direction of the S&P 500

5.3 Probability Interpretation:

Forecasting next day price using logistic regression.

```
print("Next Day - Probability Up:", probs[0][1], "Probability Down:", probs[0][0])
```

```
Next Day - Probability Up: 0.7608106065035632 Probability Down: 0.23918939349643675
```

fig: 5.3

Unlike ensemble models, neural models, and naïve bayes that are difficult to predict, logistic regression provides probabilistic interpretations that are simple to understand and reliable.

In the study it reveals a probability that indicates a significant 76% upward movement and a 23% risk of a downward price movement the next day. This is important in financial markets for investors to make better decisions since market decisions might be based on probability.

Conclusion

The main aim of this study was to forecast the direction of S&P 500 using machine learning approaches. In this study we used both classical models such as logistic regression and naïve bayes, advanced model like decision trees, random forests and multi-layer perceptron. We applied these models on the dataset taken from the 2020- 2025 with the features included of price, volume, volatility, gold, energy, oil, financial and volatility indices.

From the analysis in our study logistic regression with the highest accuracy of 77.7% shows as the strongest and reliable model compared to other models that are performed. The multi-layer perceptron with 74.1% also performed better by capturing the complex patterns from the dataset. The other models such as naïve bayes, random forest and decision tress performed less effective and unable to capture the financial time series data due to its noisy and non-linear patterns.

The most important features, according to feature importance analysis, were short-term S&P 500 indicators such volume, previous-day price, and 30-day volatility. Also external variables such as oil, gold, and sector indices also had a substantial impact, strengthening equity markets interconnection with global economic indicators.

In brief, this dissertation suggests that machine learning models can offer significant understanding into the short-term trend of the market, even though forecasting the S&P 500 is still intrinsically unpredictable. In particular, logistic regression is a strong and understandable model that achieves a balance between practical use and predicted accuracy. This study shows how data-driven approaches can enhance standard financial analysis by combining technical and macroeconomic indicators, providing valuable information for analysts, investors, and researchers.

References

- Zhang and Zhou- **arXiv:2412.11462**
- Suraj Sakaira- https://bura.brunel.ac.uk/bitstream/2438/30054/1/FulltextThesis.pdf
- Eugene Fama's **1970 article**, *"Efficient Capital Markets: A Review of Theory and Empirical Work"- https://people.hec.edu/rosu/wp-content/uploads/sites/43/2023/09/Fama-Efficient-capital-markets-1970.pdf*
- Bakary Bah- *https://lup.lub.lu.se/student-papers/search/publication/9137039*
- *Stock Market Prediction Using Machine Learning and Deep Learning Techniques: https://www.mdpi.com/2673-9909/5/3/76*
- HSBS- https://www.hsbc.co.uk/investments/what-is-the-sp-500/
- Gokmenoglu- https://scholar.google.com/scholar_lookup?title=The%20interactions%20among%20gold%2C%20oil%2C%20and%20stock%20market%3A%20Evidence%20from%20SP500&publication_year=2015&author=K.K.%20Gokmenoglu&author=N.%20Fazlollahi
- Givonni- https://www.sciencedirect.com/science/article/pii/S0169207023000729#sec1
- https://armgpublishing.com/wp-content/uploads/2024/01/FMIR_4_2023_10.pdf
- Alhnaity,- http://bura.brunel.ac.uk/handle/2438/13652
- Rish, I. et al. (2001). An empirical study of the naive bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence, volume 3, pages 41–46.
- Ahas et al- https://www.researchgate.net/publication/359855764_Artificial_Intelligence_and_Machine_Learning_in_Finance_A_Bibliometric_Review

- Patel et al- https://www.sciencedirect.com/science/article/pii/S0957417414004473?via%3Dihub
- Ghosh- https://www.researchgate.net/publication/361091690_A_hybrid_approach_to_forecasting_futures_prices_with_simultaneous_consideration_of_optimality_in_ensemble_feature_selection_and_advanced_artificial_intelligence
- Lv et al- https://www.researchgate.net/publication/357941890_Stock_Index_Prediction_Based_on_Time_Series_Decomposition_and_Hybrid_Model
- Myles et al- https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/epdf/10.1002/cem.873
- Alsubaie et al. 2019 - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8861031&utm_source=sciencedirect_contenthosting&getft_integrator=sciencedirect_contenthosting&tag=1
- Ghaith Abdulsattar A.Jabbar Alkubaisi - https://www.ccsenet.org/journal/index.php/ijsp/article/view/0/41680
- Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5-32
- Baker, S. R., Bloom, N., & Davis, S. J. (2016). Measuring economic policy uncertainty- https://ideas.repec.org/a/oup/qjecon/v131y2016i4p1593-1636..html