MSc Data Analytics, Dissertation MTHM038, 2023/24

PepsiCo Stock Price Prediction Using Time Series and Machine Learning Models

An Analytical Investigation

Arjun Bayadegere Prabhanna, ID 230850895

Supervisor: Prof. Alex Shestopaloff

A thesis presented for the degree of
Master of Science in *Data Analytics*

School of Mathematical Sciences
Queen Mary University of London

Declaration of original work

This declaration is made on September 1, 2024.

Student's Declaration: I Arjun Bayadegere Prabhanna hereby declare that the work in this thesis is my original work. I have not copied from any other students' work, work of mine submitted elsewhere, or from any other sources except where due reference or acknowledgement is made explicitly in the text. Furthermore, no part of this dissertation has been written for me by another person, by generative artificial intelligence (AI), or by AI-assisted technologies.

Referenced text has been flagged by:

1. Using italic fonts, **and**
2. using quotation marks "...", **and**
3. explicitly mentioning the source in the text.

Dedication

I dedicate this to future investors and analysts who want to learn and unlock the power of data knowledge while also predicting the complexities of future markets. I would also like to recognise the great value of publicly available data given by Yahoo Finance, which enables experts and analysts around the world to study and innovate in the field of stock market prediction.

Acknowledgement

I would like to acknowledge my supervisor, Dr. Alex Shestopaloff, for his essential guidance and insightful inputs throughout the production of this dissertation. His encouragement and persistent support contributed to forming this research. I am really grateful for the time and effort he put into mentoring me, as well as the tremendous impact his counsel had on my work.

Abstract

This dissertation conducts a time series analysis of PepsiCo stock prices using historical data from Yahoo Finance in order to estimate future trends. The study focused on the closing prices of PepsiCo shares listed on the NASDAQ. Patterns, trends, and volatility in the data were detected using preprocessing techniques such as sorting, transformation, and exploratory data analysis (EDA). A number of time series forecasting algorithms were used, as well as advanced machine learning methods. The study also examined how ensemble techniques could improve forecast accuracy. The findings are useful for investors, financial analysts, and anyone who wants to contribute to the larger field of predictive modelling in finance.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

PepsiCo is one of the world's most well-known multinational firms, operating in the food, snack, and beverage industries. According to yahoo finance, it is stated that PepsiCo, Inc. manufactures, markets, distributes, and sells a variety of beverages and convenience foods globally. The company operates in seven segments: Frito-Lay North America; Quaker Foods North America; PepsiCo Beverages North America; Latin America; Europe; Africa, the Middle East, and South Asia; and the Asia Pacific, Australia, New Zealand, and China regions. It offers dips, cheese-flavored snacks, and spreads, as well as corn, potato, and tortilla chips; cereals, rice, pasta, mixes and syrups, granola bars, grits, oatmeal, rice cakes, and side dishes; beverage concentrates, fountain syrups, and finished goods; ready-to-drink tea, coffee, and juices; dairy products; sparkling water makers and related products; and alcoholic beverages under the Hard MTN Dew brand.

According to yahoo finance and [5], the company offers its products primarily under the Lay's, Doritos, Fritos, Tostitos, BaiCaoWei, Cheetos, Cap'n Crunch, Pearl Milling Company, Gatorade, Pepsi-Cola, Mountain Dew, Quaker, Rice-A-Roni, Emperador, Diet Mountain Dew, Diet Pepsi, Gatorade Zero, Crush, Propel, Dr Pepper, Schweppes, Marias Gamesa and much more. It provides services to wholesale and other distributors, foodservice customers, grocery stores, drug stores, convenience stores, discount/dollar stores, mass merchandisers, membership stores, hard discounters, e-commerce retailers, and approved independent bottlers, within others, through a network of direct-store-delivery, customer warehouse, and distributor networks, as well as directly to consumers via e-commerce platforms and retailers. This massive corporation was formed in 1898 and is headquartered in New York.

Futhermore, PepsiCo trades on the NASDAQ stock exchange under the ticker symbol PEP. The NASDAQ is an American stock market recognised for its huge number of technology and growth-oriented companies. It is also one of the world's largest stock exchanges. PepsiCo's listing on NASDAQ emphasises its status as a key player in the stock market, with significant market capitalisation and influence on market movements. PepsiCo's stock performance is actively monitored by investors, analysts, and financial institutions. Having said that, analysing PepsiCo's stock price is critical because it reflects the market's opinion of the companies financial health and future prospects, which includes investor mood, market confidence, and growth projections. Income reports, product beginnings, changes in consumer demand, and global economic conditions all have an impact on PepsiCo's stock price and provide information about investor state of mind.

The wider market surroundings, which includes interest rates, inflation, and geopolitical events, also influences PepsiCo's stock price, with changes in commodity prices affecting profit margins. Regular stock price analysis illustrates how PepsiCo responds to internal and external forces, providing useful insights into the company's performance. Understanding these dynamics is vital for investors to make informed decisions about purchasing, holding, or selling shares, eventually contributing in portfolio optimisation and reaching financial goals.

1.1.1 Objectives

The core objective of this dissertation is to forecast PepsiCo's stock price using multiple time series and machine learning models, as well as to assess and compare their performance. This includes building forecasting models such as ARIMA, Exponential Smoothing (ETS), and advanced machine learning approaches to estimate PepsiCo's future stock prices. The dissertation will evaluate the accuracy and efficacy of these models by comparing their predicting performance to actual stock price data, and will select the most effective model based on parameters such as prediction error, accuracy, and robustness. The intent of implementing time series forecasting is to predict future price movements using historical data and statistical models to detect patterns and trends. Accurate forecasting is critical for building successful investment strategies, controlling risks, and optimising portfolios, which guide PepsiCo's investment decisions and strategic planning. Further, the study intends to provide strategic insights and suggestions based on the forecasting results, allowing marketers, investors, and financial analysts to make more educated judgements about PepsiCo's stock.

1.1.2 Importance of Returns and Volatility

Returns and volatility are key principles in financial forecasting, with a substantial impact on the accuracy and dependability of stock price projections.

Returns are the gains or losses realised on an investment over a given time period, usually stated as a percentage of the initial investment. They provide insights into a stock's performance and are critical for determining its profitability. Understanding returns allows investors to assess how well an asset has performed historically and can inform future investing decisions.

Volatility, on the other hand, quantifies the degree to which a stock's price varies or fluctuates over time. High volatility denotes huge swings in stock prices, whereas low volatility implies more consistent price movements. Volatility is critical when determining the risk associated with a stock. Understanding both returns and volatility allows investors to better anticipate the possible risks and rewards of their investments, make informed decisions, and build strategies that are in line with their risk tolerance and financial objectives. In the context of stock price forecasting, including these principles into models allows for a more thorough examination of market behaviour, resulting in more accurate and dependable predictions.

Let us understand this by a simple example taken from[4],
The graph depicts two investment funds, Fund A and Fund B, which each earned an 8% return over five years. Despite having the same yield, each funds' prices moved differently.

Fund A experienced a lot of ups and downs, indicating significant volatility. Investors in this fund experienced greater volatility in the value of their investments.

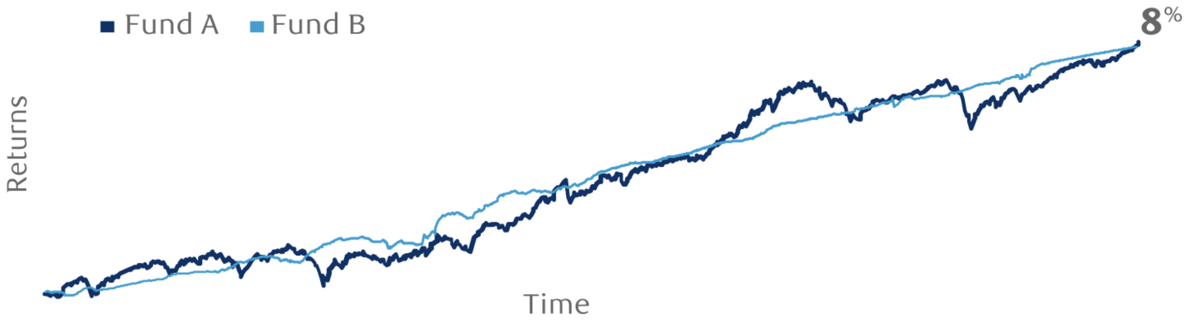

Figure 1.1: Return/Volatility over Time

Fund B had a more stable performance, indicating lower volatility. Its price fluctuated less substantially.

The key take away is that while returns are important, you should also consider how much price movement you are okay with. If you can take more ups and downs, consider Fund A. If you desire less risk and more consistency, Fund B may be a better option. Ultimately, Investors need to balance their expected returns with the anticipated volatility in their portfolio, keeping in mind their comfort level with risk, time horizon and long-term goals.

1.1.3 Close Price and Adjusted Close Price

Close price is one of the most widely utilised data points in financial research and forecasting since it represents the stock's most current market value at the end of the trading day. On the other hand, the ***adjusted close price*** is the stock's closing value changed to adjust for changes caused by business events such as stock splits and dividend payments, ensuring that historical price data is comparable and consistent.

For a visual comparison, refer to the graph depicting these price variations for PepsiCo.

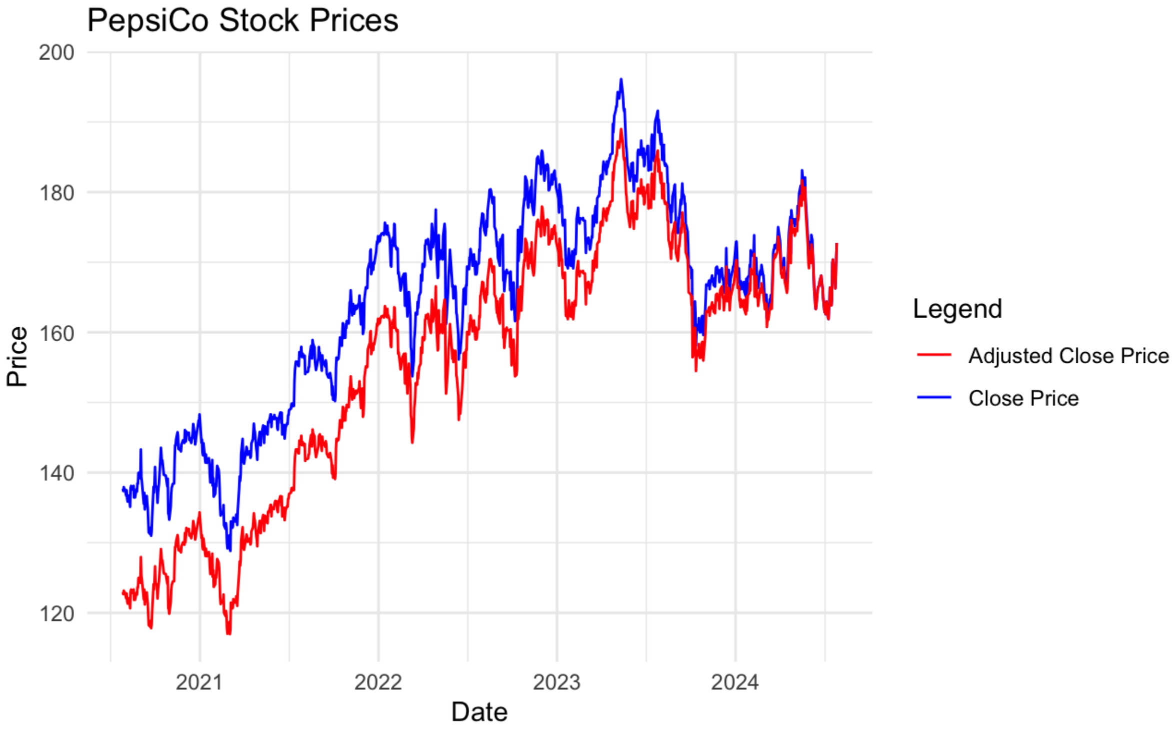

Figure 1.2: Price Comparison

Both the closing price and the adjusted close price serve unique but complimentary functions in financial research. The close price is necessary for understanding daily market circumstances and making short-term investment decisions, but the adjusted close price is critical for evaluating long-term performance and consistency.

Investors and analysts should use both indicators to get a complete picture of a stock's value, performance trends, and historical accuracy. This technique improves decision-making, strengthens financial models, and promotes successful portfolio management.

Chapter 2

Literature Review

2.1 Theoretical Background

2.1.1 Returns

The return in percentage indicates how well or poorly an asset has performed. These are taken from[1].

For example, if the value at the beginning of the period is S0 and at the end is S1, then the return is:

$$R = \frac{S_1 - S_0}{S_0} = \frac{S_1}{S_0} - 1 \tag{2.1}$$

Similarly, if we have a list of asset values over time say S0,S1,...Sm, then we likewise get a sequence of returns:

$$R_1 = \frac{S_1 - S_0}{S_0}, \quad R_2 = \frac{S_2 - S_1}{S_1}, \quad \ldots, \quad R_m = \frac{S_m - S_{m-1}}{S_{m-1}} \tag{2.2}$$

Please be notified that, the length of returns is one less than the size of the length of asset values.

Sometimes, investors do consider ***Log-returns*** than normal returns and we will see why below,

$$LR = \ln\left(\frac{S_1}{S_0}\right) \tag{2.3}$$

Log returns are neatly tallied up, that is: The log-return from 0 to 2 equals the aggregate of the log-returns from 0 to 1 and 1 to 2,

$$\ln\left(\frac{S_1}{S_0}\right) + \ln\left(\frac{S_2}{S_1}\right) = \ln(S_1) - \ln(S_0) + \ln(S_2) - \ln(S_1) \tag{2.4}$$

$$= \ln(S_2) - \ln(S_0) \tag{2.5}$$

$$= \ln\left(\frac{S_2}{S_0}\right) \tag{2.6}$$

The above form does not hold good in case or plain or normal returns, however both tend to be quite close.

According to the Taylor Expansion formula,

$$\ln(x) = (x-1) - \frac{1}{2}(x-1)^2 + \frac{1}{3}(x-1)^3 - \cdots \tag{2.7}$$

indicating that if $x - 1$ is tiny, then $\ln(x)$ is close to $x - 1$, as then $(x-1)^2$, $(x-1)^3$, and so on will be even smaller.

The log-return is given by:

$$\ln\left(\frac{S_1}{S_0}\right) = \frac{S_1 - S_0}{S_0} - \frac{1}{2}\left(\frac{S_1 - S_0}{S_0}\right)^2 + \frac{1}{3}\left(\frac{S_1 - S_0}{S_0}\right)^3 - \cdots \tag{2.8}$$

In the above equations:

- $\ln\left(\frac{S_1}{S_0}\right)$ represents the log-return.
- $\frac{S_1 - S_0}{S_0}$ represents the return.

Here, the log-return is approximated by the return, as higher-order terms like $\left(\frac{S_1-S_0}{S_0}\right)^2$, $\left(\frac{S_1-S_0}{S_0}\right)^3$, etc. become negligible when S_1 is close to S_0.

2.1.2 Volatility

Volatility, as previously described, is a statistical measure of the spread of returns for a certain securities or market index. Increased volatility indicates higher risk and returns. Alternatively, lesser volatility indicates more stable returns with lower risk.

Few types of volatility are:

- **Historical volatility**
- **Implied volatility**

- **Variance swap or volswap volatility**

Historical volatility: Historical volatility, also known as realised volatility, is the annualised standard deviation of a prior return.

For instance, for daily asset values like S0, S1, S2...Sm we take the average of returns,

$$\bar{R} = \frac{1}{m-1} \sum_{i=1}^{m-1} R_i \tag{2.9}$$

Then, we proceed onto find the volatility using the given formula:

$$\text{Volatility} = \sqrt{\frac{365}{m-1} \sum_{i=1}^{m} \left(R_i - \bar{R}\right)^2} \tag{2.10}$$

where,

- ***"m"*** represents the number of past days to consider.
- The coefficient 365 in the volatility calculation is called the ***"annualization factor"***. It is used to adjust the daily volatility to an annual scale. In practice, this factor may be replaced by 252 or any other number to account for the number of trading days in a year, considering that exchanges are closed on weekends and holidays.

Implied volatility: Implied volatility is a metric generated from option prices that indicates the market's expectations for future volatility. It is calculated using an options pricing model, such as the Black-Scholes model.

The formula is given by [3], and it is a comprehensive textbook widely used in finance courses. It provides detailed explanations and derivations of the Black-Scholes formula, including both the call and put options formulas.)

$$C = S_0\Phi(d_1) - Ke^{-rT}\Phi(d_2) \tag{2.11}$$

$$P = Ke^{-rT}\Phi(-d_2) - S_0\Phi(-d_1) \tag{2.12}$$

$$d_1 = \frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \tag{2.13}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{2.14}$$

- C = Price of the call option
- P = Price of the put option
- S_0 = Current price of the underlying asset
- K = Strike price of the option
- r = Risk-free interest rate
- T = Time to maturity of the option (in years)
- Φ = Cumulative distribution function of the standard normal distribution
- σ = Volatility of the underlying asset

The implied volatility is calculated by entering the market price of the option into ***Black-Scholes model***, solving for the volatility parameter.

Investors and traders frequently use implied volatility to determine the relative value of options. Options with higher implied volatility, may be deemed more expensive, whilst those with lower implied volatility could be viewed as affordable. Conversely, market conditions, investor sentiment, and option supply and demand dynamics can all have an impact on implied volatility.

Variance swap or Volswap volatility: A volatility swap is a type of derivative contract in which two parties trade the difference between the realised volatility of an underlying asset and a predetermined strike volatility. The realised volatility is derived using the asset's previous price data over a predetermined time period, whereas the strike volatility is determined at the start of the contract.

The payoff of a volatility swap at a specific expiration date is determined as follows:

$$\text{Payoff} = \frac{\sqrt{\frac{1}{m-1}\sum_{i=1}^{m-1}\left(R_i - \bar{R}\right)^2}}{\sqrt{365}} - \text{StrikeVol} \tag{2.15}$$

where,

- $\sqrt{\frac{1}{m-1}\sum_{i=1}^{m-1}\left(R_i - \bar{R}\right)^2}$ represents the annualized realized volatility, where R_i denotes the log returns, $\bar{R}$ is the average return, and m is the number of days used to compute the volatility.
- StrikeVol is the pre-agreed strike volatility.

- $\sqrt{365}$ is the annualization factor used to scale the realized volatility on an annual basis.

Concluding, the difference between realised volatility (annualised) and strike volatility determines the swap's reward. If the realised volatility exceeds the strike, the reward is positive; if it is less, the payoff is negative.

2.1.3 Close Price and Adjusted Close Price

Close Price: The closing price of a stock is the last price at which it trades during a typical trading session on a given trading day. This price is important because it serves as a baseline for calculating returns and provides an indicator of the stock's performance over a trading session.

When analysing stock performance, we frequently calculate returns based on the closing price. The simple return between two periods can be calculated as follows:

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{2.16}$$

where,

- R_t = Return at time t
- P_t = Close price at time t
- P_{t-1} = Close price at time $t-1$

Adjusted Close Price: The adjusted close price accounts for factors that affect the stock's price but have no bearing on its performance. These include stock splits, dividends, and rights offers. The adjusted close price changes historical closing prices to reflect business actions, ensuring that the price series is consistent over time.

Formula to calculate the Adjust Close Price is as given below:

$$P_{adj} = \frac{P_{close}}{S} \tag{2.17}$$

where,

- P_{adj} = Adjusted close price
- P_{close} = Raw close price
- S = Split or dividend adjustment factor

It is important to know that, the split or dividend adjustment factor in the adjusted close price corrects for stock splits and dividends by changing historical prices to reflect these changes, ensuring price series consistency. For stock splits, the raw close price is divided by the split ratio and for dividends, the dividend amount is deducted from the raw close price.

2.1.4 Modeling Equations and Theories

AutoRegressive Moving Average: The ARMA (AutoRegressive Moving Average) model is a fundamental time series forecasting technique that incorporates two main components, ***auto-regressive (AR)*** and ***moving average (MA)***.

The AutoRegressive (AR) component of the ARMA model captures the relationship between the time series present value and prior values. It is predicated on the assumption that the series present value can be described by a linear combination of its previous values.

The AR component is represented by the equation below and derived from [10]:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + z_t \tag{2.18}$$

where,

- y_t is the value of the time series at time t,
- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients,
- p is the order of the autoregressive process,
- z_t is the white noise error term at time t, with $\{z_t\}$ being white noise, i.e., $\{z_t\} \sim WN(0, \sigma^2)$, and z_t is uncorrelated with y_s for each $s < t$.

On the other hand, the Moving Average (MA) component represents the link between the current value of the time series and the previous error terms. It measures the effect of previous noise on the current value.

The MA component is represented as follows:

$$y_t = z_t + \theta_1 z_{t-1} + \theta_2 z_{t-2} + \cdots + \theta_q z_{t-q} \tag{2.19}$$

where,

- y_t is the value of the time series at time t,

- z_t is the white noise error term at time t,
- $\theta_1, \theta_2, \ldots, \theta_q$ are the moving average coefficients,
- q is the order of the moving average process,
- z_t is white noise with $z_t \sim WN(0, \sigma^2)$.

Finally, the ARMA model incorporates both AR and MA components to account for autoregressive and moving average effects.

The ARMA equation is as follows:

$$y_t - \phi_1 y_{t-1} - \cdots - \phi_p y_{t-p} = z_t + \theta_1 z_{t-1} + \cdots + \theta_q z_{t-q} \tag{2.20}$$

where,

- y_t is the value of the time series at time t,
- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients,
- $\theta_1, \theta_2, \ldots, \theta_q$ are the moving average coefficients,
- p is the order of the autoregressive process,
- q is the order of the moving average process,
- z_t is white noise with $z_t \sim WN(0, \sigma^2)$.

ARIMA: The AutoRegressive Integrated Moving Average (ARIMA) model is a popular statistical approach for analysing and forecasting time series data. It builds on the ARMA model by introducing differencing to handle non-stationary data.

- **Integrated (I) Component:** This component involves differencing the time series to make it stationary. The order of differencing is denoted by d.

Differencing is a technique that converts a non-stationary time series into a stationary one. A stationary time series has a constant mean and variance throughout time, with autocorrelation determined solely by the lag between observations. Non-stationarity is frequently defined by trends or seasonality. The differencing procedure subtracts the current value from the previous value to eliminate patterns and seasonality. The order of differencing, represented by d, indicates how often the differencing operation is executed.

From [10], ARIMA(p, d, q) represents the ARIMA model in its generic form. A process $\{y_t\}$ is said to follow an ARIMA(p, d, q) model when:

$$\nabla^d y_t = (1-B)^d y_t \text{ is } \mathrm{ARMA}(p,q). \tag{2.21}$$

We can then write the ARIMA(p,d,q) model as:

$$(1-\phi_1 B-\cdots-\phi_p B^p)(1-B)^d y_t = (1+\theta_1 B+\cdots+\theta_q B^q)z_t \tag{2.22}$$

where, $\{z_t\} \sim \mathrm{WN}(0,\sigma^2)$ is white noise, B is the backshift operator, and d is a nonnegative integer representing the differencing order.

Now, to fit an ARIMA model to a time series, we have to follow these general steps:

- **Identification:** Use diagnostic tools like ACF and PACF charts to find the proper values for p, d, and q.
- **Diagnostic Checking:** Check the residuals of the fitted model to ensure that they resemble white noise.

SARIMA: The Seasonal ARIMA (SARIMA) model extends the basic ARIMA model to accommodate time series data that exhibit seasonal trends. The SARIMA model, denoted as SARIMA$(p,d,q)(P,D,Q)_m$, combines both non-seasonal and seasonal components to efficiently capture complex seasonal behaviors in data.

The SARIMA model consists of the following components:

- **AR(p):** Autoregressive part of order p, which captures the relationship between the current value and its previous p values.
- **MA(q):** Moving average part of order q, which models the relationship between the current value and past error terms.
- **I(d):** Differencing of order d, which is used to make the time series stationary by removing trends.
- **AR$_m$(P):** Seasonal autoregressive part of order P, which captures the seasonal dependencies with period m (e.g., yearly seasonality with $m = 12$ for monthly data).
- **MA$_m$(Q):** Seasonal moving average part of order Q, which models the seasonal errors.
- **I$_m$(D):** Seasonal differencing of order D, which removes seasonal trends by differencing the series at the seasonal period m.

- **m:** The period of the seasonal pattern, indicating how often the seasonality repeats. For example, $m = 12$ for monthly data reflecting annual seasonality.

SARIMA models are very useful for dealing with time series data that has significant seasonal impacts. They are well-suited for scenarios in which the data exhibits regular seasonal fluctuations (for example, monthly sales data, temperature readings), traditional ARIMA models fail to capture these seasonal patterns effectively, or forecasts must account for both non-seasonal and seasonal variations.

Exponential Smoothing State Space Model (ETS): The ETS model, which stands for Error, Trend, and Seasonality, is a time series forecasting approach that employs exponential smoothing techniques. The model is designed to accommodate all three components.

- **Error (E)**: The error term represents the randomness of the data and might be additive (A) or multiplicative (M).
- **Trend (T)**: The component represents the series long-term evolution, which can be linear, exponential, or damped.
- **Seasonality (S)**: The seasonality component identifies repeating patterns at certain intervals, which can be additive or multiplicative.

The ETS model is particularly useful when the data reveals clear trends and seasonal patterns. Its capacity to handle several types of seasonality (additive and multiplicative) and trends (linear, exponential, or damped) makes it a useful forecasting tool. Furthermore, the ETS model can automatically determine the best combination of error, trend, and seasonality components, making it a dependable solution for a diverse set of time series data. The ETS approach is particularly successful for time series data with distinct seasonal patterns and trends that require adaptability over time. To ensure essential forecast accuracy, we will have to optimise both additive and multiplicative components of the model.

From [9], the ETS model can be formulated using a forecast equation and three smoothing equations that are used to update the model's internal state. The model is described as follows:

$$\hat{y}_{t+h|t} = (l_t + \phi^h b_t)s_{t+h-m(k+1)} \tag{2.23}$$

where,

- l_t is the level at time t,

- b_t is the trend at time t,
- s_t is the seasonal component at time t,
- ϕ is the damping factor,
- m is the number of seasonal periods, and
- h is the forecast horizon.

Smoothing Equations:

$$\begin{aligned}
\text{Level: } l_t &= \alpha \left(\frac{y_t}{s_{t-m}} \right) + (1-\alpha)(l_{t-1} + \phi b_{t-1}) \\
\text{Trend: } b_t &= \beta \left(l_t - l_{t-1} \right) + (1-\beta)\phi b_{t-1} \\
\text{Seasonal: } s_t &= \gamma \left(\frac{y_t}{l_t} \right) + (1-\gamma) s_{t-m}
\end{aligned}$$

where,

- α is the smoothing parameter for the level,
- β is the smoothing parameter for the trend,
- γ is the smoothing parameter for the seasonality.

Error Models: The error models specify how the true values y_t are updated:

Additive Error Model:

$$y_t = \hat{y}_{t|t-1} + \varepsilon_t$$

Multiplicative Error Model:

$$y_t = \hat{y}_{t|t-1}(1 + \varepsilon_t)$$

where, ε_t is the error term.

State Space Equations: Using these error models, the state space equations for the ETS models are:

$$\begin{aligned}
y_t &= (l_{t-1} + \phi b_{t-1}) s_{t-m} + \varepsilon_t && \text{(Additive Error)} \\
y_t &= (l_{t-1} + \phi b_{t-1}) s_{t-m}(1 + \varepsilon_t) && \text{(Multiplicative Error)}
\end{aligned}$$

Smoothing Parameters: The parameters α, β, and γ, as well as the initial states l_{-1}, b_{-1}, and $s_{-1}, \ldots, s_{-m}$, are estimated by maximizing the log likelihood of the model.

Overall, the ETS model is a great option for time series data that necessitates flexible and adaptive modelling of error, trend, and seasonal components.

TBATS: The TBATS model, which stands for Trigonometric, Box-Cox transformation, ARMA errors, Trend, and Seasonal components, is a highly effective time series forecasting tool. It excels at managing complex seasonal patterns, multiple seasonal cycles, and non-linear trends that are frequent in real-world data.

The TBATS model is particularly useful when dealing with time series data that exhibits multiple seasonalities, long seasonal durations, complex seasonal and trend components, and non-linear trends.

The TBATS model combines various components to enable versatility in modelling.

- **Trigonometric Seasonal Components:** TBATS use Fourier terms to represent various and complicated seasonal trends.
- **Box-Cox Transformation:** This adjustment reduces variation in the time series, making it easier to model.
- **ARMA Errors:** Autoregressive and Moving Average components are used to detect autocorrelations in the residuals.
- **Trend Components:** TBATS can represent both linear and nonlinear trends, as well as damped trends, which have a decreasing influence with time.
- **Seasonal Components:** The model can handle complex seasonal trends by mixing trigonometric terms and seasonal smoothing.

The combination of these attributes enables TBATS to efficiently represent a wide range of time series, particularly those with seasonal complexity and non-linear behaviour.

2.2 Review of Forecasting Techniques

The theoretical basis on returns, volatility, Close and Adjusted Close Prices gives the fundamental principles required to comprehend financial markets and estimate stock prices. Returns quantify an asset's performance by calculating the percentage change in its value over a given time period. These can be provided as basic or logarithmic results. Simple returns are determined as the difference between the asset's beginning and ending values. However, log returns are frequently preferred in financial analysis due to their mathematical features. Having said that, log returns are a convenient approach to aggregate returns across numerous periods because they sum neatly, unlike simple returns.

Volatility, on the other hand, is an important measure of the fluctuation in asset returns that reflects the asset's level of risk. Historical volatility, also known as realised volatility, is calculated using previous return data and offers information about the asset's risk levels. This metric is frequently annualised to standardise comparisons across historical periods. In contrast, implied volatility is derived from option prices and represents the market's expectation for future volatility. It is very beneficial for traders in assessing market expectations. Volatility swaps provide another perspective by comparing realised volatility to a predetermined strike volatility, so giving a means for trading volatility risk.

Similarly, the close price and adjusted close price are critical for assessing stock performance and volatility. The closing price represents the stock's final trading value each day, and it is an important factor in computing returns and judging daily market conditions. In contrast, the adjusted close price accounts for business events such as stock splits and dividend payments, resulting in a more accurate and consistent historical record for performance analysis. Both prices are necessary for improving forecasting models because they supply the raw data required to compute returns and volatility, which are crucial inputs for time series forecasting techniques.

Ultimately, these principles are vital in financial analysis, having a direct impact on forecasting models and serving as critical inputs for various time series forecasting approaches such as ARIMA, SARIMA, ETS, TBATS, and machine learning methods such as XGBoost and SVM. Understanding these metrics aids in the refinement of forecasting models by adding historical data and market expectations to increase prediction accuracy.

2.3 Comparison Studies

In the realm of financial forecasting, comparative studies are critical in determining the efficacy of various predictive models and methodologies. These studies carefully evaluate and compare the performance of various forecasting approaches to identify which strategies produce the most accurate and trustworthy results.

The major purpose of these studies is to determine the strengths and limitations of various forecasting methodologies when used in financial markets. Researchers can assess the relative effectiveness of traditional statistical models (e.g., ARIMA, Exponential Smoothing) and advanced machine learning approaches (e.g., XGBoost, Support Vector Machines) in predicting financial time series data, such as stock prices or market indices.

Comparison study typically involves:

- **Model Selection:** Selecting a collection of forecasting models to compare. These could include traditional statistical models, machine learning methods, and hybrid approaches.

- **Data Collection:** Using historical financial data to train and test models. This information could include stock prices, trading volumes, and other important financial indicators.

- **Performance Metrics:** Evaluating the models using several performance indicators such as accuracy, mean squared error, and forecast bias. These indicators help to quantify how successfully each model forecasts future market moves.

- **Analysis and Comparison:** Analysing the data to see which models perform best under certain scenarios. This could include evaluating model performance over various time periods or market situations to determine its robustness and reliability.

Overall, comparison studies provide useful information for investors, analysts, and financial institutions by assisting in the selection of the most effective forecasting approaches. Understanding which models have the best predictive performance can help you make better decisions, improve your investment methods, and optimise your portfolio.

Chapter 3

Methodology

3.1 Data Description

The dataset used in this study is made up of historical stock data for PepsiCo, Inc. (PEP) collected from Yahoo Finance via the NASDAQ market. The data spans a four-year period, beginning in July 2020 and ending in July 2024, and covers a wide range of market situations, including optimistic and pessimistic tendencies. This time period is notable because it encompasses a number of global economic events, including the influence of the COVID-19 pandemic on markets, recovery periods, and recent economic worries.

The dataset contains numerous crucial elements that are required for analysing and forecasting stock performance:

- **Date:** This column records the specific trade day for each record. It enables an ordered examination of the stock's performance, allowing anyone to observe trends, detect patterns, and perform time series analysis. The date format remains uniform, allowing for seamless interaction with time series forecasting models.

- **Open:** Represents the first price at which PepsiCo's stock was traded when the market started on a certain day. This figure is critical for evaluating market mood at the start of the trading day and is frequently used alongside the closing price to assess daily performance.

- **High:** The maximum price at which the stock traded during the trading session. This figure is very useful for analysing session volatility and market peaks. It might also signify the peak buying pressure of the trading day.

- **Low:** The stock's lowest trading price for the session. The low price, like the high price, reveals momentary volatility and the market's lowest valuation of the stock that day.

- **Close:** The stock's last trading price as the market closed for the day. The close price is one of the most commonly analysed data points since it shows the most recent market value at the end of a trading session. It is frequently applied to calculate returns and is a critical variable in financial models.

- **Adj Close:** The adjusted close price is an important measure that shows the company's closing price after accounting for business actions such as stock splits, dividends, and rights offers. This modification ensures that historical price data remains stable over time, increasing its reliability for future study. It is especially valuable for investors interested in a stock's total return, which includes retained dividends.

- **Volume:** This shows the total number of shares exchanged during the trading session. Volume is a measure of trading and liquidity. High trade volumes are frequently associated with significant price fluctuations and might serve as a market condition indicator. Analysing volume alongside price movements can provide more insight into the robustness of pricing patterns.

This dataset is crucial for a variety of financial analyses and it enables the analysts to thoroughly assess PepsiCo's stock performance over multiple time periods by examining price and volume trends. The close and adjusted close prices are especially critical for correct return calculations since they account for dividends and company actions. Furthermore, by examining the high, low, and closing prices, analysts may determine the stock's volatility, which is critical for risk management and pricing options. In addition, this data serves as a good foundation for creating time series forecasting models that predict future stock prices based on historical patterns.

3.2 Data Preprocessing

The data preprocessing stage included several essential steps for cleaning and preparing the data for analysis. The dataset, which consisted of PepsiCo stock prices, was initially loaded and processed as follows:

3.2.1 Loading and Sorting Data

The dataset was loaded from a CSV file containing PepsiCo's historical stock values. The `Date` column was changed to a date format, and a new `Year` column was extracted to allow for time-based operations. To align all time-series operations, the data was sequentially sorted using the `Date` column.

3.2.2 Handling Missing Values

After sorting the data, it was examined for missing values in both the training and testing sets. The function `colSums(is.na(data))` counted any `NA` values in the dataset's columns. There were no missing values, therefore the analysis could proceed without requiring imputation or data removal processes.

3.2.3 Splitting Data

Data was separated into two subsets, training (80%) and testing (20%) datasets. To ensure the accuracy of the time-series analysis, this split was performed in the correct order. The splitting index was set at 80% of the total number of records to train the model on past data and test it on future data.

3.2.4 Converting the data into Time Series Format

The `Close` price from the training and testing datasets was converted into a weekly-frequency time series. This conversion is vital for time series analysis since it enables the use of time series models and forecasting approaches.

Overall, these pre-treatment methods guaranteed that the data was clean, structured, and suitable for further analysis, modelling, and forecasting.

3.3 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is an important stage in determining the fundamental patterns, correlations, and behaviours in the data. In this part, we look at PepsiCo's stock returns and volatility, offering insights into how they are distributed and fluctuations over time. To investigate the correlations between major elements, many plots were created. The scatter plot of Volume vs. Closing Price, with a gradient dependent on Volume, shows how trading volume corresponds with closing prices over time. Similarly, the bar plot of Trading Volume Over Time depicts swings in trading activity, which could indicate periods of significant market interest or external events affecting the company.

Time Series Visualization: A time series plot of the Closing Price Over Time was constructed to show the temporal trends in closing prices. This graph aids in detecting long-term trends, seasonal patterns, and large swings in the stock price. To give a more detailed study of price movements, a candlestick chart was also created, which is particularly useful for visualising open, high, low, and closing values within the same plot, as shown below.

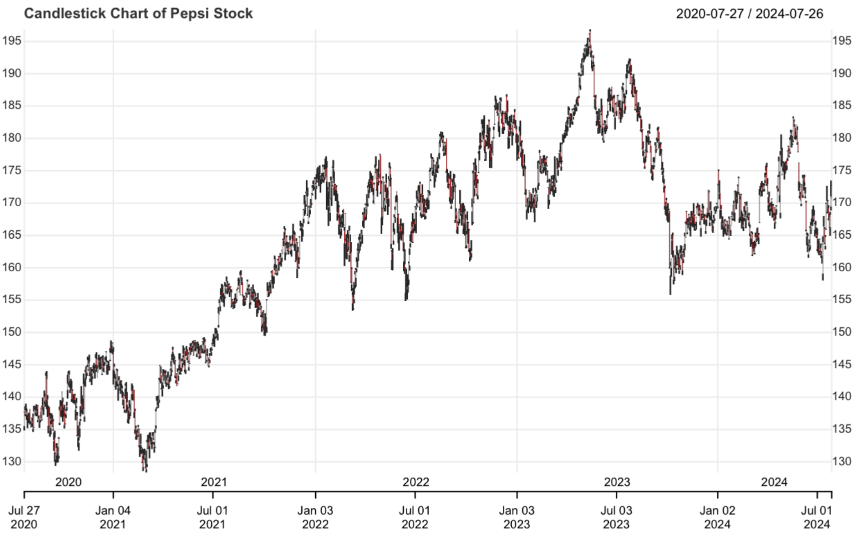

Figure 3.1: Time Series Plot for PepsiCo Stock

Returns Analysis: The percentage change in the closing price from day to day was calculated using daily returns. A time series representation of these daily returns was constructed to help discover periods of considerable volatility and assess the overall risk associated with the stock. Additionally, a histogram of daily returns was generated to analyse their distribution. This histogram shows the frequency and magnitude of positive and negative returns, allowing you to assess the stock's performance and risk, as illustrated below.

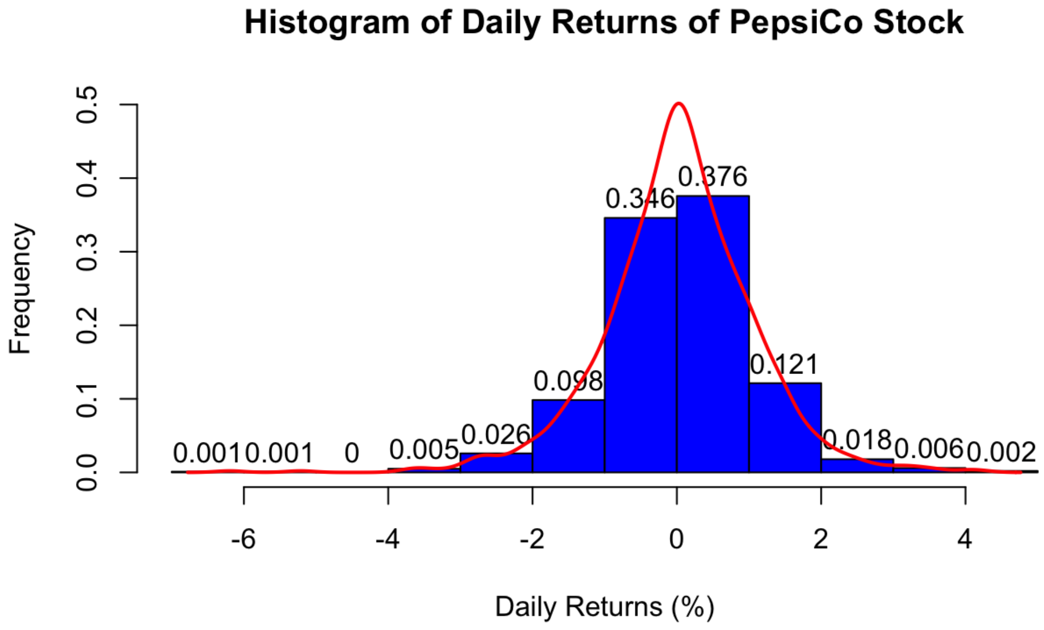

Figure 3.2: Daily Retuns Histogram Plot for PepsiCo Stock

Volatility Analysis: To assess the risk associated with PepsiCo stock, rolling volatility, namely the 30-day rolling standard deviation of returns, was calculated. This was visualised using a time series plot, which aids in understanding how the stock's volatility fluctuates over time and detecting periods of increased risk, as shown below.

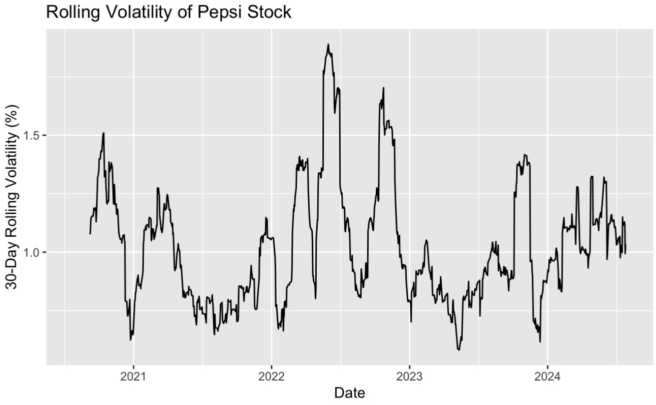

Figure 3.3: Volatility Time Series Plot

Correlation Analysis: A heatmap was used to calculate and visualise the correlation between several numerical features. This heatmap shows the intensity and direction of correlations between variables such as closing price, volume, and returns. Understanding these relationships is critical for model development and feature selection in the subsequent predictive analysis, as demonstrated below.

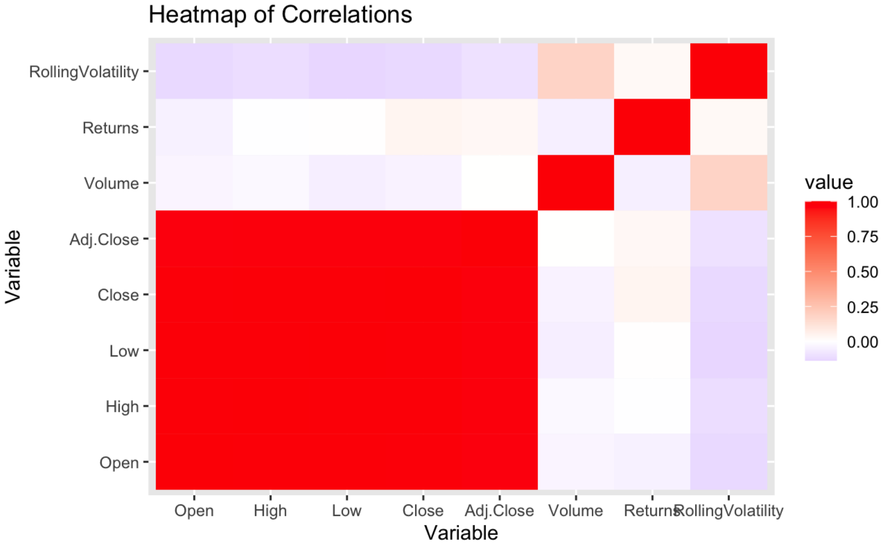

Figure 3.4: Correlation Analysis using Heatmap

In conclusion, the EDA offered a thorough understanding of the behaviour and characteristics of PepsiCo stock data, establishing the framework for the upcoming predictive modelling and forecasting tasks.

3.4 Time Series Analysis

The PepsiCo Stock Time Series Plot was decomposed in detail to better understand the underlying trends.

3.4.1 Decomposition of Time Series

Decomposing a time series yields three components: trend, seasonality, and residual (or random) noise. This decomposition helps to find underlying trends and simplifies the series for modelling. The analysis was carried out utilising PepsiCo stock price data, which was collected weekly over a four-year period. The decomposition was carried out with the `decompose` function, and the visualisation revealed clear patterns in the data.

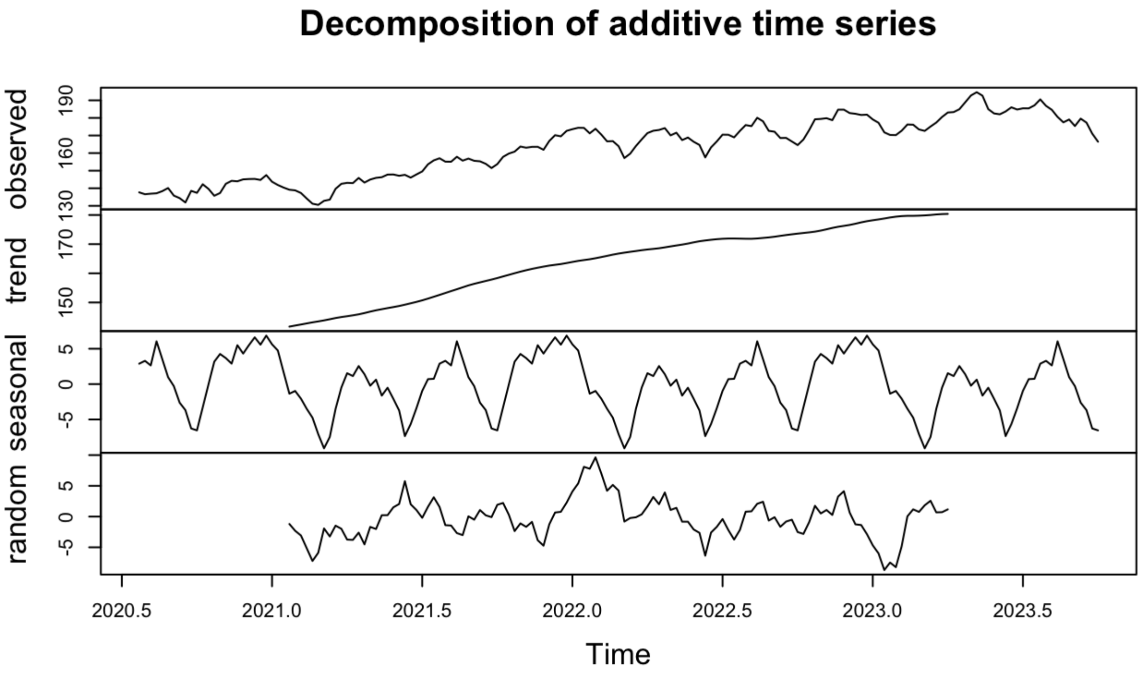

Figure 3.5: Decomposed Time Series Plot for PepsiCo Stock

The decomposition indicated that the time series followed a definite trend, with seasonal oscillations and cyclical patterns. This means that the stock price not only follows a long-term trend, but also has predictable seasonal variations.

3.4.2 Residual Analysis

Filtering the time series components, the residuals were evaluated to see if they resembled white noise. The residuals were plotted and examined as seen in figure 3.6, and it was observed that they did not resemble white noise, revealing the significant presence of seasonality and abnormalities in the time series plot. Alternatively, significant lags in the ACF plot further strengthens the seasonality argument.

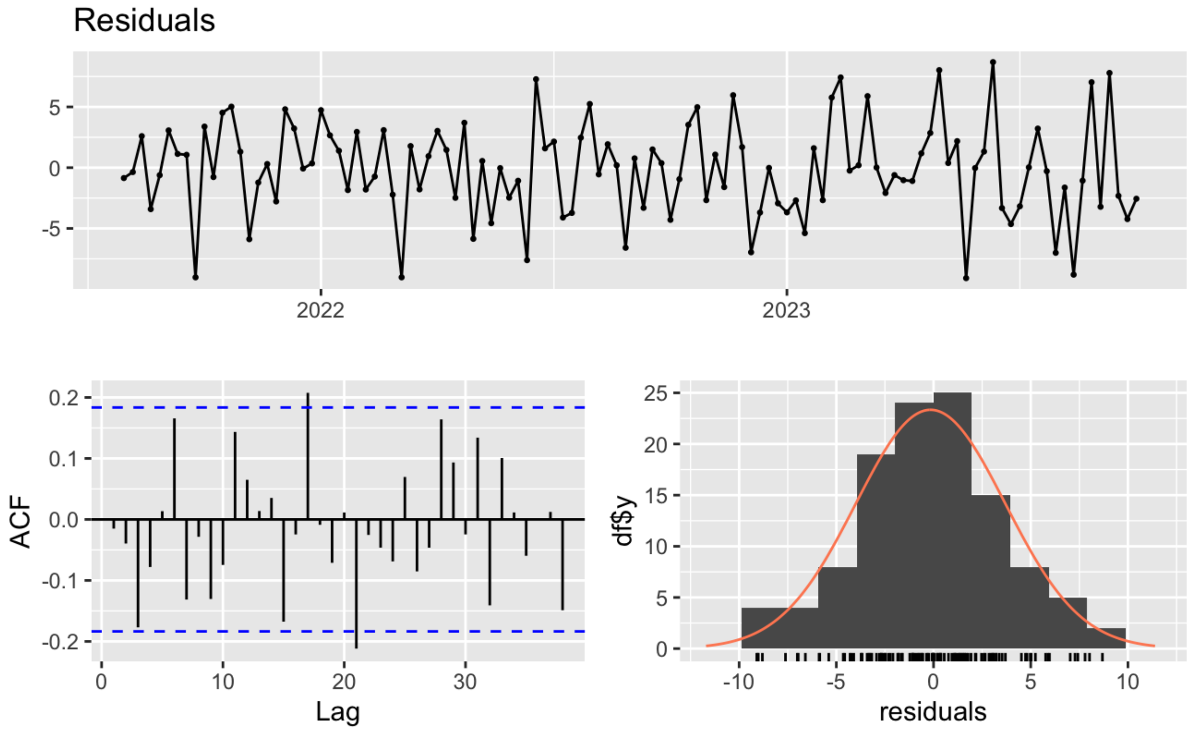

Figure 3.6: Residuals Plot for PepsiCo Stock

The `check residual` function was used to illustrate the corresponding ACF, time series residual analysis, and density chart.

3.4.3 Variance Stabilization

Variance stabilisation is a strategy used to address the issue of heteroscedasticity, which occurs when the variability of a time series changes with its level, making statistical analysis and modelling difficult. In this project, the ***Box-Cox transformation*** was used to stabilise the variance of the time series data.

The Box-Cox transformation is a powerful transformation that can solve non-constant variance by changing the data into a more uniform distribution across levels. Finally, this method not only simplifies the use of statistical models that assume constant variance, but it also improves the accuracy and interpretability of the results by making the data more homoscedastic.

Overall, a rigorous analysis of the PepsiCo stock time series revealed different trends, seasonal patterns, and cyclical behaviours across a four-year period. The data clearly showed a long-term rising trend, accompanied by frequent seasonal oscillations. Residual analysis revealed considerable seasonality and non-white noise characteristics, implying that the residuals were systematic rather than randomly distributed. To address variability, the Box-Cox transformation was used, which effectively stabilised the variance and made the data easier for subsequent statistical modelling. This thorough methodology improves the accuracy of the study and allows for a deeper understanding of the underlying stock price dynamics.

3.5 Modeling Techniques

3.5.1 ARIMA, SARIMA, ETS, TBATS

To effectively forecast and analyse time series data, a variety of statistical models can be used, each tailored to the data's specific properties. In this section, we look at various important modelling techniques, including ARMA, ARIMA, SARIMA, ETS, and TBATS. These models are critical tools for time series analysis, with each providing distinct techniques and benefits for capturing various characteristics of the data, such as trend, seasonality, and autocorrelation. By carefully fitting these models and selecting relevant parameters, we hope to enhance predicting accuracy and obtain a better understanding of the underlying patterns in time series.

3.5.2 Implementation of AR, MA, and ARMA Models

In this time series analysis, multiple models were used to represent various aspects of the data's structure and dynamics. These models are the AutoRegressive (AR) model, the Moving Average (MA) model, and the combined AutoRegressive Moving Average (ARMA) model. The `Arima` function from R `forecast` package was used to fit these models. The time series data was then fitted to a first-order autoregressive (AR) model. This model depicts the relationship between the time series current and immediate past values. The Moving Average (MA) model of order 2 was used to simulate the effect of previous error terms on the current value of the time series.

Now, to capture both autoregressive and moving average components, an ARMA model with order (1,2) was fitted. This model brings together the AR(1) and MA(2) components and the ARMA(1,2) model was specified with ARIMA of order (1,0,2), where `order = c(1,0,2)` denotes that the model has one autoregressive component and two moving average terms, with no differencing.

3.5.3 Implementation of ARIMA

The `Auto-ARIMA` function is useful in time series analysis since it automates the process of determining the ideal parameters for an ARIMA model. Automating parameter selection streamlines the modelling process and allows for more accurate projections with less personal intervention.

The `Auto-ARIMA` function is particularly useful for picking the optimum combination of parameters (p, d, q) that minimises the AIC (Akaike Information Criterion) or other criteria, which are critical for identifying the best model for a given time series data. The `forecast` package in R provides the Auto-ARIMA function. This package is frequently used for time series forecasting, and it offers a variety of model fitting and evaluation capabilities.

The key parameters in the code involves:

- `approximation = FALSE` ensures that exact calculations are used rather than approximations.
- `stepwise = FALSE` allows for a more exhaustive search for the best model rather than a stepwise approach.
- `seasonal = FALSE` specifies that the model does not include seasonal components, assuming a non-seasonal model.

For this study, the Auto-ARIMA model was applied to the training dataset, resulting in an optimised model for predicting and subsequent evaluation.

3.5.4 Implementation of SARIMA

In this research, the Seasonal ARIMA (SARIMA) model was used on the training dataset to capture both non-seasonal and seasonal patterns in PepsiCo stock price data. The model fitting was performed using the `auto.arima` function from the `forecast` package in R. This function automates the process of selecting the ideal parameters for the SARIMA model.

The `auto.arima` function is particularly useful because it identifies the best parameters for the non-seasonal components (p, d, q) as well as the seasonal components (P, D, Q) by evaluating different combinations and selecting the model with the least AIC (Akaike Information Criterion). This ensures that the model can capture the underlying seasonal trends and patterns in the data.

The SARIMA model was fitted to the training dataset using the following essential parameters:

- `seasonal = TRUE`: This parameter ensures that the SARIMA model accounts for the recurring seasonal patterns in the time series data.
- `stepwise = FALSE`: This parameter function evaluates all possible combinations of SARIMA parameters to find the best model, rather than using a stepwise approach, which might be faster but less exhaustive.
- `approximation = FALSE`: This parameter instructs the `auto.arima` function to use the exact maximum likelihood estimation method, which is more accurate but computationally intensive.

Finally, the SARIMA model was fitted to the altered training dataset using these parameters. The model effectively captured seasonal variations by selecting `seasonal = TRUE`. Disabling the stepwise and approximation options (`stepwise = FALSE` and `approximation = FALSE`) ensured a thorough search for the best model without sacrificing accuracy.

3.5.5 Implementation of ETS

To capture the underlying trends in the PepsiCo stock price time series, the ETS (Error, Trend, Seasonality) model was implemented using the `ets` function in R. This model was applied to the altered training dataset to account for the data's level, trend, and seasonality.

Using the ETS model, we may breakdown the time series into constituent parts and forecast based on the detected components. The ETS model's ability to handle a variety of trend and seasonality patterns makes it an appropriate choice for modelling PepsiCo stock data.

3.5.6 Implementation of TBATS

The TBATS model was used to capture the complicated seasonal patterns and nonlinear trends in the PepsiCo stock time series data. The TBATS model is well-suited to scenarios with numerous seasonalities and complex seasonal and trend components, making it an excellent choice for this dataset.

This model was applied on the training dataset to ensure that the discovered patterns and seasonalities were accurately represented. The analysis used TBATS to generate a robust forecast that accounts for the complexities identified in the PepsiCo stock time series data. It is implemented in R using the `tbats` function from the `forecast` package, which automatically selects appropriate model parameters for the provided data.

3.6 Conclusion on Model Selection

The Akaike Information Criterion (AIC) was used to determine the best model for anticipating PepsiCo close price values. The AIC is a statistic that weighs the model's goodness of fit against its complexity, penalising models with more parameters.

The **ARIMA(2,1,2)** model using `auto.arima` function has the lowest AIC value (812.7) and this shows that the ARIMA(2,1,2) model fits the data well and is simpler to use.

For comparison:

- ARIMA(1,0,2): AIC = 906.28
- ARIMA(0,0,2): AIC = 909.54
- ARIMA(2,1,2): AIC = 812.7
- ETS(A,N,N): AIC = 1124.386
- BATS: AIC = 1121.421

In conclusion, the `Auto-ARIMA(2,1,2)` model has the lowest AIC value, it is deemed the best model for this forecasting job. This model is favoured due to its excellent mix of fit and complexity, making it the best option for accurate and trustworthy PepsiCo stock price predictions.

3.7 Diagnostic Tests

3.7.1 Hypothesis Testing in Time Series Analysis

Hypothesis testing is a fundamental statistical procedure used to draw conclusions about a population based on samples. In time series analysis, hypothesis testing is very valuable for evaluating data assumptions like stationarity. Stationarity is an important feature of time series data in which statistical variables such as mean, variance, and autocorrelation remain constant throughout time. Testing for stationarity ensures that our forecasting models are accurate and dependable.

Augmented Dickey-Fuller (ADF) Test: The Augmented Dickey-Fuller (ADF) test is a commonly used statistical technique for determining if a time series is stationary. The null hypothesis of the ADF test is that the time series contains a unit root, implying non-stationarity. The alternative hypothesis states that the time series is stationary. To verify for stationarity in this project, the ADF test was used on the ARIMA model's residuals. The `adf.test` function from the `tseries` package was utilised for this test.

The ADF test results are as follows:

> ***Augmented Dickey-Fuller Test***
>
> Data: `arima_model$residuals`
> Dickey-Fuller = -5.1347, Lag order = 5, p-value = 0.01
> Alternative hypothesis: stationary

The negative Dickey-Fuller statistic and a **p-value of 0.01** show that the null hypothesis of non-stationarity can be rejected, implying that the residuals are stationary.

Ljung-Box (LJB) Test: The Ljung-Box test, also known as the Box-Ljung test, is used to detect whether or not a time series model's residuals include considerable autocorrelation. The null hypothesis of the Ljung-Box test is that the residuals are independently distributed, implying that there are no substantial autocorrelations. A high p-value implies that the residuals are not significantly autocorrelated, implying that the model well captures the data's structure. The `Box.test` function with type "Ljung-Box" from the `stats` package was used to perform this test.

The LJB test results are as follows:

> ***Box-Ljung Test***
>
> Data: `arima_model$residuals`
> X-squared = 45.499, df = 50, p-value = 0.6544

The **p-value of 0.6544** indicates that we fail to reject the null hypothesis, implying that the residuals have no significant autocorrelations and hence behave like white noise.

3.7.2 Function Usage and Interpretation

In the R code provided:

- The `checkresiduals` function from the `forecast` package was used to visually analyse the residuals of the ARIMA and ARMA models, producing plots of residuals and ACF/PACF to assess randomness.
- The `acf` and `pacf` functions were used to construct and visualise the residuals autocorrelation and partial autocorrelation functions.

- The residuals were tested for stationarity using the ADF function from the `tseries` package.
- To test for autocorrelations in the residuals, we utilised the `Box.test` function with the type "Ljung-Box".

These diagnostic tests ensure that the residuals of the models meet the necessary assumptions for valid forecasting, providing confidence in the accuracy and reliability of the forecasts generated.

Chapter 4

Machine Learning Models

Machine learning (ML), a subset of artificial intelligence (AI), emphasises on developing systems that learn from data and improve their performance autonomously. Unlike traditional programming, machine learning algorithms recognise patterns in data and utilise these insights to produce predictions without explicit instructions based on [6].

In time series analysis, ML models excel in detecting trends, seasonality, and complicated patterns in sequential data. Organisations can use machine learning to improve forecasting accuracy and automate decision-making processes. This approach is extremely useful for applications such as financial forecasting, demand prediction, and operational efficiency.

4.0.1 XGBoost and Support Vector Machines (SVM)

Traditional statistical models such as ARIMA and SARIMA are commonly employed in time series forecasting because they can capture linear trends and seasonality in data. However, real-world time series data can display complicated, non-linear patterns that standard models may struggle to reliably anticipate. To solve these issues, powerful machine learning models like XGBoost and Support Vector Machines (SVM) might be used. These models can capture complicated correlations in the data, making them effective tools for boosting forecast accuracy.

XGBoost and SVM are two of the most effective algorithms for time series forecasting when dealing with non-linearities, high-dimensional feature spaces, and noisy or irregular datasets. By using these models, we may overcome the constraints of linear models and make more exact and dependable predictions in complicated forecasting scenarios.

In the following sections, we will look at the theoretical concepts underlying XGBoost and SVM, their specific applications in time series forecasting, and the conditions under which they are most beneficial. This presentation will also cover how these models were deployed in the context of your project, with a focus on how they improved the forecast accuracy and resilience.

XGBoost (Extreme Gradient Boosting) referred from [8], is a sophisticated machine learning technique that relies on gradient boosting. It is well-known for its speed and performance, making it an excellent candidate for a variety of machine learning tasks, including time series forecasting. The core idea underlying XGBoost is to create an ensemble of weak learners, often decision trees, with each new tree correcting the mistakes committed by the preceding ones. The model optimises the loss function using gradient descent, resulting in a highly accurate and resilient prediction model.

XGBoost constructs trees consecutively, attempting to repair earlier model mistakes at each stage. To manage model complexity and prevent overfitting, the approach minimises a regularised objective function that comprises both a convex loss function and a regularisation term. XGBoost's regularisation feature makes it especially effective for time series data, where overfitting to trends or noise can result in poor out-of-sample predictions.

Usage in Time Series Forecasting: XGBoost is extremely beneficial in time series forecasting when dealing with nonlinear patterns and complex data interactions. It can handle big datasets with missing values and is adaptable enough to be customised for different purposes. In a time series context, XGBoost can be used to treat the problem as a supervised learning task, with past observations serving as features to predict future values. It's especially useful when the time series has complex, non-linear patterns that typical linear models may struggle with.

XGBoost is employed when time series data exhibits non-linear trends and patterns, or when the dataset is vast and potentially noisy. Its capacity to handle a variety of data anomalies and generate accurate forecasts makes it an excellent alternative when more standard methods, such as ARIMA or SARIMA, may fail to capture the data's complexity. In time series forecasting project, XGBoost would be great for detecting subtle patterns and enhancing prediction accuracy, particularly when seasonality and trend components are non-linear.

Support Vector Machines (SVM) referred from [11], are supervised learning models that may be applied to classification and regression applications. In the field of time series forecasting, SVMs are commonly utilised in their regression version, known as Support Vector Regression. The primary principle behind SVM is to identify the optimal hyperplane that maximises the margin between distinct data points (for classification) or fits the best line (for regression).

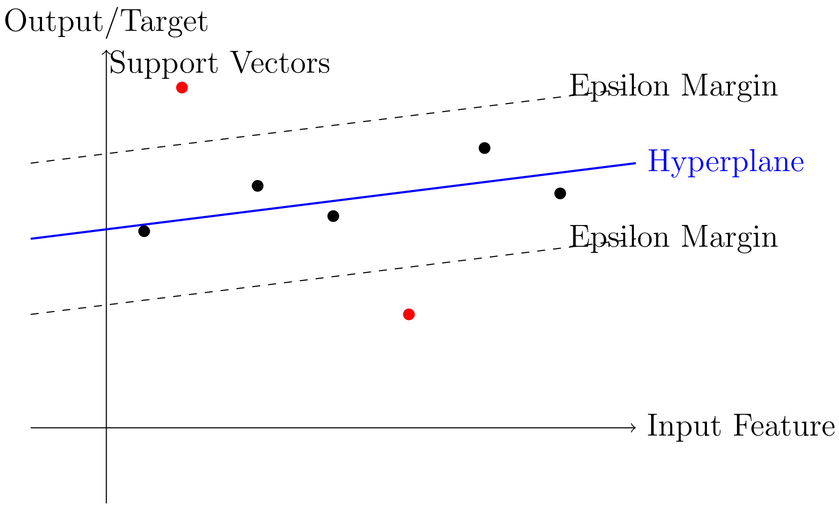

Figure 4.1: Support Vector Machine (SVM) illustration with hyperplane, margin boundaries, and support vectors.

In SVR, the goal is to discover a function that deviates from the actual observed data points by no more than a specified threshold (known as epsilon) while being as flat as possible. SVM employs kernel functions to transform the input data into a higher-dimensional space that facilitates the regression job. This transformation enables SVM to detect nonlinear correlations in the data.

Usage in Time Series Forecasting: XGBoost is extremely beneficial in time series forecasting when dealing with nonlinear patterns and complex data interactions. It can handle big datasets with missing values and is adaptable enough to be customised for different purposes. In a time series context, XGBoost can be used to treat the problem as a supervised learning task, with past observations serving as features to predict future values. It's especially useful when the time series has complex, non-linear patterns that typical linear models may struggle with.

SVM is beneficial in time series forecasting when the variables relationships are non-linear and complicated. It works particularly well when the data is not linearly separable in its original space. SVM's ability to use kernel functions enables it to adapt to the form of the data and identify patterns that may not be visible in the original feature space.

SVM is commonly employed when time series data lacks a simple linear trend and there are complicated, non-linear correlations between past observations and future values. It is effective when the dataset is not too vast, as SVM can be computationally expensive, and it excels when the time series has a high degree of variability that typical linear models cannot capture. SVM could be used in your project when the data exhibits nonlinear patterns and you require a model that can adapt to these patterns in order to offer reliable forecasts.

4.1 Ensemble Methods for Time Series Forecasting

Ensemble methods are advanced machine learning approaches that aggregate predictions from numerous models to produce a more precise and reliable forecast. The basic goal of ensemble approaches is to use the strengths of several models to compensate for their individual errors, resulting in better overall predictive performance. This method is particularly useful for complex jobs like time series forecasting.

4.1.1 Benefits of Ensemble Methods

Ensemble approaches provide a number of significant advantages for forecasting the PepsiCo stock price.

- **Improved Accuracy:** Ensemble approaches improve predicting accuracy by combining the results of numerous models. Each model may capture distinct features of the data, such as trends, seasonality, or short-term changes. Combining these several insights yields a more accurate projection of PepsiCo's stock price.
- **Robustness to Overfitting:** Overfitting happens when a model learns noise in training data rather than the genuine underlying patterns, which leads to poor performance on new data. Ensemble approaches reduce this risk by averaging out individual model errors, resulting in more robust and dependable projections, which are critical in unpredictable financial markets.
- **Model Diversity:** Time series data, such as stock prices, frequently show complicated patterns that a single model may not fully capture. Ensemble approaches can better capture the many patterns and interactions in stock price movements by incorporating a variety of models, including as ARIMA, ETS, and TBATS, each specialising on a particular aspect of data.
- **Reduction of Model Bias:** Each model has inherent biases, such as overestimating trends or underestimating abrupt shifts. Ensemble approaches mitigate these biases by averaging predictions from numerous models, resulting in more balanced and impartial projections.

Ensemble methods were used in this study to aggregate projections from different base models, such as AR, MA, ARIMA, SARIMA, ETS, and TBATS. This method enables us to use the strengths of each particular model to generate a more reliable forecast. The ensemble method utilised is stacking, which combines forecasts from basic models and feeds them into sophisticated machine learning models like XGBoost and Support Vector Machines (SVM) to get the final forecast.

By using ensemble approaches, we ensure that our forecasts are not excessively dependent on the assumptions or constraints of any one model. This leads to more reliable estimates, which are better suited for making educated financial decisions.

4.1.2 Implementation of XGBoost and SVM using Ensemble methods

In this study, I used a variety of base models to estimate future periods and then compared the results using sophisticated machine learning techniques. The models are AR, MA, ARMA, ARIMA, SARIMA, ETS, TBATS, and a local trend model. To combine the capabilities of various models and increase prediction accuracy, I used a stacking strategy that aggregated the projections from each base model into a single, more trustworthy forecast.

XGBoost, a sophisticated gradient-boosting technique, was used for stacking. The `XGBoost` model was trained with forecasts from the basis models as input features. This was accomplished using the `xgboost` package in R. To generate future forecasts, I created a `xgb.DMatrix` from the combined forecast data using the `as.matrix` function and then use the `predict` function.

Similarly, `Support Vector Machines` (SVM) in its regression form, known as Support Vector Regression (SVR), was used to compare with the stacked XGBoost model. The SVM model was developed using the `e1071` package in R, which provides a simple interface for training and forecasting with SVM.

Later, the forecasts from the basic models are combined into a data frame and used to train the SVM model. The `predict` function was used to create future forecasts.
Finally, the results were plotted to visually compare the actual historical data to the forecasts made by both the `XGBoost` and `SVM` models, providing insight into their respective performances. This method enabled the discovery of which model produced the most accurate predictions for the time series data.

Chapter 5

Results and Analysis

5.1 Model Evaluation

5.1.1 Performance Metrics

To assess the performance of the forecasting models, several critical metrics were used. These measures allow to quantify how effectively each model predicts future values relative to actual results.

The primary performance metrics used are:

- **Mean Absolute Error (MAE):** This calculates the average size of errors in a group of projections, regardless of direction. The formula from [11] is as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i| \tag{5.1}$$

  where, $\hat{y}_i$ is the forecasted value, y_i is the actual value, and n is the number of forecasts.

- metric takes the square root of the average squared difference between forecasted and actual values. It's computed as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2} \tag{5.2}$$

- **Mean Absolute Percentage Error (MAPE):** This calculates the forecasts accuracy as a percentage. It's defined as and collected from [2]:

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n} \left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{5.3}$$

where, the error is expressed as a percentage of the actual values.

- **Root Mean Squared Percentage Error (RMSPE):** This measure takes the square root of the average squared percentage error. It's provided by [7]:

$$\text{RMSPE} = \sqrt{\frac{100^2}{n}\sum_{i=1}^{n}\left(\frac{\hat{y}_i - y_i}{y_i}\right)^2} \tag{5.4}$$

  RMSPE offers information about the relative size of mistakes in percentage terms.

In this investigation, we used an ensemble method to aggregate forecasts from different models. Two methodologies were compared, ***Simple Average Forecast and Stacking Forecast***.

Simple Average Forecast: This method involves calculating the average of forecasts from numerous models to get a composite forecast. The performance indicators for the Simple Average Forecast are as follows:

- MAE: 148.3245
- RMSE: 148.3946
- MAPE: 88.18671%
- RMSPE: 88.18726%

Stacking Forecast: This method uses a machine learning model (such as XGBoost) to aggregate forecasts from various base models. The performance indicators for the Stacking Forecast are:

- MAE: 5.01844
- RMSE: 5.984601
- MAPE: 2.945837%
- RMSPE: 3.482807%

Forecasting errors are critical in determining the quality of a forecasting model. They demonstrate how closely the model's predictions correspond to the actual outcomes. The previously mentioned measures (MAE, RMSE, MAPE, and RMSPE) provide different viewpoints on forecast accuracy.

MAE is a simple measure of average prediction error, whereas RMSE highlights greater mistakes by squaring them, making it sensitive to outliers. MAPE and RMSPE provide percentage-based errors, which are valuable for analysing forecast performance in comparison to actual results.

The results show that the ***Stacking Forecast*** outperforms the ***Simple Average Forecast*** across all measures. The Stacking Forecast has lower MAE, RMSE, MAPE, and RMSPE, indicating that it makes more accurate and dependable forecasts.

In conclusion, the investigation shows that the Stacking Forecast, which combines numerous model predictions with advanced machine learning algorithms, outperforms a simple average of forecasts. This increased precision can provide useful insights for investors and marketers, allowing them to make better judgements about PepsiCo stock prices. Stakeholders can improve their strategic planning and investment decisions by using models with lower forecasting errors, resulting in better financial outcomes and more efficient resource allocation.

5.2 Validation of Forecasts on Test Data

During the validation of the forecasting models on the test dataset, a thorough examination of their predictive ability was done. The models tested were Auto Regressive (AR), Moving Average (MA), ARMA, Auto-ARIMA, SARIMA, ETS, TBATS, and a Local Trend model. Several metrics were used to assess the performance of these models, including mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and root mean squared percentage error (RMSPE). Among the models studied, the Auto-ARIMA model had the lowest error metrics overall, with an MAE of 3.85, an RMSE of 5.17, a MAPE of 2.25%, and an RMSPE of 2.97%.

These results show that the Auto-ARIMA model produced the best accurate forecasts when compared to individual models. Despite this great performance, the Auto-ARIMA model's projections did not match the needed accuracy criteria for certain applications, most likely due to overfitting or poor pattern identification which is shown in the below figure.

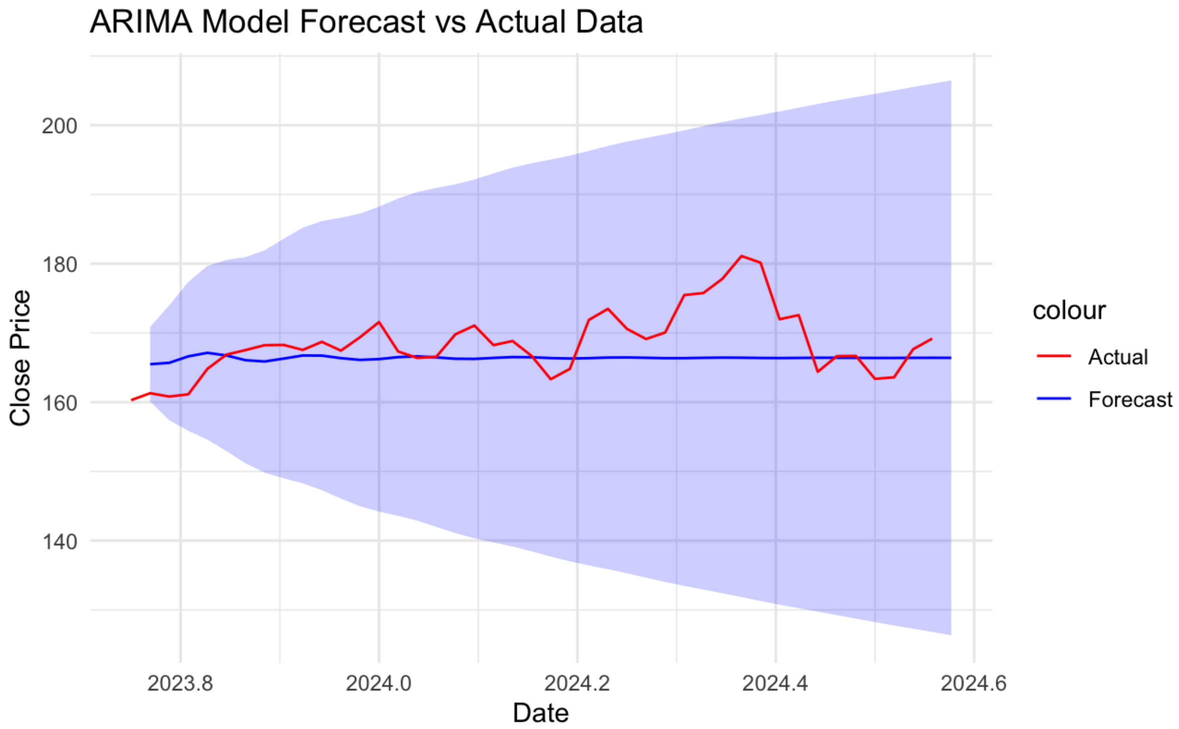

Figure 5.1: ARIMA Model Forecast

To overcome these restrictions and improve forecast accuracy, an ensemble forecasting approach was used. This method aggregated the predictions of numerous models, leveraging their various strengths while correcting for individual faults. The stacking forecast, which combines forecasts from multiple models, showed significant improvement in forecast accuracy. The ensemble approach had an MAE of 5.02, an RMSE of 5.98, a MAPE of 2.95%, and an RMSPE of 3.48 percent. These findings show a considerable reduction in errors when compared to the basic average forecast and the individual models.

The test dataset covering the period from October 2023 to July 2024.

Model	**MAE**	**RMSE**	**MAPE**	**RMSPE**
AR	169.67	169.73	100.65%	100.65%
MA	169.64	169.71	100.64%	100.64%
ARMA	169.95	170.01	100.83%	100.83%
ARIMA	3.85	5.17	2.25%	2.97%
SARIMA	168.56	168.63	100.00%	100.00%
ETS	169.67	169.74	100.66%	100.66%
TBATS	170.49	170.56	101.14%	101.14%
Local Trend	169.67	169.74	100.66%	100.66%
Simple Average Forecast	148.32	148.39	88.19%	88.19%
Stacking Forecast	5.02	5.98	2.95%	3.48%

Table 5.1: Performance metrics for different forecasting models on the test dataset.

The ensemble method's success is seen in its capacity to average out individual model errors, resulting in more trustworthy and accurate projections. This technique improves overall prediction accuracy while also providing a more robust forecasting solution by balancing the trade-offs inherent in different models. The Auto-ARIMA model performed well, but the ensemble method outperformed it in terms of forecast refinement and total accuracy. This is later shown in below sections

5.2.1 Forecasting Result and Accuracy

This section compares the performance of various forecasting models applied to PepsiCo time series data, with a focus on Simple Average, Support Vector Machine (SVM), and XGBoost. The purpose is to determine which model makes the best accurate forecasts and to highlight the benefits of merging various forecasting models using ensemble methods.

The Simple Average Forecast: The Simple Average approach combines the predictions of different models by calculating their arithmetic mean. This approach is frequently used as a baseline in ensemble learning.

- **Visual inspection**: When plotted, the Simple Average forecast showed drastically misaligned with the actual data, suggesting poor performance as seen in Figure 5.3. This disparity is clear from the significant mistake measurements given in the Performance measurements section.

- **Interpretation**: The poor performance might be attributable to the fact that averaging forecasts from models with varied biases and variances may lead to a diluted or overly smoothed forecast, failing to capture the underlying variability in the data.

Support Vector Machine (SVM) Forecast: The SVM model, recognised for its ability to handle high-dimensional data and non-linear connections, was used to forecast future values.

- **Visual inspection**: The SVM forecast produced a slightly flat trend as seen in Figure 5.4, indicating that it lacked sensitivity to dynamic changes in the dataset. While it outperformed the Simple Average technique however, it still missed several important variants.
- **Interpretation**: Because of its kernel-based methodology, SVM may have struggled to capture complicated temporal patterns, making it unsuitable for time series forecasting without careful tuning.

XGBoost Forecast: XGBoost, a sophisticated gradient boosting technique, was used as part of the ensemble strategy. XGBoost is very good at managing large datasets and complicated relationships.

- **Visual inspection**: XGBoost's forecast properly predicted peaks and troughs based on actual data. The model performed well, with few predicted errors, indicating a good fit to past data.
- **Interpretation**: XGBoost's improved performance is due to its capacity to simulate complicated non-linear interactions between variables and its robustness against overfitting using regularisation techniques.

Furthermore, XGBoost was utilised to forecast real results for the next two years, up to 2026. Figure 5.2 demonstrates the model's ability to accurately predict future trends.

The ensemble methods examined in this study demonstrate the advantages of mixing many models to improve forecast accuracy. While the Simple Average approach served as a baseline, its performance was poor due to the averaging effect, which can smooth out significant variances in data. SVM, while superior, was rather constrained in capturing the entire dynamics of the time series.

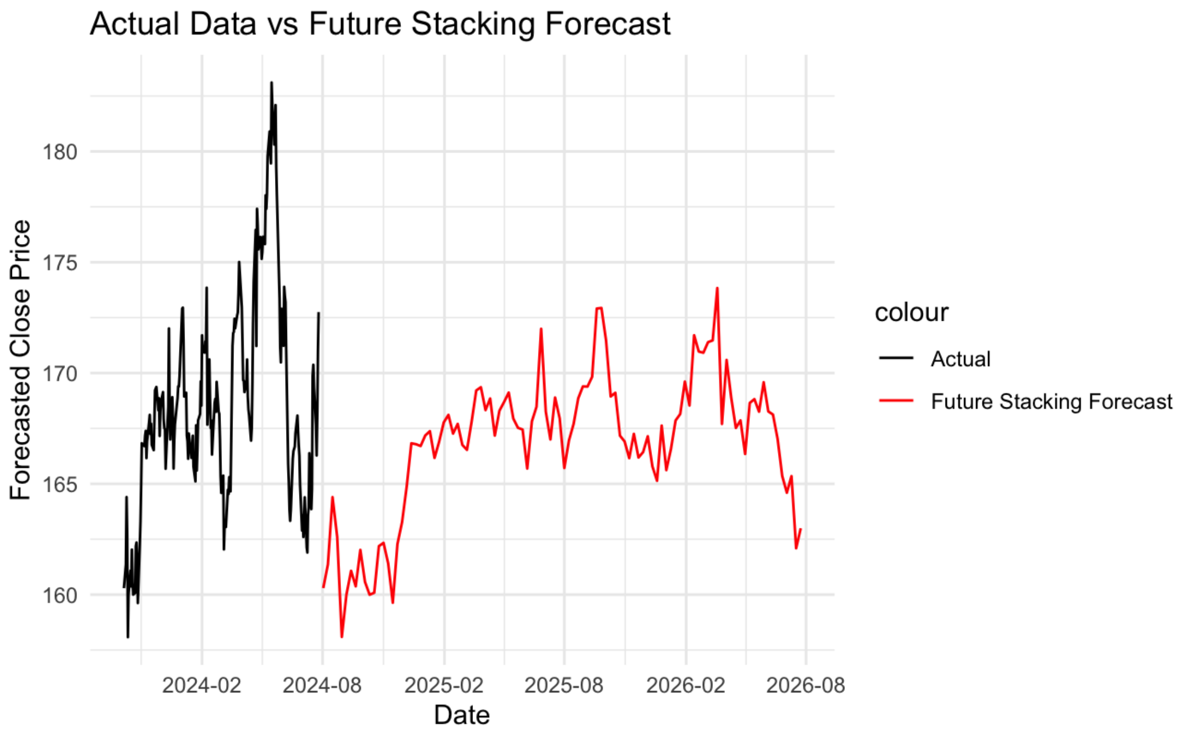

Figure 5.2: XGBoost Forecast for PepsiCo Stock Price Until 2026

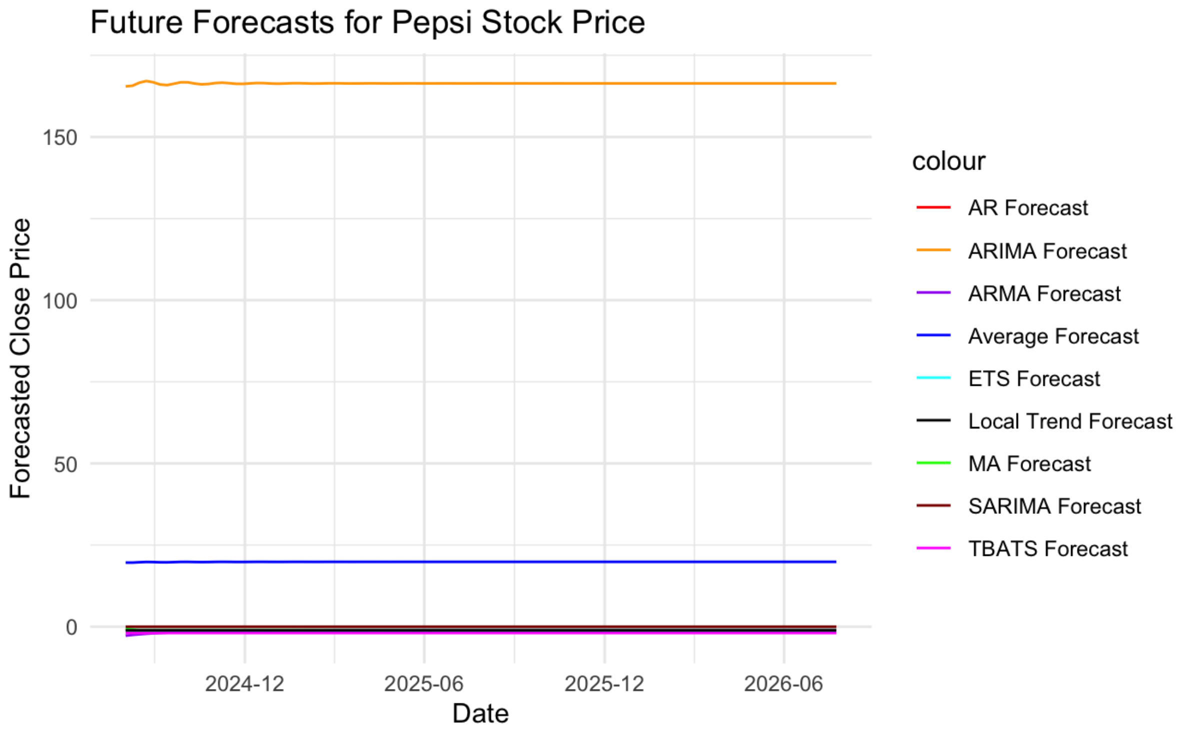

Figure 5.3: Simple Average Forecast for PepsiCo Stock Price until 2026

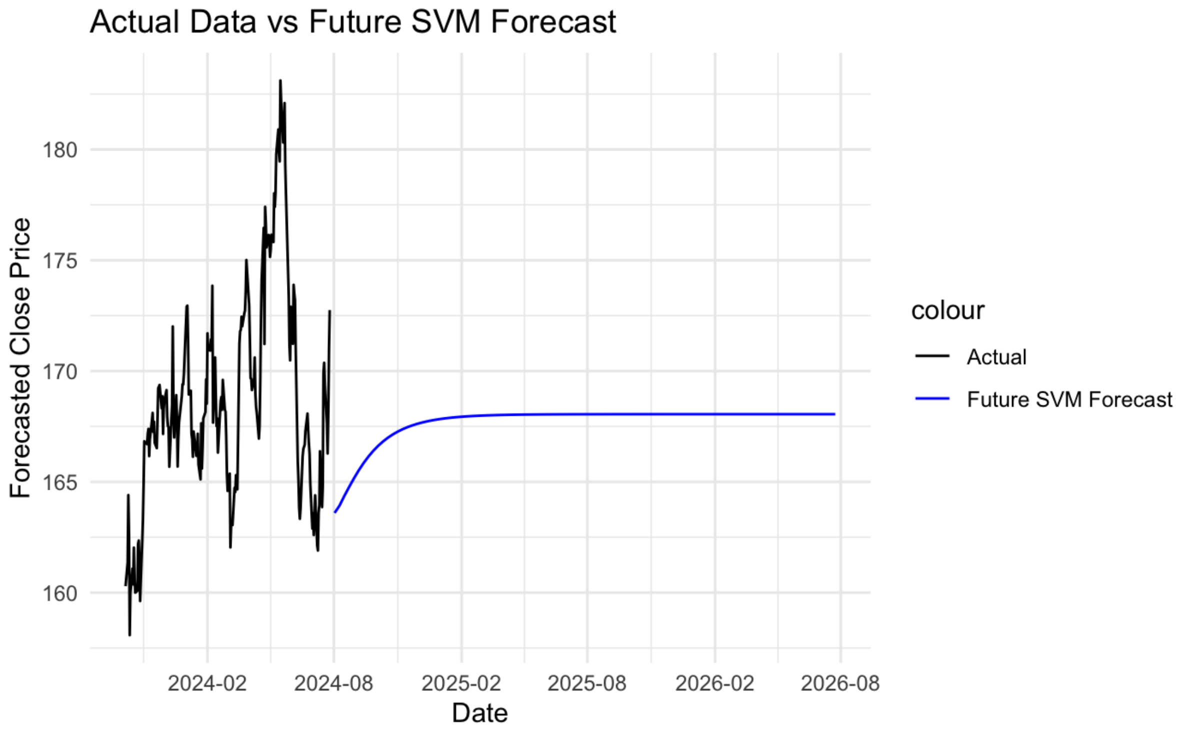

Figure 5.4: SVM Forecast for PepsiCo Stock Price until 2026

XGBoost emerged as the most effective model, giving projections that were closely related to actual values. This highlights the benefits of utilising sophisticated machine learning approaches for time series forecasting, particularly in cases with complicated patterns and trends.

Forecast accuracy: To evaluate the forecast accuracy of the XGboost stacking technique, 5% threshold was set based on the actual results. The model's accuracy inside this threshold was **86.04651%**, showing that the majority of its predictions were closely aligned with the actual values. This high level of precision demonstrates how well the Stacking XGBoost model captures the underlying patterns in the data. However, the other models as shown in the forecast figures above, these models were unable to match the accuracy of the Stacking XGBoost model, demonstrating its supremacy for this particular forecasting task.

Practical Implications: XGBoost-based forecasts can help investors, marketers, and decision-makers make more accurate projections of stock prices and economic indicators. The close alignment of XGBoost's projection with actual data implies it could be a useful tool for making educated decisions, potentially leading to better investment strategies and market positioning. Using an ensemble strategy, particularly one that includes advanced models such as XGBoost, stakeholders can produce more trustworthy forecasts, improving their capacity to foresee future trends and optimise their plans.

Chapter 6

Conclusion

6.1 Summary of Findings

This study evaluated several forecasting models for predicting PepsiCo Inc's stock price, including Simple Average, Support Vector Machine (SVM), and XGBoost. Among these, XGBoost was the most effective, accurately capturing trends and fluctuations in historical data with minimal forecasting errors. Its strength lies in handling complex non-linear correlations and its resistance to overfitting, making it suitable for both short- and long-term forecasts. In comparison, the SVM model performed better than the Simple Average but was less effective than XGBoost. While SVM identified some trends, it struggled with dynamic changes, resulting in less responsive projections. The Simple Average model, serving as the baseline, significantly lagged behind the more advanced models. Its simplistic approach failed to account for the complexities of stock price movements, leading to notable discrepancies between predicted and actual values. Ensemble techniques that combined multiple models, including XGBoost, enhanced prediction accuracy. By leveraging the strengths of individual models, these ensembles provided more robust forecasts and mitigated the limitations of any single model. XGBoost also excelled in long-term forecasting, accurately predicting stock values up to 2026. This demonstrates its value for strategic planning and investment decisions. Visual comparisons of forecasts from Simple Average, SVM, and XGBoost supported these findings, affirming XGBoost as the most accurate and reliable model.

6.2 Recommendations

Based on a thorough research and evaluation of numerous forecasting models, several major recommendations arise for investors, financial analysts, and other stakeholders interested in projecting stock market performance, specifically PepsiCo's stock price.

- **Prioritize XGBoost for Forecasting:** XGBoost excels in predicting PepsiCo's closing price due to its ability to capture complex trends and correlations. It is recommended for both short- and long-term forecasting.

- **Consider SVM for Medium-Term Forecasts:** SVM is useful for medium-term forecasts, particularly in stable market conditions with low volatility. It should be used alongside other models.
- **Avoid Sole Reliance on Simple Average:** The Simple Average method is insufficient for capturing stock price complexities and should not be used alone. It can serve as a baseline for comparison with advanced models.
- **Utilize Ensemble Methods for Enhanced Accuracy:** Ensemble approaches, especially those incorporating XGBoost, improve forecast accuracy and are recommended for reliable predictions in volatile markets.
- **Apply XGBoost for Long-Term Planning:** XGBoost's accuracy over long periods makes it suitable for strategic decisions like retirement planning and major investments.
- **Incorporate Volatility Forecasting for Risk Management:** Future research should include volatility forecasting. SVM and other models can complement XGBoost for better risk management.

6.3 Future Work

While the current research provides valuable insights into forecasting stock prices, there are various aspects that require more investigation to improve the accuracy and application of forecasting models. Exploring new machine learning techniques is one interesting area for future research. While XGBoost has demonstrated outstanding performance, additional advanced methods, such as deep learning models like Long Short-Term Memory (LSTM) networks or Gated Recurrent Units (GRU), should be researched. These models are well-suited to capture complicated, non-linear patterns in time series data and may provide additional predicting accuracy. Another key option for future research is the inclusion of external elements and attributes other than previous stock prices. Integrating macroeconomic information, market mood, trade volume, and geopolitical events may provide a more comprehensive view and improve forecasting models' predictive potential. Accounting for these extra variables may allow models to better represent the diverse influences on stock prices.

Furthermore, future research could benefit from analysing high-frequency data. The current study used weekly data, however looking at daily or even hourly pricing could enhance forecast accuracy and provide more relevant insights. Handling high-frequency data may necessitate specialised modelling approaches to successfully manage the increasing data volume and complexity.

The development of hybrid models is yet another interesting topic for future research. Combining classical statistical methods with machine learning techniques, such as merging ARIMA models with XGBoost or other sophisticated algorithms, has the potential to capitalise on both approaches' strengths. This hybrid method may solve individual model shortcomings and result in more robust forecasts. Also, rigorous cross-validation and robustness testing can be used to guarantee that forecasting models are reliable across a range of market situations. Rolling window cross-validation and out-of-sample validation are two techniques that can help check model performance and stability over multiple time periods, ensuring that forecasts stay reliable under a variety of scenarios. Incorporating volatility and risk measures into forecasting models is another important area of future research. While this study focusses on predicting stock prices, volatility forecasting using GARCH models could provide useful insights into market risk. This addition would improve risk management techniques and give a more complete picture of market dynamics.

Overall, broadening the research to include other financial products such as bonds, commodities, or cryptocurrencies could provide a more comprehensive understanding of forecasting approaches. Different asset classes may exhibit distinct features and behaviours that can be addressed using bespoke forecasting models, broadening the research's relevance and impact. In conclusion, pursuing these pathways of future study has the potential to advance forecasting approaches, improve accuracy, and provide more complex insights into financial markets.

Bibliography

[1] Financial Data Analytics. *Time Series analysis for Business*. Accessed: 2024-08-31. 2024. URL: https://qmplus.qmul.ac.uk/course/view.php?id=23739.

[2] Wikipedia Data Analytics. *Time Series analysis for Business*. Accessed: 2024-08-31. 2024. URL: Wikipedia:%20Mean%20Absolute%20Percentage%20Error.

[3] John C Hull and Sankarshan Basu. *Options, futures, and other derivatives*. Pearson Education India, 2016.

[4] *Investment Fund Performance Analysis*. Accessed: 2024-08-31. 2024. URL: https://www.rbcgam.com/en/ca/learn-plan/investment-basics/understanding-the-relationship-between-volatility-and-returns/detail#:~:text=The%20more%20the%20price%20changes,preparing%20for%20weather%20on%20vacation.

[5] Assan Jallow. "A strategic case study on PepsiCo". In: *Available at SSRN 3828353* (2021).

[6] Batta Mahesh. "Machine learning algorithms-a review". In: *International Journal of Science and Research (IJSR).[Internet]* 9.1 (2020), pp. 381–386.

[7] Wikipedia Error Metrics. *Time Series analysis for Business*. Accessed: 2024-08-31. 2024. URL: Cross%20Validated%20(Stack%20Exchange)%20discussion%20on%20RMSPE.

[8] Santhanam Ramraj et al. "Experimenting XGBoost algorithm for prediction and classification of different datasets". In: *International Journal of Control Theory and Applications* 9.40 (2016), pp. 651–662.

[9] Statsmodel. *Time Series analysis*. Accessed: 2024-08-31. 2024. URL: https://www.statsmodels.org/stable/generated/statsmodels.tsa.exponential_smoothing.ets.ETSModel.html#.

[10] *Time Series analysis for Business*. Accessed: 2024-08-31. 2024. URL: https://qmplus.qmul.ac.uk/course/view.php?id=23745.

[11] Haifeng Wang and Dejin Hu. “Comparison of SVM and LS-SVM for regression”. In: *2005 International conference on neural networks and brain*. Vol. 1. IEEE. 2005, pp. 279–283.