FMDB *Transactions on Sustainable Computer Letters*

A Comparative Study Between Statistical and Machine Learning Methods for Forecasting Retail Sales

Shanmugapriya Murugavel[1,*], Francisco Hernandez[2]

[1,2]Department of Mathematics, Munster Technological University, Cork, Munster, Ireland.
s.murugavel@mycit.ie[1], francisco.hernandez@mtu.ie[2]

Abstract: Forecasting techniques are widely used in various sectors, from predicting climatic changes, healthcare systems, agriculture crop simulation, and stock market management to business forecasting, including demand, supply, and sales forecasting. Sales prediction is one of the forecasting methods that predict the retailing of products to be sold in the future. Sales forecast additionally serves as a tool for identifying benchmarks, determining incremental results of new initiatives, planning resources to meet demand, and projecting future costs. The forecasting process helps retailers stock products based on customer requirements, improving the supply chain management process. During unexpected situations like the pandemic, sales of a few items, such as milk, bread and toilet paper, were unpredictable. This is an example where sales prediction plays a major impact. This paper compares the performance of various machine learning and statistical time series models in predicting sales. As strong seasonal fluctuations were observed, several regression approaches like LSTM, Linear Regression and Random Forests were used to predict future sales using the historically available data. Finally, the models were compared based on the RMSE scores to evaluate the performances.

Keywords: Statistical and Machine Learning Methods; Forecasting Retail Sales; Linear Regression; Supply Chain Management Process; Sales Forecast; Predicting Climatic Changes, Healthcare Systems.

Received on: 30/12/2022, **Revised on:** 27/02/2023, **Accepted on:** 07/04/2023, **Published on:** 17/04/2023

Cited by: S. Murugavel and F. Hernandez, "A Comparative Study Between Statistical and Machine Learning Methods for Forecasting Retail Sales," FMDB Transactions on Sustainable Computer Letters., vol. 1, no. 2, pp. 76 –102, 2023.

1. Introduction

Growth in technologies has widely influenced business performances across all sectors. Many industries have completely transformed their interactions and services towards their customers. These innovations in various fields have helped businesses better understand future customer needs. Sales forecasting is one of the most common and essential tasks performed in small businesses and multinational conglomerates. This also supports companies in staying on track with their desired goals, effectively allocating resources, and managing forthcoming business growth. Accurate sales forecasts help create a healthy business environment and make shareholders happy, as sales are considered the lifeline of any company. The main aim behind sales forecasting is to estimate sales performances precisely. Sales forecasts mostly use historical data and industry trends to analyse weekly, monthly, quarterly, and annual sales. This forecasted information is used for making impactful business decisions like allocating resources, hiring new staff during busy hours, increasing inventory stock, or managing future sales. Sales forecasting is not just predicting the sales numbers; it provides business leaders with the necessary information to make the right decisions to help the business achieve better goals. It is considered extensive among both the industrialists and the manufacturers, which helps in better understanding the shelf-life of the goods and determining their production. In earlier days, companies produced supplies without considering demand and sales, leading to revenue loss in surplus and shortage situations.

[*]Corresponding author.

Many factors are considered while selecting a suitable method before prediction, including the relativity and availability of the historical data, a forecasted period, the time available to make the analysis, the benefit for the company through the forecast made and the purpose of the forecast. Considering all these factors, suitable techniques are considered for the analysis, and the results are compared to choose the most appropriate technique depending on each case. However, the forecasting process is still challenging to predict as the raw data collected is mixed up with high trends, seasonal variations and the sales offers made during the period. There are several tools used in the industry for forecasting. These tools include Microsoft Excel, CRM (Customer Relationship Management), Sales analytical tools, project management tools and accounting software such as QuickBooks. The sales forecast is mainly built on assumptions. A combination of Artificial Intelligence, Time Series, and Statistics is changing traditional forecasting methods. These methods are data-driven and are mostly based on quantitative analysis of past information. Changes in retail demand over the years, the global economy, weather, and the introduction of new products are some external factors that impact these data patterns. In the past, sales prediction was seen as an arduous task. But modern forecasting techniques have become game changers. Modern techniques like Machine Learning and Artificial Intelligence mainly use probability and statistics for forecasting sales.

This paper aims to develop a sales prediction model that can accurately predict the future sales of a retail store using historical data. The dataset was collected from a local retail store. Five different techniques: i) Holt-Winters' exponential smoothing techniques; ii) Auto-Regressive Integrated Moving Average (ARIMA); iii) a statistical approach using Linear Regression; iv) a time series approach based on Long Short-Term Memory (LSTM); v) and a machine learning approach based on Random Forest were considered. These models were then compared based on their accuracy and the value of residuals. Linear regression analysis predicts a dependent variable's value based on the independent variable's value. The LSTM are a Recurrent Neural Network (RNN) with a memory that helps analyse a time series [1]. Random Forest combines the output of multiple decision trees to obtain the desired value. Having work experience in the retail industry, this field was chosen as an interest.

1.1. Time Series Regression Models

The time series regression model uses sequential data collected at different points regularly. Time series analysis helps answer questions like, what will be the effect of variable Y of a change in variable X over time? Y is the dependent variable, and X is the independent variable. In the dataset collected, the store sales are analysed over the years. So, Sales data becomes the dependent variable Y – to be predicted, and time is the independent variable X already available. Since the analysis is based on the period, Time Series analysis is chosen as the major implementation method. Sequential sales data is collected starting from October 2015 to April 2022. The data is collected over seven years in a weekly periodicity. To better explain how time series can be helpful in further sales analysis, consider the following example. To forecast monthly sales y using a total number of customers visited x as a predictor. The forecasted variable y is also called the explained or dependent variable [2]. The predictor variable x is also called the explanatory or the independent variable.

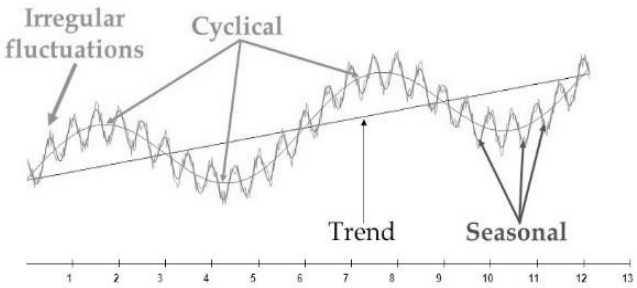

Figure 1: Components of a Time Series [3]

A time series can be identified into four components: the seasonal variation, the cyclical variation, the secular trend and the residual variation. The combination of these four components results in an irregular time series. An irregular time series is represented in Figure 1 above. Apart from these components, another term, periodicity, also plays an important role in a time series. It is usually described as the number of time points at which the data is recorded during one completion of a seasonal pattern. The periodicity can be collected hourly, daily, weekly, monthly, quarterly, or annually. Here, the dataset is of weekly and monthly periodicity.

1.2. Machine Learning/Artificial Intelligence Regression Models

Artificial Intelligence is the processes and algorithms that simulate a machine to mimic human intelligence. Personal assistant models like Alexa and Siri that mainly understand spoken language, modern web search engines, personalised ads, and self-

driving vehicles are some everyday applications of AI. AI/ML models are used in various industries, namely Healthcare, Telecommunications, Financial Services, Insurance/Recharge renewals, Automobile, etc.; the major advantage of choosing a machine learning technique for performing the sales/demand forecasting here for the available dataset is that these ML models are designed in such a way to handle thousand metrics of data at a single time and have the capacity to handle them accurately. Machine Learning involves training a model to analyse the sales from the historical data available, learning how each input factor impacts a weighted output, and finally, using the testing model to predict future sales [3]. Here, LSTM, the Long Short-Term Memory method, is chosen due to its capability to handle long-term dependencies among the data. These work on the Recurrent Neural Networks (RNN) technology, which remembers the previous information and uses it to process the current input (fig.2).

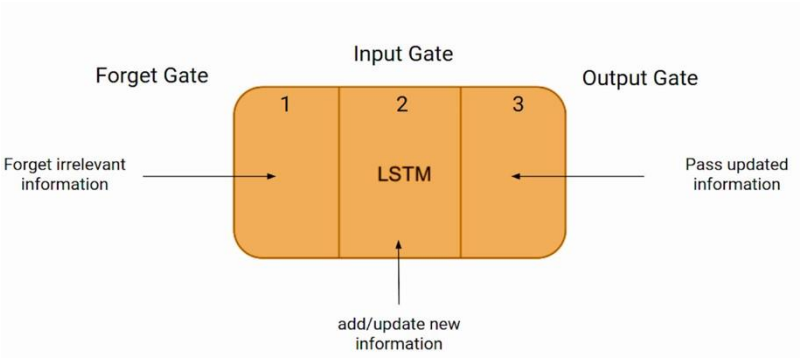

Figure 2: Architecture of an LSTM cell [4]

Here, the first part of the Forget gate decides whether the information coming from the previous timestamp is to be remembered or not. If the data feels irrelevant, the previous data is forgotten. In the second part, known as the input gate, the cell tries to learn new information from the current input. In the output gate, the information is passed from the current timestamp.

2. Literature Review

This section discusses the research papers on sales forecasting and how it is being implemented by industries of all domains, especially retail. There have been relatively few works related to this topic of study. The eXtreme Gradient Boosting (XGBoost) Machine Learning technique was used in [5] to forecast sales amounts accurately. The Walmart retail dataset from the Kaggle competition was used for this analysis. The XGBoost technique was selected because it was more scalable and efficient, making it ten times faster than other machine learning models. The algorithm's efficiency was evaluated using the RMSE score - 0.655, about 16.3% lower than other linear regression models [6] and 15.4% lower than the Ridge regression method [7].

A similar work was done in [8], which analysed the same Walmart Sales data based on the Light Gradient Boost Method (LightGBM). The results indicated the RMSE of the model was 2.09, which was better when compared to linear regression models and Support Vector Machine (SVM) models. Finally, the model with the highest accuracy was selected to make future predictions. This motivated the study to perform a comparison by designing a model using linear regression.

Many other studies have achieved the same accurate results but used different implementation techniques, such as time series modelling. One such noticeable work was from [9]. The analysis was based on historical sales data from Amazon.com, a leading e-commerce website. At first, three popular forecasting approaches were performed: Holt-Winters exponential smoothing, Autoregressive integrated moving average (ARIMA) and Neural network autoregression. The sales data forecasted Amazon's quarterly sales in 2019. Later, the forecasted sales were tested against the actual sales numbers in 2019 to compare the results. The accuracy was measured using Mean Absolute Percentage Error (MAPE) and Root Mean Square Error (RMSE). The results showed that seasonal ARIMA gave the most accurate results compared to the other two methods. Some of the standard methods used in this type of analysis are mentioned above, and a few similar approaches are to be used in this study.

In the comparative study between LSTM and ARIMA conducted in [10]. The models predicted sales for one day ahead, and the results were evaluated using RMSE and MAE together with a t-test. The study concludes that the LSTM model was much more promising in forecasting retail sales when compared to the ARIMA models. A similar approach is to be performed in this research based on LSTM models. Even though various comparative studies between traditional and modern forecasting methods have been conducted in the past, findings about the linearity of the data are mixed. For example, in a comparative study for forecasting electric load between the ARIMA model and

Few other results from [11] showed Neural Networks had better predictions than the linear ARIMA model. However, a comparative study conducted with nonlinear data [12] proved neural networks were less effective than their linear time series models. On the other hand, no study has been conducted to compare a traditional statistical model, a time series-based model, or a modern machine learning model to our knowledge. The next sections of the paper are organised to perform this comparative research. The research methodology of the study is discussed in the next section.

3. Methodology

3.1. Dataset

The sales data used in this study was obtained from a retail store based in Cork in Microsoft Excel format. The total sampling period examined is from October 2015 to May 2022. As part of this research, sales is the main component to be analysed. Figure 3 gives us a visual representation of the dataset.

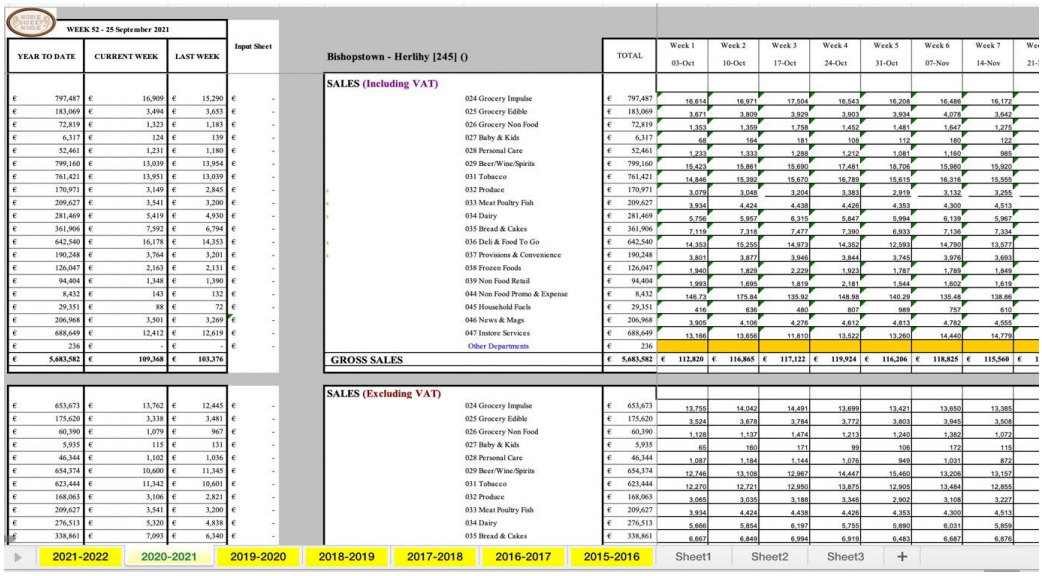

Figure 3: Sample data as received

The sales data included the VAT (Value-Added Tax), Sales excluding VAT, net Sales, Profit, Weekly Scanning Margin, Participation of various departments, Customer Count, Average Customer Spending amount, Employee wages details and weekly wastage information. Data was extracted from only the details required for the analysis. Here, the analysis is based on sales forecasting; only the Sales, including VAT details, are used. The other details, like the Sales excluding VAT, Weekly Scanning Margin, Employee wages and Participation, were removed as part of the data extraction process.

3.1.1. Software

Most statisticians prefer using R to perform these time series-based analyses. However, this study performs the complete analysis in the Python 3 environment. Python was preferred over R for easing the analysis in the same environment. Python has many libraries that can support statistical and machine-learning methods. Also, previous knowledge of working with Python was an added advantage. Jupyter Notebooks and Google Collaboratory were the tools used to perform the analysis.

3.1.2. Pre-processing on Cleaning the data

The sales data gathered was then imported into the Python working environment as a data frame. A data frame is described as a data structure of 2-dimension, often containing rows and columns with data. Once loaded, the data is first transformed into a time series format for further analysis. Check for any inconsistencies, and null values are followed. No inconsistent or missing data was found. After this, the DATE is found to be an object categorical type converted into the 'date' format using the pandas – pd.to_datetime( ) function in Python (fig.4).

```
In [57]: pd.to_datetime(Centra_sales['Date'])
Out[57]: 0      2015-03-10
         1      2015-10-10
         2      2015-10-17
         3      2015-10-24
         4      2015-10-31
                   ...
         341    2022-04-16
         342    2022-04-23
         343    2022-04-30
         344    2022-07-05
         345    2022-05-14
         Name: Date, Length: 346, dtype: datetime64[ns]
```

Figure 4: Code snap - conversion of DATE into proper datetime64 format

This converts the date into the standard YYYY-MM-DD format. The columns were renamed using lower-case letters to make the names consistent. To ease the analysis further, the weekly periodic data was converted to monthly periodicity using internal loops. Now, the observed resultant data is the monthly overall sales of the store in euros.

3.2. Exploratory Data Analysis (EDA)

The process of EDA is useful for analysing the data set using various visualization methods. The main characteristics of the dataset are summarised while performing EDA. This EDA process also helps identify the outliers or some hidden relations between the variables. Exploratory data analysis can be performed using various tools. Below are the images from the EDA process for this dataset (fig.5).

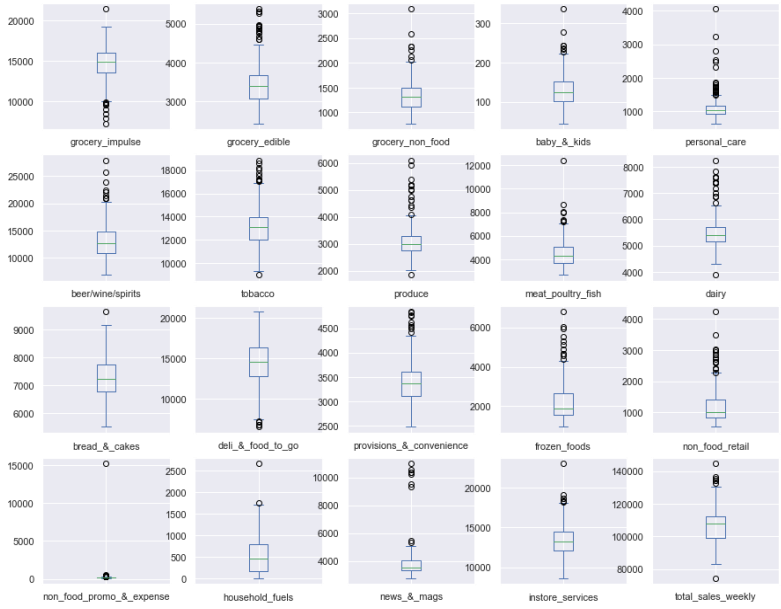

Figure 5: Boxplots of categorical sales

From the boxplots, it can be observed that Impulse – Grocery contributes a major part to the store's overall sales. The correlation coefficient matrix indicates the correlation coefficient values for different variables. The correlation coefficient indicates the strength of a relationship. Equation 1 helps calculate the correlation coefficient between two variables.

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\, n\Sigma x^2 - (\Sigma x)^2 \,][\, n\Sigma y^2 - (\Sigma y)^2 \,]}}$$

Equation 1: Correlation Coefficient

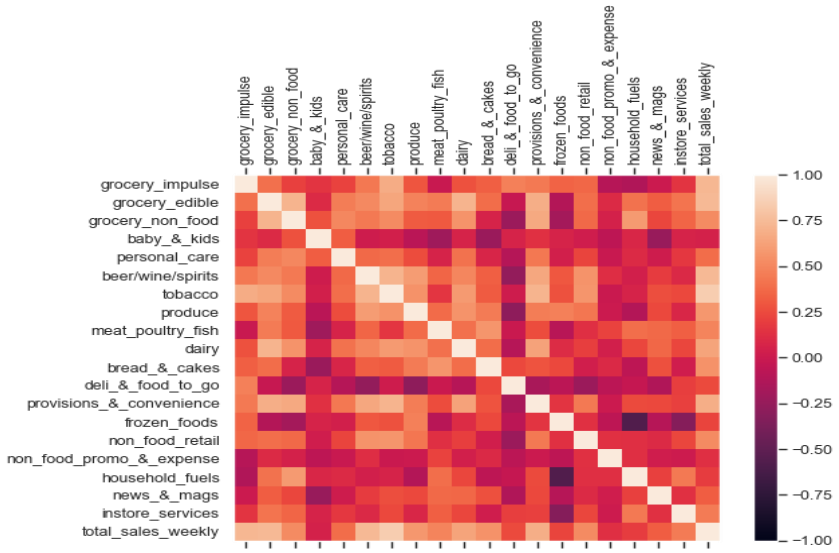

Figure 6: Correlation coefficient matrix of multivariate sales data

Figure 6 indicates the correlation coefficient matrix of the dataset. This plot is considered very powerful as it helps summarise large datasets with multiple variables. The darker the box is, the higher the correlation coefficient values are. From the above-obtained graph, it can be observed that the values 'frozen_food' and 'household_fuels' have the highest coefficients.

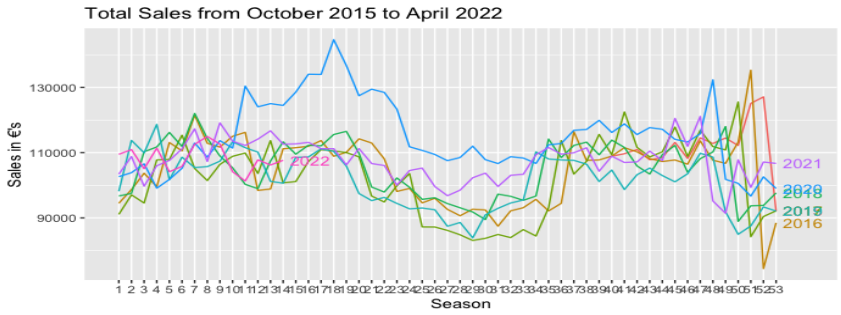

Figure 7: Year-wise sales represented in a line graph

Figure 7 shows that the sales were highest in 2020 during the pandemic. Similarly, it can be inferred that the average sales were high in 2021 compared to the overall sales.

3.3. Understanding a time-series

The time series data is plotted against its period, followed by data transformation and visualisation. To better understand the data, the seasonal decomposition of the original time series data is plotted. The seasonal_decompose( ) function separates the trend, seasonal and cyclical variations, and residual components. A time series is usually represented by equation 2.

$$y_t = f(S_t, T_t, R_t)$$

Equation 2: General equation of a time series

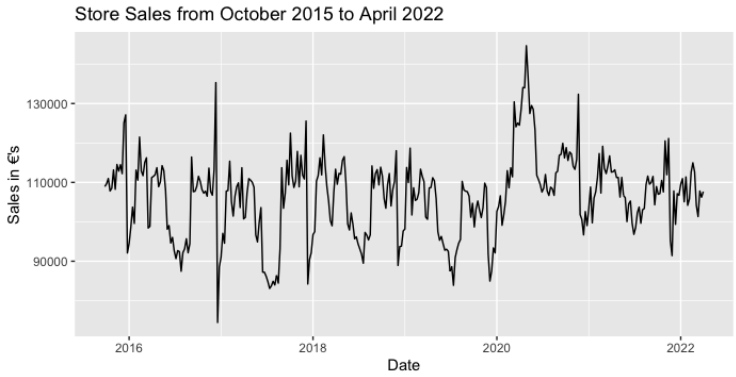

Figure 8: Weekly sales data from October 2015 to April 2022

There are no null values. As shown above figure 8, the average weekly sales of the store from 2016 to 2021 were € 106,292. The maximum weekly sales during this period were € 144,661, and the minimum weekly sales were € 74,465.

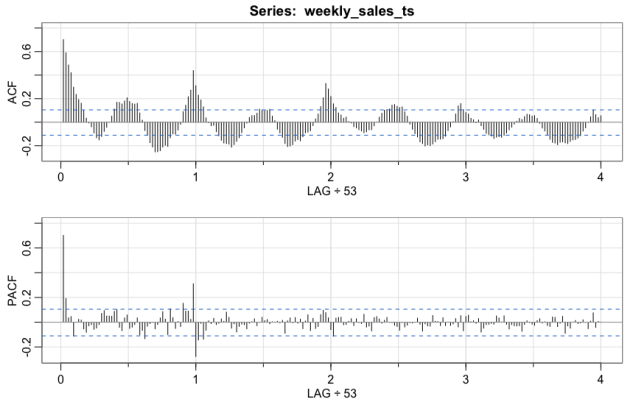

Figure 9: Autocorrelation plots for weekly sales

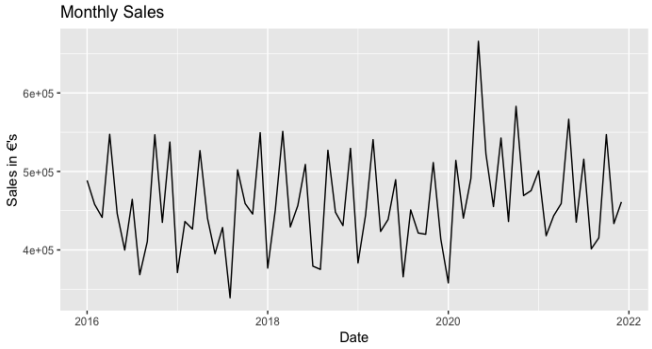

Figure 10: Monthly sales data from October 2015 to April 2022

From the above figures 9 and 10 monthly plot of the original data, it can be observed that there is a repeating pattern and a cyclical variation. The store's monthly sales were € 460,857, whereas, during peaks, they reached € 665,802.

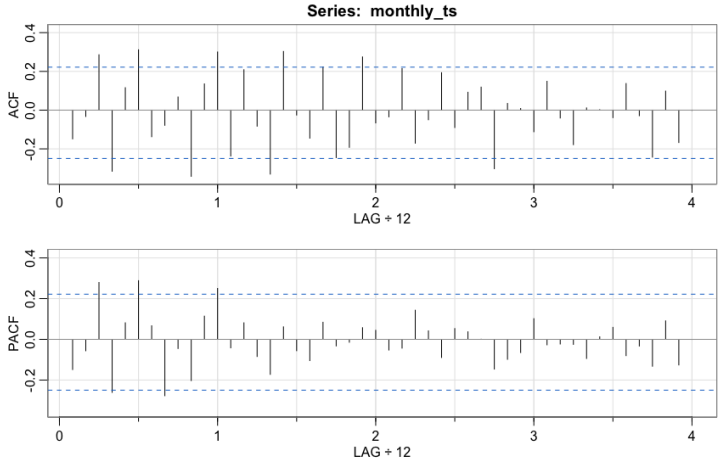

Figure 11: Autocorrelation plots for monthly sales

Interpreting the above seasonal decomposition, the trend is considered over a long time and can either be an upward/downward trend or constant. Here, from the above figure 11, it is observed that there is an increase in the trend, especially during the pandemic. Seasonal variation refers to the trend which occurs within a year. These seasonal changes are likely to repeat over the subsequent years. In the observed decomposition, a seasonality exists every year. There are peak sales during April and

December and low sales during summer. Usually, a wave-like movement is observed. This is caused by externally affecting features like the market demand, introduction of new products and offers, festivals, etc; apart from these components, there always exists some extreme values in the data, which are called outliers or residuals. These residuals build up the time series's residual variation component, which is used in ARIMA modelling.

3.4. Decomposition of the time series

Time series exhibits various patterns: trend, seasonality and cyclical variation. Composing a time series into its components improves the forecast accuracy. There are two forms of classical decomposition: additive decomposition and multiplicative decomposition. An additive decomposition subtracts the trend estimates from the time series. A multiplicative decomposition is done by dividing the series by the trend values. The additive model is useful when the seasonal variation is relatively constant. The multiplicative model is useful when the seasonal variation increases over time.

Additive Decomposition of Weekly Sales: The components are added together in additive time series. In a multiplicative time series, the individual components are multiplied together. The formula for each of the models is given as follows.

$$y_t = T_t + S_t + R_t$$

Equation 3: Additive time series formula

where,
y_t --- the data in period t
T_t --- trend component and cycle component at time t
S_t --- Seasonal component at time t
R_t --- Residual component at time t

The plot below shows the observed series, the trend line, the seasonal pattern and the random part of the time series. Here, the seasonal pattern is the regularly repeating pattern (fig.12).

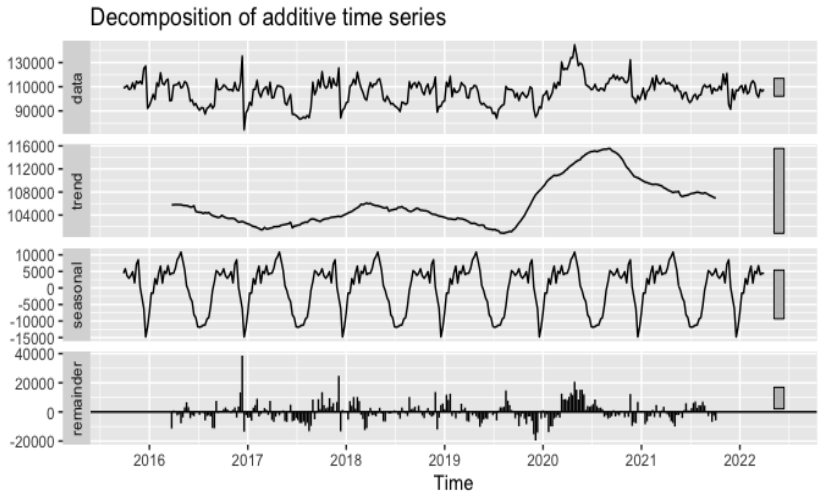

Figure 12: Additive decomposition of weekly sales

The first row is the actual time series, followed by its decomposition. The trend in the second line has a cyclical movement. Having ups and downs over a long time. In 2020, a dramatic increase was due to the pandemic. The squares at the right end of the lots describe the data's scale. The smaller the box is, the higher the scale of the data. If the square boxes are of similar size, they both have the same impact on the variability of the data. The residuals from the remainder part are the ones which lead to randomness. The smaller the box is, the less the randomness. When the seasonality is removed, a repeating pattern is observed every 12 months (fig.13).

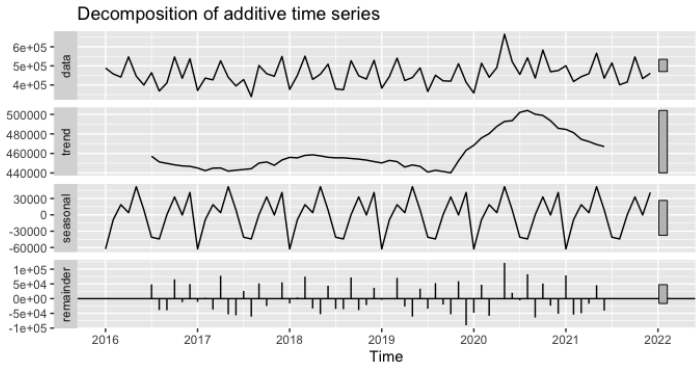

Figure 13: Additive decomposition of monthly sales

A similar pattern is observed for the monthly sales data.

Multiplicative Decomposition of Weekly Sales: The multiplicative decomposition argues that time series data is a function of the product of its components. It can usually be identified from its variation. If the magnitude of the seasonal component changes with time, then the time series is multiplicative. Here, the magnitude of the seasonal component grows as time increases.

$$y_t = T_t \times S_t \times R_t$$

Equation 4: Multiplicative time series formula

where,
y_t --- the data in period t
T_t --- trend component and cycle component at time t
S_t --- Seasonal component at time t
R_t --- Residual component at time t

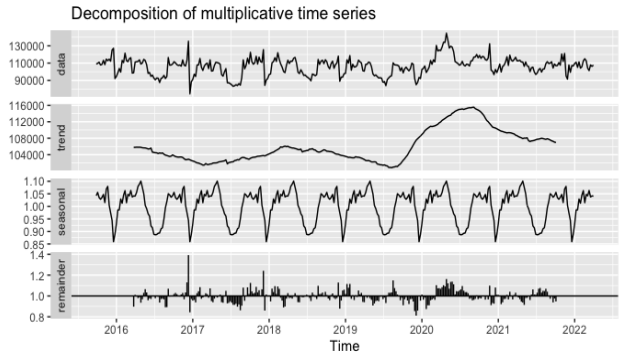

Figure 14: Multiplicative decomposition of weekly sales

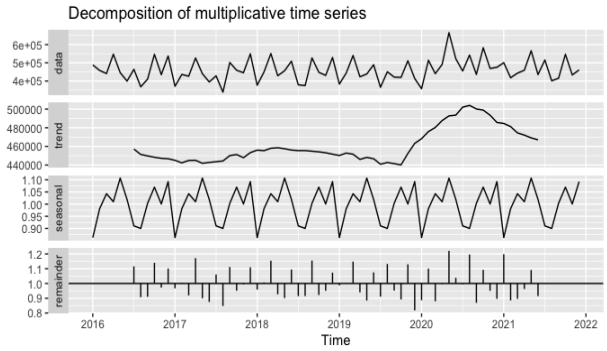

Figure 15: Multiplicative decomposition of monthly sales

From the above graph (Figures 14 and 15), it can be noticed long periods of apparent trend ups and downs are present. The time series is non-stationary to mean. Taking 1st difference may result in a stationary time series. Here, it can also be observed that the time series variance is not constant. To make it constant, logarithms are used to remove or stabilise the variance of the time series. The seasonal difference removes the peaks, and the ordinary difference removes the trend.

3.5. Statistical Methods

Some classical classical methods like HoltWinter, ARIMA, and SARIMA can be used to forecast historical data sales under traditional statistical forecasting methods. This analysis uses Holt-Winters' and ARIMA models to analyse and forecast sales.

3.5.1. Time Series Analysis using Holt-Winters' exponential smoothing:

Holt-Winters' exponential smoothing is one of the oldest methods of time series forecasting techniques that consider trend and seasonality while forecasting. This method has three major aspects for performing the predictions.

- Exponential Smoothing: Simple exponential smoothing – Forecasting when the data contains no trend or seasonality.
- Holt's Smoothing: Holt's exponential smoothing – Forecasting when the data has a trend but no seasonality.
- Holt-Winters' Smoothing: Holt-Winters' exponential smoothing – Forecasting technique includes seasonality and the trend.

Holt Winter's classical method is best suited for the Sales dataset Time series because the time series contains both trend and seasonality. This method forecasts future values by extending the simple exponential smoothing to capture the trend and seasonality of the time series using triple exponential smoothing.

3.5.2. Time Series Analysis using ARIMA: Auto-Regressive Integrated Moving Average

In the ARIMA model, a linear combination of historical values and errors is used for predicting future values. The ARIMA combines the AR, Integration, and MA parts. An ARIMA (p, d, q)(P, D, Q) model where p is the number of non-seasonal autoregressive terms (AR), q is the number of non-seasonal moving average terms (MA), and d is the degree of non-seasonal differencing and P, D, Q refers to the same for a seasonal component respectively.

Auto-Regression: The AR (p) model is given as follows:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

Equation 5: Equation for the Auto-regressive part of the ARIMA model

Integration: The I (d) is the number of differences that make the time series stationary.

Moving Average: The MA (q) model is given as follows:

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}.$$

Equation 6: Equation for the Moving-average part of the ARIMA model

Combining all three types of models above gives the resulting ARIMA (p,d,q) model.

ARIMA

$$\hat{y}'_t = c + \phi_1 y'_{t-1} + \phi_2 y'_{t-2} + \ldots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Equation 7: General equation of the ARIMA model

where,

- εt is defined as white noise or randomness.
- ϕp depends on the order p selected
- θq depends on the order q selected
- $c = (1 - \sum^p \phi) \cdot \mu$

Stationarity Tests: Stationarity describes how a particular time series variable changes over some time. Transformations are made on the original data to obtain stationarity. A time series is considered stationary when the below conditions are satisfied:

- Constant mean.
- Constant variance (Homoscedasticity).
- The autocorrelation pattern is constant throughout the time series.

ARIMA is based on the assumption that the data should be stationary and univariate. A time series is considered stationary if its mean, variance, and covariance are all invariant with time. The weekly sales data does not show a constant variance and constant mean but also some repeating patterns over time, making it different from white noise. Monthly may be considered a non-stationary process. However, transformations can always be done to remove the trend and seasonality on the original time

series data. These transformations include differentiating the time series to remove the trend and taking log transformations to stabilize the variance in the data.

Removing Trend and Seasonality: Time series datasets may contain trends and seasonality, which must be removed before further modelling. Trends can result in a varying mean over time, and seasonality can change variance over time; both define a time series as non-stationary. Stationary datasets are those that have a stable mean and stable variance. Differencing and log transformations on widely used data can make a time series stationary.

Stationary Time Series: The observations in stationary time series data are not dependent on time. Time series is stationary if they do not have a trend or seasonal effects. Summary statistics calculated on the time series are consistent over time, like the observations' mean or variance. When a time series is stationary, it can be easier to model. Statistical modelling methods assume or require the time series to be stationary.

Non-Stationary Time Series: Observations from a non-stationary time series show seasonal effects, trends, and other structures that depend on the time index. Summary statistics like the mean and variance change over time, providing a drift in the concepts a model may try to capture. Classical time series analysis and forecasting methods make non-stationary time series data stationary by identifying and removing trends and stationary effects. Differentiation and log transformations are not the only determinants of stationarity. Formal statistical tests include the Augmented Dickey-Fuller test and the Kwiatkowski–Phillips–Schmidt–Shin (KPSS) tests. These tests check for the stationarity of a series around a deterministic trend (fig. 16).

Part 1-Monthly Sales data

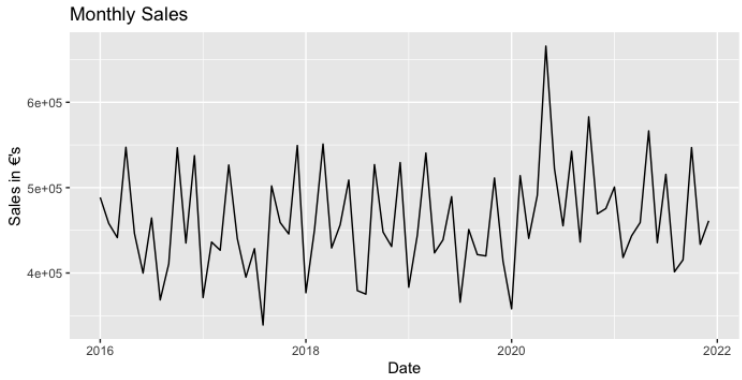

Figure 16: Monthly sales data from October 2015 to April 2022

Tests for Stationarity on Dickey-Fuller Test: To check the stationarity of the time series, an Augmented Dickey-Fuller test is performed. This can be imported from the statsmodels library using the adfuller( ) function.

```
> adf.test(monthly_ts, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  monthly_ts
Dickey-Fuller = -9.9116, Lag order = 0, p-value =
0.01
alternative hypothesis: stationary

Warning message:
In adf.test(monthly_ts, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 17: Code snap showing the ADF test for monthly time series

- The null hypothesis H_o states that the time series is not stationary.
- The alternative hypothesis H_a states that the time series is stationary.

Since the p-value (0.01) is less than the significance level (0.05), we have strong evidence to reject the null hypothesis and conclude that the time series is stationary. Figure 17 shows the results from the test for stationarity.

KPSS Test: The Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test determines if a time series is stationary around a mean or linear trend or is non-stationary due to a unit root. A stationary time series is one where statistical properties like the mean and variance are constant over time (fig.18).

- The null hypothesis H_o states that the time series is stationary with a trend.
- The alternative hypothesis H_a states that the time series is not stationary.

```
> kpss.test(monthly_ts)

        KPSS Test for Level Stationarity

data:  monthly_ts
KPSS Level = 0.28017, Truncation lag parameter = 3, p-value = 0.1

Warning message:
In kpss.test(monthly_ts) : p-value greater than printed p-value
```

Figure 18: Code snap showing KPSS test for monthly time-series

Since the p-value (0.01) is less than the significance level (0.05), we have strong evidence to reject the null hypothesis and conclude that the time series is not stationary. This contradicts the results obtained from Dicky Fuller test. Hence, the time series needs to be transformed to make it stationary. Manipulate the Time Series to make it stationary: Stationarity of the time series can be observed by looking at a line plot of the series over time. Signs of trends, seasonality or other random components in the series indicate a non-stationary series. A more accurate method would be a statistical test, such as the KPSS or Dickey-Fuller test. Differencing is one of the methods of transforming time series data. It can remove the time dependence on the data and help stabilize the time series mean by removing changes in the time series level and eliminating trends and seasonality. Differencing is performed by subtracting the previous observation from the current observation. Another such method is log transformation. Here, the difference between consecutive observations is taken. Log differences can be adjusted to suit the specific temporal structure. For a time, series with a seasonal component, the log may be expected to be the period (width) of the seasonality. Violation of stationarity creates estimation problems for ARIMA models. Below are the steps carried out to transform the time series data stationary to be fed into an ARIMA model.

Step 1: Remove the variability

Remove the variability of the time series by applying a log. A log transformation converts a multiplicate time series model into an additive one. The time series now displays constant variance, as shown below (fig.19).

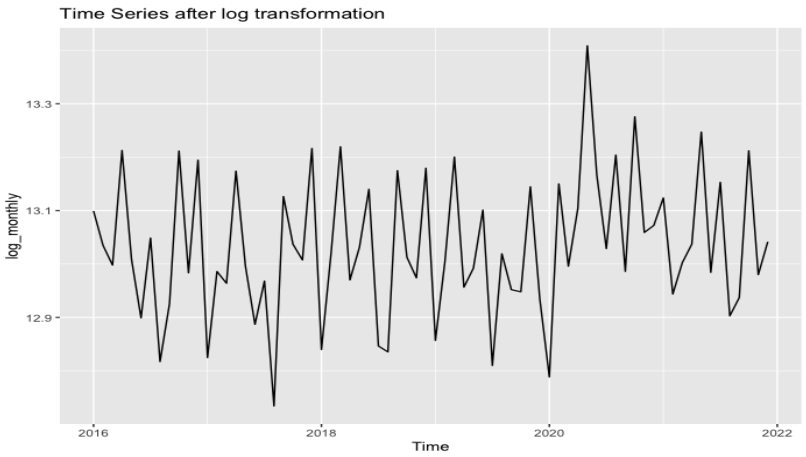

Figure 19: Time Series plot after log transformation

```
> adf.test(log_monthly, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  log_monthly
Dickey-Fuller = -9.9436, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(log_monthly, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 20: Code snap showing the ADF test after log transformation

```
> kpss.test(log_monthly)

        KPSS Test for Level Stationarity

data:  log_monthly
KPSS Level = 0.28658, Truncation lag parameter = 3, p-value = 0.1

Warning message:
In kpss.test(log_monthly) : p-value greater than printed p-value
```

Figure 21: Code snap showing the KPSS test for log-transformed data

The KPSS test confirms that the time series is not stationary with a level or trend by giving a p-value (0.1) greater than the significance level (0.05) (Figures 20 and 21). Hence, we fail to reject the null hypothesis and conclude that the time series is stationary (Figures 22 and 23).

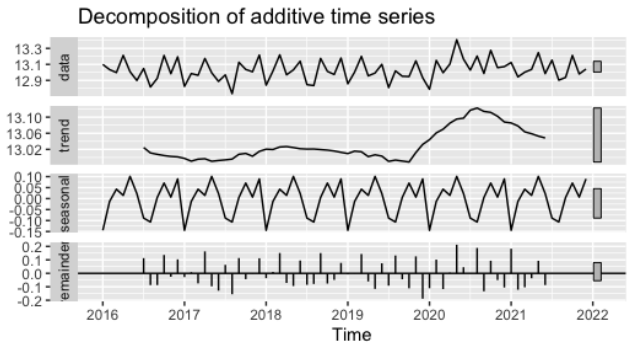

Figure 22: Additive decomposition of the log-transformed data

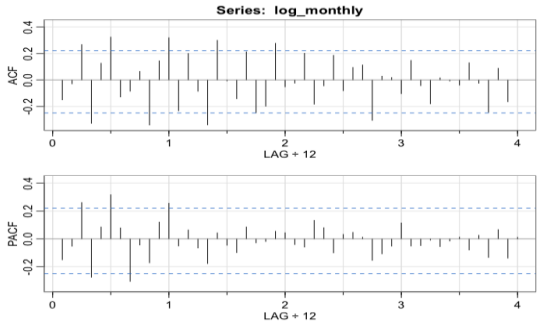

Figure 23: Autocorrelation plots for monthly sales after log transformation

Step 2: Seasonal Differencing

Remove the trend by differentiating the time series. Since we observe a linear trend, a 1st order differentiation is sufficient. If the trend is exponential, then a 2nd order differentiation is required. If the trend is cubic, then a 3rd order differentiation is required. As shown below, the trend has the least impact on the time series post-differentiation (figures 24 to 27).

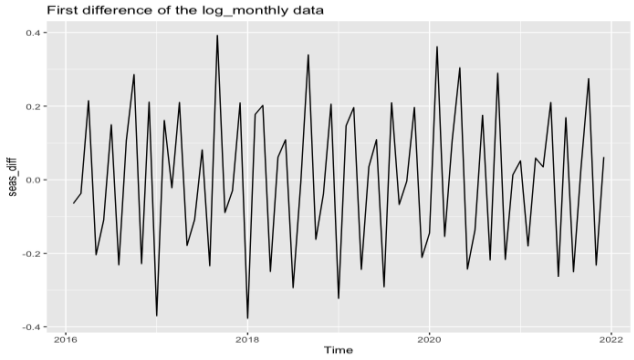

Figure 24: Time Series plot after differencing the log-transformed data

```
> adf.test(seas_diff, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  seas_diff
Dickey-Fuller = -15.248, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(seas_diff, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 25: Code snap of ADF test after the first difference on log_monthly data

```
> kpss.test(seas_diff)

        KPSS Test for Level Stationarity

data:  seas_diff
KPSS Level = 0.024286, Truncation lag parameter = 3, p-value = 0.1

Warning message:
In kpss.test(seas_diff) : p-value greater than printed p-value
```

Figure 26: Code snap of KPSS test after the first difference on log_monthly data

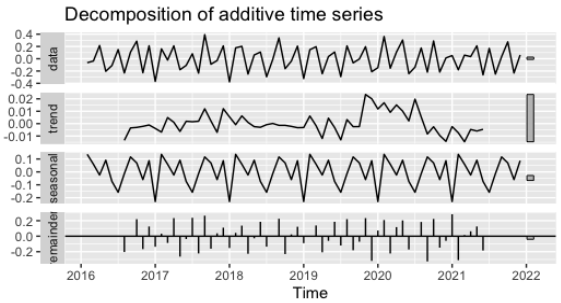

Figure 27: Additive decomposition of the differenced log-transformed data

The autocorrelation plot highlights that although most of the data are within the 95% limit, there is still a peak every four periods. This indicates periodicity in the time series; hence, the seasonality must be removed (fig.28).

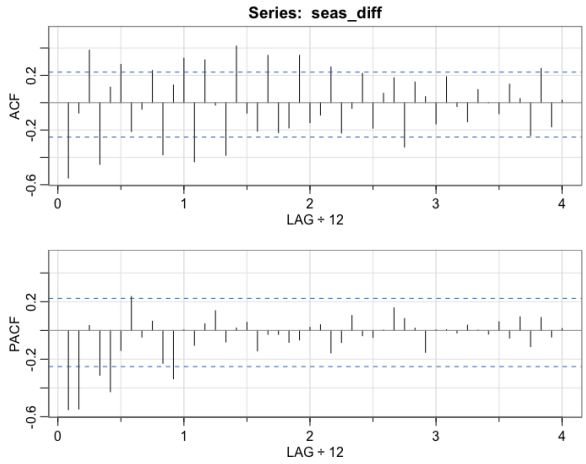

Figure 28: Autocorrelation plots after the first difference of log_monthly

Step 3: Remove the seasonality by applying differentiation. As shown below, the time series is now converted into white noise with mean 0 and constant variance (figures 29 to 31).

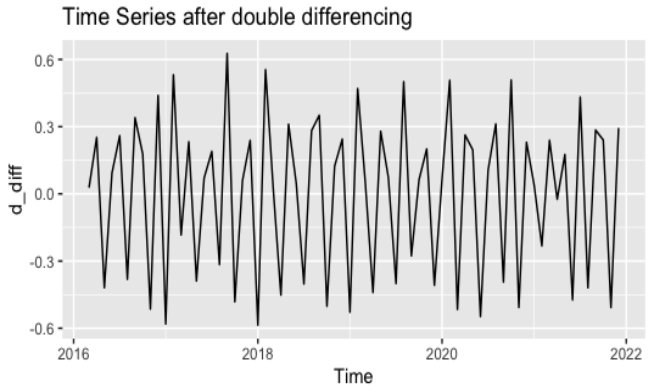

Figure 29: Time Series plot after double differencing the log-transformed data

```
> adf.test(d_diff, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  d_diff
Dickey-Fuller = -17.736, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(d_diff, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 30: Code snap of ADF test after the second difference on log_monthly

```
> kpss.test(d_diff)

        KPSS Test for Level Stationarity

data:  d_diff
KPSS Level = 0.02027, Truncation lag parameter = 3, p-value = 0.1

Warning message:
In kpss.test(d_diff) : p-value greater than printed p-value
```

Figure 31: Code snap of KPSS test after the second difference on log_monthly

The below decomposition plot highlights that trend and seasonality have been removed from the time series, and the data is ready for feeding into an ARIMA model (Figures 32 and 33).

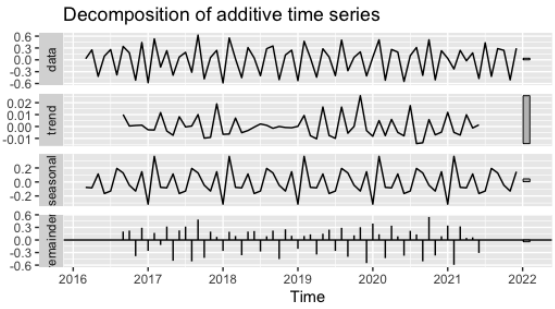

Figure 32: Additive decomposition of the double differenced log-transformed data

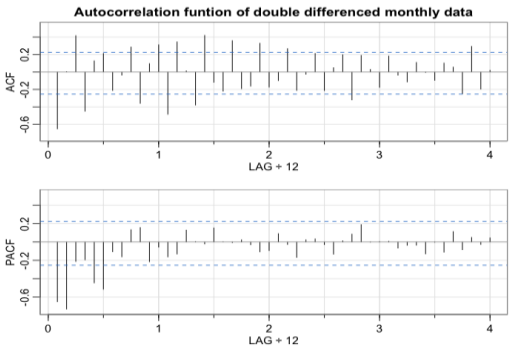

Figure 33: Autocorrelation plots after the second difference of log_monthly

The autocorrelation plot highlights that there is no periodicity, but a few bars are still outside the 95% limit (Figure 34).

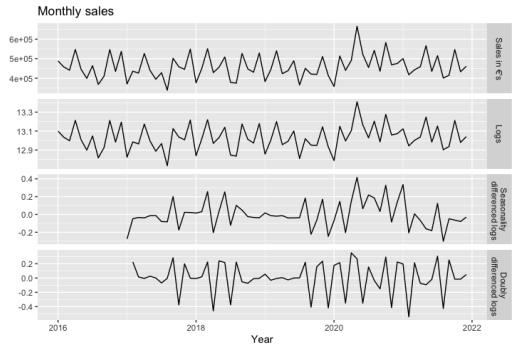

Figure 34: Comparison plot of the monthly sales

Figure 34 represents the different transformations carried to the monthly time series data. These transformations consist of a log transformation and a double differentiation to stabilize the variance in the original sales data. It can be observed that the trend is removed from all the data. Combining the log-transformed and double differenced monthly data, it became like white noise, as seen in the last row of Figure 34. After getting a stationary process, the next in ARIMA modelling is the identification of the model's p, d, and q parameters. This process requires the use of autocorrelation and partial autocorrelation functions. The ACF and PACF measure the level of dependence between observations in a time series in a set of lags. ACF helps identify which lags have significant correlations, and it also helps understand the patterns of the time series, which are of great help for time series modelling as ACF helps access the randomness and stationarity of a time series as well as the presence of trend and

seasonal patterns. The SARIMA model has been chosen to process this time series since it has a seasonal component. Many models were derived from this figure, and their training scores are summarized in the next table.

Table 1: ARIMA models for monthly data (frequency = 12)

	AIC	AICc	BIC
ARIMA(0,1,1,0,0,0,12)	-77.59	-77.41	-73.06
ARIMA(1,0,0,0,0,0,12)	-84.01	-83.41	-74.9
ARIMA(1,1,1,0,0,0,12)	-77.14	-76.54	-68.09
ARIMA(1,1,0,0,0,0,12)	-46.18	-45.82	-39.39
Auto.arima - ARIMA(0,0,0,0,0,1,12)	-91.97	-91.61	-85.14

The models are selected if their AIC and RMSE represent the minimum, and MAE is used to decide if the two models have similar RMSE and AIC. Table 1 shows the best ARIMA models for monthly periodic data. ARIMA (0,0,0,0,0,1,12), having the least AIC value of -91.97, outperforms the rest of the models. The auto.arima model was found to be the best. The Ljung Box test was performed on the mode. Following this, future predictions were made. A similar analysis of the weekly dataset was carried out as follows (Figures 35 to 52).

Part 2 – Weekly Sales data

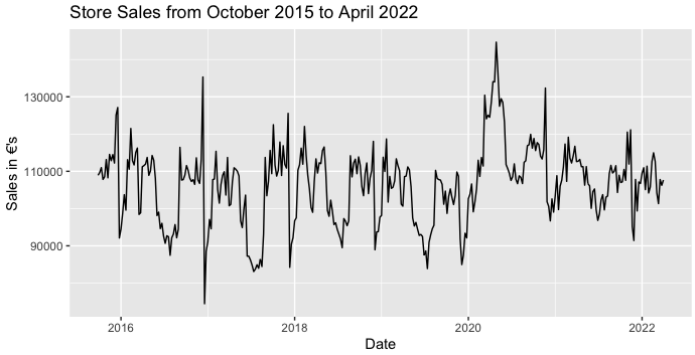

Figure 35: Weekly sales data from October 2015 to April 2022

Tests for Stationarity:

```
> adf.test(weekly_sales_ts, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  weekly_sales_ts
Dickey-Fuller = -7.8701, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(weekly_sales_ts, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 36: Code snap showing the ADF test for weekly time-series

```
> kpss.test(weekly_sales_ts)

        KPSS Test for Level Stationarity

data:  weekly_sales_ts
KPSS Level = 0.40716, Truncation lag parameter = 5, p-value = 0.07407
```

Figure 37: Code snap showing KPSS test for weekly time-series

Step 1: Remove the variability

Remove the variability of the time series by applying a log.

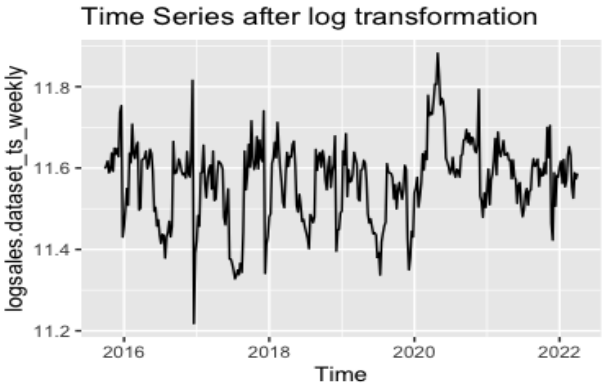

Figure 38: Time Series plot after log transformation

```
> adf.test(log_weekly, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  log_weekly
Dickey-Fuller = -7.801, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(log_weekly, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 39: Code snap showing the ADF test for log_weekly

```
> kpss.test(log_weekly)

        KPSS Test for Level Stationarity

data:  log_weekly
KPSS Level = 0.4224, Truncation lag parameter = 5, p-value = 0.0675
```

Figure 40: Code snap showing KPSS test for log_weekly

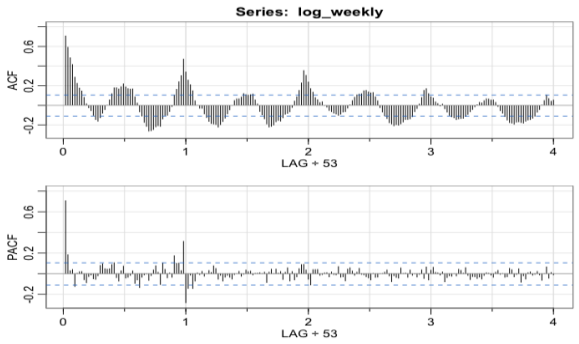

Figure 41: Autocorrelation plots for log_monthly

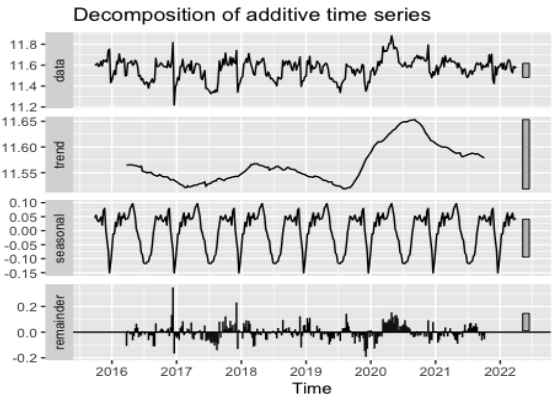

Figure 42: Additive decomposition of the log-transformed weekly data

Step 2: Seasonal Differencing

Remove the trend by differentiating the time series.

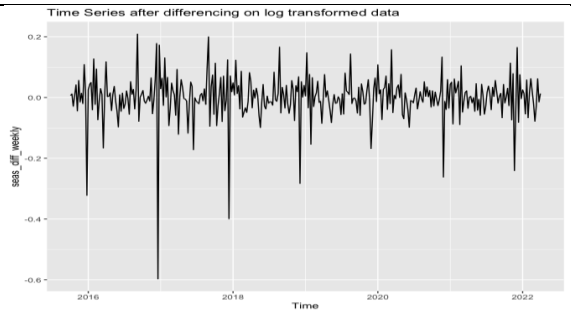

Figure 43: Differencing on log-transformed weekly data

```
> adf.test(seas_diff_weekly, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  seas_diff_weekly
Dickey-Fuller = -25.263, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(seas_diff_weekly, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 44: Code snap showing the ADF test for the first differenced log_weekly (Dickey-Fuller Test)

```
> kpss.test(seas_diff_weekly)

        KPSS Test for Level Stationarity

data:  seas_diff_weekly
KPSS Level = 0.012009, Truncation lag parameter = 5, p-value = 0.1

Warning message:
In kpss.test(seas_diff_weekly) : p-value greater than printed p-value
```

Figure 45: Code snap showing the KPSS test for the first differenced log_weekly

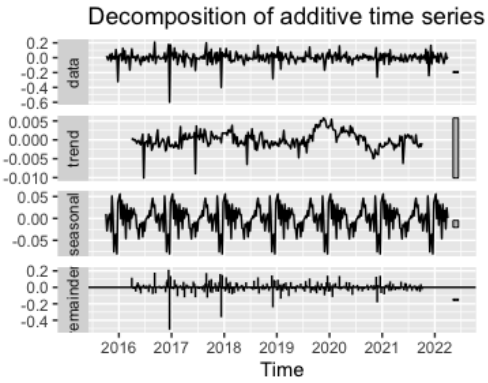

Figure 46: Additive decomposition of the above data

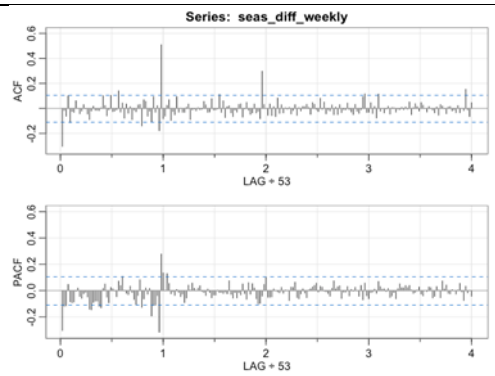

Figure 47: Autocorrelation plots for the first difference log_monthly

Step 3: Remove the seasonality by applying differentiation. Double differencing is done to the log-transformed data.

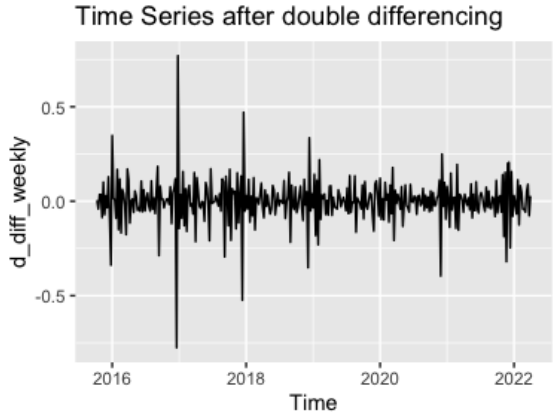

Figure 47: Time Series plot after double differencing the log-transformed data

```
> adf.test(d_diff_weekly, alternative = "stationary",k = 0)

        Augmented Dickey-Fuller Test

data:  d_diff_weekly
Dickey-Fuller = -37.485, Lag order = 0, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(d_diff_weekly, alternative = "stationary", k = 0) :
  p-value smaller than printed p-value
```

Figure 48: Code snap showing the ADF test for the second differenced log_weekly (Dickey-Fuller Test)

```
> kpss.test(d_diff_weekly)

        KPSS Test for Level Stationarity

data:  d_diff_weekly
KPSS Level = 0.0086577, Truncation lag parameter = 5, p-value = 0.1

Warning message:
In kpss.test(d_diff_weekly) : p-value greater than printed p-value
```

Figure 49: Code snap showing the KPSS test for the second differenced log_weekly

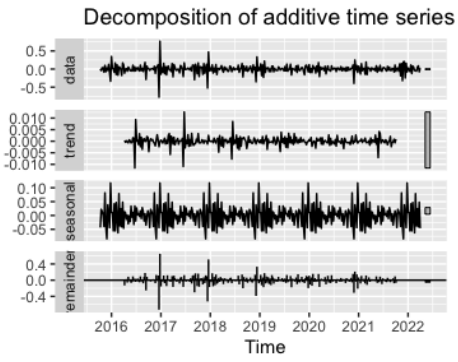

Figure 50: Decomposition after double differencing the log-transformed data

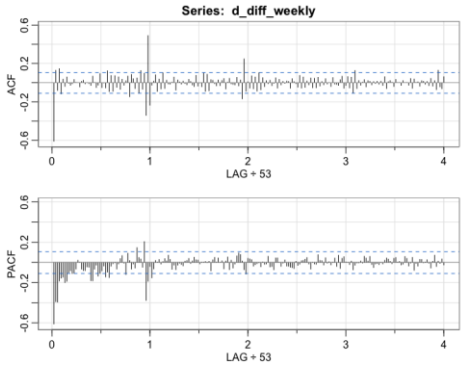

Figure 51: Autocorrelation plots for the second difference log_monthly

A comparison plot is given below in Figure 53 to better understand the transformed time series. At first, the log transformation is performed on the original weekly sales to remove variability. Following this, the log-transformed data is differentiated twice to eliminate the seasonal differencing. As a result of these transformations, a graph similar to white noise is obtained in the final row of Figure 53.

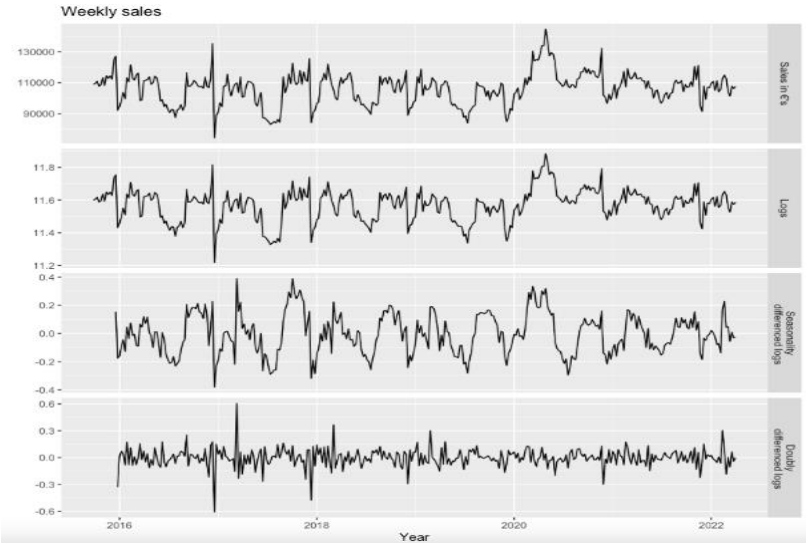

Figure 52: Comparison plot of the weekly sales

The above decomposition plot highlights that trend and seasonality have been removed from the time series, and the data is ready for feeding into an ARIMA model. ARIMA models describe the autocorrelation in the data.

Table 2: ARIMA models for monthly data (frequency = 53)

	AIC	AICc	BIC
ARIMA(0,1,1,0,0,0,53)	-844.49	-844.42	-832.96
ARIMA(1,0,0,0,0,0,53)	-860.82	-860.7	-845.44
ARIMA(1,1,1,0,0,0,53)	-844.17	-844.05	-828.8
ARIMA(3,1,0,0,0,1,53)	-844.16	-843.91	-821.1
Auto.arima - ARIMA(0,0,1,2,0,0,53)	-847.62	-847.51	-832.25

The models are selected if their AIC and RMSE represent the minimum, and MAE is used to decide if the two models have similar RMSE and AIC. Table 2 shows the best ARIMA models for weekly periodic data. ARIMA (1,0,0,0,0,0,53), having the least AIC value of -860.82, outperforms the rest of the models, including the auto.arima ARIMA(0,0,1,2,0,0,53) . The Ljung Box test was performed on the mode. Following this, future predictions were made.

3.6. Machine Learning Methods

3.6.1. Long Short-Term Memory

Long Short-Term Memory Networks or LSTMs can handle long-term dependencies that are a part of the Recurrent Neural Networks (RNN). LSTM is the widely used method among other machine learning models as it avoids the problem of long-term dependency. To know more about how LSTM works, refer to [13]. RNN is a particular form of artificial neural network that mainly performs on sequential data. Generally, when performing a time series-based analysis, the check for stationary is conducted. If the dataset is not stationary, it is first converted into a stationary dataset before performing the analysis. However, the advantage of the current neural networks is that they do not need stationary data. This is one of the major reasons for choosing LSTM over other classical models, as the dataset, in this case, is not stationary. These models can learn complex patterns of the data compared to the ARIMA and SARIMA models [14]. On the other hand, if the model cannot make good predictions, changing the dataset into a stationary form can be a good option for obtaining better results. The length of the dataset is analysed, and a split for the training and testing parts is done. The first 60 values are reserved in the training set. The last 12 months of analysis are reserved in the testing set and later will be used to compare the results with the predicted values. Using MinMaxScalar( ), the dataset is converted into a scale of 0 to 1. Converting this is important to avoid a difference in the magnitude of the data. Next, the scalar values fit into the training set and are converted to scalar training and test sets. The most challenging part of designing the LSTM model was to format the data exactly to provide input to the RNN model.

```
[8]  # splitting test and train 20:80
     train = df.iloc[:60]
     test = df.iloc[60:]

[9]  scaler.fit(train)
     scaled_train = scaler.transform(train)
     scaled_test = scaler.transform(test)

[10] scaled_train[:10]

     array([[0.63310491],
            [0.33582366],
            [0.42808304],
            [0.45759662],
            [0.36419868],
            [0.31326994],
            [0.63725817],
            [0.3280558 ],
            [0.18613477],
            [0.38373769]])

     # define generator
     input_n =  3
     features_n = 1
     generator = TimeseriesGenerator(scaled_train, scaled_train, length = input_n, batch_size= 1)
```

Figure 53: Code for scalar transformation and input transformation

This was solved using the library Timeseries Generator. It uses a sequence of the first 3 (n =3) values to predict the upcoming 4th value. The lines of codes in Figure 54 above explain them. Finally, an LSTM layer with 100 neurons and the activation function as “relu” is designed. To compare the predictions, a graph is plotted against the original values and for numerical comparison, the RMSE function from the SciKit learning libraries is used.

3.6.2. Simple Linear Regression

Linear Regression analysis defines a linear relationship between a dependent and independent variable. It also fits a straight line approximating the relationship between these variables. The mathematical equation for the simple linear regression is given as:

$$Y = a + bX + e$$

Equation 8: Simple Linear Regression formula

Y is the predicted value of the dependent variable y for every specified value of the independent variable X, a is the line intercept, b is the regression coefficient, e is the error, and X is an independent variable (fig.55).

```
[9] from sklearn.linear_model import LinearRegression
    LR_model = LinearRegression()

    x1,x2,x3,y=df['Sales_LastMonth'],df['Sales_2Months_back'], df['Sales_3Months_back'], df['SALES']
    x1,x2,x3,y=np.array(x1),np.array(x2),np.array(x3),np.array(y)
    x1,x2,x3,y=x1.reshape(-1,1),x2.reshape(-1,1),x3.reshape(-1,1),y.reshape(-1,1)
    x_final=np.concatenate((x1,x2,x3),axis=1)
    print(x_final)

    [[478939. 448795. 545926.]
     [488582. 478939. 448795.]
     [458066. 488582. 478939.]
     [441426. 458066. 488582.]
     [547283. 441426. 458066.]
     [446257. 547283. 441426.]
     [399887. 446257. 547283.]
     [464450. 399887. 446257.]
```

Figure 54: Codes from moving average calculation

Regression analysis helps the business organisation better understand their data points, making decision-making easier. In this study, analysing historical sales data can help us understand the store's top sellers and which sales areas need improvement. The data obtained is also linear, making linear regression possible. Sales are the dependent variable to be predicted based on the independent variable time. Initially, this analysis was performed with the help of the SciKit learning library, which was unsuccessful. Later, the same analysis was carried out in R using the 'tslm' function to fit a linear model with time series components. The model can fit linear models to time series, including trend and seasonality components. The moving average values were calculated to train the model. Once the model is trained and fitted, the predicted values are plotted against the actual sales to compare the performance. Similarly, the RMSE is also taken to measure the accuracy.

3.6.3. Random Forest Regression

The Random Forest algorithm is considered one of the easiest machine learning algorithms and the most widely used due to its uniqueness and simplicity of execution. The decision trees approach is carried out where each tree generated depends on the sampled random vector value. The forests considered the ensembles of the decision trees are trained using the bagging method. In the end, several decision trees are combined into a single tree. Random forest algorithms can handle overfitting, so they are much preferred over other models. While analysing high-dimensional datasets like the one in this case, the model uses random subsets of features, improving its performance (fig.56).

```
[10] from sklearn.ensemble import RandomForestRegressor
     RF_model = RandomForestRegressor(n_estimators=100, max_features=3, random_state=1)

[ ]  from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = x_final[:-30], x_final[-30:],y[:-30],y[-30:]

[ ]  RF_model.fit(X_train, y_train)
     LR_model.fit(X_train,y_train)
```

Figure 55: Fitting of the Random Forest model

Some reasons for selecting random forests over the other models were that the outliers do not influence this algorithm and, secondly, the noise in the dataset is identified properly. It is not included during the pattern recognition by the model. Once the model is fitted using the suitable libraries from SciKit learning, the performance of the model is evaluated using its accuracy, RMSE and the confusion matrix.

4. Results

A comparison of predicted sales of different models has been given below. The plots are results obtained from different models designed.

4.1. Forecasts using Holt-Winters' exponential smoothing

The first plot is the HW multiplicative, the second is the HW additive, and the third is the additive damped model. HW multiplicative model can catch up with the trend and seasonality much better than the additive model (fig.57).

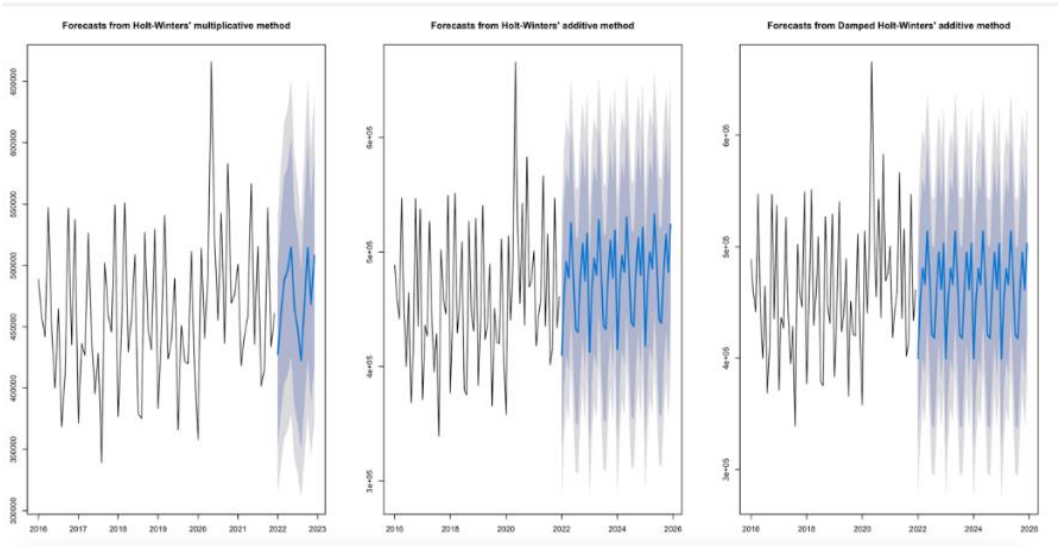

Figure 56: Forecasts from Holt-Winters' methods

As referred to in the table 3 below, the HW multiplicative model is the best for this time series since it has the lowest AIC, BIC, and RMSE.

Table 3: ARIMA models for monthly data (frequency = 53)

Model Name	Akaike Information Criterion (AIC)	Bayesian Information Criterion (BIC)	Root Mean Square Error (RMSE)
Holt-Winters' multiplicative method	1913.863	1952.567	55007.86
Holt-Winters' additive model	1915.333	1954.036	55630.13
Damped Holt-Winters' additive model	1916.213	1957.193	55199.04

4.2. Forecasts using ARIMA – monthly data

For the monthly sales time series – the auto.arima - ARIMA(0,0,0,0,0,1,12) was the best-fitted model. This model has given the lowest AIC of -91.97. For the above-stated ARIMA model, the residuals are normally distributed and have constant variance. Almost all the bars in the ACF plot are within the 95% confidence interval (Figures 58 to 63).

```
> #Auto-ARIMA
> auto.arma11_monthly = auto.arima(log(monthly_ts))
> auto.arma11_monthly
Series: log(monthly_ts)
ARIMA(0,0,0)(0,0,1)[12] with non-zero mean

Coefficients:
        sma1     mean
      0.4025  13.0322
s.e.  0.1175   0.0191

sigma^2 = 0.015:  log likelihood = 48.98
AIC=-91.97   AICc=-91.61   BIC=-85.14
```

Figure 57: Auto-ARIMA results for monthly sales

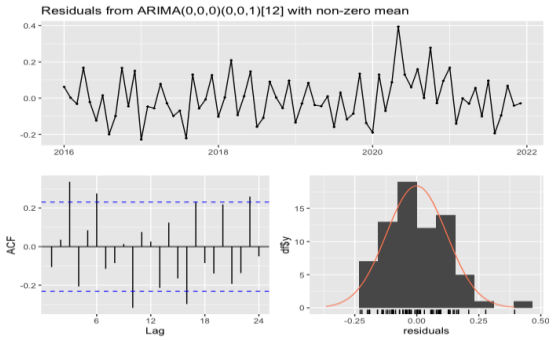

Figure 58: ARIMA residuals - Monthly data, p = 12

The Ljung-box test confirms that the residuals of the selected model are independent.

```
> checkresiduals(auto.arma11_monthly)

        Ljung-Box test

data:  Residuals from ARIMA(0,0,0)(0,0,1)[12] with non-zero mean
Q* = 36.205, df = 12, p-value = 0.0003003

Model df: 2.   Total lags used: 14
```

Figure 59: Ljung-Box test for monthly data

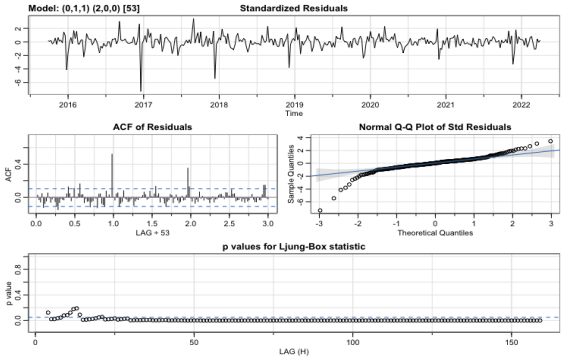

Figure 60: Residuals plot from ARIMA – monthly

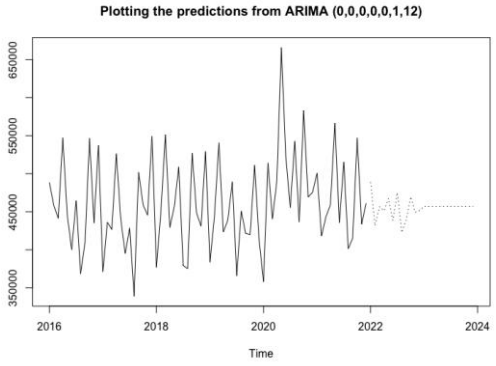

Figure 61: Predictions from ARIMA – monthly

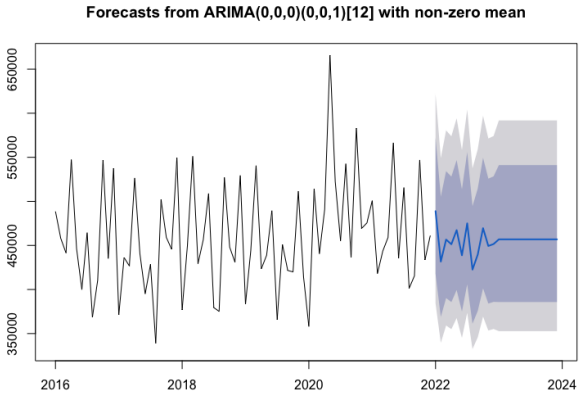

Figure 62: Forecasts from ARIMA – monthly

4.3. Forecasts using ARIMA – weekly data

For the weekly sales time series – the auto.arima - ARIMA(1,0,0,0,0,0,53) was the best-fitted model. This model has given the lowest AIC of -91.97. For the above-stated ARIMA model, the residuals are normally distributed and have constant variance (Figures 64 to 75). Almost all the bars in the ACF plot are within the 95% confidence interval (Table 4).

```
> # ARIMA models
> #AR model
> ar.mod_weekly = Arima(log(weekly_sales_ts), order=c(1, 0, 0),
+                       seasonal = c(0, 0, 0),
+                       include.drift = TRUE)
> ar.mod_weekly
Series: log(weekly_sales_ts)
ARIMA(1,0,0) with drift

Coefficients:
         ar1  intercept  drift
      0.6971    11.5434  2e-04
s.e.  0.0383     0.0241  1e-04

sigma^2 = 0.004786:  log likelihood = 434.41
AIC=-860.82   AICc=-860.7   BIC=-845.44
```

Figure 63: Auto-ARIMA results for weekly sales

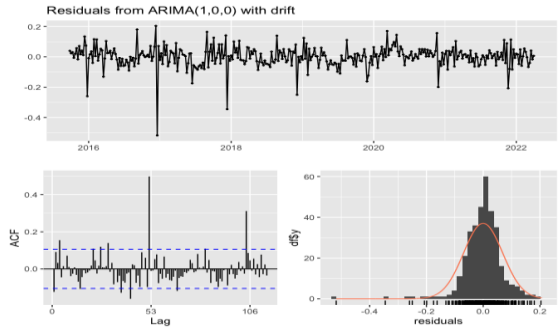

Figure 64: ARIMA residuals - Weekly data, p = 53

Equation of the ARIMA (1,0,0,0,0,1,53) : $y_{t = 11.5434 + 0.6971 * e_{t-1}}$

The Ljung-box test confirms that the residuals of the selected model are independent.

```
> checkresiduals(ar.mod_weekly)

        Ljung-Box test

data:  Residuals from ARIMA(1,0,0) with drift
Q* = 212.76, df = 66, p-value < 2.2e-16

Model df: 3.   Total lags used: 69
```

Figure 65: Ljung-Box test for weekly data

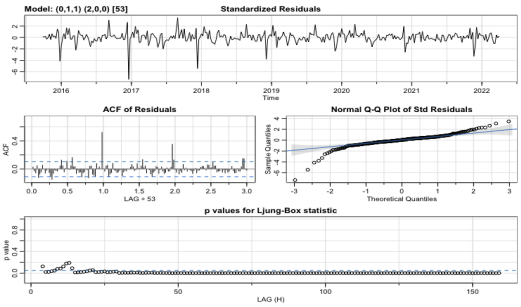

Figure 66: Residual plot of weekly data

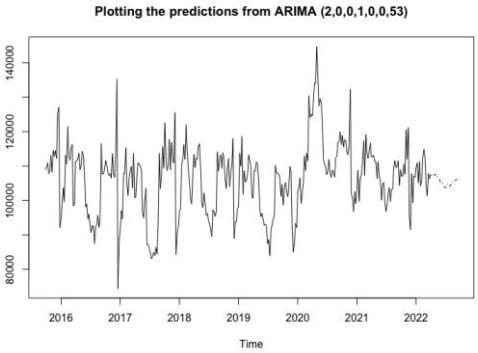

Figure 67: Predictions from ARIMA – weekly

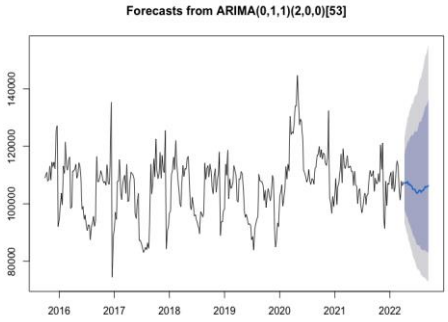

Figure 68: Forecasts from ARIMA – weekly

4.4. Forecasts using LSTM

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100)               40800

 dense (Dense)               (None, 1)                 101

=================================================================
Total params: 40,901
Trainable params: 40,901
Non-trainable params: 0
_________________________________________________________________
```

Figure 69: Results from the LSTM model

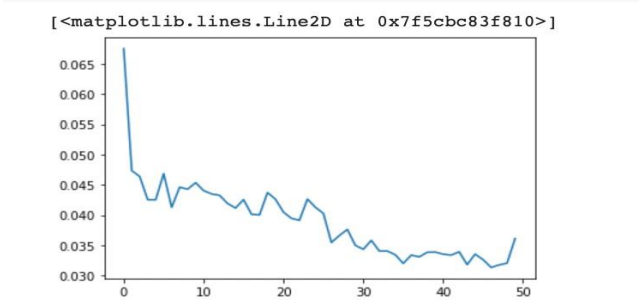

Figure 70: Plot of the losses per epoch

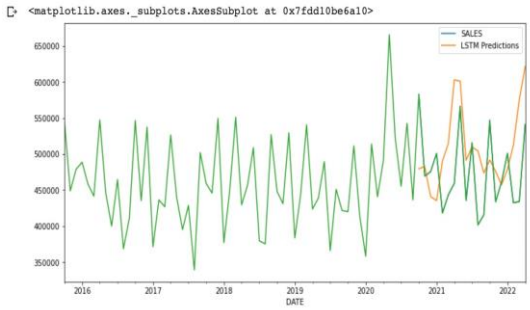

Figure 71: Time series plot of actual sales and the LSTM Predicted sales

4.5. Forecasts using Linear Regression

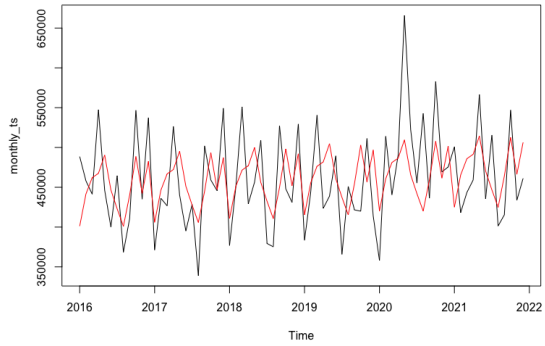

Figure 72: Sales predictions from the Linear model – monthly

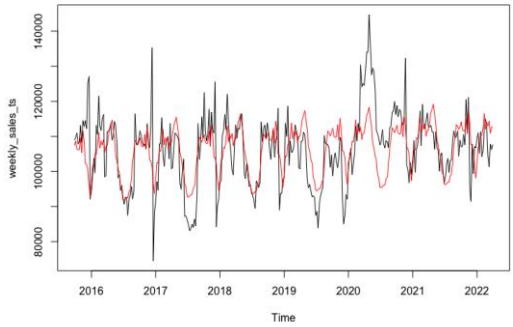

Figure 73: Sales predictions from the Linear model – weekly

4.6. Forecasts using Random Forest

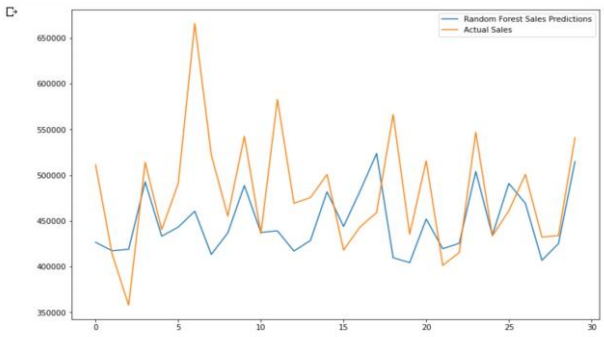

Figure 74: Sales predictions from the Random Forest model

4.7. Performance Summary of the different models

Table 4: Comparison of the algorithms' performances based on RMSE

Models	RMSE values
LSTM	61260.98
Random Forest Regressor	67903.81

5. Conclusion

The objective of this thesis was to compare the different statistical and machine learning approaches in predicting a retail store's sales based on the historical data collected. The research carried out leads to a major added advantage to the businesses. It helps in a better decision-making process where each department of good sales is tracked and helps monitor where an improvement is required. For a business to compete in the current market, it is necessary to make such sales predictions in advance to meet the customers' requirements. The RMSE values were used for the comparison of the final performances of the model. All three different models proved effective in terms of the predicted sales outputs. LSTM model proved to be the closest among the predictions. The lower the RMSE, the better the predictions could "fit" within the dataset. The same study can be conducted for the weekly periodic dataset for future work. Some other works may include more complex deep learning models and hybrid algorithms to compare with the existing models. One major limitation of this study was the computational power to run the Python codes. Python modules were processed in the Google Collaboration environment over the Jupyter notebooks to overcome this.

Appendix:

For monthly data, another best-fit model is as follows (Figures 75 and 76). ARIMA (3,1,0,0,0,1,12) has another lowest AIC of -84.11. All the lags of the ACF were within the 95% C.I.

```
> #model4
> arima310_monthly = Arima(log(monthly_ts), order=c(3, 1, 0),
+                          seasonal = c(0, 0, 1),
+                          include.drift = TRUE)
> arima310_monthly
Series: log(monthly_ts)
ARIMA(3,1,0)(0,0,1)[12] with drift

Coefficients:
          ar1      ar2      ar3    sma1    drift
      -0.8997  -0.5972  -0.0349  0.5678  -0.0021
s.e.   0.1191   0.1429   0.1205  0.1285   0.0083

sigma^2 = 0.01498:  log likelihood = 48.06
AIC=-84.11   AICc=-82.8   BIC=-70.54
> checkresiduals(arima310_monthly)

        Ljung-Box test

data:  Residuals from ARIMA(3,1,0)(0,0,1)[12] with drift
Q* = 25.419, df = 9, p-value = 0.002541

Model df: 5.   Total lags used: 14
```

Figure 75: ARIMA results for another fit model - monthly data

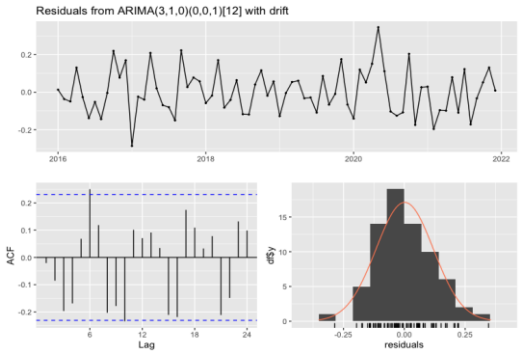

Figure 76: ARIMA residuals - Monthly data, p = 12

Acknowledgement: Our friends who patiently supported us during the research are likewise appreciated.

Data Availability Statement: This study's data, graphics, and code are accessible from the associated author upon reasonable request.

Funding Statement: No funding was received to conduct the research.

Conflicts of Interest Statement: Authors collaborate on this work and agree on its points, issues, and discoveries.

Ethics and Consent Statement: All authors agree to make this paper available for reading, teaching, and learning.

References

1. A. S. Denney and R. Tewksbury, "How to write a literature review," J. Crim. Justice Educ., vol. 24, no. 2, pp. 218–234, 2013.
2. K. Hamlett, "Statistical methods of sales forecasting," Small Business - Chron.com, 05-Jul-2010. [Online]. Available: https://smallbusiness.chron.com/statistical-methods-sales-forecasting-4692.html. [Accessed: 26-Nov-2022].
3. Linkedin.com. [Online]. Available: https://www.linkedin.com/pulse/statistical-methods-sales-forecasting-retail-industryajay-bidyarthy. [Accessed: 26-Nov-2022].
4. K. Ofoegbu, A Comparative Analysis of Four Machine Learning Algorithms to Predict Product Sales for a Retail Store". MSc in Data Analytics, Dublin Business School, pp.1-64, 2021.
5. N. Liu, S. Ren, T.-M. Choi, C.-L. Hui, and S.-F. Ng, "Sales forecasting for fashion retailing service industry: A review," Math. Probl. Eng., vol. 2013, pp. 1–9, 2013.
6. "Topics," Ibm.com. [Online]. Available: https://www.ibm.com/cloud/learn/what-isartificial-intelligence. [Accessed: 26-Nov-2022].

7. A. Krishna, Akhilesh, A. Aich, and C. Hegde, “Sales-forecasting of retail stores using machine learning techniques,” in 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), 2018.
8. J. Wang, G. Q. Liu, and L. Liu, “A selection of advanced technologies for demand forecasting in the retail industry,” in 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA), 2019.
9. M. A. I. Arif, S. I. Sany, F. I. Nahin, and A. S. A. Rabby, “Comparison study: Product demand forecasting with machine learning for shop,” in 2019 8th International Conference System Modeling and Advancement in Research Trends (SMART), 2019.
10. J. Chappelow, “Statistics in Math: Definition, Types, and Importance,” Investopedia, 30-May-2007. [Online]. Available: https://www.investopedia.com/terms/s/statistics.asp. [Accessed: 26-Nov-2022].
11. D. Richardson, “What is AI/ML and why does it matter to your business?,” Redhat.com, 04-Nov-2021.
12. R. J. Hyndman and G. Athanasopoulos, Forecasting: Principles and practice, 3rd ed. Australia: OTexts, 2021.
13. P.-F. Pai and C.-S. Lin, “A hybrid ARIMA and support vector machines model in stock price forecasting,” Omega, vol. 33, no. 6, pp. 497–505, 2005.
14. J. S. Armstrong and F. Collopy, “Integration of statistical methods and judgment for time series forecasting: principles from empirical research,” in Forecasting with Judgment, P. Wright, Ed. John Wiley & Sons Ltd, pp. 269–293, 1998.