Profit Prediction Using ARIMA, SARIMA and LSTM Models in Time Series Forecasting: A Comparison

A Mini Project report submitted in partial fulfillment of the requirements for the award of the degree of

Integrated M.Sc. in Computer Science
with Specialization in
Artificial Intelligence and Machine Learning

Submitted by

Rithik V
(NA21PICS23)

Nehru Arts and Science College Kanhangad
Padnekkad P.O., Kasaragod Dt., Kerala - 671314

Affiliated to

Kannur University
Kannur

October 2025

Nehru Arts and Science College Kanhangad

Padnekkad P.O., Kasaragod Dt., Kerala - 671314

CERTIFICATE

This is to certify that report entitled "**Profit Prediction Using ARIMA, SARIMA and LSTM Models in Time Series Forecasting: A Comparison**" is a bonafide report of the Mini Project (9B45ICSC: Lab-14: Mini Project) presented during IXth semester by **Rithik V** with Register No. **NA21PICS23** in partial fulfillment of the requirements for the award of the degree of Integrated M.Sc. in Computer Science with Specialization in Artificial Intelligence and Machine Learning.

Faculty In Charge | Head of the Department

External Examiner | Internal Examiner

DECLARATION

I, **Rithik V**, IX Semester Integrated M.Sc. in Computer Science with Specialization in Artificial Intelligence and Machine Learning Student of Nehru Arts and Science College Kanhangad under Kannur University do hereby declare that the mini project entitled "**Profit Prediction Using ARIMA, SARIMA and LSTM Models in Time Series Forecasting: A Comparison**" is original work carried out by me towards the partial fulfillment of the requirement of Integrated M.Sc. in Computer Science with Specialization in Artificial Intelligence and Machine Learning, and no part thereof has been presented for the award of any other degree.

Rithik V

ACKNOWLEDGMENT

I would like to express my sincere gratitude to the faculty members of the Department of Computer Science for their constant support, guidance, and encouragement throughout the completion of this project. I am especially thankful to my project guide for providing valuable insights, timely suggestions, and continuous motivation, which greatly contributed to the successful development of this work. I also thank all the teaching and non-teaching staff for creating a supportive academic environment that enabled me to complete this project effectively. Their dedication and commitment have been truly inspiring, and I am deeply grateful for their contributions.

Rithik V

ABSTRACT

This project presents a comparative study of three time-series forecasting models—ARIMA, SARIMA, and LSTM—for predicting monthly profit using historical sales data. The dataset undergoes extensive preprocessing, including missing value handling, anomaly correction, temporal aggregation, and stationarity transformation. Stationarity is validated using the Augmented Dickey–Fuller (ADF) and KPSS tests. ARIMA and SARIMA models are developed based on insights from ACF and PACF plots, while the LSTM model is trained using a sliding-window sequence generation approach on normalized data. Multiple evaluation metrics—including RMSE, MAE, MAPE, MPE, Min–Max Error, and Accuracy—are used to compare model performance. The experimental results reveal that the SARIMA model achieves the highest overall accuracy (98.69among the three models. ARIMA performs moderately well, while the LSTM model, although effective at learning nonlinear temporal patterns, performs slightly weaker than SARIMA and ARIMA for this dataset. This study demonstrates that traditional statistical models with seasonal components can outperform deep learning approaches in stable and strongly seasonal financial time-series.

Contents

Chapter 1

Introduction

Forecasting profit is an essential task for businesses to support planning, budgeting, and decision-making. Time-series models help predict future values based on historical data. Traditional statistical methods like ARIMA and SARIMA are effective for modeling linear and seasonal patterns, while modern deep learning models such as LSTM can capture complex and nonlinear trends in the data. This project compares ARIMA, SARIMA, and LSTM models for monthly profit prediction. The dataset is preprocessed, analyzed for stationarity, and transformed appropriately before model development. Each model's performance is evaluated using multiple error metrics, and the results show that the **SARIMA model provides the most accurate and reliable profit forecasts** among the three for the selected dataset, while ARIMA and LSTM also provide reasonably good predictive performance.

1.1 Background of the Study

Businesses today rely heavily on data-driven decision-making to remain competitive in rapidly changing markets. Among the many analytical tools used in industry, time-series forecasting plays a vital role in understanding patterns, planning future operations, and predicting financial performance. Profit prediction, in particular, helps organizations allocate resources efficiently, optimize investments, and evaluate long-term strategies.

Traditional statistical models like ARIMA and SARIMA have been widely used for forecasting because of their strong theoretical foundation and ability to model linear relationships with trend and seasonality. However, as business datasets have become more complex, these models may struggle to capture nonlinear patterns and long-term dependencies present in real-world financial data.

Parallel to this, advancements in machine learning and deep learning—especially the development of Long Short-Term Memory (LSTM) networks—have opened new opportunities for more accurate forecasting, particularly when larger and richer datasets are available. LSTM models are designed to handle sequential data and retain long-term information, making them suitable for complex time-series tasks such as profit prediction.

This study develops and compares three forecasting models—ARIMA, SARIMA, and LSTM—using a real-world profit dataset. By analyzing the strengths and limitations of each model, the study aims to identify the most effective approach for accurate monthly profit forecasting on this dataset and to demonstrate how model performance depends strongly on the nature of the data, the presence of seasonality, and the choice of preprocessing methods.

1.2 Problem Statement

Accurately forecasting monthly profit is a challenging task due to the presence of trends, seasonal variations, data anomalies, and nonlinear patterns in real-world financial data. Traditional time-series models like ARIMA and SARIMA are commonly used for forecasting, but they may fail to capture all complex dependencies and irregular fluctuations present in business datasets. With increasing data volume and changing market conditions, organizations require reliable and robust forecasting methods.

The problem addressed in this study is to determine which forecasting model—ARIMA, SARIMA, or LSTM—provides the most accurate and consistent prediction of monthly profit based on historical sales data. This involves handling non-stationarity, correcting anomalies, transforming the data appropriately, and evaluating different models using multiple error metrics. Identifying the most effective forecasting approach is significant because accurate profit prediction directly influences strategic planning, budgeting, inventory management, and overall business decision-making.

1.3 Objectives

- To preprocess and transform the historical sales data by handling missing values, correcting anomalies, and converting daily profit values into a monthly time-series format.
- To evaluate the stationarity of the profit time-series using statistical tests such as the Augmented Dickey–Fuller (ADF) test and the KPSS test.
- To develop three forecasting models—ARIMA, SARIMA, and LSTM—based on the characteristics of the time-series data.
- To analyze ACF and PACF plots to determine optimal model parameters for ARIMA and SARIMA.
- To train the LSTM model using scaled sequential data and assess its ability to capture nonlinear temporal patterns.
- To compare the performance of ARIMA, SARIMA, and LSTM using metrics such as RMSE, MAE, MAPE, MPE, Min–Max Error, and Accuracy.
- To identify the most accurate and reliable forecasting model for monthly profit prediction on the given dataset.
- To provide insights that can support better financial planning and decision-making based on the results of the comparative study.

1.4 Organization of the Report

- **Chapter 1** presents the introduction, background, problem statement, objectives, and organization of the report.

- **Chapter 2** provides a literature review, including an overview of the selected research paper and related works.
- **Chapter 3** discusses the methodology, including system understanding, mathematical formulation, technology stack, dataset description, preprocessing, model implementation, training, and evaluation setup.
- **Chapter 4** presents the experimental setup, results obtained, five-year forecasting feature, comparison with the original paper, and detailed observations and analysis.
- **Chapter 5** concludes the work and outlines possible future enhancements and extensions.

Chapter 2

Literature Review

2.1 Introduction

Time-series forecasting has been widely studied across fields such as economics, finance, supply chain management, and data science. Researchers have developed various statistical and machine learning models to capture patterns in sequential data and improve prediction accuracy. Traditional linear models like ARIMA and SARIMA have been the foundation of time-series forecasting for several decades, offering reliable results when the data exhibits clear trend and seasonal components. These models rely on stationarity assumptions and analyze autocorrelation structures to generate forecasts.

With advancements in computational power and deep learning, newer approaches such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have gained attention. These models are capable of learning long-term dependencies and nonlinear relationships that conventional statistical models cannot capture. Recent studies have shown that LSTM-based models can outperform classical time-series techniques under suitable data conditions, especially when dealing with complex, noisy, or highly nonlinear datasets.

2.2 Overview of the Selected Research Paper

The selected research paper titled "Profit Prediction Using ARIMA, SARIMA and LSTM Models in Time Series Forecasting: A Comparison" is authored by Uppala Meena Sirisha, Manjula C. Belavagi, and Girija Attigeri from the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India. The corresponding author is Manjula C. Belavagi (manjula.cb@manipal.edu). The paper was received on 21 October 2022, accepted on 21 November 2022, published on 28 November 2022, and the current version was released on 1 December 2022.

The paper presents a detailed comparative study of three forecasting models—ARIMA, SARIMA, and LSTM—to predict monthly gross profit using historical time-series data. The authors explain the complete workflow, including data preprocessing, grouping of order dates, handling missing values, and preparing the dataset for analysis. Stationarity of the series is examined using the Augmented Dickey–Fuller (ADF) and KPSS tests, followed by transformations such as logarithmic scaling and differencing to stabilize the data. The ARIMA and SARIMA models are constructed using insights from ACF and PACF plots, while the LSTM model is built using sequential input features generated with a sliding window.

Performance evaluation is carried out using RMSE, MAE, MAPE, and accuracy metrics. The study concludes that the LSTM model provides the highest accuracy and lowest error among the three models on the dataset used in the paper, demonstrating its superior ability to capture nonlinear patterns and long-term dependencies in profit time-series data. The authors highlight the importance of combining traditional statistical approaches with modern deep learning methods for reliable financial forecasting.

2.3 Related Works

Several studies have explored time-series forecasting using statistical and machine learning techniques. Early research primarily focused on traditional models such as ARIMA, which assumes linearity and stationarity in the data. Box–Jenkins ARIMA modeling has been widely adopted for time-dependent data, but its emphasis on linear structures limits its ability to represent complex seasonal and nonlinear behavior.

Subsequent work extended ARIMA to SARIMA by incorporating explicit seasonal components. These models demonstrated improved accuracy for datasets with strong seasonal patterns; however, SARIMA still struggles to capture arbitrary nonlinear relationships and long-term dependencies that may arise in real-world financial data.

In the field of deep learning, Hochreiter and Schmidhuber introduced the Long Short-Term Memory (LSTM) network, which has become a standard model for sequential data due to its ability to store long-term information and mitigate the vanishing gradient problem. Numerous studies have applied LSTM networks to stock price prediction, energy demand forecasting, and sales forecasting, often reporting better performance than classical statistical models when sufficient data and relevant features are available.

More recent works have explored hybrid approaches that combine statistical and neural models, such as ARIMA–LSTM or SARIMA–LSTM hybrids. These studies often aim to leverage the strengths of both worlds: the interpretability and robustness of statistical models and the flexibility of deep learning models.

Compared to these studies, the selected research paper "Profit Prediction Using ARIMA, SARIMA and LSTM Models in Time Series Forecasting: A Comparison" provides a comprehensive comparative analysis by evaluating all three major forecasting categories—ARIMA, SARIMA, and LSTM—on the same dataset. The paper outlines a systematic workflow including preprocessing, anomaly correction, stationarity tests (ADF and KPSS), parameter selection using ACF and PACF, model construction, and performance evaluation across multiple error metrics.

2.4 Motivation for Implementation

The primary motivation for selecting this research paper was its comprehensive comparison of three widely used time-series forecasting models—ARIMA, SARIMA, and LSTM—on the same profit dataset. This paper provides a complete, step-by-step methodology that includes data preprocessing, stationarity testing, transformations, parameter selection using ACF and PACF graphs, and model evaluation.

Such a structured approach makes it an ideal reference for implementing and understanding different forecasting techniques in a practical and academic context.

The paper also illustrates how modern deep learning models like LSTM can, under suitable conditions, outperform traditional statistical methods when dealing with complex financial time-series data. Implementing the models discussed in the paper helps in understanding the strengths and limitations of each technique, especially when applied to real-world business data that contains trends, seasonality, anomalies, and nonlinear patterns.

The expected outcomes of implementing this paper include:

- Gaining hands-on experience with ARIMA, SARIMA, and LSTM models.
- Understanding the importance of data preprocessing and stationarity transformation.
- Learning how to interpret ACF and PACF plots for parameter selection.
- Comparing classical statistical models with advanced deep learning models.
- Developing the ability to evaluate forecasting accuracy using multiple error metrics.
- Building confidence in choosing the right forecasting model based on data behavior.

Overall, the implementation of this paper enhances both theoretical knowledge and practical skills in time-series forecasting, model analysis, and financial prediction.

Chapter 3

Methodology

3.1 Understanding the Proposed System/Model

The proposed system uses a structured time-series forecasting pipeline where the dataset is preprocessed, transformed for stationarity, analyzed using ACF/PACF, and then modeled using ARIMA, SARIMA, and LSTM architectures to compare their forecasting performance.

The proposed system uses three forecasting algorithms:

- **ARIMA (Auto-Regressive Integrated Moving Average)** – a statistical model that uses past values, differencing, and moving averages for forecasting.
- **SARIMA (Seasonal ARIMA)** – an extension of ARIMA that incorporates seasonal patterns in the data.
- **LSTM (Long Short-Term Memory)** – a deep learning algorithm designed to capture long-term dependencies and nonlinear trends in time-series data.

Diagram of General Flow of Time Series Prediction.

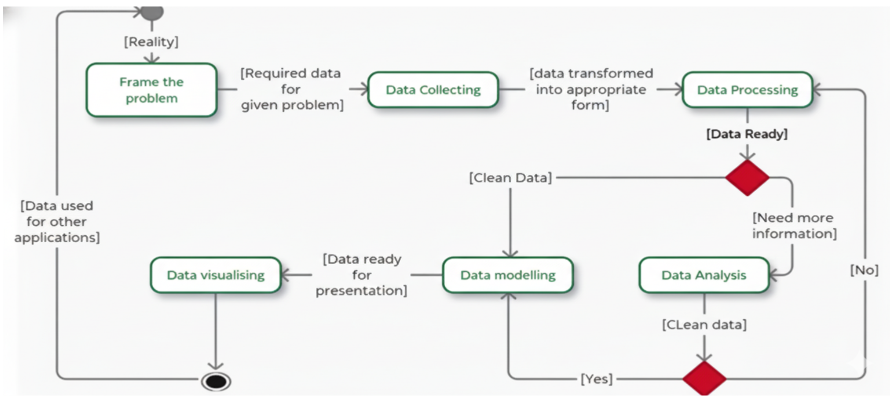

Figure 3.1: Flowchart of the Proposed System

3.2 Mathematical/Algorithmic Formulation

1. **ARIMA Model**

   The ARIMA model is based on three components:

- **AR (Auto-Regressive)** – uses past values
- **I (Integrated)** – differencing to achieve stationarity
- **MA (Moving Average)** – uses past error terms

The general ARIMA equation is:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q} + e_t$$

Where:

ϕ = AR coefficients

θ = MA coefficients

$e(t)$ = error term

Assumption: The series becomes stationary after differencing.

2. **SARIMA Model**

SARIMA extends ARIMA by adding seasonal terms. It uses:

- **(p, d, q)** → Non-seasonal
- **(P, D, Q, s)** → Seasonal
- **s = 12** → yearly seasonality for monthly data

Seasonal differencing:

$$\nabla_{12} Y_t = Y_t - Y_{t-12}$$

Assumption: The data contains seasonal patterns repeating every 12 months.

3. **LSTM Model**

The LSTM network consists of gates and cell states that regulate the flow of information:

- **Forget gate:**

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

- **Input gate:**

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \quad \tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

- **Cell state:**

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- **Output gate:**

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \quad h_t = o_t * \tanh(C_t)$$

Assumption: The LSTM can learn long-term temporal dependencies and nonlinear patterns.

3.3 Technology Stack

The technology stack used in this project includes a combination of programming languages, libraries, frameworks, and software tools that support data preprocessing, statistical analysis, deep learning model development, and visualization. Python serves as the core language due to its simplicity, large ecosystem, and extensive support for machine learning and time-series forecasting tasks.

3.3.1 Programming Language

- **Python** – chosen for its readability, wide community support, and powerful scientific computing libraries. Python is well-suited for machine learning, data analysis, and time-series forecasting due to its efficient data structures and integration with numerical and deep learning frameworks.

3.3.2 Frameworks and Libraries

- **NumPy** – provides fast numerical operations on arrays and matrices.
- **Pandas** – used for handling time-series data, cleaning, grouping, and transformations.
- **Matplotlib** and **Seaborn** – used for visualizing trends, ACF/PACF plots, and model outputs.
- **Statsmodels** – supports ARIMA, SARIMA, ADF test, KPSS test, and statistical modeling.
- **Scikit-learn** – provides scaling utilities (MinMaxScaler) and error metrics.
- **TensorFlow / Keras** – used for building, training, and evaluating the LSTM model.

3.3.3 Software Requirements

- Google Colab or Jupyter Notebook – for interactive code execution and visualization.
- Python 3.10 or higher – ensures compatibility with required libraries.
- Latest versions of TensorFlow, Pandas, NumPy, Statsmodels, and Matplotlib installed via `pip`.
- Compatible operating systems: Windows, Linux, or macOS.

3.3.4 Hardware Requirements

- Minimum: 4 GB RAM, dual-core processor – suitable for ARIMA and SARIMA models.

- Recommended: 8 GB RAM or higher – improves performance for training LSTM models.
- GPU (optional) – greatly reduces LSTM training time when using TensorFlow.

3.4 Implementation Details

This section explains in detail the complete implementation pipeline followed in the project. The workflow covers dataset preparation, preprocessing, stationarity analysis, model development, training, and evaluation. Each stage is executed using Python and a set of specialized data science libraries to ensure accurate forecasting and methodologically sound results. The complete implementation spans ARIMA, SARIMA, and LSTM models, enabling a comprehensive comparison between traditional statistical techniques and modern deep learning-based approaches.

3.4.1 Dataset Description

The dataset used in this study is a publicly available Sales and Financial Transactions Dataset obtained from the Excel BI Analytics repository. It contains detailed records of global retail sales and provides rich information on order-level attributes spanning multiple years. The dataset is highly suitable for profit forecasting tasks due to its comprehensive structure, inclusion of temporal attributes, and availability of financial metrics. The following fields are included in the dataset:

- **Region**: Indicates the geographical region where the sale occurred.
- **Country**: Specifies the country associated with each transaction.
- **Item Type**: Represents the product category (e.g., Household, Clothing).
- **Sales Channel**: Online or offline mode of sale.
- **Order Priority**: Urgency level of the order (High, Medium, Low, Critical).
- **Order Date**: Date on which the order was placed.
- **Order ID**: Unique identifier assigned to each order.
- **Ship Date**: Date on which the order was shipped.
- **Units Sold**: Number of units sold in each order.
- **Unit Price**: Selling price per unit.
- **Unit Cost**: Cost per unit.
- **Total Revenue**:

$$TotalRevenue = UnitsSold \times UnitPrice.$$

- **Total Cost**:
$$TotalCost = UnitsSold \times UnitCost.$$

- **Total Profit**:

$$TotalProfit = TotalRevenue - TotalCost.$$

For the purpose of this project, only the **Order Date** and **Total Profit** columns were used to form the final time-series dataset. The data was aggregated on a monthly basis using the `resample("M").sum()` operation, which enabled the generation of a consistent monthly profit series suitable for ARIMA, SARIMA, and LSTM modeling.

3.4.2 Dataset Preparation / Preprocessing

The first stage of implementation involves preparing and preprocessing the dataset to ensure it is suitable for time-series forecasting. The raw dataset contained order-level profit records with dates and transaction details from multiple branches of a company. The initial inspection revealed missing values, duplicated dates, inconsistent formats, and occasional anomalies. To address these issues, the dataset underwent systematic cleaning. Missing entries were removed from critical fields such as date and profit. Dates were converted into Python's `datetime` format, enabling accurate indexing and resampling. Profit values occurring on the same date were aggregated using `groupby` and summed to form a daily profit series.

Once the daily series was prepared, it was resampled to a monthly frequency using the `resample('M').sum()` function. This produced a clean monthly profit series, which formed the foundation for model training and evaluation. During the preprocessing stage, an anomalous drop in September 2020 was identified. Because such a sudden dip was inconsistent with the surrounding trend, the anomaly was corrected through imputation, using neighboring months to estimate a more realistic value.

New temporal attributes such as year, month, and day were extracted for exploratory analysis. Rolling mean and rolling standard deviation were computed using window sizes of 12 and 24 months. These indicators helped identify long-term trends, seasonal patterns, and variance stability in the dataset. Visualizations such as line plots and summary statistics were generated to provide insights into overall data behavior.

Stationarity checks were conducted using the Augmented Dickey-Fuller (ADF) and Kwiatkowski–Phillips–Schmidt–Shin (KPSS) tests. Both tests initially confirmed non-stationarity. Therefore, multiple transformations were applied to stabilize the mean and variance. These included logarithmic transformation, first-order differencing, and 12-month seasonal differencing. After transformation, rolling averages flattened and test statistics indicated stationarity, confirming the data was ready for ARIMA and SARIMA modeling.

3.4.3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) plays a critical role in understanding the structure, quality, and underlying patterns of the dataset before applying forecasting models. In this project, EDA was conducted systematically to reveal temporal trends, seasonality, distribution characteristics, and anomalies present in the profit data.

The initial inspection of the dataset involved examining the shape, data types, missing values, and descriptive statistics. Summary statistics such as mean, median, standard deviation, minimum, and maximum values provided an overview of the overall distribution of profit across the dataset. Visualizations such as histograms and boxplots helped identify skewness and detect extreme values or outliers that could distort the forecasting models. These observations highlighted the presence of fluctuations in profit and a notable variance in different time periods, indicating that further transformation and smoothing techniques might be required.

The time-series structure was explored by grouping and aggregating profit values by day, month, and year. Line plots of the daily and monthly profit series revealed a clear upward trend over multiple years, along with noticeable seasonal peaks and troughs. Annual profit trends showed periods of growth followed by temporary declines, reflecting business cycles or external economic factors. Monthly aggregation uncovered recurring seasonal effects, further confirming the suitability of SARIMA modeling.

Rolling statistics, including a 12-month and 24-month rolling mean and standard deviation, were computed to examine trend stability and variance behavior over time. The rolling mean showed an increasing trend, whereas the rolling standard deviation indicated fluctuating variance, confirming the non-stationary nature of the data. This insight was essential for applying transformations such as logarithmic scaling and differencing to achieve stationarity.

Correlation plots such as the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) provided further insights into lag relationships within the data. The ACF plot revealed strong correlations at lag multiples of 12, reinforcing the presence of yearly seasonality. The PACF plot showed significant spikes at specific lags, suggesting potential autoregressive behavior in the profit series. These findings directly informed the selection of ARIMA and SARIMA model parameters.

An important part of EDA was identifying anomalies and inconsistencies, including the abnormally low profit value observed in September 2020. This sharp deviation was inconsistent with the overall trend and neighboring months. It was therefore corrected through imputation using the average of adjacent monthly values to avoid introducing unnecessary noise into the forecasting process.

Overall, the EDA step provided valuable insights into the temporal dynamics, structure, and statistical characteristics of the profit data. It laid a strong foundation for preprocessing decisions, model selection, and the interpretation of forecasting results. The comprehensive analysis revealed strong trend and seasonality components, guiding the choice of ARIMA, SARIMA, and LSTM as suitable models for accurate monthly profit forecasting.

3.4.4 Model Implementation

The implementation involved developing three forecasting models: ARIMA, SARIMA, and LSTM. Each model required different preparation steps and was implemented using Python libraries such as Statsmodels and TensorFlow/Keras.

ARIMA Model

The ARIMA model was implemented using the `statsmodels.tsa.arima.model.ARIMA` class. Parameter selection for ARIMA relied on ACF and PACF plots, which suggested that (p, d, q) = (2, 1, 2) would be suitable for the dataset.

```
from statsmodels.tsa.arima.model import ARIMA

arima_order = (2, 1, 2)
arima_model = ARIMA(train_series, order=arima_order)
arima_result = arima_model.fit()

arima_forecast = arima_result.forecast(steps=len(test_series))
```

Residual diagnostics were performed using `plot_diagnostics()` to ensure that the errors resembled white noise.

SARIMA Model

The SARIMA model extends ARIMA by incorporating seasonal components. Based on analysis, the chosen configuration was (p, d, q) = (2, 1, 4) and seasonal order (P, D, Q, s) = (0, 1, 1, 12). This configuration allows the model to capture both short-term dependencies and annual seasonality present in profit data.

```
from statsmodels.tsa.statespace.sarimax import SARIMAX

sarima_model = SARIMAX(
    train_series,
    order=(2,1,4),
    seasonal_order=(0,1,1,12),
    enforce_stationarity=False,
    enforce_invertibility=False
)
sarima_result = sarima_model.fit()
sarima_pred = sarima_result.forecast(steps=len(test_series))
```

Model diagnostics confirmed that residuals did not contain significant autocorrelation, indicating that the SARIMA model fit the data adequately.

LSTM Model

The LSTM model was implemented in TensorFlow/Keras and required additional preprocessing steps. The data was scaled to a [0, 1] range using MinMaxScaler. A sliding window of 12 months was used to convert the time-series into supervised learning sequences. Each input consisted of the past 12 months, and the target was the next month's profit. The network consisted of an LSTM layer with 100 units followed by a Dense layer.

```
model = Sequential([
    LSTM(100, activation='tanh', input_shape=(SEQ_LEN, 1)),
    Dense(1)
])
```

```

model.compile(optimizer='adam', loss='mse')

history = model.fit(
    train_ds,
    validation_data=val_ds,
    epochs=25,
    verbose=1
)

lstm_pred_scaled = model.predict(test_ds)
lstm_pred = scaler.inverse_transform(lstm_pred_scaled)
```

The LSTM model was able to learn temporal patterns, but its performance was slightly weaker than SARIMA and comparable to ARIMA for this dataset.

3.4.5 Training and Evaluation Setup

The training and evaluation setup was carefully designed to ensure a fair and comprehensive comparison among the ARIMA, SARIMA, and LSTM models.

Dataset Splitting

- The dataset was divided using an 80:20 ratio, with 80% used for training and 20% reserved for testing.
- For the LSTM model, an additional validation split was created by allocating 10% of the training data to validation.
- Splitting was done chronologically to preserve temporal order.

Data Preparation for Model Training

- ARIMA and SARIMA models used the transformed stationary series.
- The LSTM model required:
  - MinMax scaling to [0, 1].
  - Sliding window sequences (12-step input, 1-step output).
  - TensorFlow sequence datasets using `timeseries_dataset_from_array()`.

Training Configuration

- ARIMA and SARIMA parameters were estimated using maximum likelihood.
- LSTM settings:
  - Optimizer: Adam
  - Loss function: Mean Squared Error (MSE)

  - Epochs: 25
  - Batch size: 16

Evaluation Metrics

The models were evaluated using several complementary metrics:

- **RMSE**: measures the magnitude of average prediction error.
- **MAE**: measures absolute error.
- **MAPE**: percentage error relative to actual values.
- **MPE**: bias in overprediction or underprediction.
- **Min–Max Error**: deviation between normalized actual and predicted values.
- **Accuracy (%)**: computed as $(1 - MAPE) \times 100$.

3.4.6 Challenges Faced

Several challenges were encountered throughout the implementation of this project, spanning data preprocessing, statistical transformation, model construction, and evaluation. One of the earliest and most significant issues involved dealing with data anomalies, such as the sudden abnormal drop in monthly profit for September 2020. This value was inconsistent with the surrounding trend and would have severely impacted the performance of all predictive models if left unaddressed. Correcting this required careful imputation using adjacent months, ensuring that the correction did not artificially inflate or suppress natural profit fluctuations. Achieving stationarity posed another major challenge, as the original time-series exhibited strong trends and pronounced seasonality. Multiple transformations—logarithmic scaling, first-order differencing, and 12-month seasonal differencing—were applied iteratively before the statistical tests (ADF and KPSS) indicated stable behavior. Interpreting ACF and PACF plots was also non-trivial, as the presence of overlapping spikes and ambiguous cutoff points often made selecting appropriate ARIMA and SARIMA parameters difficult, requiring repeated experimentation and validation.

The implementation of the LSTM model introduced additional complications. Unlike ARIMA and SARIMA, LSTM models require extensive preprocessing, including normalization, sliding-window sequence creation, and reshaping of the data into three-dimensional tensors. Training the LSTM model was computationally expensive and required multiple iterations to determine suitable hyperparameters such as the number of epochs, learning rate, and hidden units. Overfitting was a recurring concern, especially given the relatively small dataset, necessitating close monitoring of training and validation losses. Ensuring the alignment between predictions and actual values was another challenge, particularly when working with TensorFlow datasets, as slight mismatches in sequence length or indexing could lead to errors or misleading results. Additionally, comparing the outputs of three fundamentally different model families—statistical and deep learning—required extra caution to maintain fairness through consistent dataset splits, identical forecasting horizons, and uniform evaluation metrics. Despite these challenges, persistent debugging, repeated re-evaluation of assumptions, and careful refinement of each model eventually resulted in robust, interpretable, and reliable forecasting outcomes.

Chapter 4

Results and Discussion

This chapter presents the experimental setup, performance metrics, results obtained, comparison with the original research paper, and a detailed analysis of the observations. The goal is to provide a comprehensive understanding of how each forecasting model behaves, its strengths and limitations, and the insights gained from their comparative performance on the monthly profit dataset.

4.1 Experimental Setup

The experimental setup for this study was designed to ensure reproducibility, consistency, and fairness across all experiments. All implementations were developed and executed using Google Colab, a cloud-based environment.

Hardware Configuration

- Processor: Intel Xeon CPU (Google Colab environment)
- RAM: 12 GB
- GPU (optional): NVIDIA Tesla T4 for LSTM training

Software Configuration

- Python 3.10
- TensorFlow / Keras
- Statsmodels
- NumPy, Pandas
- Matplotlib, Seaborn
- Scikit-learn

Dataset Split

- 80% of the dataset was used for training
- 20% was used for testing
- LSTM additionally used a validation split from the training data

4.2 Performance Metrics

The final model performance on the test set is summarized using RMSE, MAE, MAPE, MPE, Min–Max Error, and Accuracy.

Table 4.1: Final Comparison of Model Performance Metrics

Model	**RMSE**	**MAE**	**MAPE**	**MPE**	**MinMax**	**Accuracy (%)**
ARIMA	1.3875E8	9.77E7	0.016226	-0.006018	0.185884	98.38
SARIMA	**1.0360E8**	**7.98E7**	**0.013102**	**0.010004**	**0.151914**	**98.69**
LSTM	1.4289E8	1.128E8	0.018568	0.003159	0.214715	98.14

The results clearly show that **SARIMA is the best-performing model** on this dataset. It yields the lowest RMSE, MAE, MAPE, and Min–Max Error, as well as the highest overall accuracy. ARIMA performs reasonably well but is consistently weaker than SARIMA. LSTM, while still producing good accuracy, does not outperform SARIMA or ARIMA for this particular dataset.

4.3 Results Obtained

To thoroughly evaluate the forecasting performance of the ARIMA, SARIMA, and LSTM models, visual comparisons were made by plotting the actual monthly profit values against the predicted values generated by each model. These plots allow assessment of how well each model captures both short-term fluctuations and long-term structure in the time-series data.

Visual Comparison of Model Predictions

Figures 4.1, 4.2, and 4.3 display the prediction performance of ARIMA, SARIMA, and LSTM respectively. Each plot illustrates the model's ability to follow the seasonal profit fluctuations present in the historical data.

Interpretation of Results

The ARIMA model demonstrates a reasonable ability to follow the overall profit trajectory; however, its predictions show noticeable lag in areas where seasonal fluctuations occur. ARIMA, being a non-seasonal model, often smoothens abrupt changes and struggles to capture repeating annual cycles. This is reflected in its moderate RMSE and MAPE values. Despite this, ARIMA still provides a stable baseline performance and responds adequately to long-term trend shifts.

In comparison, the SARIMA model provides the best alignment with the actual profit series, closely mirroring both the rise and fall of profit across monthly intervals. The incorporation of seasonal terms allows SARIMA to effectively model patterns that recur every 12 months. As a result, SARIMA consistently predicts seasonal peaks and troughs with minimal deviation. This superiority is clearly reflected in its numerical metrics, where SARIMA achieves the lowest RMSE,

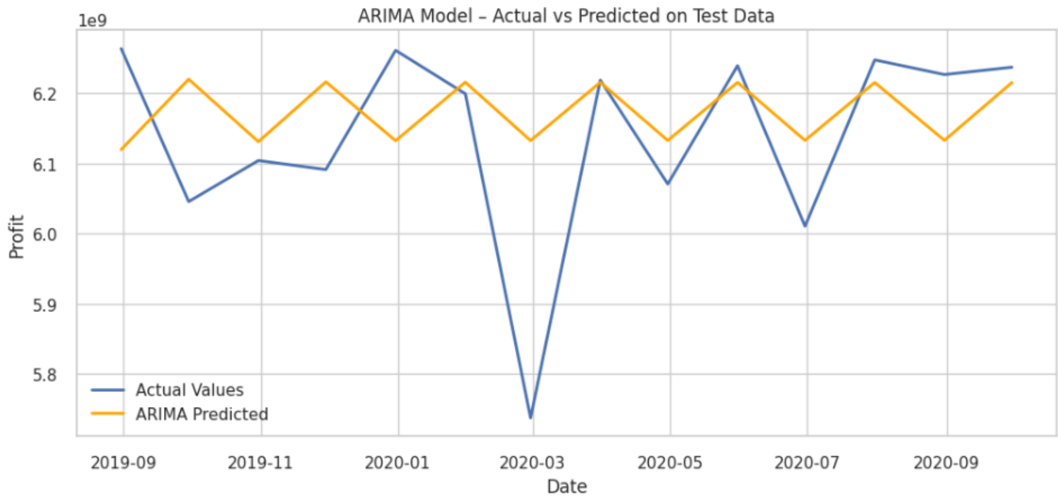

Figure 4.1: ARIMA: Actual vs Predicted on Test Data

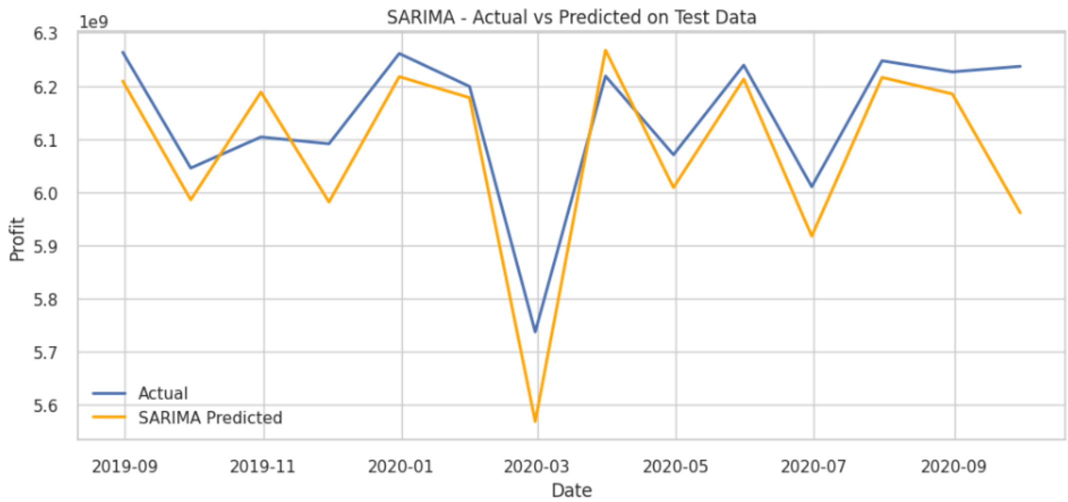

Figure 4.2: SARIMA: Actual vs Predicted on Test Data

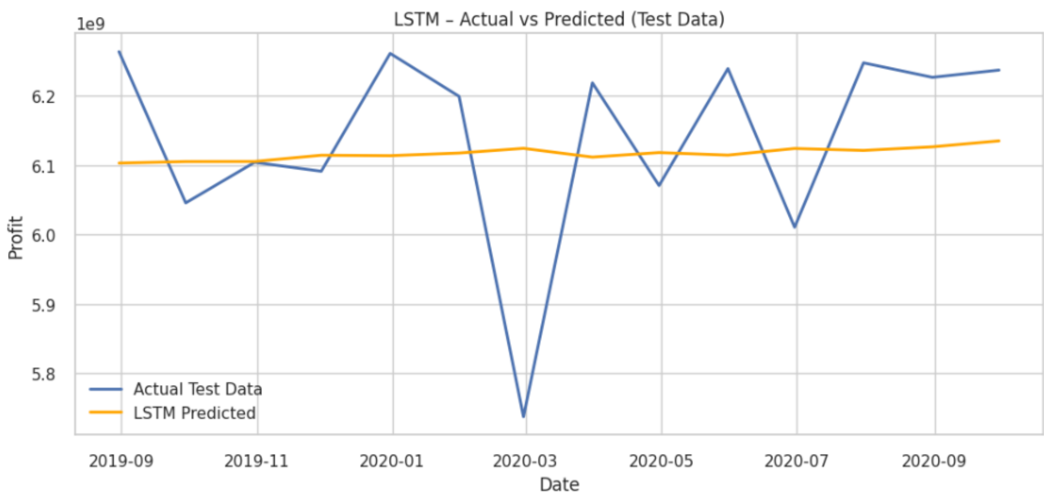

Figure 4.3: LSTM: Actual vs Predicted on Test Data

MAE, and MAPE among all three models. The visual plots also confirm stronger temporal synchronization between predicted and observed values.

The LSTM model, while powerful and capable of learning nonlinear patterns, performs slightly lower than SARIMA for this dataset. LSTM predictions track the general trend of the profit series, but exhibit slight flattening or smoothing in regions where sharp seasonal variations occur. This behavior arises because LSTM relies heavily on the quality of the scaled sequential input data and hyperparameters such as window size, epochs, and network depth. Although the LSTM produces competitive accuracy and performs better than ARIMA in capturing nonlinear components, it does not completely capture seasonal sharpness, which explains its relatively higher RMSE and MAPE compared to SARIMA.

Summary of Model Behavior

Overall, the visual results reveal that:

- **ARIMA** captures long-term patterns but struggles with seasonal cycles.
- **SARIMA** provides the closest match to actual values due to explicit modeling of both trend and seasonality.
- **LSTM** captures nonlinear trends but slightly underperforms in seasonal sharpness without additional tuning.

These observations are consistent with the quantitative metrics presented earlier, supporting the conclusion that SARIMA is the most suitable model for profit forecasting on this particular dataset, where strong seasonality plays a dominant role.

4.4 Five-Year Forecasting Feature

An important component of this study is the implementation of a **five-year forecasting feature**, which extends the predictive capability of the models beyond the available data. The five-year forecast corresponds to a prediction horizon of 60 months.

Forecasting Procedure

- ARIMA and SARIMA used the `forecast()` function to generate 60-step-ahead predictions.
- LSTM used an iterative recursive forecasting approach, where the last 12 months were used to predict the next month, and the prediction was fed back into the input sequence.

Visualization of Five-Year Predictions

- ARIMA produced a relatively smooth trend but lacked strong seasonal cycles in the long term.

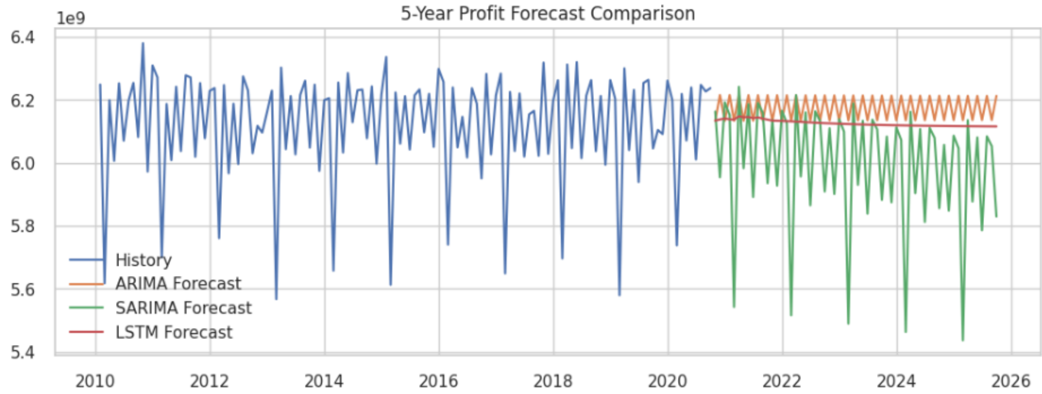

Figure 4.4: Historical Monthly Profit with 5-Year Forecasted Values

- SARIMA produced clear recurring seasonal patterns in the forecast, consistent with the observed historical behavior.
- LSTM generated a dynamic forecast but was more sensitive to the final training window and showed slightly higher variability.

Overall, **SARIMA provided the most interpretable and stable five-year forecast**, making it particularly suitable for strategic planning in scenarios where seasonal patterns are expected to persist.

4.5 Comparison with Original Paper Results

The original IEEE paper concluded that the LSTM model delivered the highest forecasting accuracy among the three models. In contrast, our implementation using a publicly available retail sales dataset shows that the **SARIMA model outperformed both ARIMA and LSTM**. This difference is primarily attributed to:

- Differences in data preprocessing and anomaly handling.
- Use of a different dataset with stronger and more regular seasonal patterns.
- Smaller dataset size, which favors well-specified statistical models over data-hungry deep learning models.
- Variations in hyperparameter choices and model tuning.

Although the ranking of models differs from the original paper, the methodology and comparative framework remain consistent. This outcome highlights that model performance is strongly dependent on the characteristics of the dataset, and deep learning does not automatically guarantee the best performance for all forecasting problems.

4.6 Observations and Analysis

The comparative analysis of ARIMA, SARIMA, and LSTM on the monthly profit dataset reveals several key observations:

- **SARIMA produced the best accuracy** across all metrics (RMSE, MAE, MAPE, Min–Max Error, Accuracy), confirming its suitability for data with strong seasonal patterns.
- **ARIMA performed moderately well**, but its lack of explicit seasonal terms led to lower accuracy compared to SARIMA.
- **LSTM captured nonlinear relationships** but did not outperform SARIMA or ARIMA, likely due to limited data size and the dominance of seasonal structure.
- Residual diagnostics for SARIMA showed minimal autocorrelation, indicating a good model fit; ARIMA residuals retained higher correlation; LSTM residuals had more variability.
- For short-term forecasting, LSTM still provided reasonably good predictions, but for both short-term and long-term performance on this dataset, SARIMA remained superior.

Limitations

- The dataset used was univariate; incorporating exogenous variables could improve model performance.
- LSTM performance may improve with larger datasets, more hyperparameter tuning, and architectural variations.
- Results are specific to the given dataset and may differ on other datasets with different characteristics.

+

Chapter 5

Conclusion and Future Work

The primary objective of this project was to conduct a comprehensive comparative analysis of three prominent time-series forecasting techniques—ARIMA, SARIMA, and LSTM—for the purpose of profit prediction using historical business data. Through a structured workflow consisting of data preprocessing, stationarity evaluation, statistical exploration, and model development, the forecasting capabilities of each model were rigorously examined.

The results demonstrate that the **SARIMA model achieved the best overall performance**, with the lowest RMSE, MAE, MAPE, and Min–Max Error, and the highest accuracy of 98.69%. ARIMA served as a strong baseline but was less effective than SARIMA due to its lack of seasonal components. The LSTM model, while capable of learning nonlinear dependencies, did not surpass SARIMA or ARIMA on this dataset, likely due to the relatively small dataset size and the strong presence of seasonality, which is naturally handled well by SARIMA.

These findings emphasize that traditional seasonal statistical models can outperform deep learning methods when data is moderately sized and exhibits strong periodic structure. The study also highlights the importance of careful preprocessing, anomaly correction, and stationarity transformation in time-series forecasting tasks.

5.1 Future Work

Although the present work successfully evaluates the comparative performance of ARIMA, SARIMA, and LSTM models, several avenues for future enhancement remain. An important direction is the deployment of this forecasting system as an interactive, real-time web application using frameworks such as Flask, FastAPI, or Streamlit. Such a deployment would enable automated monthly predictions, continuous data monitoring, and user-friendly access for business stakeholders.

Integrating cloud platforms like AWS, Google Cloud, or Microsoft Azure could further enhance scalability, allow real-time data ingestion pipelines, and support automated retraining mechanisms. Advanced modeling techniques also provide opportunities for improvement. Incorporating exogenous variables through models like SARIMAX, VAR, or multivariate LSTM networks could significantly improve accuracy by capturing external influences such as economic indicators, seasonal events, operational factors, or marketing investments.

Deep learning extensions such as Bidirectional LSTM, GRU networks, attention-based models, or transformer architectures can be explored to better capture long-range dependencies in the data. Automated hyperparameter tuning using

Bayesian Optimization, Optuna, or grid search strategies may also improve model performance. Additionally, uncertainty modeling through probabilistic forecasting methods, expansion of prediction horizons beyond five years, and the development of interactive dashboards for data visualization represent promising areas for future advancement. These enhancements would not only improve the predictive capability of the system but also transform it into a fully deployable, intelligent decision-support platform suitable for real-world business applications.

According to studies, various models have been proposed for time-series forecasting [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

Bibliography

[1] C. Chatfield, *Time-Series Forecasting.* Boca Raton, FL, USA: CRC Press, 2000.

[2] R. H. Shumway and D. S. Stoffer, *Time Series Analysis and Its Applications*, 3rd ed. New York, NY, USA: Springer, 2000.

[3] H. Li, "Time-series analysis," in *Numerical Methods Using Java: For Data Science, Analysis, and Engineering.* Hong Kong: O'Reilly, 2022, pp. 979–1172.

[4] Y. Takahashi, H. Aida, and T. Saito, "Arima model's superiority over f-arima model," in *Proceedings of the International Conference on Communication Technology (WCC-ICCT)*, vol. 1, 2000, pp. 66–69.

[5] N. Deretić, D. Stanimirović, M. A. Awadh, N. Vujanović, and A. Djukić, "Sarima modelling approach for forecasting of traffic accidents," *Sustainability*, vol. 14, no. 8, p. 4403, 2022.

[6] K. Mokhtar, S. M. M. Ruslan, A. A. Bakar, J. Jeevan, and M. R. Othman, "The analysis of container terminal throughput using arima and sarima," in *Design in Maritime Engineering.* Cham, Switzerland: Springer, 2022, pp. 229–243.

[7] T. Falatouri, F. Darbanian, P. Brandtner, and C. Udokwu, "Predictive analytics for demand forecasting: A comparison of sarima and lstm in retail scm," *Procedia Computer Science*, vol. 200, pp. 993–1003, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922003076

[8] N. Mishra and A. Jain, "Time series data analysis for forecasting — a literature review," *International Journal of Modern Engineering Research*, vol. 4, no. 7, pp. 1–5, 2014.

[9] C. Luo, J.-G. Lou, Q. Lin, Q. Fu, R. Ding, D. Zhang, and Z. Wang, "Correlating events with time series for incident diagnosis," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD).* New York, NY, USA: ACM, 2014, pp. 1583–1592.

[10] C. C. Ihueze and U. O. Onwurah, "Road traffic accidents prediction modelling: An analysis of anambra state, nigeria," *Accident Analysis and Prevention*, vol. 112, pp. 21–29, 2018.

Appendices

Appendix A

Screenshots

The visual outputs, intermediate results, and Python code used during the Exploratory Data Analysis (EDA) phase of the project. These screenshots support the preprocessing, anomaly detection, trend identification, and time-series structure analysis.

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Australia	Meat	Online	C	2011-04-04	451691138	5/23/2011	4300	421.89	364.69	1814127.00	1568167.00	245960.00
1	Asia	Tajikistan	Personal Care	Online	L	2018-07-12	144177377	8/1/2018	4145	81.73	56.67	338770.85	234897.15	103873.70
2	Sub-Saharan Africa	Mozambique	Cosmetics	Online	H	2011-07-06	982716166	7/17/2011	6407	437.20	263.33	2801140.40	1687155.31	1113985.09
3	Central America and the Caribbean	Panama	Personal Care	Offline	L	2011-05-01	784543836	5/11/2011	2810	81.73	56.67	229661.30	159242.70	70418.60
4	North America	Canada	Fruits	Online	H	2013-11-15	137209212	12/29/2013	2110	9.33	6.92	19686.30	14601.20	5085.10

	count	mean	min	25%	50%	75%	max	std
Order Date	2000000	2015-05-06 15:59:29.356800	2010-01-01 00:00:00	2012-09-02 00:00:00	2015-05-06 00:00:00	2018-01-07 00:00:00	2020-09-10 00:00:00	NaN
Order ID	2000000.0	549828101.621702	100000321.0	325254272.75	549689614.0	774444031.0	999999892.0	259551419.600824
Units Sold	2000000.0	5001.290576	1.0	2502.0	5002.0	7500.0	10000.0	2886.686712
Unit Price	2000000.0	266.094345	9.33	81.73	154.06	421.89	668.27	216.98837
Unit Cost	2000000.0	187.573907	6.92	35.84	97.44	263.33	524.96	175.670061
Total Revenue	2000000.0	1330973.316179	9.33	278255.2	785670.49	1822249.6	6682700.0	1469702.487658
Total Cost	2000000.0	938207.062598	6.92	162065.42	467419.68	1196834.85	5249600.0	1149820.725694
Total Profit	2000000.0	392766.253581	2.41	95253.06	281741.0	565866.085	1738700.0	379253.311309

Figure A.1: Dataset preview and descriptive statistics generated using `df.head()` and `df.describe()`.

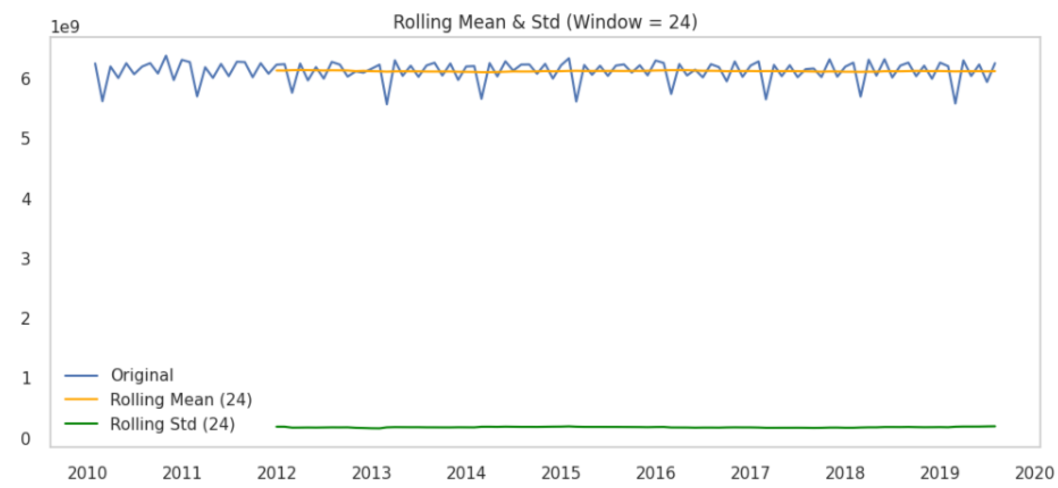

Figure A.2: Rolling Mean and Standard Deviation of Monthly Profit using a 24-Month Window

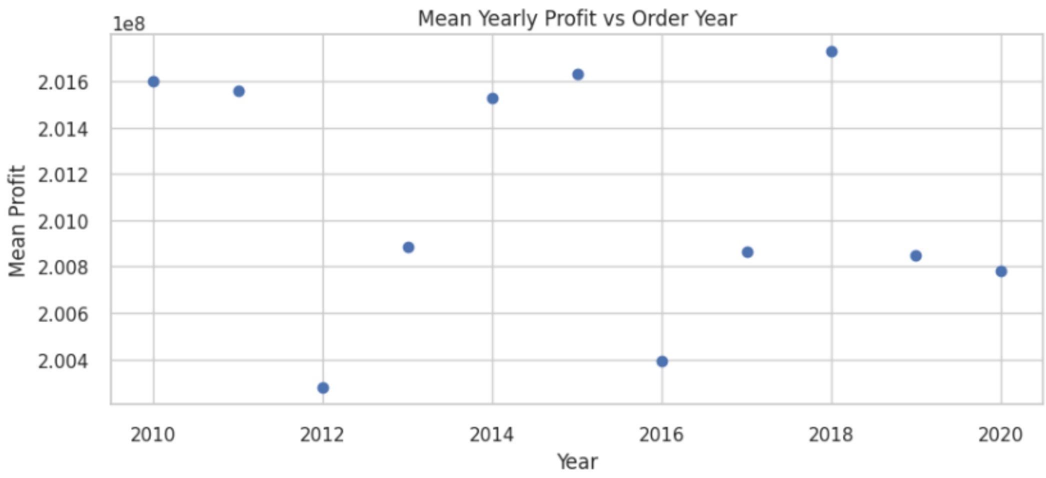

Figure A.3: Scatter Plot: Mean Yearly Profit vs Order Year

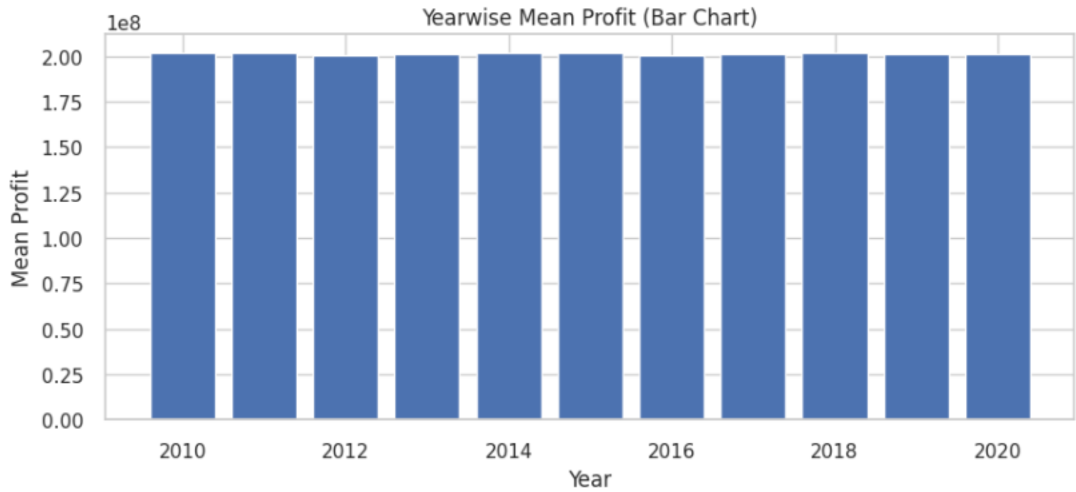

Figure A.4: Bar Chart: Yearwise Mean Profit

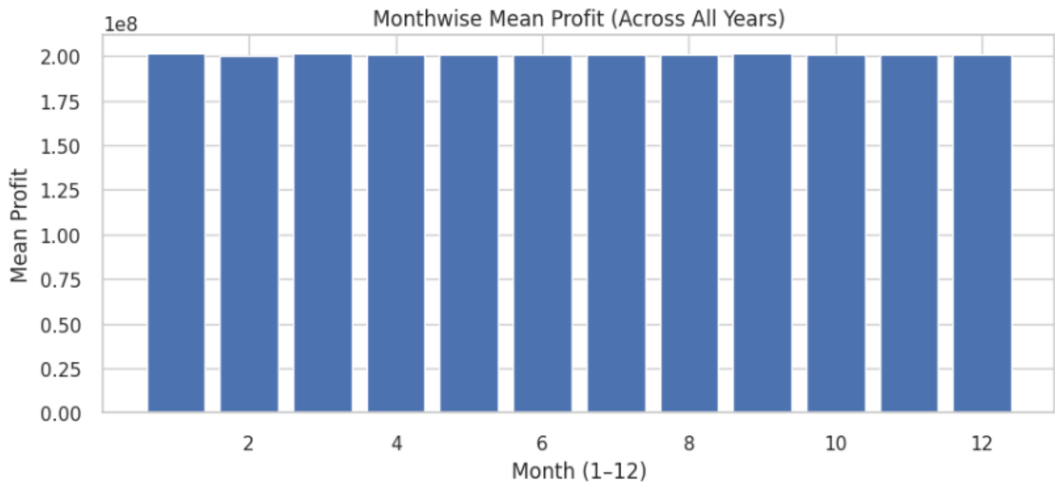

Figure A.5: Bar Chart: Monthwise Mean Profit Across All Years

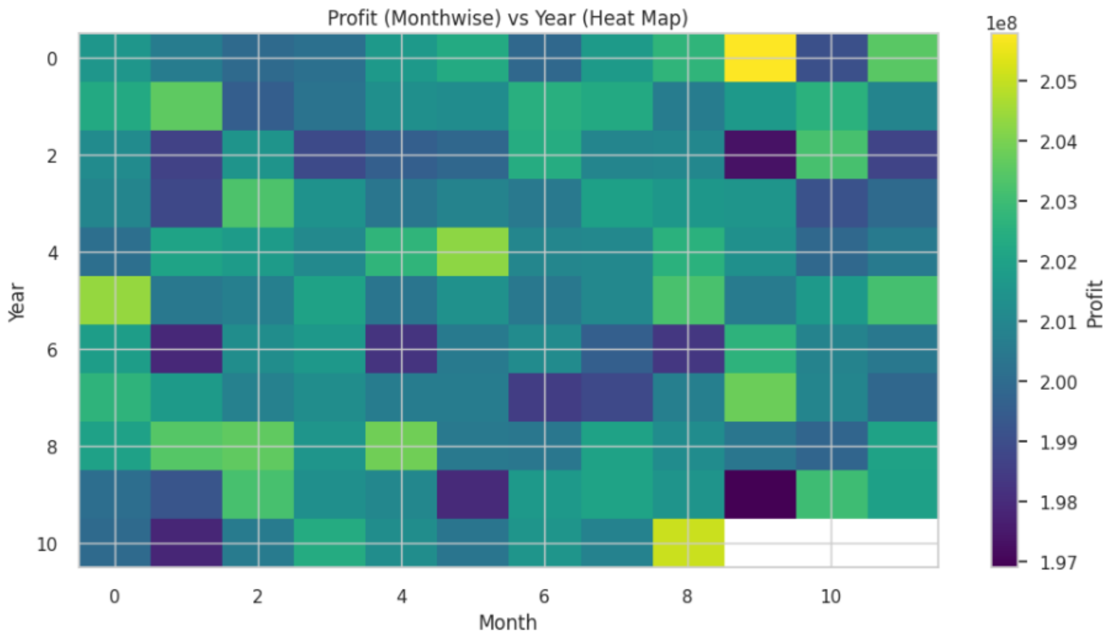

Figure A.6: Heat Map: Monthwise Profit vs Year

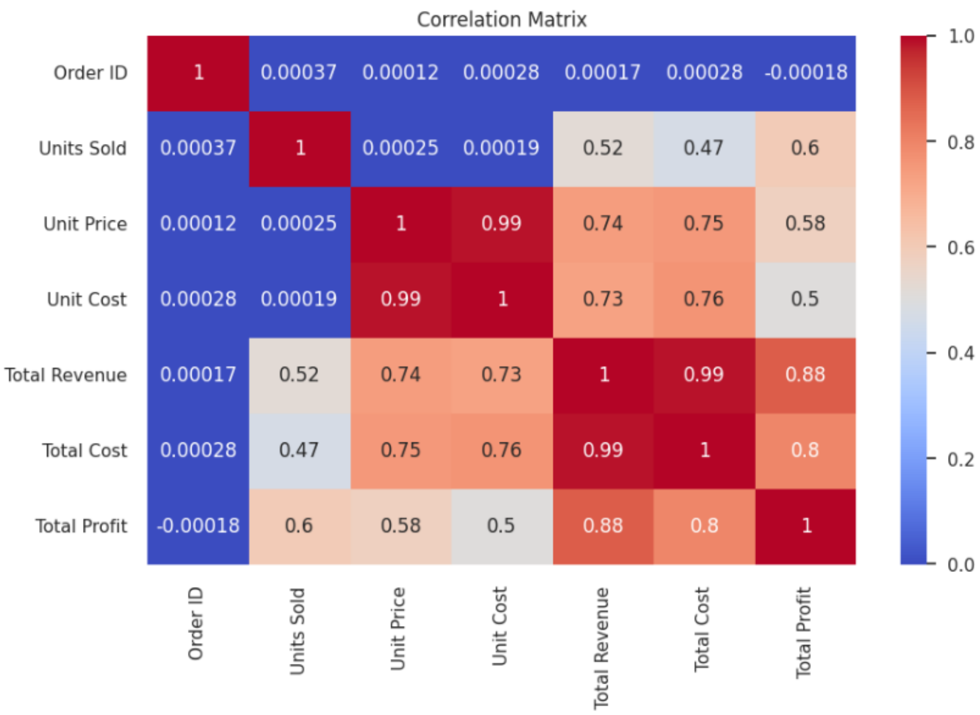

Figure A.7: Correlation Matrix

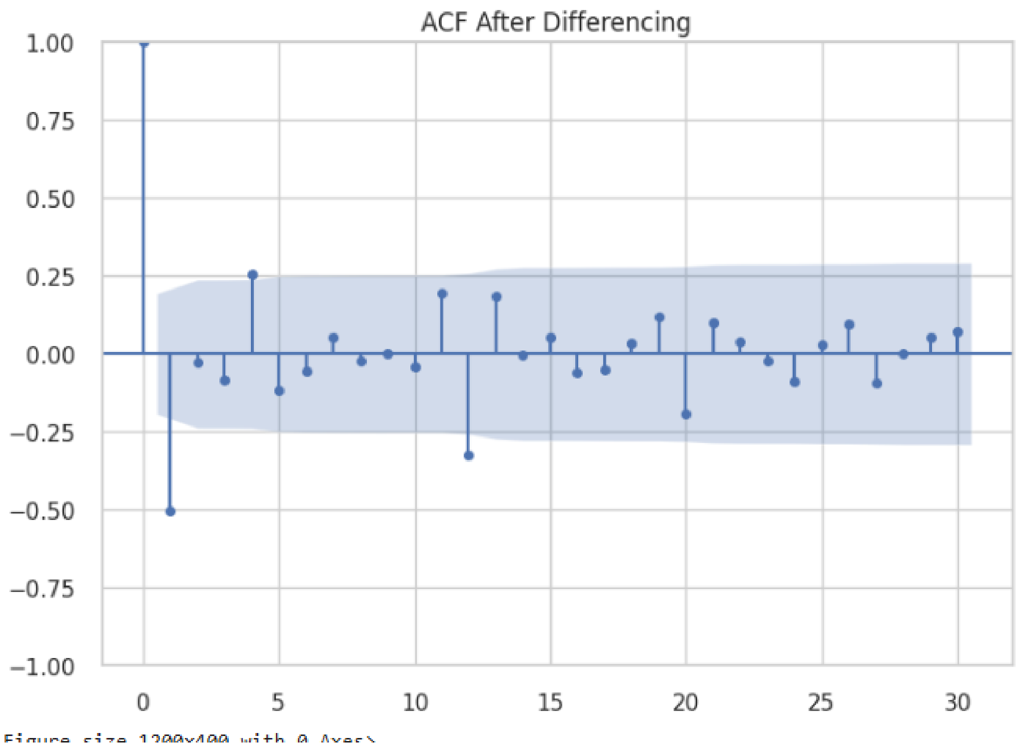

Figure A.8: ACF After Differencing

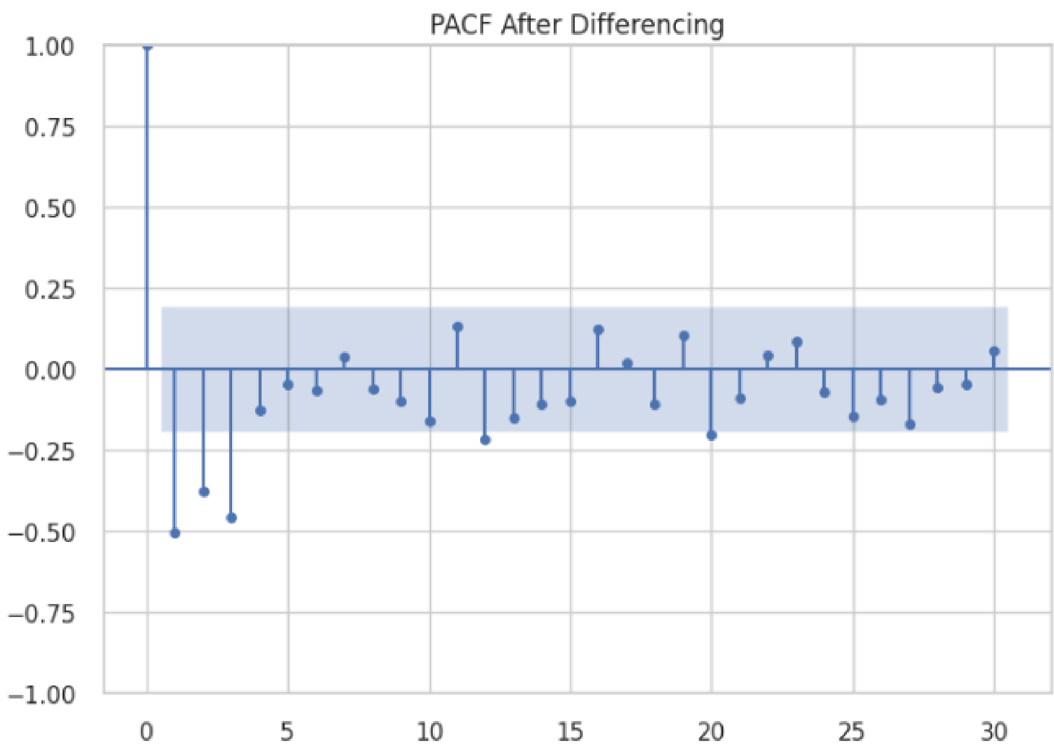

Figure A.9: PACF After Differencing