# Incorporate Machine Learning in Analyse Amazon Sales Datasets to Improve Operational Strategy

BY

ONG QI HAO

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS) INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology

(Kampar Campus)

June 2025

## COPYRIGHT STATEMENT



This Final Year Project report is submitted in partial fulfillment of the requirements for the degree of Bachelor of Information Systems (Honours) Information Systems Engineering at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project report represents the work of the author, except where due acknowledgment has been made in the text. No part of this Final Year Project report may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

## ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Abdulkarim Kanaan Jebna for his invaluable guidance, encouragement, and constructive feedback throughout the course of this Final Year Project. His support has been instrumental in helping me grow both technically and professionally, and I am truly grateful for the opportunity to work on a meaningful project that aligns with my interests in data science and machine learning.

Finally, I am deeply thankful to my parents and family for their endless love, support, and motivation throughout my university life. This accomplishment would not have been possible without their constant encouragement and belief in me.

## ABSTRACT

This project is in the field of data analytics and machine learning, focusing on developing a short-term e-commerce sales forecasting system. The problem addressed is the difficulty for online retailers to predict sales trends due to seasonality, promotion-induced peaks, and irregular demand, and thereby leading to ineffective inventory planning and operations [2][3][4][13]. In order to overcome these challenges, three prediction models were used: Bidirectional Long Short-Term Memory (BiLSTM), Temporal Convolutional Network (TCN), and XGBoost. The study was carried out following the CRISP-DM methodology, beginning with data preprocessing and feature engineering from an openly accessible Amazon sales dataset (2017–2020) [28]. The data were condensed to daily sales figures, cleaned and converted, and subsequently modeled. BiLSTM modeled nonlinear temporal dependencies using a moving average, TCN learned long- and short-term patterns using causal convolutions, and XGBoost utilized lag features, rolling statistics, and calendar effects for interpretable tree-based forecasting. The models were evaluated using four metrics: $R^2$, RMSE, MAE, and MAPE. Results indicated that BiLSTM yielded the most balanced and accurate predictions ($R^2$ = 0.8481, MAPE = 12.91%), TCN had the highest explanatory power ($R^2$ = 0.9336, MAPE=24.35%) but overestimated peaks, whereas XGBoost performed poorly ($R^2$ = 0.6586, MAPE = 70.62%) despite being interpretable. To enhance practical adoption, an interactive Streamlit dashboard was developed, enabling users to upload sales data, select models, visualize forecasts, and receive AI-driven business insights. The novelty of this work lies in the combination of cutting-edge deep learning models and a decision-support dashboard that bridges the gap between predictive modeling and actionable strategy. In short, the system produces reliable short-term forecasts and interpretable recommendations, thereby forming an effective tool for operational and strategic planning for e-commerce.



Area of Study: Machine Learning, Data Analytics

Keywords: Sales Forecasting, BiLSTM, TCN, XGBoost, CRISP-DM, Streamlit, Amazon Dataset, AI Assistant, Time Series, E-commerce

# TABLE OF CONTENTS


TITLE PAGE I
COPYRIGHT STATEMENT II
ACKNOWLEDGEMENTS III
ABSTRACT IV
TABLE OF CONTENTS V
LIST OF FIGURES VIII
LIST OF TABLES XI
LIST OF ABBREVIATIONS XII
CHAPTER 1 1
1.1 Project Background 1
1.1.1 Sales Forecasting in Retail 1
1.1.2 From Statistical Models to Machine Learning 1
1.1.3 From Machine Learning to Deep Learning 2
1.1.4 Key Concepts 3
1.1.5 Data-Driven Forecasting in Modern E-Commerce 4
1.2 Problem Statement and Motivation 5
1.2.2 Motivation 6
1.3 Research Objectives 7
1.4 Project Scope and Direction 7
1.5 Contributions 8
1.6 Report Organization 9
CHAPTER 2 10
Literature Reviews 10
2.1 Review of Technologies 10
2.1.1 Introduction to XGBoost 10
2.1.2 Historical Development and Popularity 11
2.1.3 Key Technical Innovations in XGBoost 11
2.1.4 System Architecture of XGBoost 13
2.1.5 Introduction to BiLSTM 13
2.1.6 Historical Development and Popularity of BiLSTM 13
2.1.7 Key Technical Innovations in BiLSTM 14
2.1.8 System Architecture of BiLSTM 14
2.1.9 Introduction to TCN 15
2.1.10 Historical Development and Popularity of TCN 15

2.1.11 Key Technical Innovations in TCN 15
2.1.12 System Architecture of TCN 16
2.2 Review of Existing System 17
2.2.1 Sales Forecasting Dashboard by Afroz Sameer Ahmad 17
2.2.2 Sales Forecast and Risk Dashboard by Vishal 19
2.2.3 Sales Analysis & Prediction Dashboard by JCortes 22
2.2.4 Summary of the Existing Systems 25
2.3 Related Works 26
2.3.1 Classical Approaches (ARIMA, SARIMA) 26
2.3.2 Machine Learning Approaches (XGBoost, RF, LightGBM, etc.) 27
2.3.3 Deep Learning Models (BiLSTM, TCN, Hybrids) 28
2.4 Critical Remarks on Previous Works 29
CHAPTER 3 32
3.1 Business Understanding Plan 32
3.2 Data Understanding Plan 32
3.3 Data Visualization Plan 33
3.4 Data Preprocessing Plan 33
3.5 Modeling Plan 35
3.6 Evaluation Plan 36
3.7 Deployment 37
Chapter 4 39
4.1 System Overview 39
4.2 Dashboard Specifications 39
4.3 System Architecture 40
4.4 System Features and Functions 42
4.5 System Workflow 43
4.6 System Design Considerations 45
CHAPTER 5 47
5.1 Business Understanding 47
5.2 Data Understanding 48
5.3 Data Visualization 49
5.4 Data Preprocessing 55
5.5 Modeling and Fine-Tuning 58
5.5.1 BiLSTM 58
5.5.2 TCN 61

5.5.3 XGBoost 62
5.6 Evaluation 63
5.6.1 BiLSTM 63
5.6.2 TCN 65
5.6.3 XGBoost 67
5.6.4 Comparative Analysis 69
5.7 Deployment 70
CHAPTER 6 74
6.1 System Testing and Performance Metrics 74
6.2 Testing Setup and Result 75
6.2.1 Strengths and Weaknesses of Each Model 76
6.2.2 Implications for Business Decision-Making 78
6.2.3 Dashboard Results and Analysis 79
6.3 Project Challenges 85
6.4 Objectives Evaluation 85
6.5 Conclusion Remark 86
CHAPTER 7 87
7.1 Conclusion 87
7.2 Recommendations 88
REFERENCES 90
APPENDIX A-1
POSTER A-1

## LIST OF FIGURES


| Figure Number | Title | Page |
|---|---|---|
| Figure 2.2.1 | Sales Forecasting Dashboard developed by Afroz Sameer Ahmad | 17 |
| Figure 2.2.2 | Sales Forecasting Dashboard – About section | 18 |
| Figure 2.2.3 | Screenshot of the Sales Forecasting Dashboard showing missing column error message | 19 |
| Figure 2.2.4 | Sales Forecast and Risk Dashboard – overviews | 19 |
| Figure 2.2.5 | Sales Forecast and Risk Dashboard – Filtered sales table with sorting options | 20 |
| Figure 2.2.6 | Sales Forecast and Risk Dashboard – AI Assistant input field | 20 |
| Figure 2.2.7 | Sales Forecast and Risk Dashboard – KeyError when processing uploaded dataset | 21 |
| Figure 2.2.8 | Sales Forecast and Risk Dashboard – AuthenticationError when using AI Assistant | 21 |
| Figure 2.2.9 | Sales Analysis & Prediction Dashboard – Overview page | 22 |
| Figure 2.2.10 | Sales Analysis & Prediction Dashboard – Forecast period selection | 22 |
| Figure 2.2.11 | Sales Analysis & Prediction Dashboard – Weekly sales prediction graph (next 30 days) | 23 |
| Figure 2.2.12 | Sales Analysis & Prediction Dashboard – Weekly sales prediction table (next 30 days) | 23 |
| Figure 2.2.13 | Sales Analysis & Prediction Dashboard – Business insights and recommendations | 24 |
| Figure 4.3.1 | System Architecture of the Proposed Amazon Sales Forecasting Dashboard | 42 |
| Figure 4.5.1 | Planned Workflow of the Lung Cancer Detection System | 45 |
| Figure 5.2.1 | Pearson Correlation Heatmap | 49 |
| Figure 5.3.1 | The Total Sales Amount by Quarter | 51 |
| Figure 5.3.2 | The Quarterly Sales Amount by Year | 52 |
| Figure 5.3.3 | The Quarterly Sales Trend by Category | 53 |
| Figure 5.3.4 | The Category Sales Ranking in Quarter 1 | 54 |

| | | |
|---|---|---|
| Figure 5.5.1a | Validation Forecast vs Actual Sales (averaged MA7) | 59 |
| Figure 5.5.1b | Original vs Smoothed Sales with BiLSTM next 7-day Forecast | 60 |
| Figure 5.5.1c | Forecasted Results for Future 7 days | 60 |
| Figure 5.5.1d | Hyperparameter Tuning Results for BiLSTM (Validation $R^2$ by hidden units and batch size) | 60 |
| Figure 5.5.2 | Hyperparameter Tuning Results (Validation $R^2$ by filters and window size) | 62 |
| Figure 5.6.1a | Test Set Forecast vs Actual (Best BiLSTM, Averaged) | 64 |
| Figure 5.6.1b | Next 7-Day Forecasted Sales Amounts (Best BiLSTM) | 64 |
| Figure 5.6.1c | Original vs Smoothed Sales and 7-Day BiLSTM Forecast | 65 |
| Figure 5.6.2a | The Comparison Plot of Predicted Versus actual test values | 66 |
| Figure 5.6.2b | Predicted Sales Amount for Next 7 days Forecast (TCN) | 66 |
| Figure 5.6.2c | Original Sales with Best TCN 7-Day Forecast | 67 |
| Figure 5.6.3a | Test Set Forecast vs Actual (Best XGBoost) | 68 |
| Figure 5.6.3b | Next 7-Day Forecasted Sales Amounts (Best XGBoost) | 68 |
| Figure 5.6.3c | Historical Sales and 7-Day XGBoost Forecast | 69 |
| Figure 5.7.1 | Streamlit dashboard interface for dataset upload and model selection | 72 |
| Figure 5.7.2 | Dataset successfully loaded and previewed in the dashboard | 72 |
| Figure 5.7.3 | Model performance metrics and 7-day forecast results | 72 |
| Figure 5.7.4 | Visualization of next 7 days predicted sales | 72 |
| Figure 5.7.5 | Business insights panel showing forecasted peak day and recommendations | 73 |
| Figure 5.7.6 | AI assistant recommendations for inventory, promotions, and staffing | 73 |
| Figure 5.7.7 | Interactive Q&A with AI assistant for tailored sales strategies | 73 |
| Figure 6.2.3a | Dataset Upload and Successful Loading (showing first 5 rows of sales data) | 81 |
| Figure 6.2.3b | Error Handling: Invalid Dataset Format | 81 |

| | | |
|---|---|---|
| Figure 6.2.3c | Dashboard Output for TCN Model | 82 |
| Figure 6.2.3d | Dashboard Output for BiLSTM Model | 82 |
| Figure 6.2.3e | Dashboard Output for XGBoost Model | 82 |
| Figure 6.2.3f | Next 7 Days Forecast (tabular display of daily predicted sales) | 83 |
| Figure 6.2.3g | Forecast Visualization (bar chart of next 7 days predicted sales) | 83 |
| Figure 6.2.3h | Forecast Export as CSV (downloaded Excel file with 7-day predictions) | 83 |
| Figure 6.2.3i | Business Insight Module (trend analysis and recommendation summary) | 84 |
| Figure 6.2.3j | AI Assistant Recommendation (default recommendations for inventory, promotions, and operations) | 84 |
| Figure 6.2.3k | AI Assistant Custom Answer (user query: "how to increase daily sales" with tailored strategy output) | 84 |

## LIST OF TABLES


| Table Number | Title | Page |
|---|---|---|
| Table 2.1.1 | Comparative Summary of Forecasting Technologies | 17 |
| Table 2.2.1 | Comparison of Existing Sales Forecasting Dashboards | 26 |
| Table 2.4.1 | Comparative Analysis of Forecasting Approaches and Proposed Solutions | 31 |
| Table 5.4 | Data Preprocessing Steps for BiLSTM, TCN, and XGBoost | 56 |
| Table 5.6 | Performance Comparison of Forecasting Models on Test Set | 69 |
| Table 6.1 | Strengths and Weaknesses of BiLSTM, TCN, and XGBoost Models | 77 |
| Table 6.2 | System Test Cases and Outcomes | 79 |

## LIST OF ABBREVIATIONS

| | |
|---|---|
| ARIMA | AutoRegressive Integrated Moving Average |
| ML | Machine Learning |
| LSTM | Long Short-Term Memory |
| XGBoost | Extreme Gradient Boosting |
| BiLSTM | Bidirectional Long Short-Term Memory |
| TCN | Temporal Convolutional Networks |
| AI | Artificial Intelligence |
| CNNs | Convolutional neural networks |
| RNN | Recurrent Neural Network |
| MAPE | Mean Absolute Percentage Error |
| $R^2$ | Coefficient of Determination |
| MAE | Mean Absolute Error |
| RMSE | Root Mean Squared Error |
| MA7 | Seven-Day Moving Average |

# CHAPTER 1
# Introduction

This chapter lays the foundation for the project by providing its background, problem statement, motivation, objectives, scope, and contributions. This chapter begins with a background of sales forecasting in retail and e-commerce industries, tracing the history from traditional statistical models to recent machine learning and deep learning models. The discussion thereafter introduces the key forecasting models adopted in this project—BiLSTM, TCN, and XGBoost—demonstrating their suitability to overcome the complexities of e-commerce sales data. The chapter also introduces the limitations of existing forecasting systems and addresses the motivation for developing a more flexible framework. The chapter finally outlines the aims, scope, and expected contributions of the project, as well as introducing the organization of this report.

## 1.1 Project Background

### 1.1.1 Sales Forecasting in Retail

Sales forecasting refers to the scientific prediction of future sales over a defined time horizon and has long been recognized as a fundamental component of retailing. As an integral part of business planning, forecasting enables firms to anticipate demand, allocate resources effectively, and establish financial budgets. Accurate forecasts provide organizations with the ability to make evidence-based decisions, influencing key areas such as inventory management, workforce scheduling, and marketing strategies. By anticipating future sales patterns, retailers can avoid the costly consequences of over-forecasting, such as excess stock that locks up capital, or under-forecasting, which risks stockouts and customer dissatisfaction [2], [11]. Traditionally, such forecasts have relied heavily on historical sales records and statistical trend analysis, based on the assumption that past patterns can inform future outcomes [2].

### 1.1.2 From Statistical Models to Machine Learning

Traditionally, companies relied on statistical models for sales forecasting, with the AutoRegressive Integrated Moving Average (ARIMA) model being a widely adopted example. ARIMA and other classical time-series approaches were extensively used in

retail forecasting for several decades due to their mathematical simplicity and flexibility [2]. These models employ historical sales information to generate forecasts and perform well in environments where demand patterns are stable. However, they exhibit inherent limitations. Most notably, they assume that underlying structures such as seasonality and trends persist indefinitely, and they struggle to incorporate external factors like changing market conditions, promotional activities, or unforeseen disruptions. Such constraints become particularly problematic when demand is volatile, for example in cases involving new products with no historical records or sudden shifts in consumer behavior [11].

With advancements in computational power and the exponential growth of data, forecasting has increasingly shifted towards machine learning (ML) approaches [2]. Unlike traditional statistical models, ML algorithms can automatically learn complex and nonlinear relationships within data, making them well-suited to capture the diverse factors influencing retail performance. By integrating historical sales with exogenous variables such as prices, holidays, and promotions, ML-based models can uncover patterns that handcrafted statistical models often fail to detect. Empirical studies and systematic reviews further confirm that AI- and ML-based forecasting techniques generally outperform traditional models across a wide range of applications [2], [4]. Nevertheless, this flexibility comes with trade-offs, as ML models require larger datasets, careful parameter optimization, and robust methodological practices in data preparation and algorithm selection to achieve optimal results.

### 1.1.3 From Machine Learning to Deep Learning

Machine learning methods such as Random Forests, Gradient Boosting, and Support Vector Machines have demonstrated utility in sales forecasting; however, their applicability to sequential, high-dimensional time-series data is limited. These algorithms typically assume feature independence and rely heavily on manual feature engineering (e.g., lagged variables or rolling statistics) to approximate temporal dependencies, which risks overlooking complex dynamics inherent in retail data [4], [11].

Deep learning models address these shortcomings by directly learning temporal dependencies and nonlinear interactions from raw or minimally processed sequences. Long Short-Term Memory (LSTM) networks were introduced to overcome the vanishing gradient problem in recurrent architectures and enable effective modeling of long-range dependencies [5]. Bidirectional LSTM (BiLSTM) further enhances this capability by processing sequences in both forward and backward directions, thereby capturing contextual information from past and future simultaneously [6].

More recently, Temporal Convolutional Networks (TCNs) have been proposed as an alternative, using causal and dilated convolutions to efficiently capture both short- and long-range dependencies with stable training dynamics [7]. Unlike traditional ML approaches, these deep learning architectures automatically extract salient temporal features, making them well-suited for volatile, nonlinear, and large-scale e-commerce environments [12].

### 1.1.4 Key Concepts

Extreme Gradient Boosting (XGBoost) is an advanced implementation of the gradient boosting framework that has become one of the most widely adopted algorithms for predictive modeling on structured data. It constructs an ensemble of decision trees in a sequential manner, where each new tree attempts to correct the residual errors of the previous ones, thus progressively improving model accuracy [8]. XGBoost distinguishes itself through system-level optimizations such as parallelized tree construction, regularization to mitigate overfitting, and efficient handling of sparse features, which make it highly scalable for large and complex datasets [8], [11]. In both forecasting competitions and real-world retail applications, XGBoost has consistently demonstrated superior performance by capturing nonlinear feature interactions and accommodating heterogeneous inputs (e.g., categorical features such as holidays combined with numerical features such as discounts) [2], [9]. These properties render XGBoost particularly effective in e-commerce sales forecasting, where predictive accuracy and computational efficiency are critical.

BiLSTM extends the Long Short-Term Memory (LSTM) architecture by introducing bidirectional processing, whereby the input sequence is analyzed in both forward and

backward directions. This design allows the model to learn dependencies not only from preceding time steps but also from future contexts, thereby enhancing its ability to capture long-range temporal relationships [5], [6]. Compared to standard recurrent neural networks, BiLSTM mitigates the vanishing gradient problem while retaining the capacity to model complex sequential dynamics. In sales forecasting, this dual contextual learning is particularly advantageous in capturing patterns influenced simultaneously by past behaviors (e.g., seasonality) and future-oriented factors (e.g., upcoming promotions). Consequently, BiLSTM has emerged as a robust architecture for handling volatile and nonlinear retail time series.

TCNs represent a convolutional alternative to recurrent architectures, specifically designed for sequence modeling tasks. They employ causal convolutions to preserve temporal order, ensuring that predictions at a given time step are influenced only by present and past observations. In addition, the use of dilated convolutions enables TCNs to capture long-range dependencies without requiring excessively deep networks, thereby improving computational efficiency [7]. Compared to recurrent models, TCNs offer advantages in terms of stable training dynamics, parallelization, and scalability to large datasets. For sales forecasting, TCNs can effectively model both short-term seasonal variations and long-term demand trends, offering a powerful alternative to LSTM-based methods in dynamic e-commerce environments [4], [7].

### 1.1.5 Data-Driven Forecasting in Modern E-Commerce

E-commerce growth has amplified the importance of data-driven forecasting, as online platforms such as Amazon must manage vast inventories, rapid fluctuations in consumer demand, and diverse determinants ranging from web traffic and marketing campaigns to external shocks. In this environment, accurate forecasting is directly tied to both customer satisfaction and operational efficiency. Companies must estimate stock requirements, balance inventory across warehouses, and schedule deliveries in accordance with forecasted demand. To achieve this, leading e-commerce firms have invested heavily in machine learning and big data analytics, which allow the integration of historical sales with contextual information such as macroeconomic indicators, competitor pricing, social media sentiment, and weather patterns [9].

The benefits of such approaches are substantial. Improved forecasting prevents stockouts of high-demand items and avoids overstock of slower-moving products, thereby reducing holding costs. Furthermore, it enables the implementation of adaptive strategies such as just-in-time replenishment and targeted marketing, driven by accurate demand prediction. Amazon provides a clear illustration of this transformation. Its earlier forecasting systems, built upon conventional time-series models, were effective for stable demand but proved inadequate for newly launched or highly seasonal products. In response, Amazon has progressively adopted artificial intelligence-driven forecasting models, leveraging machine learning and deep learning techniques to achieve higher predictive accuracy across its diverse product portfolio [10].

This evolution reflects a broader industry trend: data-driven forecasting is no longer optional but essential. Accurate sales prediction, underpinned by robust feature engineering and advanced algorithms such as XGBoost, has become foundational for strategic decision-making in modern retail. Accordingly, this project seeks to apply machine learning techniques to Amazon sales datasets, with the aim of demonstrating how data-driven predictive methods can enhance planning and operational efficiency in the e-commerce sector [9], [11].

### 1.2 Problem Statement and Motivation

With the rapid development of the e-commerce industry and the advent of the big data era, sales forecasting has become a critical determinant of both competitiveness and profitability. However, when faced with massive, complex, and noisy datasets, traditional statistical forecasting methods exhibit serious limitations in terms of data processing and adaptability [4].

A persistent challenge is the imbalance in sales distributions: products and categories often vary widely in demand, leading forecasting models to become biased toward high-volume categories while neglecting low-demand ones. This imbalance reduces the overall accuracy of forecasts, particularly for niche products where traditional models tend to underperform [4].

Moreover, e-commerce sales are inherently volatile, influenced by promotions, discounts, and seasonal campaigns, which introduce irregular fluctuations. Classical approaches such as ARIMA and exponential smoothing assume linearity and stationarity, making them inadequate for capturing nonlinear dependencies, seasonal peaks, and sudden spikes caused by promotional events [2][13].

For global e-commerce leaders such as Amazon, major shopping festivals like Black Friday, Cyber Monday, and Singles' Day (November 11) account for a disproportionate share of annual revenue. During these periods, companies face the challenge of allocating labor, logistics, and inventory to meet sudden demand surges. Earlier forecasting systems, relying primarily on aggregated trend analysis and rule-based heuristics, proved incapable of handling such scale and complexity, resulting in delayed and inaccurate predictions [3].

In summary, conventional sales forecasting methods struggle with three core limitations: (i) they are biased toward high-volume categories and fail to generalize across imbalanced product distributions [4], (ii) they lack the flexibility to model nonlinear and volatile demand patterns [2][13], and (iii) they exhibit reduced accuracy when forecasting niche or low-demand products due to imbalance in sales distributions [4] and (iv) they cannot cannot effectively address sudden demand surges and operational complexity during major shopping festivals such as Black Friday, Cyber Monday, and Singles' Day [3].These shortcomings underscore the urgent need for advanced forecasting frameworks—capable of leveraging modern machine learning and deep learning techniques—to deliver higher predictive accuracy and actionable insights for e-commerce enterprises.

### 1.2.2 Motivation

This project is motivated by the need to overcome the limitations of existing e-commerce forecasting systems, which struggle with limited data processing capability, imbalanced sales distributions, volatile demand patterns, and the complexity of large-scale shopping events [2][3][4][13]. To address these challenges, three advanced models are incorporated into the forecasting framework. Extreme Gradient Boosting (XGBoost) is selected for its ability to integrate lag features, rolling statistics, and

calendar effects, thereby improving accuracy across both high-volume and low-demand categories while providing interpretability through feature importance. Bidirectional Long Short-Term Memory (BiLSTM) is employed to capture both past and future temporal dependencies, making it effective in modeling nonlinear and volatile patterns caused by promotions and irregular campaigns. Temporal Convolutional Networks (TCN) are incorporated with the ability to learn from long historical windows using dilated causal convolutions, which enhance robustness against sudden surges during shopping festivals such as Black Friday, Cyber Monday, and Singles' Day. By combining these three approaches within a single unified framework, the project aims to deliver more accurate, explainable, and actionable forecasts that support inventory optimization, logistics coordination, and promotional strategy in e-commerce operations.

## 1.3 Research Objectives

The objectives of the proposed project are shown below:

1. To develop and implement three forecasting models, BiLSTM, TCN, and XGBoost, for short-term daily sales forecasting of seven days in advance using Amazon sales data.
2. To evaluate and compare the performance of these models using quantitative metrics including MAE, RMSE, MAPE and $R^2$ .
3. To develop an interactive Streamlit-based dashboard that allows users to upload datasets, select forecasting models, visualize predictions, and download results.
4. To provide actionable business insights and AI-generated recommendations from forecast results to support inventory planning and operational strategy.

## 1.4 Project Scope and Direction

This project focuses on the design and development of an e-commerce sales forecasting system using Amazon sales data as the case study. Specifically, the system leverages a publicly available Amazon sales transaction dataset (2017–2020) containing daily transactional records with attributes such as invoice date, item category, sales quantity, sales cost, and sales amount [28]. The scope covers data preprocessing, feature engineering, and the implementation of three forecasting models—BiLSTM, TCN, and XGBoost—each tailored to handle different aspects of time-series prediction. The

deliverables of this project include (i) trained forecasting models with evaluated performance, (ii) a comparative analysis of model accuracy using MAE, RMSE, MAPE and $R^2$, and (iii) an interactive Streamlit-based dashboard that integrates the models into a user-friendly interface.

The dashboard allows users to upload sales data in CSV format, generate short-term forecasts (7-day ahead), visualize the results through tables and charts, download the prediction outputs, and obtain business insights such as peak demand days and trend direction. In addition, the system incorporates an Artificial Intelligence (AI) Assistant Recommendation module, which automatically generates strategic suggestions (e.g., inventory allocation, promotion planning, and operational adjustments) based on the forecast results. This feature extends the system beyond prediction to decision-support, making it directly useful for managerial applications.

The target users of the system are business analysts, e-commerce managers, and decision-makers who require accurate and timely sales forecasts to support inventory planning, logistics coordination, and promotional strategy. However, the project scope does not extend to real-time data integration with Amazon's internal systems, product-level forecasting, or advanced ensemble methods beyond the three selected models. The work is limited to daily sales aggregation, and external factors such as competitor activity, macroeconomic indicators, or customer sentiment analysis are excluded from this study.

## 1.5 Contributions

The main contribution of this project lies in its integration of multiple advanced forecasting approaches—BiLSTM, TCN, and XGBoost—within a single unified framework tailored for e-commerce sales forecasting. While many existing studies focus on applying a single model or traditional statistical techniques, this project provides a comparative perspective that demonstrates the strengths and trade-offs of both deep learning and machine learning approaches when applied to complex, volatile, and imbalanced sales data. Unlike conventional systems that offer limited interpretability or fail to capture temporal dependencies [4][2][13], this project not only achieves higher forecast accuracy but also incorporates explainable features through

XGBoost. In addition, the development of an interactive Streamlit dashboard distinguishes this work from prior research by bridging the gap between technical forecasting models and practical business decision-making. The dashboard empowers users to directly interact with the models by uploading datasets, generating 7-day forecasts, visualizing predictions, and downloading results, making the system both accessible and actionable for non-technical business stakeholders. Furthermore, the dashboard is enhanced with an AI recommendation module that generates strategic suggestions based on forecast results, such as peak demand days and sales trend analysis. This project goes beyond pure prediction to deliver a decision support tool that can assist enterprises in inventory optimization, resource allocation, and promotional planning. Collectively, these contributions make the project not only a technical advancement but also a practical solution that enhances operational strategy for e-commerce businesses.

### 1.6 Report Organization

This proposal is organized into six chapters. Chapter 1 introduces the project, presenting its background, problem statement, motivation, objectives, scope, and contributions. Chapter 2 reviews related works on time series forecasting approaches, comparing traditional statistical methods with modern machine learning and deep learning models, and identifying research gaps in sales forecasting for e-commerce. Chapter 3 outlines the methodology by adopting the CRISP-DM framework, describing the phases of business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Chapter 4 presents the system design, including the overall workflow, system architecture, dashboard specifications, and planned features. Chapter 5 details the system implementation, covering dataset preprocessing, feature engineering, model development for BiLSTM, TCN, and XGBoost, and the integration of forecasting models into a Streamlit-based dashboard. Chapter 6 discusses the system evaluation and challenges, comparing model performance against objectives and highlighting implications for business decision-making. Finally, Chapter 7 concludes the proposal and provides recommendations for future improvements and potential real-world deployment.

# CHAPTER 2
# Literature Reviews

This chapter reviews the theoretical foundations, key technologies, and existing systems relevant to sales forecasting in the context of e-commerce. It begins with an overview of widely used forecasting technologies, including classical statistical approaches, machine learning methods such as XGBoost, and deep learning models such as BiLSTM and TCN, highlighting their historical development, technical innovations, and system architectures. Following this, the chapter evaluates several existing sales forecasting dashboards, identifying their strengths and weaknesses in terms of functionality, usability, and applicability to real-world business environments. A comparative summary is then presented to contrast the reviewed systems and technologies. Finally, related works from prior research are examined to position the current project within the broader body of knowledge, culminating in a set of critical remarks that outline existing gaps and motivate the proposed solutions of this study.

## 2.1 Review of Technologies

### 2.1.1 Introduction to XGBoost

XGBoost (eXtreme Gradient Boosting) is an open-source distributed library that provides an optimized gradient boosted decision tree implementation [13]. It belongs to the family of ensemble tree methods that build a prediction model incrementally by adding new decision trees one at a time to correct the errors of the existing ensemble. Unlike deep learning models that excel with unstructured data (images, text), XGBoost is ideally adapted for structured/tabular data and will typically get state-of-the-art results on those tasks [13]. By implementing an effective learning algorithm and system optimizations, XGBoost is able to deal with large-scale datasets and produce precise models with comparatively quick training times. XGBoost is now a de facto standard for most machine learning competitions and real-world applications in classification, regression, and ranking problems [13].

### 2.1.2 Historical Development and Popularity

XGBoost was first released in 2014 by Tianqi Chen as a standalone command-line tool for gradient boosted trees [14]. The popularity of XGBoost came very soon after it achieved the best performance in the 2014 Kaggle Higgs Boson Challenge [14], with additional momentum generated by publishing Python and R interfaces that introduced it into mainstream analytics pipelines [15]. By 2015, XGBoost was reigning supreme on Kaggle competitions, where it was utilized in more than half of the champion solutions, topping or augmenting deep learning techniques [13]. Its triumph wasn't limited to competitions like the KDDCup 2015, where every top-10 team utilized XGBoost [13]. Its competitive performance, its ability to scale, and its ability to handle missing data made XGBoost a de facto standard solution to industry tasks as varied as sales forecasting and fraud detection to click-through rate prediction [13]. Most fundamentally to its adoption in widespread use was its computational efficiency, reported at more than an order of magnitude better than earlier implementations of gradient boosting and scaling nicely to distributed clusters of billions of examples [13].

### 2.1.3 Key Technical Innovations in XGBoost

[13] showcased several technical improvements in XGBoost that led to model accuracy and computational speed that were radically better than traditional gradient boosting methods. Founded on advances in optimization theory and systems engineering, XGBoost utilizes an array of novel techniques for addressing real-world challenges in large-scale machine learning.

One of the innovations is the application of a second-order Taylor approximation. Classical gradient boosting, as put forward by [16], merely employs first-order gradient information to model decision trees sequentially. XGBoost, on the other hand, utilizes both first-order gradients and second-order Hessians of the loss function to construct trees, which leads to more accurate updates and quicker convergence in optimization [13]. In particular, XGBoost conducts a second-order Taylor expansion of the loss function at every boosting iteration, utilizing both the gradient gi and hi to compute the optimal contribution of the newly added tree.

The second significant innovation is the introduction of a regularized objective function to manage model complexity. Conventionally, gradient boosting only considers training error minimization, whereas XGBoost introduces an additional regularization term $\Omega(f)$ that penalizes both the number and the size of leaf weights [13]. This method effectively avoids overfitting by favoring simpler, more generalizable models, especially when dealing with noisy or small data.

Shrinkage, i.e., adjustment of the learning rate, is also included in the training process. After fitting each new tree, XGBoost shrinks its contribution by a factor $\eta$ ($0 < \eta \leq 1$), preventing over-correction in the early iterations and giving a boost to the model's generalization capability. While shrinkage was first suggested by [16], its imposition in XGBoost additionally strengthens its utility by combining it with regularized objectives for robustness and stability.

Among the key contributions of [13] is the sparsity-aware split finding algorithm. Noticing that a lot of real-world data sets contain missing or sparse values, XGBoost has native support for handling sparsity without explicit imputation. When building trees, the algorithm learns default directions for missing values via optimal loss reduction, thereby improving efficiency and maintaining predictive performance even with incomplete data.

Finally, XGBoost's emphasis on system-level optimization via parallelization and scalability differentiates it from previous approaches. The algorithm parallelizes split-finding over features by using a columnar data format, which enhances memory access patterns and supports multi-threaded computation on modern CPUs. Additionally, XGBoost provides distributed training on clusters and out-of-core computation support, allowing it to process datasets larger than a single machine's memory capacity [13].

Cumulatively, these improvements enable XGBoost to provide superior performance on big and challenging datasets, rendering it a critical tool in machine learning and predictive modeling tasks today.

### 2.1.4 System Architecture of XGBoost

XGBoost is not just a boosting algorithm, but also a modular system optimized for speed, scalability, and flexibility [13]. XGBoost contains a number of booster modules such as tree-based models (gbtree), linear models (gblinear), and dropout-boosted counterparts (dart). One such innovation is decoupling objective function and the boosting process, such that users can employ normal or their own objectives as long as gradients and Hessian can be computed [13]. During training, XGBoost employs an optimized greedy algorithm with second-order Taylor expansion, incorporates regularization and shrinkage to balance accuracy and complexity, and employs sparsity-aware split finding to natively handle missing values [13]. System-level optimisations complement performance with columnar data storage, multi-threaded computation, distributed training over clusters, and out-of-core computation for data sizes larger than memory. This synergy between algorithmic and engineering optimisations has made XGBoost one of the most scalable and efficient predictive modelling tools for structured data [13].

### 2.1.5 Introduction to BiLSTM

Bidirectional Long Short-Term Memory (BiLSTM) is a type of recurrent neural network (RNN) that extends the standard Long Short-Term Memory (LSTM) architecture by processing input sequences in both forward and backward directions [5], [6]. Unlike traditional RNNs, which only capture information from past time steps, BiLSTM also incorporates future context, enabling the model to learn richer temporal dependencies. This dual flow is especially beneficial in tasks where both past and future information influence the prediction, such as speech recognition, text processing, and sales forecasting. In time-series forecasting, BiLSTM is highly effective at modeling nonlinear and volatile data patterns, such as those caused by promotions or seasonal variations in retail.

### 2.1.6 Historical Development and Popularity of BiLSTM

The LSTM network was introduced by Hochreiter and Schmidhuber in 1997 to address the vanishing gradient problem in RNNs [5]. LSTMs quickly gained popularity in sequence modeling tasks across domains such as natural language processing, speech recognition, and financial forecasting. Recognizing that standard LSTMs could only

capture past dependencies, Schuster and Paliwal (1997) proposed the BiLSTM architecture, which augmented LSTMs with a backward-processing layer to capture future temporal information [6]. Over time, BiLSTMs demonstrated superior performance in sequential data applications, particularly where contextual information from both directions enhanced prediction accuracy. In retail forecasting, BiLSTMs have been successfully applied to model sales patterns with complex seasonality and irregular fluctuations, outperforming conventional statistical models in many empirical studies [17].

### 2.1.7 Key Technical Innovations in BiLSTM

BiLSTM inherits the core LSTM mechanism of gating units (input, output, and forget gates) that regulate the flow of information through the network [5]. These gates allow LSTMs to retain long-term dependencies while filtering irrelevant information, mitigating the vanishing gradient issue common in traditional RNNs. The key innovation of BiLSTM is the combination of two LSTM layers that process the sequence in opposite directions: one forward (past → future) and one backward (future → past). The outputs of these two layers are concatenated or merged at each time step, providing the model with a more complete representation of the sequence [6]. This bidirectional processing enables BiLSTM to capture dependencies that span across long horizons, which is particularly advantageous for volatile time-series like e-commerce sales, where both prior and upcoming events (e.g., promotions, holidays) affect demand.

### 2.1.8 System Architecture of BiLSTM

The architecture of BiLSTM consists of two parallel LSTM layers: a forward layer and a backward layer. Each layer has its own set of gating mechanisms and memory cells, and the outputs are combined at every time step [6]. Typically, the combined output is passed through additional dense (fully connected) layers to map the learned temporal features into predictions. In sales forecasting applications, BiLSTM networks are often trained on sequences of historical sales data, with input features including lagged demand, price changes, and calendar indicators. Training requires sequential backpropagation through time (BPTT), which can be computationally expensive but is manageable with modern GPUs. Despite the higher complexity compared to ARIMA

or XGBoost, BiLSTM offers significant improvements in accuracy for datasets with nonlinear patterns and long-range dependencies [17].

### 2.1.9 Introduction to TCN

The Temporal Convolutional Network (TCN) is a deep learning architecture designed for sequence modeling using convolutional operations instead of recurrence. Unlike RNNs or LSTMs, which process sequences step-by-step, TCNs leverage one-dimensional causal convolutions to capture temporal dependencies while preserving the order of inputs [7]. This design makes TCNs parallelizable, efficient to train, and less prone to issues such as vanishing gradients. TCNs are especially suited for time-series forecasting tasks that require modeling both short- and long-term dependencies, such as sales forecasting in e-commerce, where demand is influenced by daily trends, seasonal cycles, and promotional spikes.

### 2.1.10 Historical Development and Popularity of TCN

Convolutional neural networks (CNNs) were originally developed for spatial data, particularly computer vision tasks. Their extension to temporal data came with the development of dilated causal convolutions, which allowed CNNs to process sequences without violating temporal causality [7]. Bai et al. (2018) formally introduced the TCN architecture, demonstrating that it could outperform LSTM and GRU models in a range of sequence modeling benchmarks [7]. Since then, TCNs have gained traction in fields such as speech synthesis, anomaly detection, and financial forecasting. In the context of e-commerce, TCN-based models have been explored for demand prediction, with studies showing improved robustness and stability compared to recurrent architectures [9].

### 2.1.11 Key Technical Innovations in TCN

The key innovation of TCN is the use of dilated causal convolutions. Causal convolutions ensure that predictions at time t depend only on inputs from time t and earlier, thus preserving the temporal order of the data. Dilation expands the receptive field of the convolution by skipping input steps, allowing the network to capture long-range dependencies without requiring very deep architectures [7]. Residual connections are also incorporated into TCNs, which help stabilize training and improve gradient

flow. These innovations make TCNs highly scalable and efficient for long sequence modeling tasks, addressing some of the limitations of RNNs and LSTMs.

### 2.1.12 System Architecture of TCN

A typical TCN architecture consists of stacked dilated causal convolutional layers, residual connections, and normalization layers [7]. The dilation factor increases exponentially across layers, enabling the model to capture both local (short-term) and global (long-term) dependencies. For sales forecasting, input sequences of daily sales are passed through these convolutional layers, which extract hierarchical temporal features. The resulting features are often fed into fully connected layers for final predictions. Compared to BiLSTM, TCNs offer faster training due to their parallelizable structure, greater stability, and the ability to handle very long sequences without performance degradation. Recent studies applying TCNs in retail forecasting, including hybrid TCN-LSTM architectures, have reported significant improvements in predictive accuracy and robustness [9].

The summary of the critical remarks of previous works are shown at table 2.1.1.

| Technology | Strengths | Weaknesses | Relevance to Project |
|---|---|---|---|
| XGBoost [11], [13], [19] | - Highly effective on structured/tabular data<br>- Captures nonlinear interactions<br>- Robust to missing values<br>- Efficient and scalable for large datasets<br>- Strong interpretability compared to DL | - Relies on heavy feature engineering (lags, rolling stats, calendars)<br>- Limited ability to capture sequential temporal dependencies | Used as a baseline ML model for Amazon sales forecasting, providing robust performance and interpretability for business decision-making. |
| BiLSTM [5], [6], [20] | - Captures long-term temporal dependencies<br>- Processes sequences in both directions (past & future) | - Requires large datasets and high computational resources<br>- Training can be slow | Applied to time-series sales forecasting to capture bidirectional dependencies and improve accuracy for |

| | | | |
|---|---|---|---|
| | - Effective for volatile and nonlinear patterns<br>- Outperforms ARIMA and standard LSTM in many cases | (sequential BPTT)<br>- Less interpretable compared to ML models | volatile e-commerce demand. |
| TCN [7], [21] | - Uses dilated causal convolutions to capture both short- and long-range dependencies<br>- Highly parallelizable, faster to train than BiLSTM<br>- Stable and scalable for very long sequences<br>- Robust against sudden demand spikes | - Requires careful tuning of dilation factors and kernel size<br>- Lower interpretability compared to tree-based methods<br>- Relatively newer, fewer established retail applications | Implemented to evaluate convolutional sequence modeling, offering scalable and efficient forecasting for large-scale Amazon sales data. |

Table 2.1.1: Comparative Summary of Forecasting Technologies

## 2.2 Review of Existing System

### 2.2.1 Sales Forecasting Dashboard by Afroz Sameer Ahmad

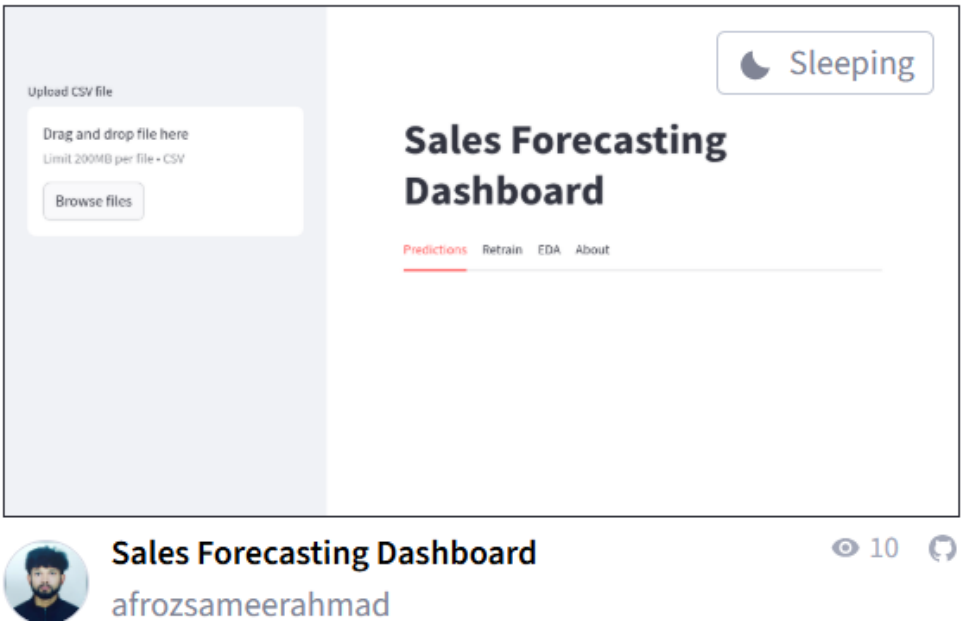



Figure 2.2.1 Sales Forecasting Dashboard developed by Afroz Sameer Ahmad [23]

As shown in Figure 2.2.1, the Sales Forecasting Dashboard provides a simple interface for uploading datasets and performing predictions. An existing sales forecasting application was identified from an online Streamlit dashboard developed by Afroz Sameer Ahmad [23]. The system integrates multiple forecasting models, including Random Forest, XGBoost, and ARIMA, providing users with several options for model comparison.

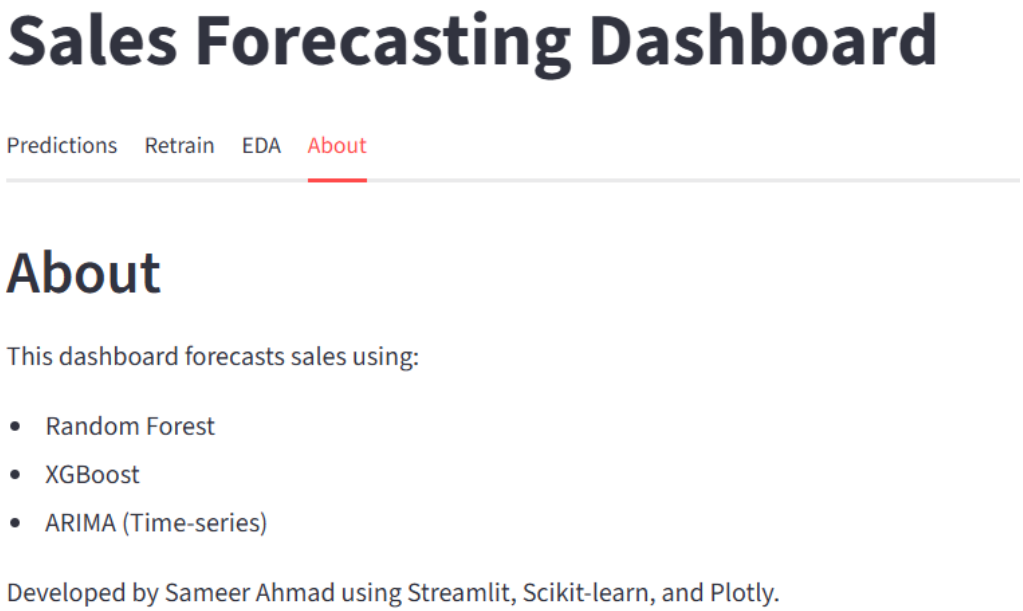



Figure 2.2.2 Sales Forecasting Dashboard – About section [23]

As shown in Figure 2.2.2, the “About” section of the Sales Forecasting Dashboard provides an overview of the forecasting models used, including Random Forest, XGBoost, and ARIMA. One key advantage of this system is its clean and minimalist interface. The dashboard is divided into modular sections—Predictions, Retrain, EDA, and About—which cover the essential workflow from data exploration to model retraining. Its simplicity ensures that first-time users are not overwhelmed, while the use of Streamlit enables real-time interactivity, allowing datasets to be uploaded and processed instantly. Additionally, offering multiple forecasting models grants flexibility for users to evaluate different approaches.

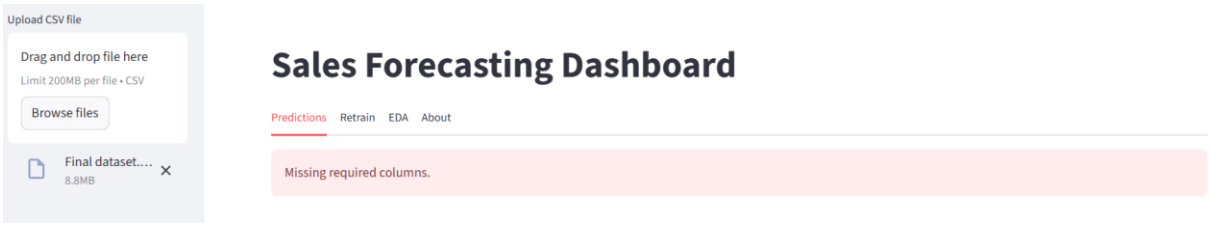



Figure 2.2.3 Screenshot of the Sales Forecasting Dashboard showing missing column error message [23]

As illustrated in Figure 2.2.3, when a dataset lacking the required columns is uploaded, the system only displays a generic error message without specifying which columns are missing. Despite these strengths, the system suffers from several usability and functional limitations. The most critical issue is its inadequate error feedback: when required columns are missing in the uploaded dataset, the system only displays a generic message ("Missing required columns") without specifying which columns are needed. This makes it difficult for users to correct their data.

### 2.2.2 Sales Forecast and Risk Dashboard by Vishal

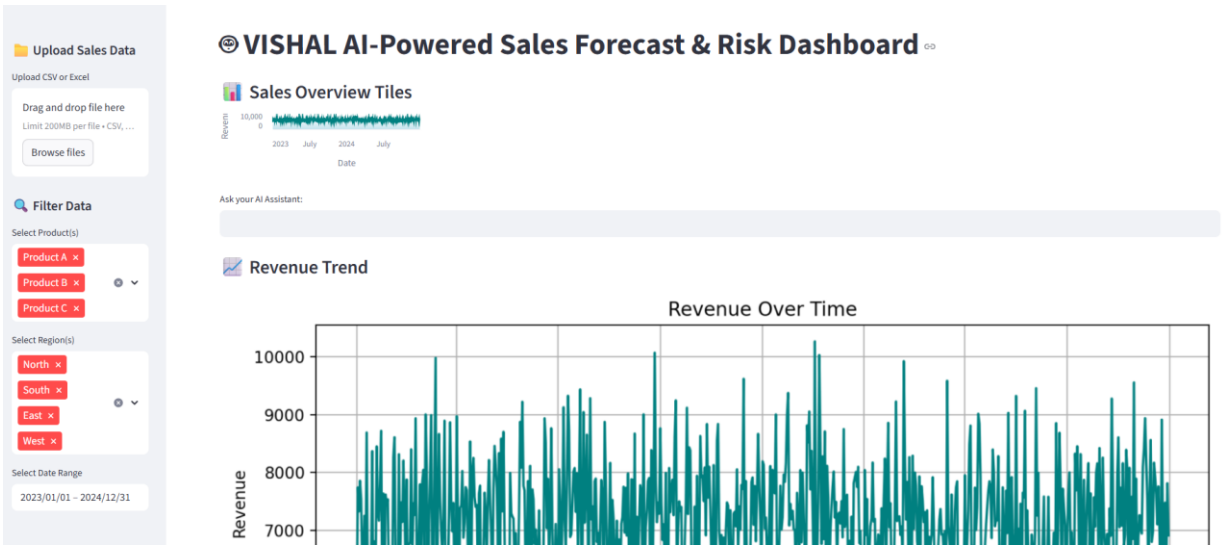



Figure 2.2.4 Sales Forecast and Risk Dashboard – overviews [24]

Another sales forecasting system was reviewed from an online Streamlit application developed by Vishal [24]. The dashboard integrates multiple features such as revenue trend visualization, regional and product-level filtering, risk warnings, and AI-generated insights, thereby offering users an interactive and multifunctional experience, as illustrated in Figure 2.2.4.

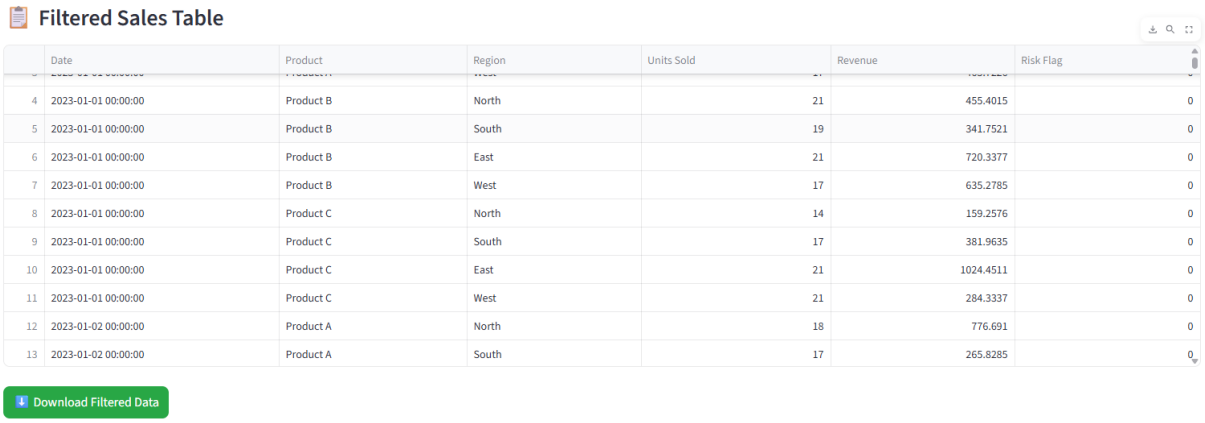



Figure 2.2.5 Sales Forecast and Risk Dashboard – Filtered sales table with sorting options [24].

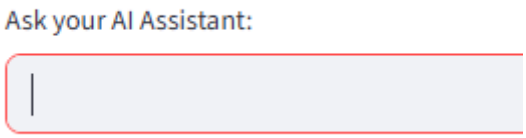



Figure 2.2.6 Sales Forecast and Risk Dashboard – AI Assistant input field [24].

The primary strength of this system lies in its high level of interactivity and multifunctionality. Users can dynamically filter data by product, region, and date range, while instantly viewing corresponding results through visualizations and tables. The dashboard also incorporates “Risk Warnings” and “AI Insights,” which provide additional contextual information about sales stability and potential risks. In addition, the sales table supports both ascending and descending sorting, enabling users to quickly organize data for easier interpretation, as shown in Figure 2.2.5. Another notable feature is the inclusion of an AI Assistant, which allows users to directly ask questions about the sales data, further enhancing interactivity and user engagement, as illustrated in Figure 2.2.6.


Figure 2.2.7 Sales Forecast and Risk Dashboard – KeyError when processing uploaded dataset [24]


Figure 2.2.8 Sales Forecast and Risk Dashboard – AuthenticationError when using AI Assistant [24]

Despite its interactive design, the system suffers from several critical drawbacks. Most importantly, it lacks transparency regarding its underlying methodology: the dashboard does not specify which predictive models are used, what type of datasets it supports, or the contexts in which it is most applicable. Without such information, users may find it difficult to evaluate the reliability of forecasts or to determine whether the tool aligns with their specific business needs. Furthermore, while the dashboard includes an AI Assistant, it appears to be non-functional due to configuration or authentication issues, as shown in Figure 2.2.7 and Figure 2.2.8, which limits its practical utility. These limitations significantly reduce the overall usefulness of the system for professional or academic applications.

### 2.2.3 Sales Analysis & Prediction Dashboard by JCortes

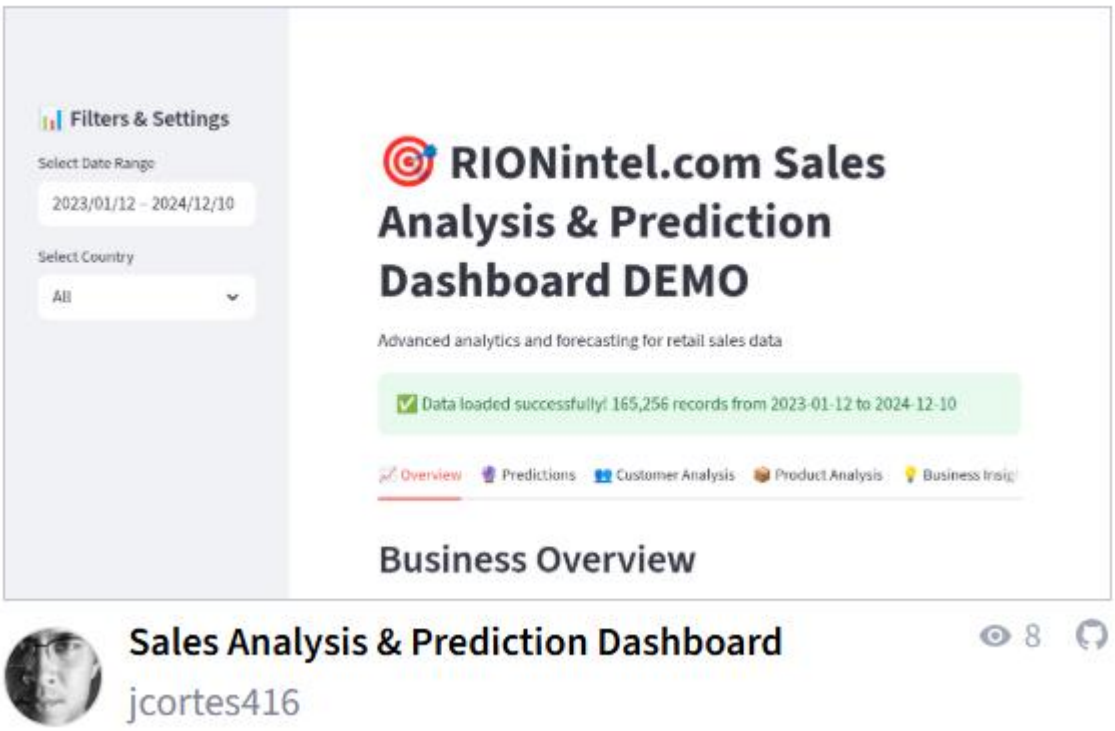



Figure 2.2.9 Sales Analysis & Prediction Dashboard – Overview page [25]

Another reviewed system is the Sales Analysis & Prediction Dashboard developed by JCortes [25]. This dashboard provides users with a range of analytical and forecasting functionalities, including sales predictions, customer analysis, product analysis, and business insights, as illustrated in Figure 2.2.9.

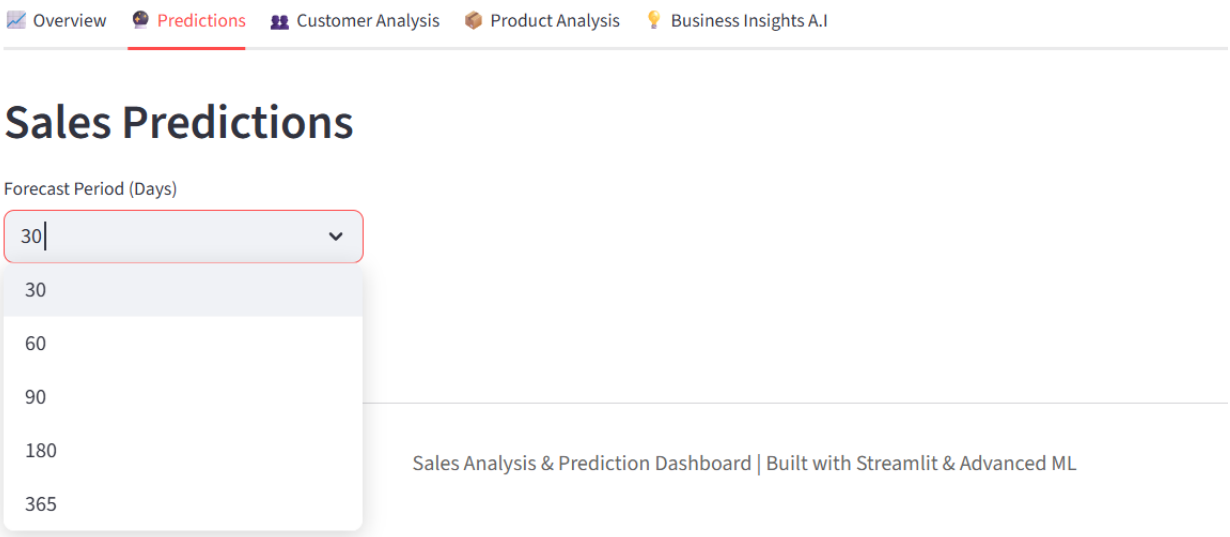


Figure 2.2.10 Sales Analysis & Prediction Dashboard – Forecast period selection [25]

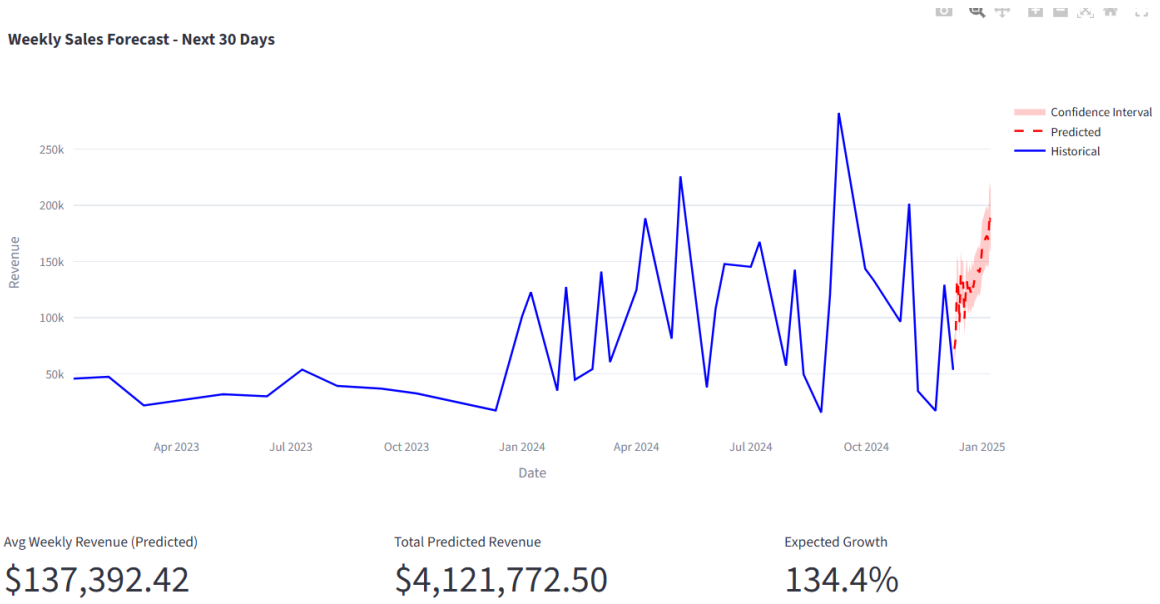



Figure 2.2.11 Sales Analysis & Prediction Dashboard – Weekly sales prediction graph (next 30 days) [25]

**Detailed Weekly Predictions**

| | date | prophet_prediction | xgboost_prediction | hybrid_prediction | confidence_lower | confidence_upper |
|---|---|---|---|---|---|---|
| 0 | 2024-12-10 | 106512.93 | 20644 | 72165.3594 | 61340.5586 | 82990.1563 |
| 1 | 2024-12-11 | 110460.14 | 37988.4414 | 81471.4609 | 69250.7422 | 93692.1797 |
| 2 | 2024-12-12 | 114414.25 | 162209.4688 | 133532.3438 | 113502.4922 | 153562.1875 |
| 3 | 2024-12-13 | 118383.44 | 147312.0938 | 129954.9063 | 110461.6719 | 149448.1406 |
| 4 | 2024-12-14 | 122377.36 | 55368.7695 | 95573.9219 | 81237.8438 | 109910.0078 |
| 5 | 2024-12-15 | 126406.29 | 155451.75 | 138024.4688 | 117320.7969 | 158728.1406 |
| 6 | 2024-12-16 | 122274.48 | 141452.4063 | 129945.6563 | 110453.8125 | 149437.5 |
| 7 | 2024-12-17 | 134607.86 | 127440.5313 | 131740.9375 | 111979.7969 | 151502.0781 |
| 8 | 2024-12-18 | 138796.03 | 39641.5781 | 99134.25 | 84264.1172 | 114004.3828 |
| 9 | 2024-12-19 | 143048.75 | 93888.4297 | 123384.6172 | 104876.9375 | 141892.3125 |

Figure 2.2.12 Sales Analysis & Prediction Dashboard – Weekly sales prediction table (next 30 days) [25]

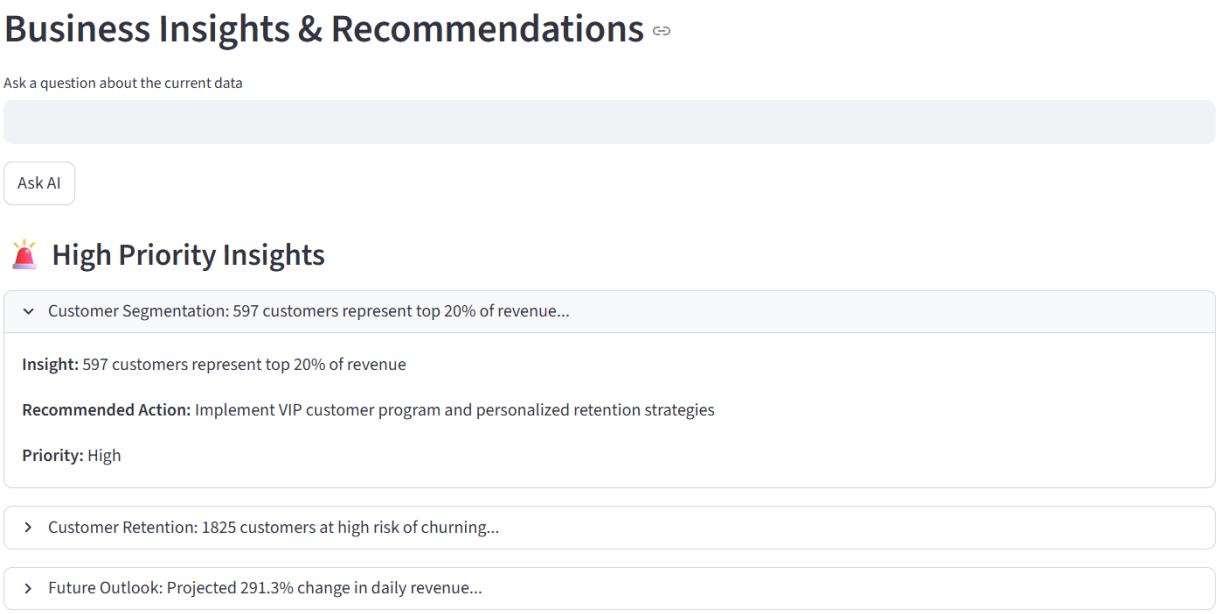



Figure 2.2.13 Sales Analysis & Prediction Dashboard – Business insights and recommendations [25]

A notable strength of this system is its ability to present actual prediction results in both graphical and tabular formats. Users can conveniently select different forecast horizons such as 30, 60, 90, 180, or 365 days depending on their business requirements, as shown in Figure 2.2.10. This flexibility enhances usability, allowing businesses to plan for both short-term and long-term sales strategies. Additionally, the dashboard integrates clear and intuitive data visualizations—such as line charts, tables to facilitate easy interpretation (Figure 2.2.11 and Figure 2.2.12). The inclusion of the Business Insights section is another advantage, providing simplified and prioritized recommendations for decision-making (Figure 2.2.13). Similar to other modern dashboards, it also features an AI Assistant, which enables users to interact with the system through natural language queries, further improving accessibility and engagement.

Despite its comprehensive design, the system has a significant limitation: it does not allow users to upload their own datasets for forecasting. Instead, it relies on preloaded data, which restricts its adaptability and practical application in diverse real-world business contexts.

### 2.2.4 Summary of the Existing Systems

The reviewed sales forecasting dashboards demonstrate varying levels of functionality, usability, and practical application. The first system by Afroz Sameer Ahmad focuses on simplicity and basic forecasting with Random Forest, XGBoost, and ARIMA models, making it accessible for beginners but limited by poor error feedback. The second system by Vishal enhances interactivity through filtering, risk warnings, and an AI Assistant, but lacks methodological transparency and suffers from technical issues with its AI component. The third system by JCortes offers a more comprehensive solution with flexible forecasting horizons, rich visualizations, and business insights, though it is constrained by the inability to upload custom datasets for forecasting.

Overall, the comparison highlights that while simplicity ensures accessibility, advanced interactivity and forecasting flexibility contribute significantly to practical usability. However, common shortcomings such as missing error guidance, lack of methodological clarity, and restricted dataset integration limit the broader applicability of these systems in real-world business environments.

The comparison of the strengths and weaknesses of the reviewed sales forecasting dashboards is presented in Table 2.2.1.

| System | Strengths | Weaknesses |
|---|---|---|
| Afroz Sameer Ahmad – Sales Forecasting Dashboard [23] | • Simple and beginner-friendly interface<br>• Supports Random Forest, XGBoost, and ARIMA<br>• Modular sections (Predictions, Retrain, EDA, About) | • Poor error feedback (“Missing required columns” without details)<br>• No advanced visualization or insights<br>• Limited business context integration |
| Vishal – Sales Forecast & Risk Dashboard [24] | • High interactivity (filters by product, region, date)<br>• Risk Warnings & AI Insights<br>• Sortable sales table<br>• Includes AI Assistant | • No transparency on predictive models or data requirements<br>• AI Assistant non-functional due to configuration issues<br>• Forecasting methodology unclear |

| | | |
|---|---|---|
| JCortes – Sales Analysis & Prediction Dashboard [25] | • Flexible forecast horizons (30/60/90/180/365 days)<br>• Rich visualizations (charts, tables, segmentation pie charts)<br>• Business Insights with recommendations<br>• AI Assistant included | • Cannot upload own datasets<br>• Fully dependent on preloaded data<br>• Limited adaptability to different business contexts |

Table 2.2.1 Comparison of Existing Sales Forecasting Dashboards

## 2.3 Related Works

### 2.3.1 Classical Approaches (ARIMA, SARIMA)

Classical time series prediction techniques like ARIMA (Auto-Regressive Integrated Moving Average) have been the norm for sales forecasting. Box and Jenkins formulated the ARIMA model, which gained popularity as it was a statistically based method to model seasonality and autocorrelations in time series [18]. ARIMA and its seasonal extensions (e.g., SARIMA) have been widely applied over the past decades since they are comparatively simple to interpret and comprehend [5], [6]. But there are some crucial limitations in these conventional models' application for sophisticated retail sales data. ARIMA is essentially a univariate approach – it works optimally for predicting one time series (i.e., the sales of a single item) and isn't designed easily for multiple inputs [18]. In practical life, fitting ARIMA on a large retailer (like Amazon, which sells hundreds of categories of products) would mean modeling individually for each product or group, a very time-consuming process that is difficult to make it capture cross-series information [18]. Furthermore, ARIMA presumes the time series may be stationary (typically through differencing) and also future patterns will adhere to comparable ones to the past. This makes it less suitable to handle structural change or sophisticated nonlinear trends in consumer demand. Traditional time series models also do not make room for external impacts (e.g. promotions, holidays, competitor activity) – even though ARIMA with exogenous inputs are feasible, they are still limited in their capability to model intricate interactions of several features [2], [11]. Problems such as outliers or missing values in sales data also challenge such classical methods, generally needing large amounts of pre-processing [2], [11]. Thus, while techniques such as ARIMA once represented a "best alternative" to forecast time series, their performance

in increasingly volatile retail situations always lags when all available signals and data are not leveraged [18].

### 2.3.2 Machine Learning Approaches (XGBoost, RF, LightGBM, etc.)

More recently, the introduction of machine learning techniques has significantly expanded the structured sales forecasting arsenal. Researchers have increasingly framed sales forecasting as a supervised learning regression problem rather than only a time series problem [16]. This is motivated by the capacity of more recent algorithms to automatically discover intricate patterns and take advantage of large, detailed datasets. Specifically, tree-based ensemble techniques have become more popular. Random forests (Breiman, 2001) and gradient boosting decision trees (Friedman, 2001) are able to capture nonlinear relationships and interactions that ARIMA is not equipped to deal with [13]. Chen and Guestrin (2016) proposed XGBoost, an effective gradient boosted trees algorithm that soon gained popularity at data science competitions and industrial applications [13]. XGBoost's regularization strategies and high performance enable it to deal with high-dimensional structured data (e.g., Amazon quarterly sales data with many product and seasonal variables) and not overfit, so it is very suitable for prediction problems with a high number of predictors. In contrast to ARIMA, where refitting for every time series is required, a single XGBoost model can be fitted to an entire set of products, implicitly learning in-series relationships and cross-series information through product IDs and time features. This type of "global" model has been demonstrated to enhance accuracy in retail forecasting [19]. For instance, Pavlyshenko (2019) illustrated that framing a multi-store sales dataset as a regression problem and using ensembles (like stacking several models) yielded more predictive power than typical time series approaches [13]. Machine learning models readily incorporate calendar effects (quarterly seasonality, holidays), prices, marketing, and other covariates into the forecasting process, removing the exogenous-factor limitation of ARIMA [13]. In addition, they can detect subtle nonlinear effects (i.e., diminishing returns of promotions or product demand-seasonal trend interactions) that linear models are unable to.

### 2.3.3 Deep Learning Models (BiLSTM, TCN, Hybrids)

While machine learning models such as XGBoost have demonstrated strong performance in structured sales forecasting, they still rely heavily on feature engineering and fail to model sequential dependencies intrinsic to time-series data. Deep learning models address these limitations by discovering temporal patterns and nonlinear relationships from raw sequences automatically. Recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks were introduced to model sequential dependencies and mitigate vanishing gradient issues [5]. Building on this, Bidirectional LSTM (BiLSTM) is a variant of the standard LSTM that runs the sequences in both forward and backward directions, thereby incorporating the past and future context simultaneously. This makes BiLSTM well-suited for retail sales forecasting, in which demand patterns are unpredictable owing to promotions, seasonality, and other time-related factors [6], [20]. At the same time, convolution-based models such as the Temporal Convolutional Network (TCN) were put forward. TCNs use dilated causal convolutions to effectively model short- and long-range dependencies, with stable training and amenability to large datasets [7]. Empirical studies have confirmed that TCNs achieve better results than recurrent models in sequence modeling, and recent applications in e-commerce forecasting, such as hybrid TCN-LSTM models, have demonstrated significant improvements over conventional methods [21]. These advances suggest deep learning and BiLSTM and TCN in particular as a promising direction for modern sales forecasting research in modeling complex, nonlinear, and long-range dependencies that cannot be effectively addressed by conventional and machine learning methods.

In general, the literature of sales forecasting exhibits a clear transition from common statistical models such as ARIMA and SARIMA towards machine learning models such as XGBoost and more recently deep learning models such as BiLSTM and TCN. Such conventional models are straightforward to interpret and easy but are not capable of dealing with nonlinear relationships, extraneous variables, and inter-series relationships in complex retail data sets [18]. Machine learning models improve predictability by modeling nonlinear interaction and accepting high sets of features, as evidenced by performance in competitions like M5 [13]. However, they are feature-engineering-dependent. Deep models overcome these issues by learning long-term dependencies

and automatically extracting temporal features, achieving deeper performance in chaotic and large-scale e-commerce environments [6], [7], [20], [21]. Together, these articles highlight the growing necessity to combine state-of-the-art ML and DL methodologies for robust, accurate, and effective sales forecasting in today's retail.

### 2.4 Critical Remarks on Previous Works

Conventional approaches such as ARIMA and SARIMA have long been the basis of sales forecasting, appreciated for their interpretability and seasoned statistical underpinning [18], [22]. They are effective at handling simple, stationary, and univariate time series with apparent seasonal or trend components. However, their reliance on stationarity assumptions, univariate modeling, and series-by-series parameter estimation makes them far less scalable to modern retail datasets comprising thousands of interconnected product categories, atypical seasonality, and external shocks. In particular, ARIMA is ill-suited to capture nonlinear interactions or large numbers of covariates such as price promotions, holiday effects, and competitor activity, which diminishes its applicability in large-scale e-commerce environments [2], [11].

Machine learning approaches, especially tree-based ensemble models like Random Forests, Gradient Boosted Trees, and XGBoost, have also significantly enhanced the forecasting landscape. They achieve their power from being capable of modeling nonlinear relationships, high-dimensional structured data, and implicitly learning cross-series dependencies by framing forecasting as a supervised regression problem [13]. Moreover, they are naturally accommodating of exogenous variables such as promotions, holidays, and price data with an edge over purely statistical methods. Empirical evidence from competitions such as M5 has confirmed that models such as XGBoost offer superior accuracy and robustness compared to classical techniques [19]. Yet, these methods continue to depend heavily on manual feature engineering, where lagged values, rolling means, and calendar features must be handcrafted by practitioners. This reliance contributes to added complexity and might not be able to detect hidden temporal patterns, making them less adaptive to volatile and rapidly changing sales environments.

Deep learning models represent the latest advancement in forecasting methodology with the capacity for automatically learning temporal dependencies and nonlinear relations straight from raw sequences. BiLSTM models are able to learn from future and past contextual information, while TCNs leverage dilated causal convolutions to model long-term dependencies effectively with stable training [6], [7], [20], [21]. They overcome inherent limitations of both statistical and machine learning methods in that they eliminate the need for hand-engineering temporal features and are resilient to irregular seasonality and sudden demand shifts. However, deep learning methods also pose challenges: they demand large volumes of training data, substantial computational resources, and diligent hyperparameter tuning to achieve stable performance. Their complexity also makes them less interpretable than traditional models, which can slow adoption in decision-making settings where transparency is required [13].

To overcome these gaps, this project adopts a comparative framework that integrates BiLSTM, TCN, and XGBoost, combining the strengths of both machine learning and deep learning approaches. Furthermore, by embedding the models within an interactive Streamlit dashboard, the system ensures that accurate, explainable, and actionable forecasts are directly accessible to business users for operational decision-making.

The summary of the critical remarks of previous works are shown at table 2.4.1.

| Research Work | Strengths | Weaknesses | Proposed Solutions |
|---|---|---|---|
| Classical Models (ARIMA, SARIMA) [18], [22] | - Strong statistical foundation<br>- Easy to interpret<br>- Effective for simple, stationary univariate series | - Poor scalability to large-scale retail datasets<br>- Struggle with nonlinear and volatile demand<br>- Cannot naturally integrate multiple covariates (e.g., promotions, holidays, competitor actions)<br>- Sensitive to missing/outlier data | Incorporate XGBoost and deep learning models (BiLSTM, TCN) that can model nonlinear interactions, cross-series dependencies, and handle external variables directly without strong |

| | | | |
|---|---|---|---|
| | | | stationarity assumptions. |
| Machine Learning Models (RF, XGBoost, LightGBM) [13], [19] | - Handle nonlinear relationships and high-dimensional data<br>- Learn cross-series relationships<br>- Flexible with external regressors (holidays, prices, promotions)<br>- Proven success in M5 competition | - Heavy reliance on manual feature engineering (lags, rolling means, calendar effects)<br>- May overlook hidden temporal dependencies<br>- Less adaptive to sudden spikes and irregular seasonality | Employ BiLSTM and TCN to automatically extract temporal dependencies and nonlinear patterns from raw sequences, reducing dependence on handcrafted features. |
| Deep Learning Models (LSTM, BiLSTM, TCN) [6], [7], [20], [21] | - Automatically learn temporal features<br>- BiLSTM captures both past and future contexts<br>- TCN efficiently models long-range dependencies<br>- Strong performance on volatile, sequential data | - Require large datasets and high computational resources<br>- Hyperparameter tuning is complex<br>- Lower interpretability compared to classical methods | Combine BiLSTM, TCN, and XGBoost in a comparative framework: DL models improve accuracy on volatile sequences, while XGBoost provides interpretability and efficiency. Integrating them into a Streamlit dashboard ensures practical usability for decision-makers. |

Table 2.4.1: Comparative Analysis of Forecasting Approaches and Proposed Solutions

# CHAPTER 3
# System Methodology/Approach

This chapter outlines the planned methodology for designing the Amazon sales forecasting system, following the CRISP-DM framework. Business understanding begins the strategy in order to define goals and scope, followed by data understanding in order to explore dataset characteristics. Data preprocessing will prepare and engineer features for modeling. Three models—BiLSTM, TCN, and XGBoost—will be developed and tuned during modeling. They will be validated against MAE, RMSE, MAPE, and $R^2$ before being deployed within a Streamlit dashboard that supports interactive forecasting and AI-driven recommendations.

## 3.1 Business Understanding Plan

Throughout the stage of business understanding, we plan to begin with a literature review and works related to it in an attempt to learn more about how sales forecasting problems have been addressed within traditional statistical approaches as well as from recent machine learning approaches. Through this review, we shall be able to clear the problem definition as well as the gaps our project shall cover. Based on the lessons learned, we will then move on to set the business objectives of the project so that these are aligned with Amazon's operational concerns such as inventory forecasting, promotion planning, and long-term sales forecasting. Next, we will define the scope of the project by specifying the levels of forecasting required and the intended uses of the results. Finally, we will define the expected deliverables from this step, which will be a structured problem definition, a set of business objectives, a clearly defined project scope, and a project plan to guide the subsequent steps of the CRISP-DM process.

## 3.2 Data Understanding Plan

The data understanding phase will focus on developing a thorough knowledge of the Amazon sales dataset to ensure its readiness for modeling. The plan is to begin by examining the dataset’s structure, attributes, and dimensions, identifying relevant numerical and categorical features. Data quality validation will then be performed to check for missing values, duplicates, or zero-critical values in the critical columns of

Sales Price, List Price, and Sales Cost Amount. Unnecessary columns that do not contribute to forecasting will be considered for removal. For temporal analysis capability, new temporal features such as Invoice_Year, Invoice_Month, and Invoice_Quarter will be included. The correlation analysis will also be planned with statistical measures such as Pearson's coefficient and heatmap visualization in order to analyze sales-related variables' correlations and thus identify the most influential features to predict. The output of this step is to develop a neat, formatted dataset suitable for further visualization and predictive modeling.

### 3.3 Data Visualization Plan

The data visualization phase will aim to explore the dataset through multiple perspectives in order to uncover sales patterns and business insights. The plan initially is to first develop item-level analysis such as best-selling items by sales and by margin through the utilization of bar charts, pie charts, and cumulative percentage tables. The same kind of visualization will then be developed at the category level to highlight high-performing product groups and their corresponding contribution towards revenue and profitability. Profitability will also be analyzed by ranking items and categories based on sales margin percentage, which will be depicted through the use of horizontal bar charts. For time-series reporting, sales trends will be accumulated by quarter and month to illustrate year-to-year trends and seasonal routines, with line and bar charts being utilized to identify highs and lows. Furthermore, sales by category over time will be visualized to understand which product groups dominate during different quarters. These visualization techniques will allow for the identification of seasonal trends, heavy-impact categories, and underperforming stock, offering a foundation for efficient forecasting and business operations.

### 3.4 Data Preprocessing Plan

Data preprocessing will be focused on transforming the raw Amazon sales dataset into a suitable form for different forecasting models. Since the project involves deep learning models (BiLSTM and TCN) and a machine learning model (XGBoost), different preprocessing methods will be developed in accordance with the needs of the models.

For the BiLSTM model, transactional data will first be aggregated into daily sales totals to construct a continuous time series. To reduce short-term fluctuations and noise, a seven-day moving average (MA7) will be used. Then, a sliding window system will be used where 30 days of past sales will be used as input to predict the following seven days. The data will be split chronologically into training, validation, and test splits, and MinMaxScaler will be fitted on the training data only to prevent data leakage. Both inputs and outputs will be scaled, and sequences will be reshaped into three-dimensional arrays (samples, timesteps, features) to match BiLSTM's expected input format.

For the TCN model, preprocessing will preserve raw daily sales figures without smoothing to maintain spikes and variability in demand. Once more, similar to BiLSTM, a 30-day window of input will be utilized to predict the following seven days. Scaling will be done by fitting MinMaxScaler to training targets, which would then be uniformly applied to validation and testing sets. The data will be reshaped in a suitable form to Conv1D layers (samples, timesteps, features). The data will be split into training, validation, and testing sets chronologically to preserve temporal consistency.

For the XGBoost model, preprocessing will focus on feature engineering instead of sequential processing. Upon aggregation of daily sales, calendar features such as day, month, year, weekday, weekend indicators, and month start/end indicators will be extracted. Lag variables (1, 2, 3, 7, 14, and 30 days) will be added to reflect autoregressive effects, and rolling statistics such as seven-day and thirty-day moving average and standard deviation will be added to reflect trends and volatility. Any missing values that result from lagging or rolling operations will be discarded, and the data will be split chronologically into training and test sets to preserve time order.

This preprocessing approach ensures that BiLSTM and TCN are given structured sequential inputs, while XGBoost is provided with engineered tabular features. By using preprocessing approaches tailored to model-specific requirements, the project aims at providing maximum predictive accuracy as well as interpretability.

### 3.5 Modeling Plan

The modeling phase will focus on developing three forecasting approaches: BiLSTM, TCN, and XGBoost, each chosen for their individual capabilities in handling temporal trends in sales data. The first plan is to establish baseline parameters for each model prior to systematically tuning their hyperparameters for maximizing precision as well as generalizability.

For the BiLSTM model, the intention is to create a stacked architecture of bidirectional LSTM layers, which will be able to learn both past and future relationships in sequential sales time series. The model will get trained with a sequence-to-sequence configuration, where a fixed size input window of previous days predicts a multi-day output window. Regularization techniques like dropout will be utilized, and the Adam optimizer will be utilized for efficient training. The model's hyperparameters like hidden units, dropout rates, learning rate, batch size, and training epochs will be experimented with using RandomizedSearchCV with cross-validation to find the best combination.

For TCN, the model will employ a convolutional configuration to leverage dilated causal convolutions for capturing both short- and long-term dependencies in sales sequences. The approach will be to begin with a simple architecture of stacked Conv1D layers and iteratively tweak parameters such as input window size, number of filters, learning rate, and batch size. The settings will be systematically tried using a grid search. The objective will be to determine whether increasing the input window or convolution depth brings improvement in validation accuracy.

For XGBoost, learning through feature-engineering-driven will take precedence. Engineered inputs such as lag features, rolling statistics, and calendar effects will be utilized as model inputs in order to capture autoregressive behavior and seasonality. Baseline configurations will involve a medium number of estimators and depth, and hyperparameter tuning will be conducted with time-series cross-validation using GridSearchCV. Learning rate, maximum depth, number of estimators, subsampling ratios, and regularization terms will be tuned in order to ensure predictive performance as well as preventing overfitting.

Overall, this modeling strategy will ensure that BiLSTM and TCN leverage sequential dependencies in the time series, while XGBoost provides interpretable forecasts based on feature engineering. Comparative evaluation across the three approaches will allow for identifying the most effective method for short-term daily sales forecasting.

### 3.6 Evaluation Plan

The evaluation of the forecasting models will be carried out using a set of well-established quantitative metrics that are suitable for both academic and business contexts. There are four metrics of comparison that will be utilized: Mean Absolute Percentage Error (MAPE), Coefficient of Determination ($R^2$), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). The reason for choosing these metrics is that they complement each other by capturing different aspects of model performance.

MAPE will be used to express forecast error as a percentage, so that it will be easily interpretable from a business point of view, with values under 20% typically being acceptable for practical forecasting purposes [11]. $R^2$ will measure how much variance in sales is explained by the models, with values above 0.7 typically being accepted as an indicator of strong explanatory power [26]. MAE will give a simple indication of the mean difference between actual and predicted sales figures, whereas RMSE will weigh larger errors more significantly, enabling model robustness to be evaluated against extreme variability. A normalized RMSE within 20% of the mean value will be taken as acceptable [27].

The plan is to evaluate all models—BiLSTM, TCN, and XGBoost—on a held-out test set that was not available during training or validation to evaluate their capacity to generalize. In addition, forward-predicting 7-day forecasts will be generated for all models to simulate real-world use, where future forecasts guide inventory decisions and operational planning. The model outputs will be compared not only by quantitative performance but also through visualization of predicted vs. actual sales curves to evaluate trend-following ability.

Finally, side-by-side comparison will be performed on the three models by showing their test-set performances in a summary table. The comparative evaluation will provide insight into the trade-offs between accuracy, robustness, and interpretability, and will allow the selection of the most appropriately suited forecasting model for short-term daily e-commerce sales forecasting.

### 3.7 Deployment

The deployment of the proposed forecasting system will be done by developing an interactive dashboard in Streamlit. The goal of the deployment stage will be to transform the trained forecasting models—BiLSTM, TCN, and XGBoost—into a easy-to-use decision-support application. This dashboard will allow users to upload their own Amazon sales datasets in CSV format, select a forecasting model of their choice, and immediately obtain short-term sales predictions.

The system will be designed to load the best-performing pre-trained models (saved in .keras and .pkl formats) and run the appropriate preprocessing pipeline depending on the selected model. Once the model is executed, the dashboard will display evaluation metrics such as MAE, RMSE, and $R^2$ to provide transparency on predictive accuracy. A 7-day ahead forecast will then be generated and presented both as a table of predicted values and as visualizations, such as line and bar charts, to facilitate intuitive understanding of sales fluctuations.

Beyond prediction, the deployment plan also includes integration of a business insights module. This module will also show key information such as the direction of trends (increasing or decreasing sales), the best selling day, and estimated average demand for the upcoming week. To maximize decision-making further, an AI assistant module will also be embraced. The assistant will review forecasted results and provide actionable recommendations on inventory planning, promotion timing, and staffing requirements. The users will be able to input their own business-related questions, the answers for which the assistant will give tailored suggestions.

Combining predictive modeling, visualization, and AI-driven suggestions in a single application, the deployed dashboard will act as a comprehensive decision-support

system. This manner, even non-technical users can leverage advanced forecasting models and derive actionable, business-oriented insights to guide operational strategy.

# Chapter 4
# System Design

The system design phase outlines the overall architecture, functional specifications, operations, and workflow of the proposed Amazon Sales Forecasting Dashboard. This chapter aims to describe how the three forecasting models—BiLSTM, TCN, and XGBoost—were integrated into a unified platform, structured to provide accurate predictions, business insights, and decision support for inventory and operational planning.

## 4.1 System Overview

The proposed system is implemented as a lightweight, interactive, and modular forecasting dashboard built with Streamlit. It integrates three machine learning and deep learning forecasting models (BiLSTM, TCN, XGBoost) into a single interface, enabling users to upload Amazon sales data, perform forecasts, and generate actionable insights. The dashboard was designed to support decision-making by providing sales forecasts, visualization of future trends, AI-generated recommendations, and exportable results.

## 4.2 Dashboard Specifications

The dashboard was developed with clear functional and technical specifications to ensure reliability, usability, and scalability. The frontend was implemented in Streamlit, providing an interactive web-based interface that can be launched in any browser without additional installation. This design ensures accessibility for both technical and non-technical business users.

At the backend, three forecasting models were integrated: BiLSTM, TCN, and XGBoost. The BiLSTM and TCN models were implemented using TensorFlow/Keras, enabling efficient handling of sequential time-series data. For BiLSTM, a 7-day moving average was employed to emphasize long-term sales trends, while TCN utilized raw daily sales values to capture short-term fluctuations. The XGBoost model was implemented using the scikit-learn API with joblib serialization, relying on engineered

features such as calendar indicators, lagged variables, and rolling statistics to achieve interpretable and accurate forecasts.

The system also incorporates automated preprocessing pipelines. For deep learning models, sales data is normalized into the [0,1] range with MinMaxScaler, while for XGBoost, feature engineering routines are executed to enrich raw data with time-aware variables. The evaluation module automatically computes MAE, RMSE, and $R^2$, presenting the results in real-time. Visualization modules generate both tabular reports and graphical outputs including bar charts and trend lines, enhancing interpretability.

In addition, the dashboard integrates an AI Assistant Recommendation Module, powered by a locally deployed large language model (Gemma3:4b) via API calls. This module transforms raw numerical forecasts into actionable business recommendations for inventory, promotions, and operational planning. To facilitate business use, forecast outputs can be exported in CSV format, enabling seamless integration into existing workflows.

Overall, the specifications prioritize lightweight deployment, modularity, and business practicality, making the system suitable for real-world decision support in e-commerce environments.

### 4.3 System Architecture

As illustrated in Figure 4.3.1, the architecture of the Amazon Sales Forecasting Dashboard begins with the user input layer, which enables users to upload sales datasets in CSV format. The system automatically validates the uploaded files to ensure that the essential fields, namely Invoice Date and Sales Amount, are included. This validation process guarantees that the data is structured appropriately for subsequent analysis, preventing errors and ensuring smooth workflow execution.

Once the dataset is validated, it is passed to the preprocessing module, where it is transformed according to the requirements of the selected forecasting model. For BiLSTM and TCN, the preprocessing pipeline normalizes the data using MinMaxScaler, applies a sliding window to generate time-series sequences, and

reshapes the data into three-dimensional arrays suitable for sequential models. For XGBoost, the preprocessing emphasizes feature engineering by creating calendar-based indicators, lag features, and rolling statistics, thereby enriching the dataset with predictive variables.

The processed data is then directed to the model selection and forecasting module, where the user chooses one of the three available models: BiLSTM, TCN, or XGBoost. Pre-trained models are loaded into memory, and the data is used to generate both back-testing predictions on historical values and forward-looking forecasts for the subsequent seven days. This modular design ensures that each model operates independently, while the system provides a unified interface for model execution.

Following forecasting, the results are handled by the evaluation and visualization module, which calculates performance metrics including MAE, RMSE, and $R^2$. These metrics are displayed in real-time to provide transparent evaluation of model accuracy. Additionally, forecasts are presented in both tabular and graphical forms, such as bar charts and line plots, which highlight essential insights including peak sales days, average forecasted values, and overall trend direction. These outputs transform raw numerical predictions into interpretable insights for business decision-making.

Finally, the recommendation and export module enhances the practical utility of the dashboard. An AI Assistant powered by a locally deployed large language model interprets forecast results and generates actionable recommendations for inventory planning, promotional timing, and operational adjustments. The system also allows forecasts and insights to be exported in CSV format, supporting integration into existing enterprise workflows. This final layer ensures that the system not only provides accurate predictions but also delivers strategic guidance, bridging the gap between technical forecasting and business application.

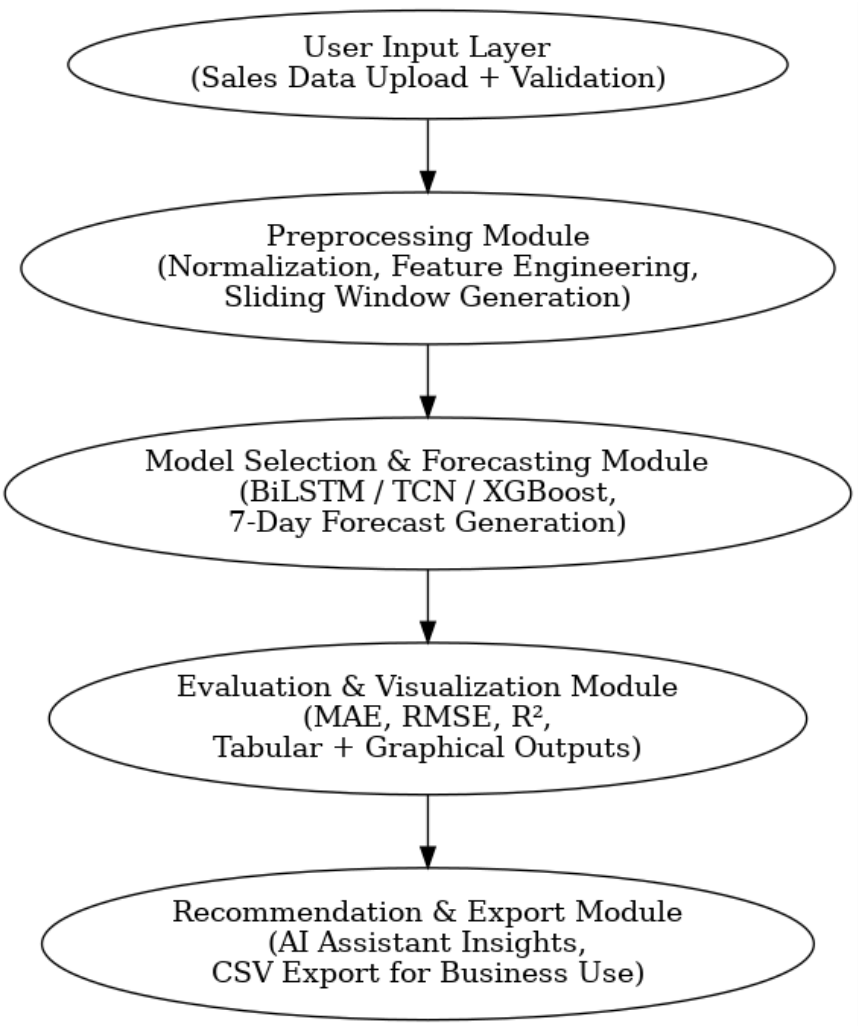



Figure 4.3.1 System Architecture of the Proposed Amazon Sales Forecasting Dashboard

**4.4 System Features and Functions**

The Amazon Sales Forecasting Dashboard was designed with several planned features and functions to ensure usability, interpretability, and practicality for business users. The first feature is the Data Upload and Validation Module, which allows users to upload sales records in CSV format. The system automatically verifies the presence of essential columns, namely Invoice Date and Sales Amount, ensuring that the dataset is correctly formatted before proceeding to model execution. This initial validation step reduces user errors and enhances the robustness of the workflow.

At the core of the system lies the Model Selection and Forecasting Module, which offers three forecasting options: BiLSTM, TCN, and XGBoost. Each model provides a

different analytical perspective, enabling users to explore both deep learning–based sequential predictions and feature-driven machine learning outputs. The module is designed to be flexible, allowing users to switch between models and compare their predictive capabilities within the same platform.

To ensure transparency and user confidence, the system incorporates a Performance Evaluation and Visualization Module. This feature automatically computes standard regression metrics such as MAE, RMSE, and $R^2$, and presents them in real-time. In addition, forecast results are displayed in both tabular format and visual representations such as bar charts and line plots. The visualization component highlights important business indicators including peak sales days, weekly averages, and forecast trends, thereby transforming raw predictions into actionable insights.

Another planned function is the AI Assistant Recommendation Module, which provides decision-support features by interpreting forecast outputs and generating business recommendations. Powered by a locally deployed large language model, this module delivers concise strategic suggestions related to inventory management, promotional campaigns, and operational planning. By integrating predictive analytics with prescriptive insights, the system bridges the gap between technical forecasting and practical business decision-making.

Finally, the dashboard offers a Result Export and Integration Module, which enables users to download forecast results and insights in CSV format. This feature supports integration with enterprise resource planning systems and other business intelligence tools, ensuring that the forecasting outputs can be directly utilized within broader organizational workflows.

### 4.5 System Workflow

As illustrated in Figure 4.5.1, the workflow of the Amazon Sales Forecasting Dashboard is designed as a structured sequence of steps that guide users from data input to actionable insights. The process begins with the data upload and validation stage, where users provide sales data in CSV format. The system checks for required fields (Invoice Date and Sales Amount) and rejects invalid files to maintain data integrity.

Validated data is then processed according to the chosen model. BiLSTM applies a 7-day moving average to smooth sales fluctuations, TCN uses raw daily values to capture short-term variations, and XGBoost relies on feature engineering with calendar indicators, lag values, and rolling statistics. These tailored preprocessing pipelines ensure that each model receives optimized inputs.

At the model execution stage, the selected model (BiLSTM, TCN, or XGBoost) generates both back-testing predictions and a 7-day forward forecast. Model performance is automatically evaluated using MAE, RMSE, and $R^2$.

The results are then presented in the visualization stage, combining tables and charts to highlight key business signals such as peak sales days, weekly averages, and overall trend direction. This makes forecasts more interpretable for decision-makers.

Finally, the recommendation and export stage provides AI-driven business insights, suggesting actions for inventory, promotions, and logistics. Forecast results and recommendations can be exported in CSV format for integration with business workflows. This streamlined workflow ensures that forecasting outputs are accurate, interpretable, and actionable for real-world decision-making.

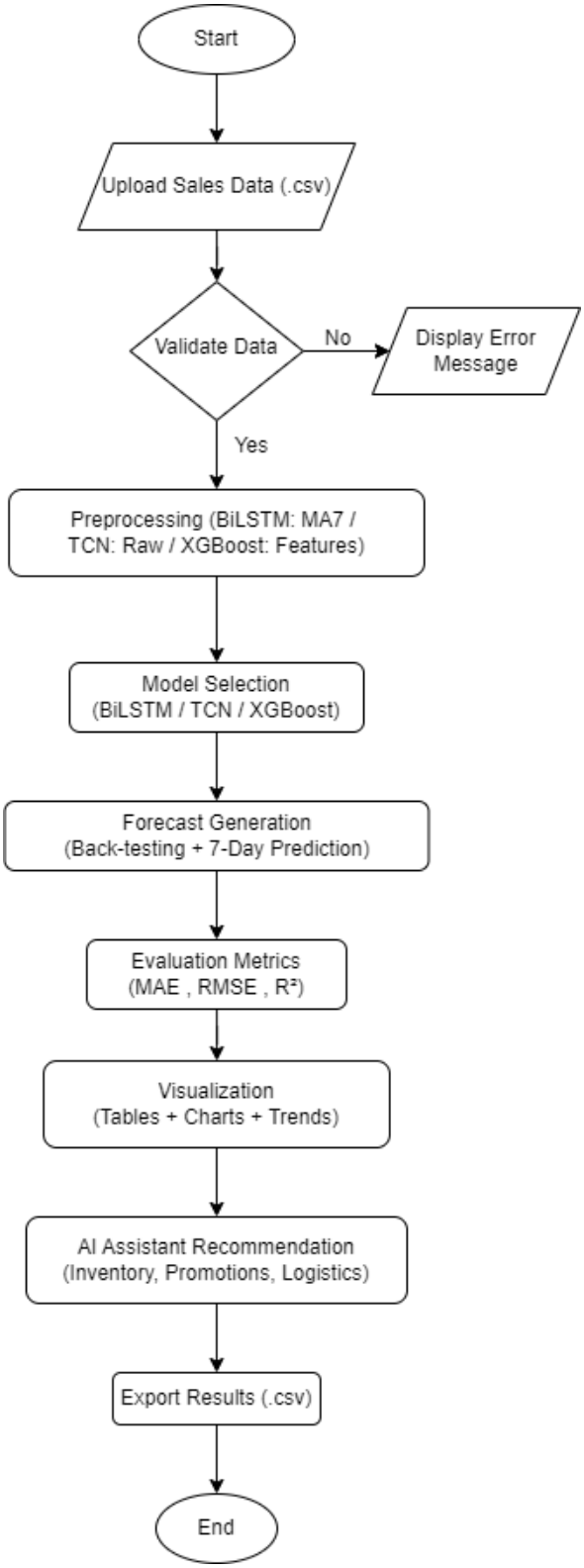



Figure 4.5.1 Planned Workflow of the Lung Cancer Detection System

## 4.6 System Design Considerations

The design of the Amazon Sales Forecasting Dashboard was guided by several key considerations to balance technical accuracy with business practicality. First,

lightweight deployment was prioritized by using Streamlit for the frontend and pre-trained models that can run efficiently on standard hardware, ensuring accessibility for small and medium enterprises. Second, modularity was emphasized in system architecture, allowing each component—data preprocessing, forecasting, evaluation, visualization, and recommendation—to operate independently while remaining integrated in a unified workflow.

Third, the system was designed for interpretability and usability. Forecast results are not only presented numerically but also visualized through charts and trend indicators, making outputs clear and actionable for non-technical users. The inclusion of performance metrics (MAE, RMSE, $R^2$) further enhances transparency and user trust. Fourth, business relevance was ensured by embedding an AI Assistant that translates predictions into prescriptive insights, directly supporting decisions on inventory, marketing, and logistics.

Finally, scalability was considered in the design. While the current implementation supports CSV input and local execution, the architecture leaves room for future integration with larger data platforms, cloud services, or enterprise-level ERP systems. This ensures that the dashboard can evolve from a proof-of-concept tool into a scalable decision-support solution.

# CHAPTER 5
# System Implementation

This chapter describes the Amazon sales forecasting system deployment through the CRISP-DM process. It begins with business understanding to define goals and scope, followed by data understanding and visualization to reveal patterns and trends. Data preprocessing is specified for BiLSTM, TCN, and XGBoost to prepare inputs unique to each model. The modeling and fine-tuning process is presented, highlighting baseline performance and improvements through hyperparameter optimization. Model comparison compares the three approaches on the basis of MAE, RMSE, MAPE, and $R^2$, while the deployment section presents the Streamlit dashboard that integrates forecasting, visualization, and AI-driven recommendations into a simple-to-use decision-support tool.

## 5.1 Business Understanding

In the business understanding stage of the CRISP-DM methodology, this project set its foundation by evaluating academic articles, and case studies on e-commerce forecasting. The evaluation highlighted the limitations of traditional statistical methods and justified the application of advanced machine learning and deep learning algorithms such as XGBoost, BiLSTM, and TCN that can better understand nonlinear dynamics, long-term dependencies, and volatile sales fluctuations. Based on these findings, the project made its background and problem statement and emphasized problems introduced by Amazon sales data such as promotional spiking, seasonal campaigns, and biased demand patterns.

The following project objectives were established: to design and deploy BiLSTM, TCN, and XGBoost models for seven-day short-term daily sales forecasting with Amazon sales data; to compare and assess their performance through MAE, RMSE, and $R^2$; to create an interactive Streamlit-based dashboard for data upload, model selection, forecast visualization, and result download; and to generate actionable business insights and AI-driven recommendations from forecast results to assist in inventory planning and operational strategy.

The scope of the project was restricted to daily sales forecasting and included data preprocessing, feature engineering, model training, model evaluation, and deployment. Competitor activity, macroeconomic, and customer sentiment were excluded as external variables. Performed forecasting models with evaluated performance, comparative accuracy analysis, and a Streamlit-based dashboard with observations such as sales trends, peak demand days, and AI-based recommendations were the primary deliverables. In general, the business knowledge phase ensured that the project not only addressed a research problem in academia but also produced practical tools for strategic decision-making for e-commerce.

## 5.2 Data Understanding

Data understanding began with the examination of the dataset structure, identification of missing values, and elimination of unnecessary columns such as Item Class and Item Number. Rows with missing or zero-critical values in key fields such as Sales Price, List Price, and Sales Cost Amount were eliminated for data quality maintenance. Time-based features such as Invoice_Year, Invoice_Month, and Invoice_Quarter were created for simpler seasonal analysis. Then, a cleaned dataset (pf1) was created that contained only numerical and categorical features relevant to sales forecasting.

A Pearson correlation heatmap shown at figure 5.2.1 was created to help visualise feature correlations. Sales Amount and variables such as Sales Quantity (r = 0.83), Sales Cost Amount (r = 0.99), and Sales Amount Based on List Price (r = 0.94) showed substantial positive correlations, suggesting that these are important factors influencing overall income. A significant performance measure, sales margin amount also showed a strong association with sales amount (r = 0.97) and sales cost amount (r = 0.94). The weak link between List Price and Sales Price and target metrics, on the other hand, indicates that price by itself would not be sufficient to drive sales if quantity or discounts are not taken into account. These realisations were essential for directing feature selection in later stages of the modelling process.

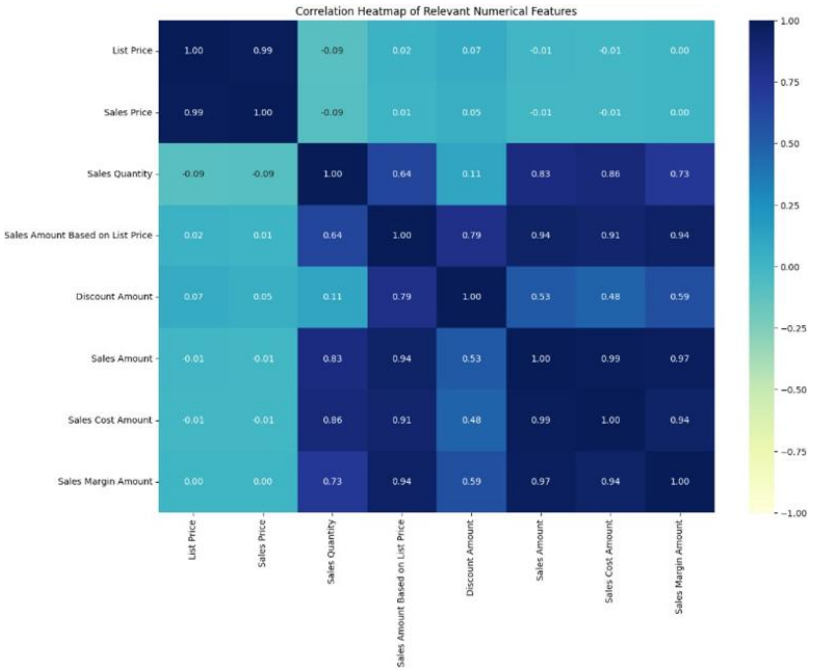



Figure 5.2.1 Pearson Correlation Heatmap

## 5.3 Data Visualization

Data visualization was done in various segments to analyze the dataset from different perspectives and uncover actionable business insights:

### Part 1: Top 10 Items by Sales and Margin

The first analysis focused on identifying the top 10 individual items by both sales amount and sales margin amount. With pie charts and grouped bar charts, this part helped visualize which single products contributed most to overall revenue and profit. Cumulative percentage tables were also built to show the proportion of sales/margin these top items represented, with an "other" group for the remaining items. This helped understand item-level concentration and long-tail effects better.

### Part 2: Top 10 Categories by Sales and Margin

In the second part, the same process was repeated at the category level. Sales and margin figures were rolled up at the category level, and the top 10 performers were determined. Bar and pie charts plotted their contribution to the overall performance. Tabular summaries gave their percentage shares and also showed the rolled-up share of

other remaining "other" categories. This view supports operational decisions such as targeting specific product categories in inventory planning.

**Part 3: Top 10 Items and Categories by Sales Margin Percentage**

Here, profitability was analyzed by calculating sales margin percentages (margin as a percentage of sales) of items and categories. The top 10 performers with the highest margin percentages were graphed in horizontal bar charts to give an idea of which products/categories are most cost-effective to sell. This is helpful in strategic pricing as much as marketing decisions.

**Part 4: Sales Amount Over Time (Monthly & Quarterly Trends)**

Temporal analysis was conducted by aggregating sales by month and quarter, both overall and broken down by year. Bar plots and line graphs were drawn to reveal seasonality and inter-annual trends. Further breakdowns by category over time were graphed to visualize the performance of each category over months and quarters.

To get a better understanding of sales trends and help develop the forecasting model, below are some series of visualizations with quarterly tendencies. The investigation began with an overview of total sales throughout the four quarters. From the bar chart of total quarters, it was observed that Q1 always registers the highest sales compared to the other quarters significantly. This suggests strong consumer activity at the beginning of the year, maybe due to season restocking or holiday consumption patterns. On the other hand, Q2 had the lowest overall sales, so it is most likely a low-demand quarter where marketing interventions can be targeted to improve performance. Figure 5.3.1 shows the total sales amount by quarter.

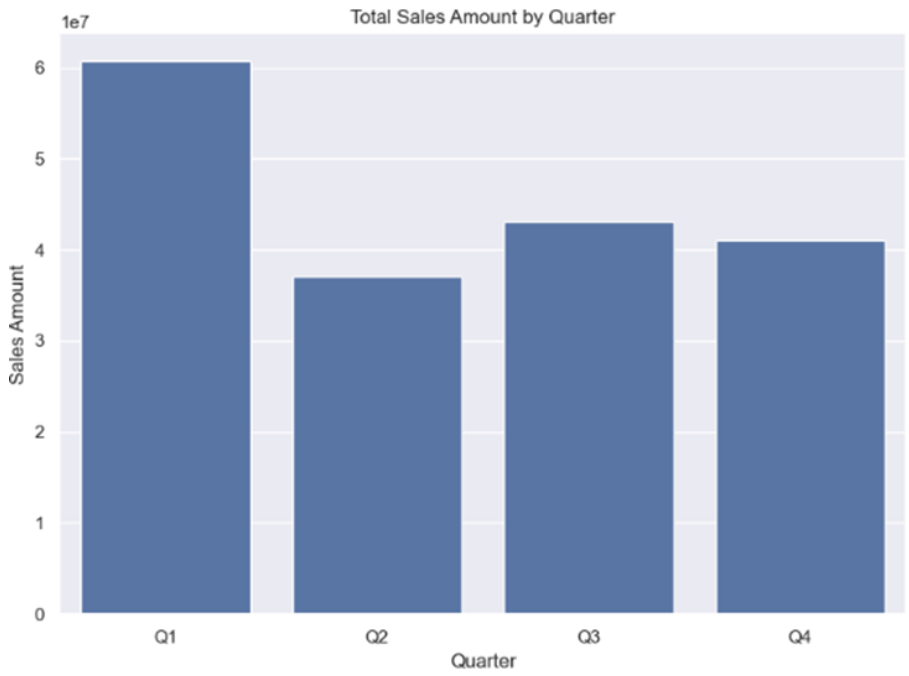



Figure 5.3.1 The Total Sales Amount by Quarter

Further year-over-year segmentation showed Q1 sales equally strong in years 2017, 2018, and 2019, further strengthening its seasonality leadership. Somewhat unexpectedly, Q3 too displayed similarly steady performance, while Q4 demonstrated mixed trends, potentially influenced by year-end sale promotions or seasonal shopping patterns different every year. Such year-over-year comparisons are particularly crucial in helping to ascertain the most steady quarters and make sure that the model captures regular and anomalous trends alike. Figure 5.3.2 shows the quarterly sales amount by year.

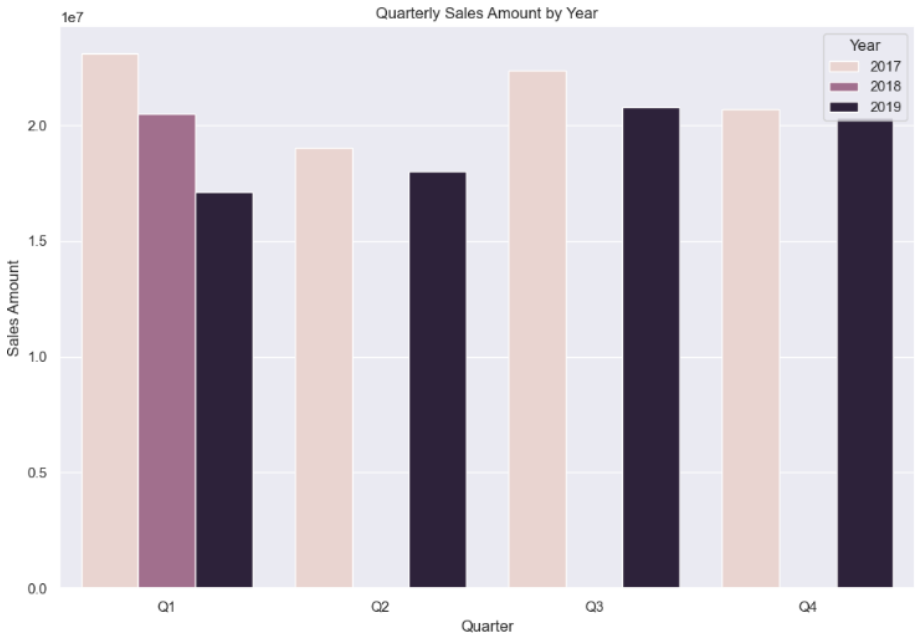



Figure 5.3.2 The Quarterly Sales Amount by Year

The pattern of quarterly sales by category demonstrates clear seasonality of demand from consumers. Canned Food leads every other category in all seasons of the year, reaching its peak in Q1 and Q3, dropping slightly in Q2 and Q4. Meat and Beverage show a similar trend, showing peak functionality in Q1 and Q3, suggesting that these months are optimum for higher consumption or marketing. Dairy, Frozen, and Bakery experience steady mid-range performance during the year, with Bakery experiencing a big surge in Q4, which necessarily has to be due to year-end holiday season demand. Overall, Q2 experiences overall decreasing sales across all the categories, indicating a post-peak season period of cooling off. In contrast, Q4 experiences partial recovery for some of the categories such as Bakery, Condiments & Spices, and Desserts. On the other hand, categories such as Oil, Legumes, Nuts & Seeds, and Soup have low volatility and sales across all quarters, reflecting low demand or product obsolescence. These findings suggest that operational focus needs to be on top-performing categories in high-performing quarters (Q1 and Q3), cost-saving initiatives and stock reductions in Q2, and targeted promotions in Q4. Lower-performing categories can maybe justify reducing SKUs or testing seasonal repackaging or bundling initiatives. Figure 5.3.3 shows the quarterly sales trend by category.

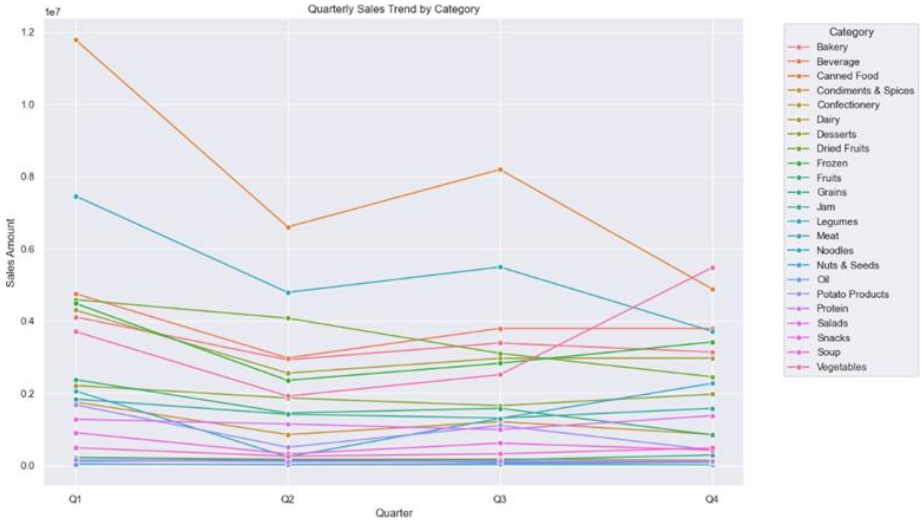



Figure 5.3.3 The Quarterly Sales Trend by Category

For Q1, Canned Food was the top-performing category and clearly dominated the sales chart. Its consistent demand is explained by the fact that it has a long shelf life, is easy to store, and is suitable for bulk purchase, making it a staple stock for the majority of consumers. Meat ranked second, reflecting strong seasonal demand that may be driven by festive consumption patterns or promotional offers that are usually available at the beginning of the year. Following these two are a group of middle-ranking high-performing categories that include Beverage, Dried Fruits, Frozen, Dairy, Bakery, and Vegetables. These categories are a mix of everyday essentials—such as Dairy and Vegetables—and seasonal or periodic demand items such as Frozen products and Dried Fruits. Their performance suggests a trade-off between usual consumption needs and seasonal highs, indicating the need to have a well-diversified and strategically stocked inventory at the start of the year. Figure 5.3.4 shows the category sales ranking in quarter 1.

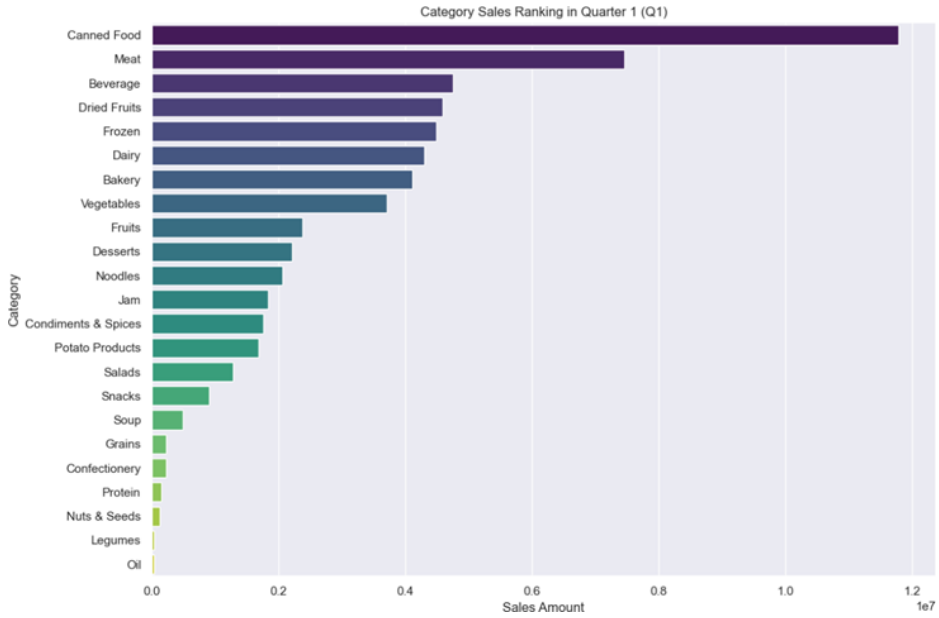



Figure 5.3.4 The Category Sales Ranking in Quarter 1

Last but not least, the graphical outcomes clearly demonstrate that Q1 is strongest in terms of overall sales, while Q2 is weakest, maybe worthy of promotional efforts. Among all product categories, the 'Canned Food' category realizes the strongest overall sales, particularly in Q1 and Q3, followed by the 'Meat' and 'Beverage' category. These findings directly influence the model's quarterly forecast approach and business operation strategy design in the dashboard.

## 5.4 Data Preprocessing

For the BiLSTM model, preprocessing started with aggregating the transactional data into daily sales totals to construct a continuous time series. A seven-day moving average (MA7) of daily sales was computed and used as the main input sequence in an attempt to reduce noise and smooth changes. A sliding window approach with a window length of 30 days and an output horizon of seven days was utilized to generate multi-output supervised sequences, for which each input window of 30 days forecasted the next seven days. The data were partitioned into training, validation, and test sets in an 80:10:10 ratio, with the validation set being sampled from the training set. Significantly, MinMaxScaler was only fit on the training set to prevent data leakage and subsequently used uniformly across all validation and test sets. Inputs and outputs were both scaled, and generated sequences were reshaped into three-dimensional arrays (samples, timesteps, features) so as to be in accordance with the expected input format of BiLSTM.

For the TCN model, daily sales data in raw form was used as it was without computing moving averages so the model could learn from typical variations and spikes in demand. A 30-day sliding window was utilized to forecast values for the subsequent seven days, first applied to the series without scaling to prepare the target values for fitting the scaler. The MinMaxScaler was applied only to the training targets (y_train_raw) and then applied to scale the entire dataset. The input sequences were then rebuilt after scaling for ensuring parity between features and targets. The sequences were reshaped into the form required by the Conv1D layer (samples, timesteps, features) in order to enable the TCN to learn both long-range and short-range dependencies using dilated causal convolutions. The data was chronologically divided into 70% training, 10% validation, and 20% testing.

For the XGBoost model, the preprocessing depended on feature engineering instead of sequence modeling. Following aggregation of daily sales values, calendar features were extracted such as day, month, year, weekday, weekend flags, and month start/end indicators. Lag variables at 1, 2, 3, 7, 14, and 30 days were added to capture autoregressive effects, and seven-day and thirty-day moving averages and standard deviations were added as rolling statistics to capture recent trend in demand and

volatility. Lag- and rolling-induced missing values were removed, and the data were chronologically divided into a training set and a testing set, where 80% of the data were reserved for training. This preprocessing allowed XGBoost to combine temporal patterns and calendar effects within a tabular structure, enabling interpretable and efficient forecasts. This table 5.4 summarizes the preprocessing steps applied to each forecasting model together with their intended purposes. It highlights the differences between sequence-based preprocessing used in BiLSTM and TCN, and feature-engineering-based preprocessing applied in XGBoost.

| **Model** | **Preprocessing Method** | **Purpose** |
|---|---|---|
| BiLSTM | - Aggregated transactional data into daily sales totals.<br>- Applied 7-day moving average (MA7) to smooth short-term fluctuations.<br>- Used a sliding window (30 days → 7 days ahead) to create multi-output sequences.<br>- Split into train (80%) / validation (10%) / test (10%), ensuring temporal order.<br>- Applied MinMaxScaler, fitted only on training data, then scaled val/test.<br>- Reshaped sequences into 3D arrays . | - Provide consistent daily time series.<br>- Reduce noise while preserving overall trend.<br>- Prepare supervised learning sequences for sequence-to-sequence prediction.<br>- Prevent data leakage and preserve temporal integrity.<br>- Normalize values for stable gradient-based training.<br>- Match BiLSTM's expected input format. |
| TCN | - Aggregated data into daily sales totals (no smoothing).<br>- Generated raw input-output sequences (30 days → 7 days ahead).<br>- Fitted MinMaxScaler only on training targets, then scaled the entire dataset.<br>- Re-created sequences using scaled values for consistency.<br>- Reshaped data into Conv1D input.<br>- Split into 70% train / 10% validation / | - Preserve raw variability and spikes in sales.<br>- Capture both short-term and long-term dependencies.<br>- Avoid data leakage while ensuring correct scaling.<br>- Maintain alignment between inputs and scaled targets.<br>- Enable TCN's dilated causal convolutions. |

| | | |
|---|---|---|
| | 20% test. | - Ensure fair model evaluation with chronological splits. |
| XGBoost | - Aggregated daily sales totals.<br>- Created calendar features (day, month, year, weekday, weekend, month start/end).<br>- Added lag features (1, 2, 3, 7, 14, 30 days).<br>- Computed rolling statistics (7-day and 30-day averages and standard deviations).<br>- Dropped missing values from lag/rolling operations.<br>- Split chronologically into 80% train / 20% test.<br><br>- Aggregated data into daily sales totals.<br>- Extracted calendar features (day, month, weekday, weekend, month start/end).<br>- Created lag features (1, 2, 3, 7, 14, 30 days).<br>- Added rolling statistics (7-day and 30-day averages and standard deviations).<br>- Dropped missing values from lag/rolling ops.<br>- Split data into chronological train/test sets. | - Provide a structured feature set for tabular learning.<br>- Encode seasonal, weekly, and monthly patterns.<br>- Capture autoregressive dependencies in demand.<br>- Reflect recent demand trends and volatility.<br>- Ensure valid samples without NaNs.<br>- Preserve temporal order and avoid data leakage. |

Table 5.4 Data Preprocessing Steps for BiLSTM, TCN, and XGBoost

## 5.5 Modeling and Fine-Tuning

The suggested project adopted BiLSTM, TCN, and XGBoost as the predictive frameworks based on their respective strengths. BiLSTM was selected since it excels in extracting nonlinear temporal patterns in short-term sales predictions, while TCN provides a high-performance convolutional alternative capable of extracting both short- and long-term patterns. XGBoost was introduced as a machine learning baseline that leverages engineered features in order to give explainable and efficient predictions. Together, these models enable all-rounded comparison of deep learning versus machine learning approaches,

### 5.5.1 BiLSTM

The model architecture was two stacked bidirectional LSTM layers of 128 and 64 hidden units respectively, with dropout layers (0.3) in between for regularization. There was a fully connected dense output layer for prediction of the seven-day result. The model was optimized using the Adam optimizer and Huber loss function, which is robust to outliers but sensitive to smaller errors.

It was trained up to a maximum of 50 epochs with a batch size of 32, with early stopping and learning rate(0.001) reduction callbacks to prevent overfitting and stable convergence. Training and validation set testing gave good accuracy, where training $R^2$ = 0.578, and validation $R^2$ = 0.521. As illustrated in Figure 5.5.1a, the validation forecast generally followed the upward trend of actual sales, but it exhibited higher fluctuations and consistently tended to overestimate the demand. Similarly, Figure 5.5.1b shows the full time series comparison between original sales, smoothed MA7 values, and the BiLSTM forecast. The model closely followed the smoothed series and successfully predicted an increasing trajectory for the next seven days, although short-term spikes in the raw data were not fully captured due to the smoothing process. The forecast table confirmed a steady growth in predicted sales, starting from 290,161 on the first forecasted day and reaching 366,830 by the seventh day shown in figure 5.5.1c.

For the performance improvement, hyperparameter tuning was performed using RandomizedSearchCV with three-fold cross-validation. Search space included the

number of units in each BiLSTM layer (64, 128, 256 for the first and 32, 64, 128 for the second), dropout rates (0.2, 0.3, 0.5), learning rates (0.01, 0.001, 0.0005), batch sizes (16, 32), and epochs (20, 50). The randomized search attempted 10 parameter combinations simultaneously, and model evaluation was performed based on the $R^2$ scores.

Tuning results identified the best configuration as 128 units for the initial BiLSTM layer, 64 units for the second, a dropout rate of 0.5, learning rate of 0.01, batch size of 16, and 50 epochs. This configuration achieved a strong performance with an $R^2$ of 0.8679 on the training set and 0.9267 on the validation set. These results indicate that the tuned model significantly outperformed the baseline setup. As shown in Figure 5.5.1d, the tuning visualization confirmed that larger hidden units combined with a smaller batch size delivered the strongest validation performance.

In summary, the BiLSTM model was able to capture sequential dependencies in the sales data and benefited from hyperparameter fine-tuning. While the baseline achieved moderate performance, the optimized BiLSTM significantly improved validation accuracy and demonstrated stronger generalization capability for short-term daily sales forecasting.

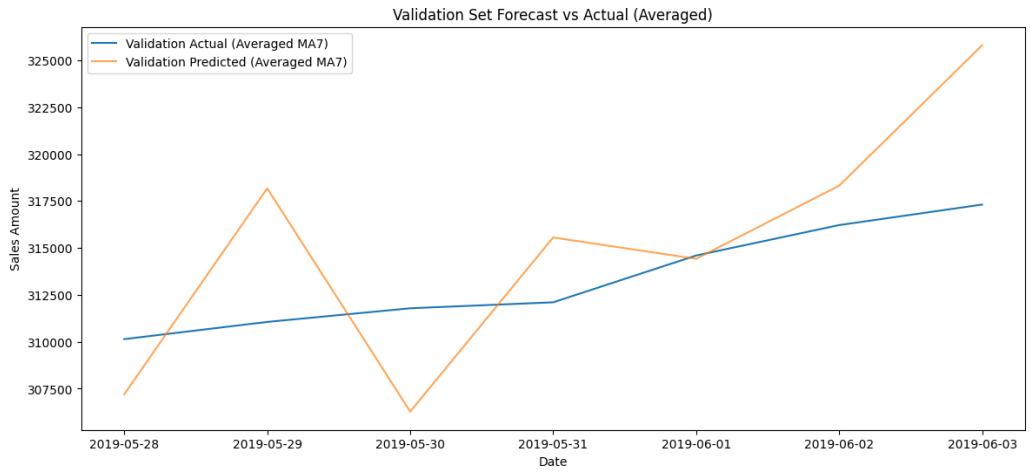



Figure 5.5.1a Validation Forecast vs Actual Sales (averaged MA7)

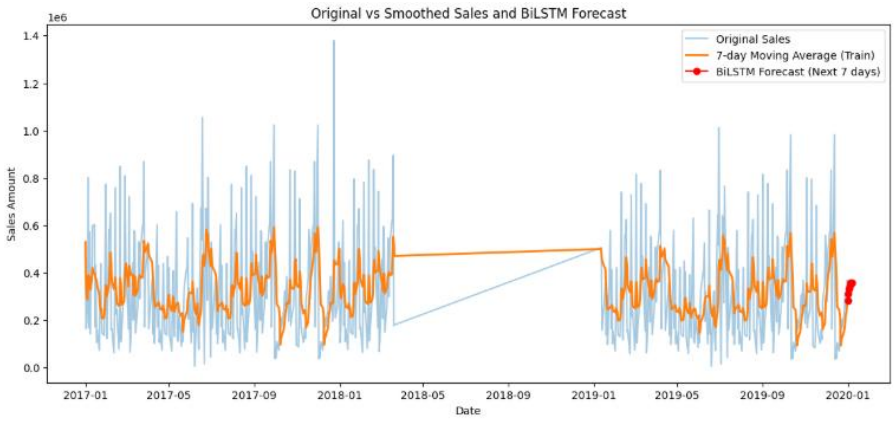



Figure 5.5.1b Original vs Smoothed Sales with BiLSTM next 7-day Forecast

```
Next 7 Days Forecast:
        Date  Predicted Sales Amount (MA7)
0 2020-01-01                290160.81250
1 2020-01-02                334980.21875
2 2020-01-03                339814.40625
3 2020-01-04                362742.90625
4 2020-01-05                363229.00000
5 2020-01-06                368983.75000
6 2020-01-07                366830.31250
```

Figure 5.5.1c Forecasted Results for Future 7 days

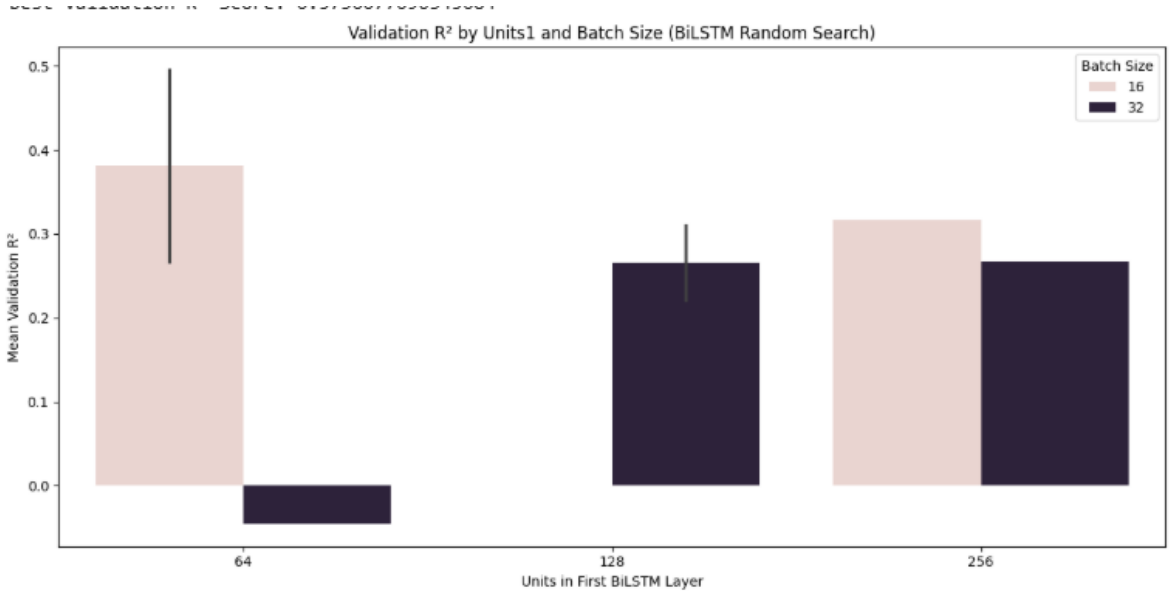



Figure 5.5.1d Hyperparameter Tuning Results for BiLSTM (Validation $R^2$ by hidden units and batch size)

### 5.5.2 TCN

Temporal Convolutional Network (TCN) was used to learn sequential relationships in the sequence of daily sales using causal convolutions. Two Conv1D layers were stacked with causal padding to preserve temporal structure, followed by a dropout layer to regularize against overfitting, and a dense output layer to produce the seven-day forecast period. The model was trained using the Adam optimizer and the mean squared error (MSE) loss, and early stopping and learning rate reduction callbacks for encouraging stability and preventing overfitting.

The baseline model was trained with a 30-day input horizon, 64 filters, the learning rate as 0.001, and batch size of 32. As seen from the evaluation results of the baseline model, the model's training $R^2$ was 0.844 and validation $R^2$ was 0.752.These results indicated that the model learned temporal dependencies effectively but could be further optimized, especially validation accuracy.

For further enhancing performance, grid search hyperparameter tuning was performed over multiple configurations, i.e., input window size (30 vs. 45 days), filters number (32 vs. 64), learning rate (0.001 vs. 0.0005), and batch size (16 vs. 32). Results, as described in the tuning visualization (Figure 5.5.2), showed that increasing the input window to 45 days from 30 days always improved validation performance, confirming that a larger past context allowed the model to learn temporal relations more effectively. Furthermore, the combination of 64 filters, 45-day window, 0.001 learning rate, and 16 batch size also recorded the highest validation performance with a validation $R^2$ of approximately 0.82. When 64 filters were used, increasing accuracy still further. These results indicate that both a more extensive receptive field (through longer sequences of input) and increased filter capability were beneficial to the TCN model, enabling it to pick up more complex patterns of demand volatility and promotional peaks in the sales data.

In summary, the TCN model worked well for short-term sales forecasting by leveraging causal convolutions to encode temporal relationships. While the baseline performed

well at initialization, the fine-tuned configuration significantly enhanced validation accuracy, demonstrating the importance of proper hyperparameter tuning in high-quality forecasting performance.

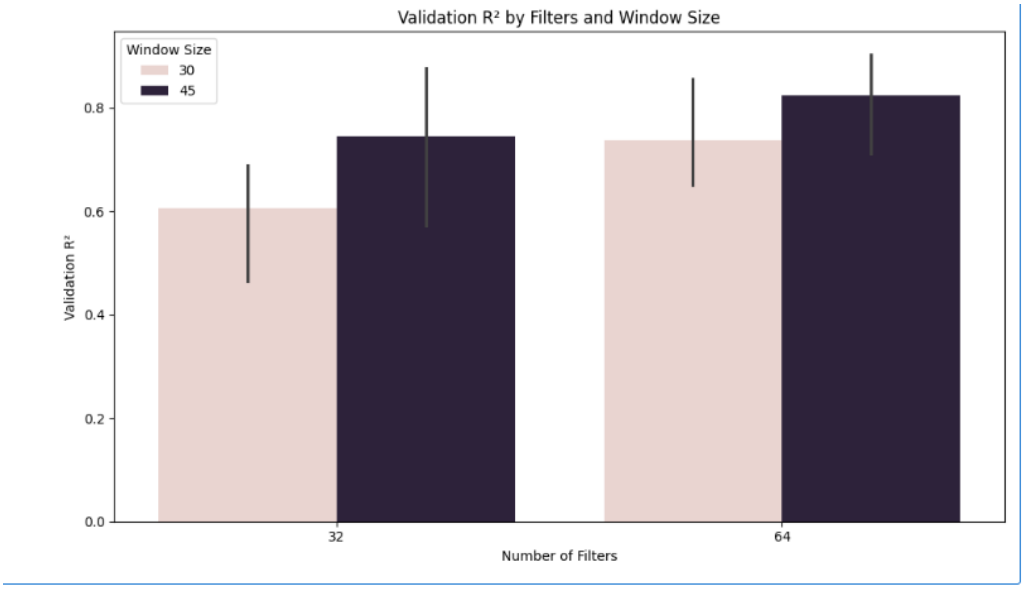



Figure 5.5.2 Hyperparameter Tuning Results (Validation $R^2$ by filters and window size)

### 5.5.3 XGBoost

The XGBoost model was employed as a baseline machine learning algorithm for daily sales forecasting. The baseline machine was configured with 100 estimators, max depth of 3, 0.1 learning rate, 0.8 subsample ratio, column sampling ratio of 0.8, and regularization parameters $\lambda = 5$ and $\alpha = 1$. The model was trained on lagged sales features, rolling statistics, and calendar features obtained during preprocessing. Baseline model performance gave a training $R^2$ of 0.850 and a validation $R^2$ of 0.701. These results showed that while the baseline XGBoost was good at capturing patterns, the validation accuracy drastically fell, which was a sign of overfitting.

In a bid to enhance model generalizability, grid search hyperparameter tuning was carried out through three-fold time-series cross-validation. The hyperparameter search space included the significant hyperparameters like the number of estimators (100, 200), maximum depth (2, 3, 4), learning rate (0.05, 0.1), subsample ratios (0.7, 0.8, 1.0), column sampling ratios (0.7, 0.8, 1.0), minimum child weight (3, 5, 7), and

regularization parameters λ (5, 10, 20) and α (0, 1, 5). The tuning process identified the optimal values as 200 estimators, max depth of 4, learning rate of 0.1, subsample ratio of 0.8, column sampling ratio of 0.7, min child weight of 3, λ = 20, and α = 0.

The hyperparameter-tuned model improved validation accuracy to $R^2$ = 0.796, from baseline validation $R^2$ of 0.701. The training performance was greatly improved with $R^2$ = 0.962. These results confirmed that hyperparameter tuning helped XGBoost to recognize sales dynamics more effectively while reducing overfitting, and hence be more accurate for forecasting in the short term.

## 5.6 Evaluation

This study employed four evaluation metrics, namely MAPE, $R^2$, MAE, and RMSE, to provide a balanced assessment of forecasting performance. MAPE was used because it is easily interpreted in business practice as it expresses average error as a percentage, with less than 20% being generally desirable [11].$R^2$ was used to measure the explanatory power of the models, with values above 0.7 indicating a good fit [26]. MAE was included as a straightforward measure of average deviation in sales units, while RMSE was applied to penalize larger errors more heavily, with a normalized RMSE below 20% of the mean considered acceptable in forecasting practice [27]. Together, these metrics ensured that model evaluation captured both statistical accuracy and business relevance.

### 5.6.1 BiLSTM

The optimized Bidirectional Long Short-Term Memory (BiLSTM) model was evaluated on the reserved test dataset for identifying its generalization ability. Test assessment provided an $R^2$ of 0.8481, RMSE of approximately 41,337, MAE of 33,150, and MAPE of 12.91%. These metrics are a significant improvement in predictive performance over baseline outcomes, with BiLSTM capturing sequential dependencies and reducing overall forecast error efficiently.

The plot comparison of the averaged predicted versus actual sales across the forecast horizon showed that the BiLSTM model tracked the overall level of sales but tended to overestimate sales when there were slightly increasing trends as shown in figure 5.6.1a.

However, the high value of $R^2$ and relatively low values of error suggest the model's capacity to capture nonlinear temporal dynamics in Amazon's daily sales data.

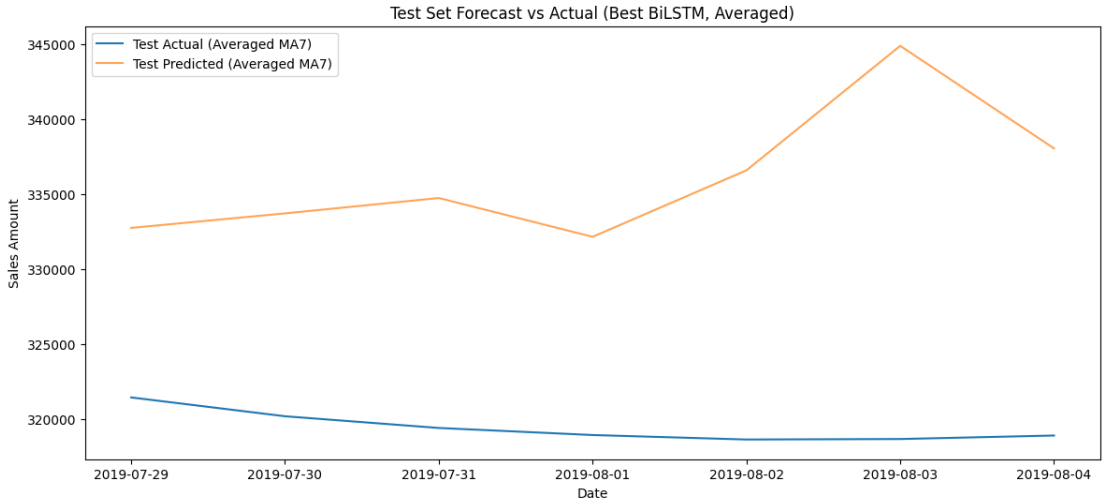



Figure 5.6.1a Test Set Forecast vs Actual (Best BiLSTM, Averaged)

In addition to retrospective evaluation, the best BiLSTM model was also utilized to conduct a forward 7-day forecast. The forecasted sales ranged from approximately 307,193 units on 1st January 2020 to 459,095 units on 7th January 2020 shown in figure 5.6.1b. The results exhibited a clear rising trend over the week, suggesting likely sales growth in this short-term interval. The overlay of the historical sales, smoothed moving averages, and 7-day BiLSTM prediction highlighted the model's ability to extrapolate trends from past behavior into actionable future predictions as shown in figure 5.6.1c.

```
Next 7 Days Forecast (Best BiLSTM):
        Date  Predicted Sales Amount (MA7)
0 2020-01-01                  307193.06250
1 2020-01-02                  349138.84375
2 2020-01-03                  407998.37500
3 2020-01-04                  451188.43750
4 2020-01-05                  463801.25000
5 2020-01-06                  468214.21875
6 2020-01-07                  459095.34375
```

Figure 5.6.1b Next 7-Day Forecasted Sales Amounts (Best BiLSTM)

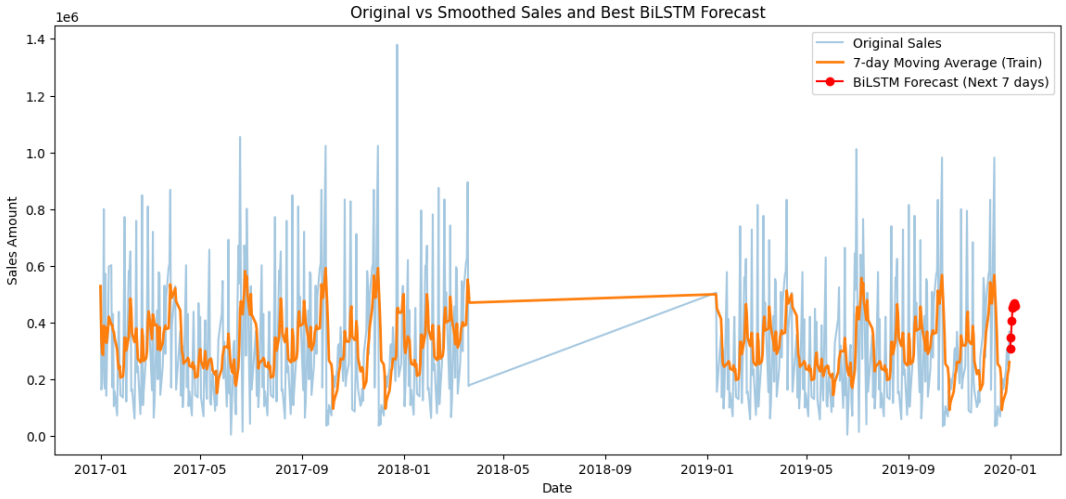



Figure 5.6.1c Original vs Smoothed Sales and 7-Day BiLSTM Forecast

Overall, BiLSTM testing indicated that the model handled unseen test data well alongside generating reasonable forward predictions. With its good balance between accuracy and time learning ability, BiLSTM qualifies as a satisfactory model for short-term demand forecasting in e-commerce applications.

### 5.6.2 TCN

After the Temporal Convolutional Network (TCN) model was optimized, the best configuration was selected and tested further on the unseen test set. In fairness, the dataset was rebuilt according to the best input window size determined by grid search, and the data were split chronologically into test, validation, and training sets. The final model was trained over the training and validation subsets, and the final model's generalization performance was estimated with the held-out test set.

For evaluation, the optimized TCN yielded an $R^2$ of 0.9336, RMSE of approximately 60,503, MAE of approximately 40,739, and MAPE of 24.35% on the test set. These results revealed profound predictive performance with reasonably low magnitudes of error relative to the baseline configuration.

To better interpret the model’s output, forecasts were visualized against the actual sales values. Since the model produced overlapping sequences for multi-step forecasting, predictions were averaged across test samples to generate a single consistent 7-day

forecast horizon. The comparison plot (Figure 5.6.2a) of predicted versus actual test values demonstrated that the TCN successfully captured the general upward and downward movements in sales, although it slightly overestimated peak fluctuations in certain days. This means that while the model captured temporal correlations and seasonality very well, it was still sensitive to exceptionally high spikes in sales.

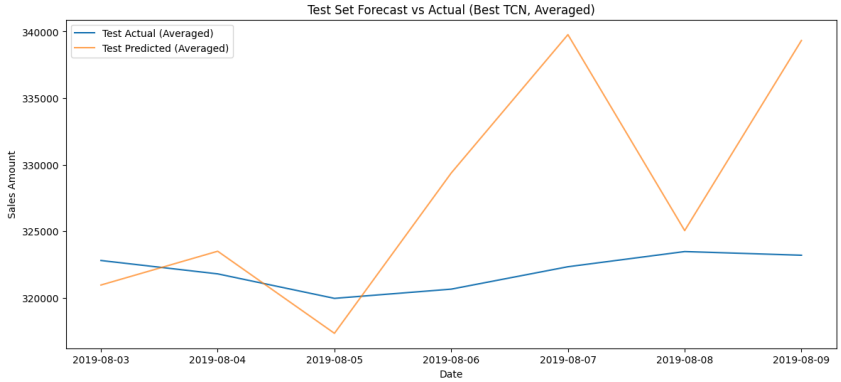



Figure 5.6.2a The Comparison Plot of Predicted Versus actual test values

In addition to evaluation on historical data, the best TCN model was also used to generate a forward-looking 7-day forecast beyond the available dataset. The predicted values ranged from approximately 320,188 to 765,726 units over the forecast horizon shown in Figure 5.6.2b. Visualization of the results showed that the model projected a rising trend during the forecast window, highlighting potential demand surges shown in Figure 5.6.2c. The forecast plot placed these future values alongside the historical sales series, clearly marking the transition from observed data to predicted outcomes.

```
Next 7 Days Forecast (Best TCN):
        Date  Predicted Sales Amount
0 2020-01-01           320188.12500
1 2020-01-02           510268.09375
2 2020-01-03           522899.34375
3 2020-01-04           471668.59375
4 2020-01-05           567791.18750
5 2020-01-06           705725.93750
6 2020-01-07           653650.62500
```

Figure 5.6.2b Predicted Sales Amount for Next 7 days Forecast (TCN)

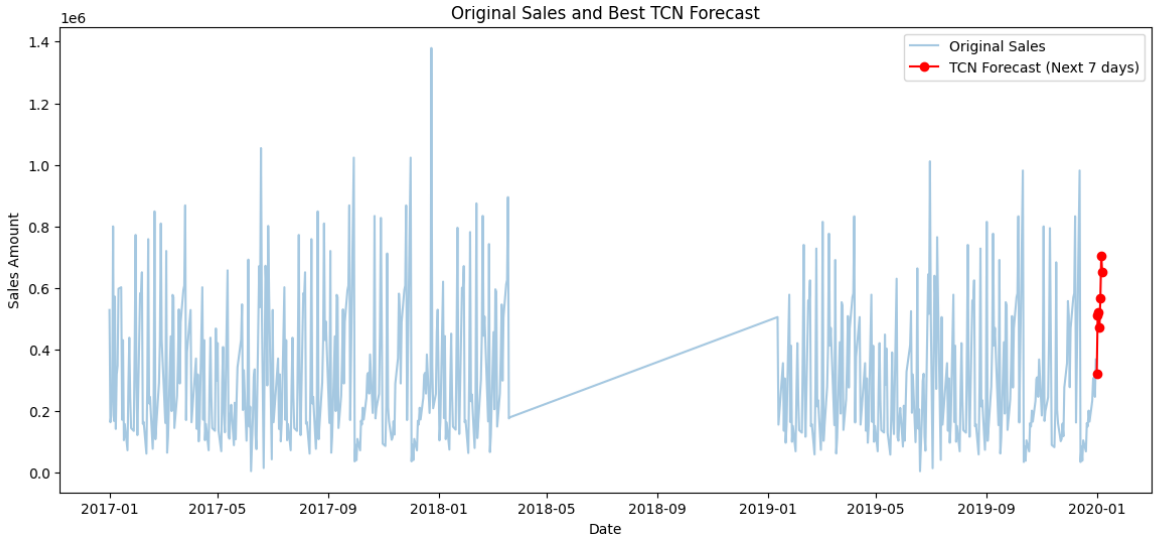



Figure 5.6.2c Original Sales with Best TCN 7-Day Forecast

Overall, the test confirmed that the TCN model fine-tuned generalized very accurately to unseen data and made reliable short-term forecasts that are suitable for operational planning in e-commerce environments.

### 5.6.3 XGBoost

The XGBoost model, which was optimized with GridSearchCV, was then run against the test set to find its predictive effectiveness. The final measure gave an $R^2$ of 0.6586, RMSE of approximately 134,140, MAE of 98,478, and MAPE of 70.62%. These results indicate that while XGBoost may be able to explain a moderate level of variance in the sales data, its error magnitudes were bigger than those of the deep learning models. The difference in performance arises from the fact that XGBoost relies on engineered features, which may not be able to capture all intricate nonlinear temporal dependencies that are represented in the sales data.

The test set forecast visualization compared actual sales with the model's predictions shown in figure 5.6.3a. The results showed that XGBoost managed to copy the overall sales patterns, identifying peaks and troughs in the test data. It had a higher variance and struggled with periods of extreme fluctuation, under/overestimating when demand was highest or at its lows. Despite this, the model provided reasonable interpretability through its feature importance values, which pointed toward the contribution of lagged sales values, rolling statistics, and calendar variables towards the prediction outcomes.

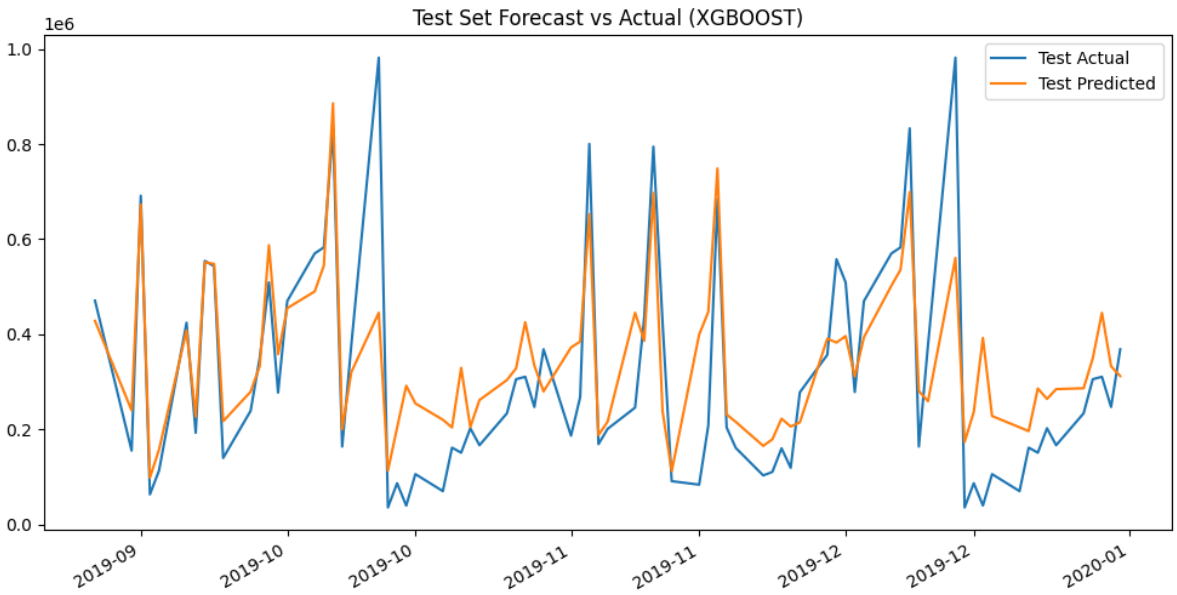



Figure 5.6.3a Test Set Forecast vs Actual (Best XGBoost)

Apart from test evaluation, the best XGBoost model was also employed to generate a forward-looking 7-day forecast. The predicted sales data ranged between roughly 268,784 and 685,164 units across the forecasting period (figure 5.6.3b). Visualization of the predictions with past sales indicated that the model had forecasted short-term spikes in demand, although with significant volatility across the 7 days (figure 5.6.3c). In contrast to BiLSTM and TCN, whose smoother trajectory predictions were observed, XGBoost's predictions more drastically fluctuated as a consequence of being highly sensitive towards engineered features and recent historical values.

```
Next 7 Days Forecast (Best XGBoost):
        Date  Predicted Sales Amount
0 2020-01-01             311899.1875
1 2020-01-02             268784.3125
2 2020-01-03             422165.3750
3 2020-01-04             677435.5625
4 2020-01-05             382252.6875
5 2020-01-06             685164.4375
6 2020-01-07             339517.5625
```

Figure 5.6.3b Next 7-Day Forecasted Sales Amounts (Best XGBoost)

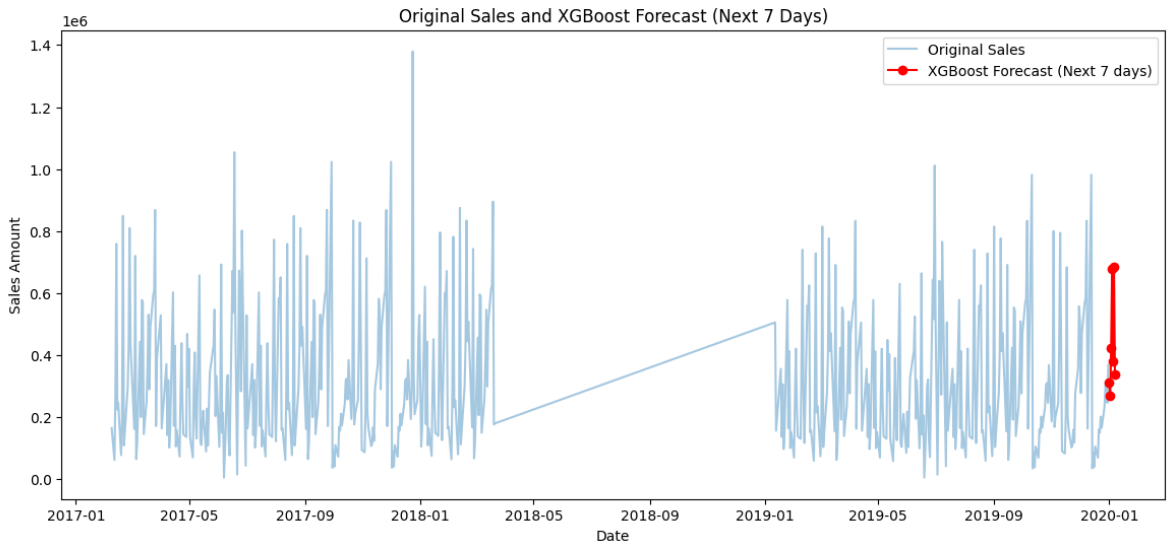



Figure 5.6.3c Historical Sales and 7-Day XGBoost Forecast

In summary, while XGBoost was inaccurate in comparison to BiLSTM and TCN, it was still valuable owing to its interpretability and low computational complexity. Its feature importance analysis provides decision-makers with tangible recommendations for making decisions, and its predictions, while being unstable, can still serve as an auxiliary perspective when combined with deep learning models for hybrid decision-support systems.

### 5.6.4 Comparative Analysis

From the evaluation results shown in table 5.6, it is evident that the deep learning models (BiLSTM and TCN) outperformed the traditional machine learning approach (XGBoost) in terms of accuracy.

| Model | Test $R^2$ | Test RMSE | Test MAE | Test MAPE |
|---|---|---|---|---|
| BiLSTM | 0.8481 | 41,337 | 33,150 | 12.91% |
| TCN | 0.9336 | 60,503 | 40,739 | 24.35% |
| XGBoost | 0.6586 | 132,140 | 98,478 | 70.62% |

Table 5.6 Performance Comparison of Forecasting Models on Test Set

The BiLSTM model achieved a good balance between accuracy and stability with comparatively small RMSE, MAE, and the lowest MAPE (12.91%), and hence the most

reliable model for short-term daily prediction. Its output was smooth and stable, perfect for the representation of nonlinear patterns in sales data.

TCN had the highest $R^2$ (0.9336), with excellent explanatory power and high correlation with sales dynamics. However, its measures of error (RMSE and MAPE) were higher than BiLSTM, indicating that while it closely followed trends, it overestimated peak amplitudes occasionally. The wider spread in its 7-day forecast also reflects this sensitivity.

XGBoost, though interpretable and fast, was not good enough with a poorer $R^2$ score (0.6586) and bigger error values. Its predictions were more error-prone and unstable, perhaps due to its dependence on human-designed features which did not do so well in picking up long-term temporal dependencies. Its feature importance plot was nevertheless useful in providing business insights into what generates sales, and thus was a good complementing tool.

Overall, BiLSTM was the most accurate of the three, TCN had the best trend capture performance, and XGBoost offered interpretability and optimality. A possible way forward may be marrying these models in a hybrid ensemble architecture to exploit their respective strengths: BiLSTM's low error rate, TCN's trend sensitivity, and XGBoost's interpretability.

### 5.7 Deployment

For deployment, an Amazon sales forecasting dashboard was developed using Streamlit for deploying the trained models in a production environment and providing business-meaningful insights for making decisions. The app allows users to upload their data in the CSV format, select a model to forecast (BiLSTM, TCN, or XGBoost), and get instant forecasts. The back-end loads pre-trained models (best_bilstm_model.keras, best_tcn_model.keras, and best_xgb_model.pkl), runs preprocessing pipelines for each individual model, and returns evaluation metrics as well as a 7-day forecast.The design emphasizes usability, enabling business managers without technical expertise to interact with the models and interpret results through clear metrics, visualizations, and recommendations.

The first dashboard screen (Figure 5.7.1) provides an upload interface and model selection panel, ensuring flexibility for different forecasting approaches. After loading the dataset (Figure 5.28), the dashboard displays the first five rows for verification. Once a model is selected, the system evaluates its performance on test data and shows key performance indicators—MAE, RMSE, and $R^2$—summarized at the top of the results page (Figure 5.7.2). These metrics provide immediate feedback on model accuracy and robustness.

The dashboard then generates a 7-day forecast table (Figure 5.7.3) with forecasted sales values and marks significant findings such as the latest forecast, peak sales day, and 7-day average. The forecasts are also illustrated in the form of a bar chart (Figure 5.7.4), facilitating users' easier interpretation of short-term demand fluctuations. To connect forecasting with business strategy, the system also provides business insights (Figure 5.7.5), where the increasing or decreasing sales trends and peak days are identified, and inventory planning and promotion recommendations are proposed. Additionally, an AI assistant module (Figure 5.7.6) introduces interpretability by providing personalized recommendations on inventory, promotions, and staffing. Users can even input their own business questions (e.g., "How to increase daily sales?"), and the assistant generates customized plans (Figure 5.7.7).

This deployment framework not only gives accurate forecasts but also bridges the gap between predictive modeling and real decision-making. By integrating forecasting results, visualization, and actionable insight, the dashboard creates a holistic decision-support tool for e-commerce activities.

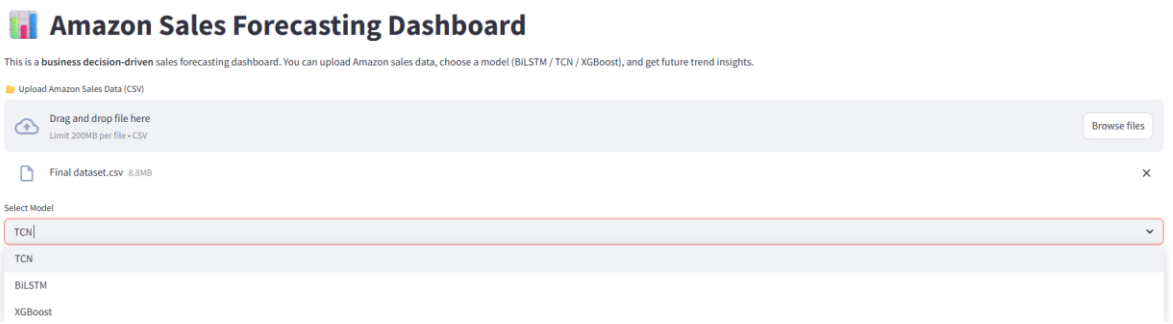
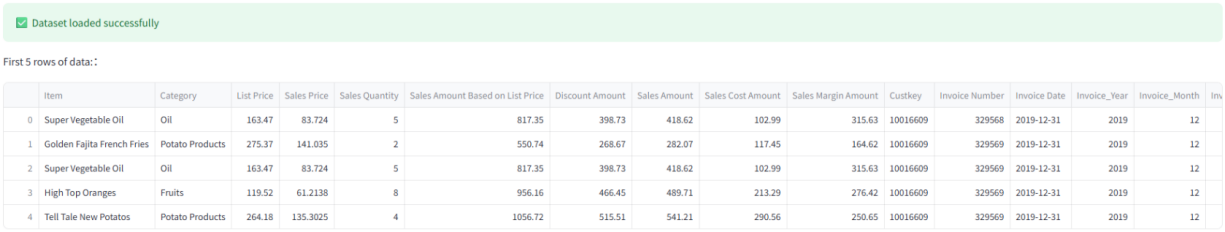


Figure 5.7.1 Streamlit dashboard interface for dataset upload and model selection

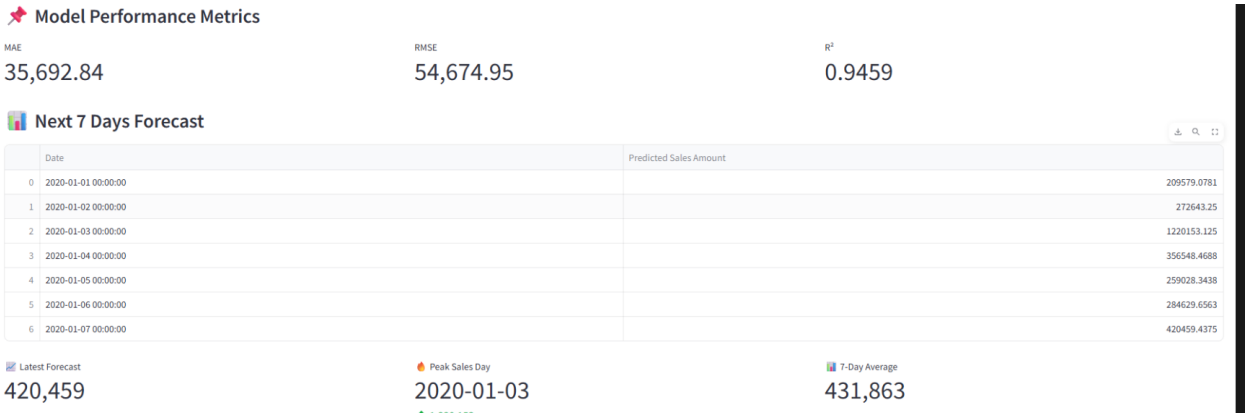


Dataset loaded successfully

First 5 rows of data::

| | Item | Category | List Price | Sales Price | Sales Quantity | Sales Amount Based on List Price | Discount Amount | Sales Amount | Sales Cost Amount | Sales Margin Amount | Custkey | Invoice Number | Invoice Date | Invoice_Year | Invoice_Month |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Super Vegetable Oil | Oil | 163.47 | 83.724 | 5 | 817.35 | 398.73 | 418.62 | 102.99 | 315.63 | 10016609 | 329568 | 2019-12-31 | 2019 | 12 |
| 1 | Golden Fajita French Fries | Potato Products | 275.37 | 141.035 | 2 | 550.74 | 268.67 | 282.07 | 117.45 | 164.62 | 10016609 | 329569 | 2019-12-31 | 2019 | 12 |
| 2 | Super Vegetable Oil | Oil | 163.47 | 83.724 | 5 | 817.35 | 398.73 | 418.62 | 102.99 | 315.63 | 10016609 | 329569 | 2019-12-31 | 2019 | 12 |
| 3 | High Top Oranges | Fruits | 119.52 | 61.2138 | 8 | 956.16 | 466.45 | 489.71 | 213.29 | 276.42 | 10016609 | 329569 | 2019-12-31 | 2019 | 12 |
| 4 | Tell Tale New Potatos | Potato Products | 264.18 | 135.3025 | 4 | 1056.72 | 515.51 | 541.21 | 290.56 | 250.65 | 10016609 | 329569 | 2019-12-31 | 2019 | 12 |

Figure 5.7.2 Dataset successfully loaded and previewed in the dashboard



Model Performance Metrics

MAE
35,692.84

RMSE
54,674.95

$R^2$
0.9459

Next 7 Days Forecast

| | Date | Predicted Sales Amount |
|---|---|---|
| 0 | 2020-01-01 00:00:00 | 209579.0781 |
| 1 | 2020-01-02 00:00:00 | 272643.25 |
| 2 | 2020-01-03 00:00:00 | 1220153.125 |
| 3 | 2020-01-04 00:00:00 | 356548.4688 |
| 4 | 2020-01-05 00:00:00 | 259028.3438 |
| 5 | 2020-01-06 00:00:00 | 284629.6563 |
| 6 | 2020-01-07 00:00:00 | 420459.4375 |

Latest Forecast
420,459

Peak Sales Day
2020-01-03

7-Day Average
431,863

Figure 5.7.3 Model performance metrics and 7-day forecast results

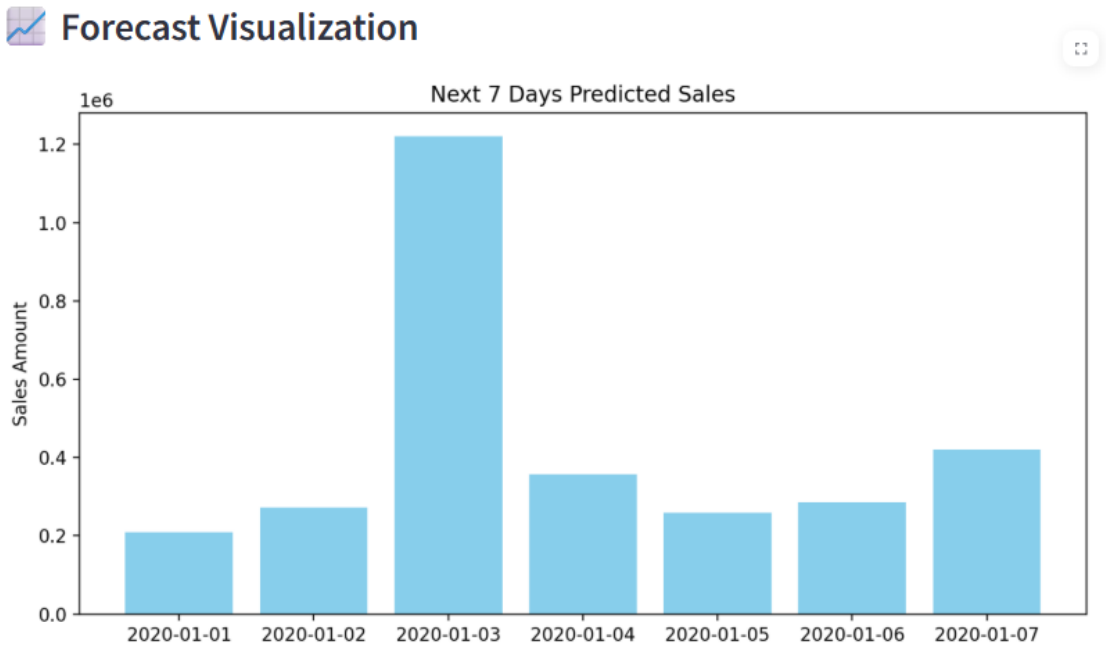



Figure 5.7.4 Visualization of next 7 days predicted sales


Figure 5.7.5 Business insights panel showing forecasted peak day and recommendations

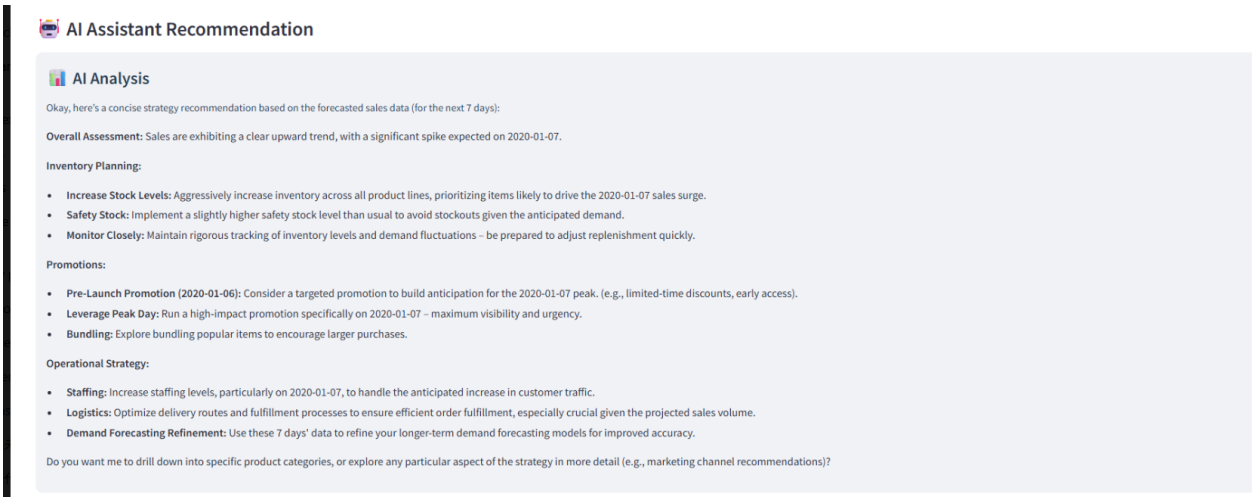



Figure 5.7.6 AI assistant recommendations for inventory, promotions, and staffing

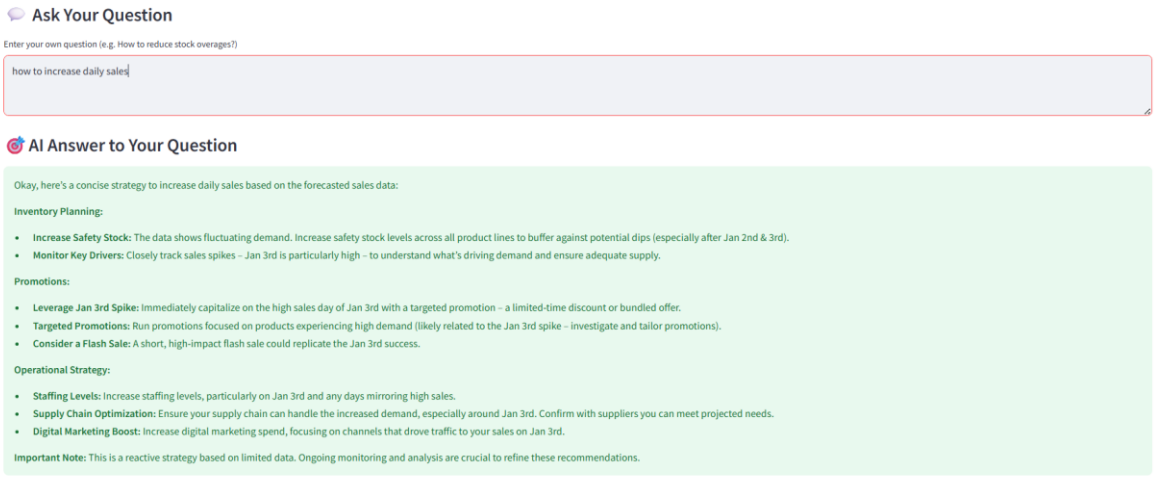



Figure 5.7.7 Interactive Q&A with AI assistant for tailored sales strategies

# CHAPTER 6
# System Evaluation And Discussion

This chapter evaluates the developed sales forecasting system in terms of accuracy, reliability, and business applicability. It begins with system testing and performance metrics to assess the predictive accuracy of BiLSTM, TCN, and XGBoost using $R^2$, RMSE, MAE, and MAPE. Testing setups and results are then treated, highlighting each model's weaknesses and strengths, along with implications for business decision-making. The dashboard is also tested functionally to validate dataset management, model execution, forecast plotting, and AI assistant recommendations. Project challenges encountered during data preparation, model optimization, deployment, and AI integration are discussed, followed by an evaluation of how well the project objectives were achieved. Finally, concluding remarks summarize major findings and suggest improvements for the future.

## 6.1 System Testing and Performance Metrics

The system was tested to ensure both prediction accuracy and operational reliability of the deployed forecasting dashboard. Testing was conducted in two main phases: (i) model evaluation using performance metrics on the test set, and (ii) system-level validation of the Streamlit dashboard functionalities, including dataset upload, preprocessing, model inference, forecast visualization, and AI assistant recommendations.

For performance of the model, four standard evaluation metrics were used: $R^2$, RMSE, MAE, and MAPE. $R^2$ indicates the variance explained by the model and indicates overall fit, with higher values indicative of greater predictive power [26]. RMSE calculates the square-root of the average squared errors and penalizes extreme deviations, while MAE provides a direct average of absolute errors and is easier to interpret in terms of actual sales units. MAPE improves on these measures by providing prediction error in percentage form, which is helpful for comparing inter-sales magnitudes. As with forecasting practice, an $R^2$ value above 0.7, MAPE below 20%, and comparatively low RMSE/MAE values are good indications of the quality of the model [11][26][27].

System testing across the three models—BiLSTM, TCN, and XGBoost—confirmed that they had been integrated successfully and produced consistent predictions when deployed. The dashboard presented a real picture with correct evaluation measurements, provided 7 days' worth of predictions ahead, and represented them via simplistic graphs and tables. The AI assistant module also provided actionable recommendations on inventory planning, promotions, and staffing, effectively demonstrating the system's ability to interpret technical forecasting outputs into support for business decisions.

In summary, the system testing validated that the proposed framework is technically sound as well as applicable in reality, meeting the twofold demands of prediction precision and business usefulness.

## 6.2 Testing Setup and Result

Table 5.6 (in Chapter 5) summarizes the test set performance of the three prediction models — BiLSTM, TCN, and XGBoost — using four metrics: $R^2$, RMSE, MAE, and MAPE. The outcomes provide a basis for a comparison of the predictive accuracy and generalization ability of the models.

The BiLSTM model showed stable and balanced performance on all metrics. With an $R^2$ of 0.8481, BiLSTM explained a lot of the variance in the sales data. More importantly, it had the least RMSE (41,337) and MAE (33,150), and the least MAPE (12.91%), indicating that its predictions were quite near the actual-world sales data. This is proof of BiLSTM's ability to model temporal dependencies extremely well while producing stable predictions, and thus it is highly reliable for real-world usage.

TCN outperformed the others in terms of $R^2$ (0.9336), reflecting its excellent ability to capture variance and follow the overall sales trends. However, its higher RMSE (60,503), MAE (40,739), and MAPE (24.35%) compared to BiLSTM indicate poorer accuracy of point forecasting. While TCN did an excellent job capturing the dynamics of the time series, it tended to overestimate fluctuations and thus incurred greater relative errors.

Conversely, the worst-performing model was XGBoost, with the lowest $R^2$ (0.6586) and highest error metrics (RMSE = 132,140, MAE = 98,478, MAPE = 70.62%). Although XGBoost was able to pick up on some of the seasonal peaks and troughs, it underperformed the deep learning models consistently with regards to handling nonlinear temporal dependencies. This highlights its limitations when applied to complex, high-variance sales data without extensive feature engineering.

Overall, the evaluation suggests that BiLSTM is the most effective model for accurate sales forecasting, given its strong balance between trend alignment and prediction accuracy. TCN is valuable when capturing broader variance and trend-following is prioritized, but its tendency to overestimate requires caution. XGBoost is not recommended for this particular forecasting problem, as its performance falls significantly behind the deep learning approaches.

### 6.2.1 Strengths and Weaknesses of Each Model

The performance analysis revealed individual strengths and weaknesses of each forecasting model for each error metric. BiLSTM had the highest accuracy across all error metrics, with minimum RMSE, MAE, and MAPE. This reflects its high capability in producing stable and robust forecasts that are strongly correlated with actual sales. Its bidirectional nature allowed the model to extract temporal relations successfully, which suits it to be well-suited for short-term demand forecasting. However, BiLSTM had a slightly lower $R^2$ than TCN, showing that it explained less variance in the dataset as a whole. Also, its training was computationally more intensive and took more run times compared to XGBoost.

The $R^2$ for the TCN model was the highest, which meant it performed the best in capturing sales variance and overall trends. Its dilated convolutional-based convolutional architecture allowed efficient handling of long-range dependencies in the sequences of sales. TCN attained higher RMSE and MAPE but lower precision values compared to BiLSTM, which showed its predictions were less precise at the level of individual data points. It also exhibited bias in overestimation of sales towards times of variability, which can limit its reliability during times of volatility.

On the other hand, XGBoost was the weakest model predictively, with the lowest $R^2$ and highest errors. Despite that, it had some advantages: it was computationally efficient, learned faster than the deep learning models, and showed interpretability through its tree format. XGBoost was also able to capture some seasonal peaks and troughs. But it struggled with the nonlinear time dependencies of the data and required extensive feature engineering to match the performance of deep learning approaches.
Table 6.1 summarizes the strengths and weaknesses of the three forecasting models evaluated in this study.

| **Model** | **Strengths** | **Weaknesses** |
|---|---|---|
| BiLSTM | - Achieved the lowest RMSE, MAE, and MAPE, indicating the highest forecast accuracy.<br>- Effectively captured temporal dependencies with stable predictions.<br>- Balanced performance across all evaluation metrics. | - Slightly lower $R^2$ compared to TCN, meaning it explains less overall variance.<br>- Requires more computational resources and longer training time compared to XGBoost. |
| TCN | - Achieved the highest $R^2$ (0.9336), effectively capturing sales variance and overall trends.<br>- Handles long-term dependencies efficiently through dilated convolutions.<br>- Robust sequence length variations. | - Higher RMSE and MAPE than BiLSTM, reflecting less precise point forecasts.<br>- Tendency to overestimate sales during fluctuations, leading to forecast instability. |
| XGBoost | - Faster training and lower computational cost compared to deep learning models.<br>- Performed relatively well in capturing some seasonal peaks and troughs.<br>- Highly interpretable due to tree-based structure. | - Lowest $R^2$ (0.6586) and highest errors (RMSE, MAE, MAPE).<br>- Struggled with nonlinear temporal dependencies in sales data.<br>- Requires extensive feature engineering to improve performance. |

Table 6.1 Strengths and Weaknesses of BiLSTM, TCN, and XGBoost Models

### 6.2.2 Implications for Business Decision-Making

The findings of the comparison among BiLSTM, TCN, and XGBoost models have a direct influence on business decision-making in the context of e-commerce companies such as Amazon. BiLSTM, with the most accurate forecasts and lowest error values, is particularly helpful in short-term demand prediction. Accurate short-term forecasting allows managers to optimize inventories, having high-demand items always in stock and reducing risks of overstocking low-demand products. This directly results in improved warehouse efficiency and reduced holding costs.

TCN, while it has the tendency to overestimate sales, had the highest $R^2$ and can therefore be utilized for long-term trend analysis. As it is able to capture broader variance, it is suitable for strategic planning-level activities such as evaluating seasonal campaigns, promotional events, and year-over-year sales growth. TCN outputs can be used by business managers to identify probable demand surges, which can cross-validate with BiLSTM forecasts for tactical execution.

Conversely, the worse performance of XGBoost suggests its less generalizability to high-variance sales prediction. While being computationally quicker, its lower accuracy makes it not ideal as the primary model for making decisions. However, its interpretability and faster training time might still provide value in the course of exploratory analysis, quick prototyping, or in an ensemble approach where tree-based intuition is combined with deep learning predictions.

From an operational strategy perspective, the results suggest that a hybrid forecast model would be ideal: BiLSTM forecasts can be relied upon for day-to-day stock control and tactical promotions, with TCN informing broader strategic planning and trend monitoring. The two-model solution ensures that decision-makers balance accuracy with longer-term visibility, reaffirming Amazon's ability to respond to both near-term market changes and evolving patterns of consumer demand.

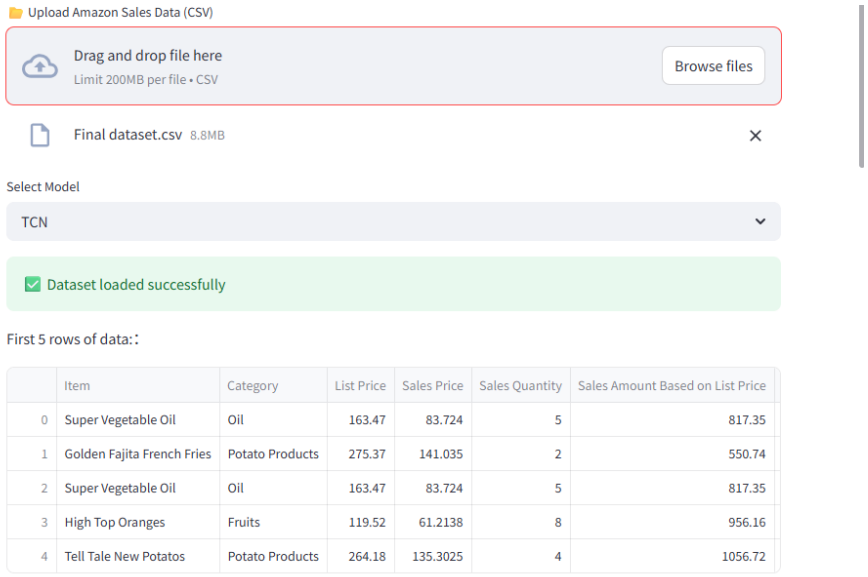
### 6.2.3 Dashboard Results and Analysis

System testing confirmed that the dashboard handled valid and invalid input appropriately. When a correctly formatted CSV was imported, the system could import the dataset successfully and show the first five rows for verification (figure 6.2.3a). When the dataset did not have the Invoice Date and Sales Amount columns required, the system generated an error message to inform the user but also displayed the first five rows for clarity (figure 6.2.3b). Model selection tests revealed that all three forecasting models—BiLSTM, TCN, and XGBoost—were executed without issue, and the dashboard correctly reported MAE, RMSE, and $R^2$ measures (figure 6.2.3c-e). The 7-day forecast outputs were also always displayed in tabular as well as graphical format, and the output could be downloaded as CSV(figure 6.2.3f-h). Additionally, the business insight module also offered important trend analysis as well as peak day identification, and the AI assistant provided default suggestions as well as user-specific answers to questions (figure 6.2.3i-k). These outcomes collectively verified that the dashboard integrates data validation, forecasting, visualization, and decision support into a reliable end-to-end system. Table 6.2 shows the system test cases and outcomes.

| **Test Scenario** | **Expected Output** | **Actual Output** |
|---|---|---|
| Upload sales dataset (CSV) | Dataset loads, first 5 rows preview displayed | CSV successfully uploaded, first 5 rows displayed in table format |
| Dataset missing required columns (Invoice Date, Sales Amount) | System shows error message indicating missing columns, while still displaying first 5 rows | Error message displayed and preview still shown |
| Model selection (BiLSTM / TCN / XGBoost) | Selected model loads without error | All three models loaded correctly from pre-trained files |

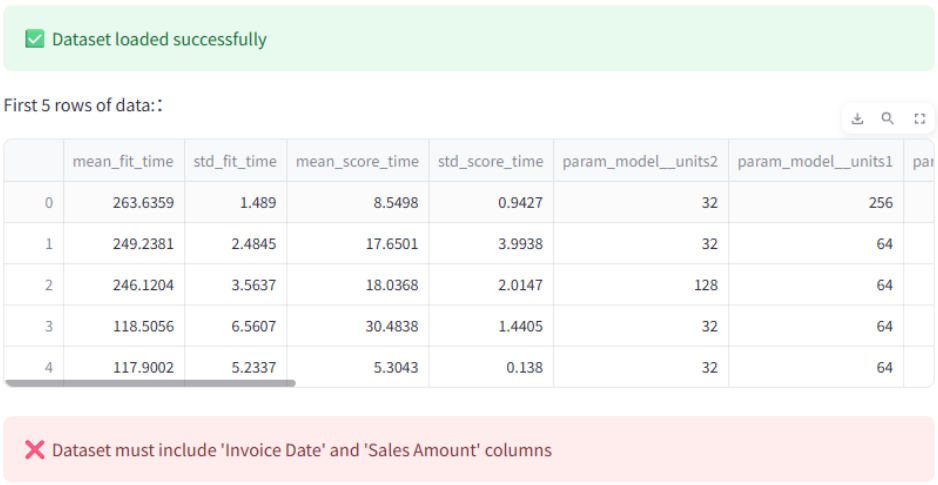
| | | |
|---|---|---|
| Forecast generation | Dashboard produces next 7-day sales forecast | Forecast table with correct dates and sales predictions displayed |
| Performance metrics calculation | Display $R^2$, RMSE, MAE | Metrics calculated and displayed accurately for each model |
| Forecast visualization | Bar charts show predicted sales for future 7 days | Graphs generated correctly |
| Download forecast results | User can export 7-day forecast in CSV format | CSV downloaded and opened in Excel successfully |
| AI assistant recommendation | Generate actionable insights based on forecasted sales | Recommendations on inventory, promotions, and staffing provided |
| Business insight summary | Highlight trend (upward/downward), peak sales day, and average forecast | Insights displayed correctly with date and peak sales value |
| AI assistant custom query | Returns tailored answer based on user question | Relevant recommendations provided |

Table 6.2 System Test Cases and Outcomes




| | Item | Category | List Price | Sales Price | Sales Quantity | Sales Amount Based on List Price |
|---|---|---|---|---|---|---|
| 0 | Super Vegetable Oil | Oil | 163.47 | 83.724 | 5 | 817.35 |
| 1 | Golden Fajita French Fries | Potato Products | 275.37 | 141.035 | 2 | 550.74 |
| 2 | Super Vegetable Oil | Oil | 163.47 | 83.724 | 5 | 817.35 |
| 3 | High Top Oranges | Fruits | 119.52 | 61.2138 | 8 | 956.16 |
| 4 | Tell Tale New Potatos | Potato Products | 264.18 | 135.3025 | 4 | 1056.72 |

Figure 6.2.3a – Dataset Upload and Successful Loading (showing first 5 rows of sales data)




| | mean_fit_time | std_fit_time | mean_score_time | std_score_time | param_model__units2 | param_model__units1 | par |
|---|---|---|---|---|---|---|---|
| 0 | 263.6359 | 1.489 | 8.5498 | 0.9427 | 32 | 256 | |
| 1 | 249.2381 | 2.4845 | 17.6501 | 3.9938 | 32 | 64 | |
| 2 | 246.1204 | 3.5637 | 18.0368 | 2.0147 | 128 | 64 | |
| 3 | 118.5056 | 6.5607 | 30.4838 | 1.4405 | 32 | 64 | |
| 4 | 117.9002 | 5.2337 | 5.3043 | 0.138 | 32 | 64 | |



Figure 6.2.3b – Error Handling: Invalid Dataset Format

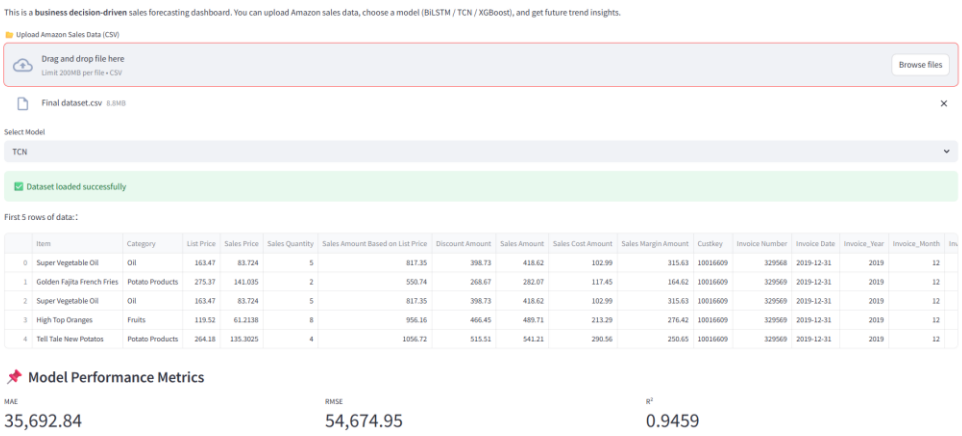



Figure 6.2.3c – Dashboard Output for TCN Model

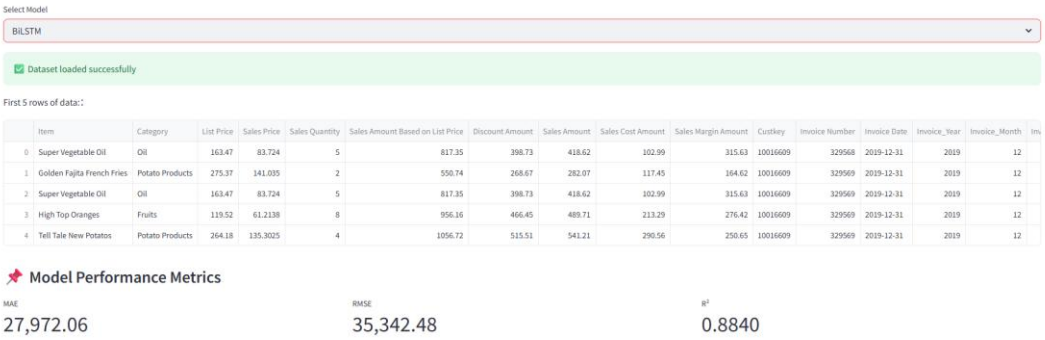



Figure 6.2.3d – Dashboard Output for BiLSTM Model

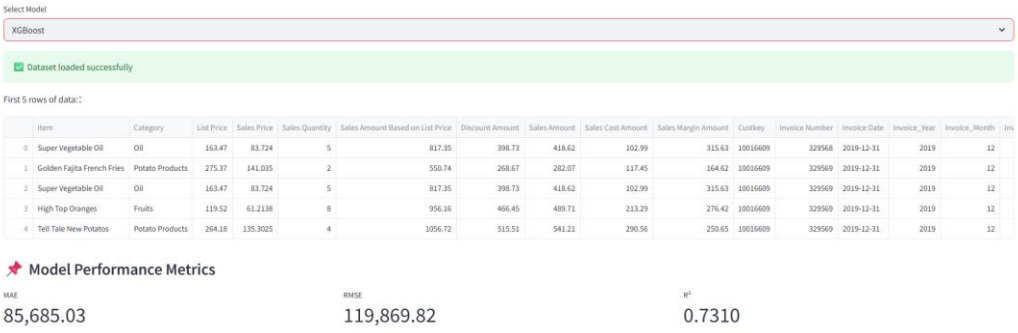



Figure 6.2.3e – Dashboard Output for XGBoost Model

Next 7 Days Forecast

| | Date | Predicted Sales Amount |
|---|---|---|
| 0 | 2020-01-01 00:00:00 | 209579.0781 |
| 1 | 2020-01-02 00:00:00 | 272643.25 |
| 2 | 2020-01-03 00:00:00 | 1220153.125 |
| 3 | 2020-01-04 00:00:00 | 356548.4688 |
| 4 | 2020-01-05 00:00:00 | 259028.3438 |
| 5 | 2020-01-06 00:00:00 | 284629.6563 |
| 6 | 2020-01-07 00:00:00 | 420459.4375 |

Figure 6.2.3f – Next 7 Days Forecast (tabular display of daily predicted sales)

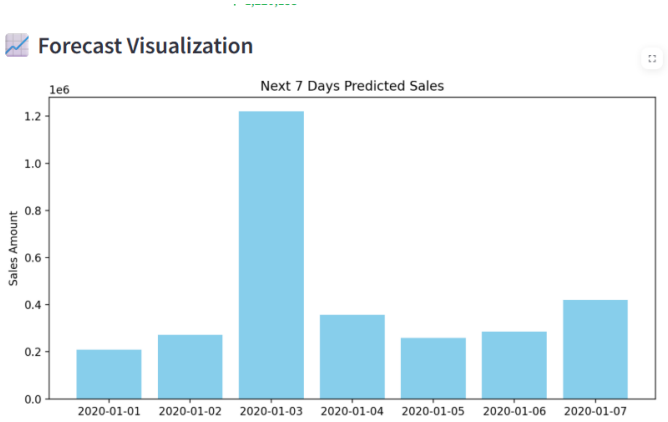



Figure 6.2.3g – Forecast Visualization (bar chart of next 7 days predicted sales)

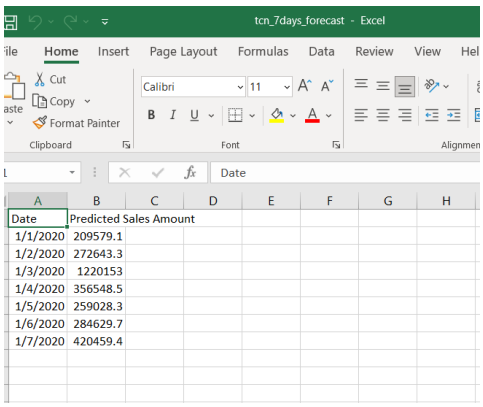



Figure 6.2.3h – Forecast Export as CSV (downloaded Excel file with 7-day predictions)

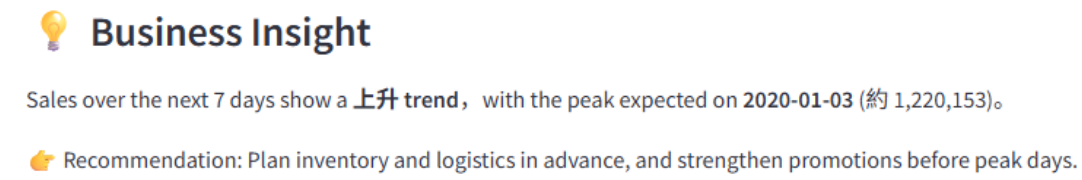



Figure 6.2.3i – Business Insight Module (trend analysis and recommendation summary)

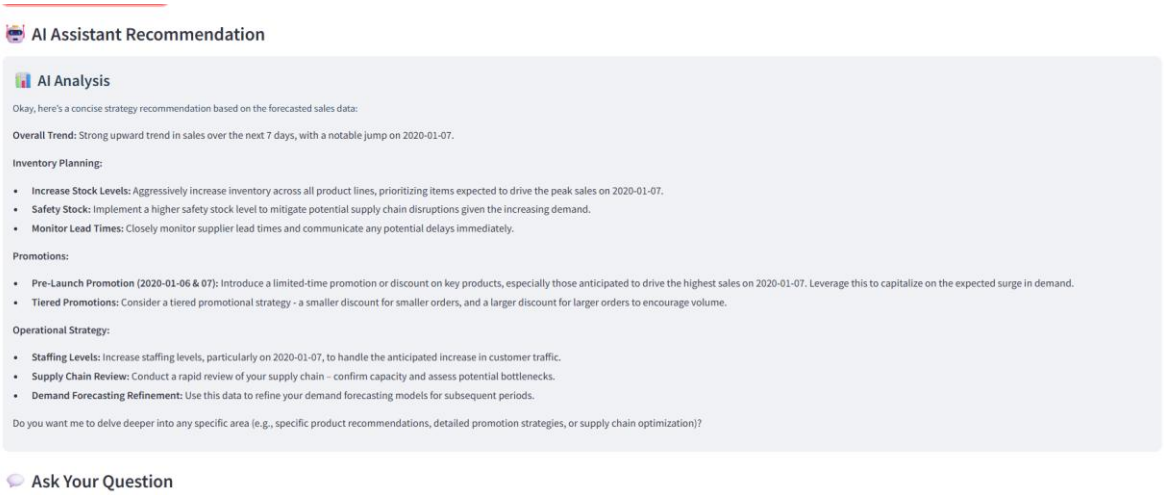



Figure 6.2.3j – AI Assistant Recommendation (default recommendations for inventory, promotions, and operations)

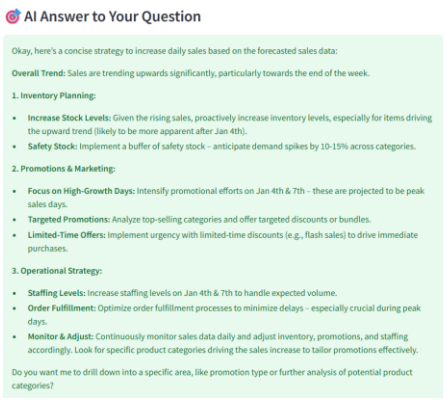



Figure 6.2.3k – AI Assistant Custom Answer (user query: "how to increase daily sales" with tailored strategy output)

### 6.3 Project Challenges

During the development of this project, there were a few challenges encountered. The initial challenge was data preprocessing and quality. The raw Amazon sales dataset contained missing values, outliers, and inconsistent formats, which required extensive cleaning, transformation, and feature engineering before being suitable for forecasting models. A second challenge was optimizing the model since TCN and BiLSTM required hyperparameter tuning with RandomizedSearchCV, which was computationally intensive and time-consuming. In addition, model generalization without overfitting was challenging, particularly for deep learning models when there are small training samples. There was another challenge at deployment, where integrating three heterogeneous models (BiLSTM, TCN, and XGBoost) under a unified Streamlit dashboard needed careful handling of preprocessing pipelines, input-output formats, and evaluation metrics. Finally, the implementation of the AI Assistant module introduced challenges in prompt engineering and ensuring that recommendations were both contextually relevant and business-oriented, rather than generic.

### 6.4 Objectives Evaluation

Project objectives were accomplished. Three forecasting models, namely BiLSTM, TCN, and XGBoost, were built and implemented for short-term daily sales prediction of seven days ahead. All the models were evaluated with MAE, RMSE, MAPE, and $R^2$, which confirmed that BiLSTM offered the optimal performance, then TCN, and XGBoost had moderate performances. A Streamlit interactive dashboard was also made to enable users to upload data, select forecasting models, view the performance measures, display forecasts as charts and tables, and download results in CSV. Furthermore, the AI Assistant module was successfully integrated, automatically providing actionable business insights and allowing interactive queries for decision support in areas such as inventory planning, promotions, and staffing. Thus, all objectives stated at the beginning of the project were fulfilled.

## 6.5 Conclusion Remark

By this project, it was demonstrated that machine learning and deep learning models could be used efficiently to e-commerce sales forecasting. The results indicated that BiLSTM is the most suitable model for short-term daily forecasting, achieving higher accuracy compared to TCN and XGBoost. Beyond forecasting, the integration of an interactive Streamlit dashboard and AI Assistant converted the system into a decision-support system by bridging the gap between technical forecast outcomes and actual business plans. By enabling managers to view forecasts, measure model performance, and receive AI-driven recommendations, the system not only improves the reliability of forecasts but also improves operational planning and decision-making. Future work could focus on extending the forecasting horizon, incorporating additional external variables such as seasonality or promotional campaigns, and enhancing the AI Assistant with more domain-specific knowledge for advanced business strategy support.

# CHAPTER 7
# Conclusion and Recommendation

Chapter 7 concludes the project by summarizing the key findings and reflecting on how the objectives were achieved. It highlights that BiLSTM delivered the most accurate short-term forecasts, TCN effectively captured broader trends, and XGBoost provided interpretability but with lower accuracy. The chapter also emphasizes the added value of the Streamlit dashboard and AI Assistant in translating forecasts into actionable business insights. Finally, it outlines recommendations for future work, including extending the forecasting horizon, incorporating external variables, improving model efficiency, enhancing the AI Assistant, and scaling deployment for enterprise adoption.

## 7.1 Conclusion

The objective of this project was to address the issue of sales forecasting in the context of Amazon e-commerce data with advanced machine learning and deep learning techniques. It was intended to develop, validate, and evaluate three forecasting models—BiLSTM, TCN, and XGBoost—under one Streamlit-based dashboard integrating predictive analytics and business decision support.

The findings confirmed that deep learning models produced greater predictive accuracy compared to the machine learning baseline. Specifically, the BiLSTM model achieved the best balanced performance, which featured the lowest error values (MAE, RMSE, MAPE) with stable and reliable predictions. The TCN model achieved the highest explanatory power ($R^2$) and delivered good performance in the capture of temporal variation and trend movements but with overestimation in periods of volatility. Conversely, the computationally efficient and interpretable XGBoost lagged the deep learning models in terms of accuracy, presenting the limitation of feature-engineering-based approaches to handling intricate time series forecasting.

Beyond model performance, the successful development of the interactive forecasting dashboard demonstrated the project's practical contributions. The dashboard accommodated dataset upload, auto-preprocessing, model selection, performance evaluation, visualization of the forecasted data, and CSV export. In addition, integration

of the AI Assistant module elevated the system from a technical forecasting tool to a business decision platform. With its capacity to transform forecasts into business-meaningful suggestions for inventory planning, promotions, and staffing, the system bridged the data science output-practical business strategy gap.

In conclusion, the project validated the efficiency of employing BiLSTM and TCN for short-term e-commerce sales forecasting, with XGBoost acting as a reliable interpretability benchmark. It further demonstrated how predictive analytics can be used within an accessible system, allowing managers to make data-informed decisions based on not only quantitative performance metrics but also qualitative business insights.

## 7.2 Recommendations

Despite the fact that the project was able to achieve its objectives, several recommendations are provided to further enhance its contribution and address existing limitations. First, the forecasting horizon can be expanded beyond the current 7-day short-term focus to monthly or quarterly predictions, thereby supporting both operational and strategic decision-making. The inclusion of such external variables as holiday calendars, promotion dates, competitor activity, and macroeconomic information would further increase accuracy by capturing external influences on sales trends.

At the model level, the exploration of more complex architectures such as Transformers or hybrid ensembles involving deep learning and tree-based methods could assist in both accuracy and stability. Since BiLSTM and TCN required extensive computational resources during hyperparameter fine-tuning, future research could use efficiency-gaining optimisation techniques such as transfer learning, Bayesian optimisation, or GPU training in the cloud to make it quicker.

The AI Assistant module, while efficient, can be enhanced further with domain-specific rules, real-time inventory and supply chain data, and reinforcement learning to generate more context-based recommendations.

From a deployment perspective, scaling the system from Streamlit to cloud-native platforms like AWS SageMaker or Google Vertex AI would improve enterprise-level integration with ERP and CRM systems. Finally, conducting real business users' testing of the dashboard would provide an effective feedback loop to improve usability, validate forecast accuracy, and generate recommendations applicable to true operational requirements.

## REFERENCES

[1] A. Esposito and A. Esposito, “Harnessing AI: Transforming sales forecasting for greater accuracy and strategic action,” *Demand Gen Report*, Mar. 14, 2025. [Online]. Available: https://www.demandgenreport.com/demanding-views/harnessing-ai-transforming-sales-forecasting-for-greater-accuracy-and-strategic-action/48964/

[2] V. I. Kontopoulou, A. D. Panagopoulos, I. Kakkos, and G. K. Matsopoulos, “A Review of ARIMA vs. Machine Learning Approaches for Time Series Forecasting in Data Driven Networks,” *Future Internet*, vol. 15, no. 8, p. 255, Jul. 2023, doi: 10.3390/fi15080255.

[3] S. Writer, “The history of Amazon’s forecasting algorithm - Amazon Science,” *Amazon Science*, Nov. 14, 2024. https://www.amazon.science/latest-news/the-history-of-amazons-forecasting-algorithm

[4] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, “Financial time series forecasting with deep learning : A systematic literature review: 2005–2019,” *Applied Soft Computing*, vol. 90, p. 106181, Feb. 2020, doi: 10.1016/j.asoc.2020.106181.

[5] S. Hochreiter and J. Schmidhuber, “Long Short-Term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[6] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681, 1997.

[7] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” *arXiv (Cornell University)*, Jan. 2018, doi: 10.48550/arxiv.1803.01271.

[8] “About XGBoost,” xgboost.ai. [Online]. Available: https://xgboost.ai/about

[9] L. Xie, J. Liu, and W. Wang, “Predicting sales and cross-border e-commerce supply chain management using artificial neural networks and the Capuchin search algorithm,” *Scientific Reports*, vol. 14, no. 1, Jun. 2024, doi: 10.1038/s41598-024-62368-6.

[10] “SALES FORECASTING USING MACHINE LEARNING TECHNIQUES,” *International Research Journal of Modernization in Engineering Technology and Science*, Mar. 2023, doi: 10.56726/irjmets34569.

[11] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, “Statistical and Machine Learning forecasting methods: Concerns and ways forward,” *PLoS ONE*, vol. 13, no. 3, p. e0194889, Mar. 2018, doi: 10.1371/journal.pone.0194889.

[12] M. Perez, “Artificial neural networks and bankruptcy forecasting: a state of the art,” *Neural Computing and Applications*, vol. 15, no. 2, pp. 154–163, Jan. 2006, doi: 10.1007/s00521-005-0022-x.

[13] T. Chen and C. Guestrin, “XGBoost,” *Computer Science*, pp. 785–794, Aug. 2016, doi: 10.1145/2939672.2939785.

[14] “Article: Fitting data with XGBoost | Institute and Faculty of Actuaries.” [Online]. Available: https://www.actuaries.org.uk/news-and-insights/news/article-fitting-data-xgboost#:~:text=XGBoost%20was%20first%20released%20in,poorly%20supported%20by%20existing%20software

[15] T. Chen, “Story and lessons behind the evolution of XGBoOST,” Mar. 10, 2016. https://tqchen.com/old_post/2016-03-10-story-and-lessons-behind-the-evolution-of-xgboost

[16] J. H. Friedman, “Greedy function approximation: A gradient boosting machine.,” *The Annals of Statistics*, vol. 29, no. 5, Oct. 2001, doi: 10.1214/aos/1013203451.

[17] K. Bandara, C. Bergmeir, and S. Smyl, “Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach,” *Expert Systems With Applications*, vol. 140, p. 112896, Aug. 2019, doi: 10.1016/j.eswa.2019.112896.

[18] M. Geurts, G. E. P. Box, and G. M. Jenkins, “Time Series Analysis: Forecasting and control,” *Journal of Marketing Research*, vol. 14, no. 2, p. 269, May 1977, doi: 10.2307/3150485.

[19] B. Pavlyshenko, “Machine-Learning Models for Sales time Series Forecasting,” *Data*, vol. 4, no. 1, p. 15, Jan. 2019, doi: 10.3390/data4010015.

[20] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural Networks*, vol. 18, no. 5–6, pp. 602–610, Jul. 2005, doi: 10.1016/j.neunet.2005.06.042.

[21] M. A. Rafi *et al.*, "A Hybrid Temporal Convolutional Network and Transformer Model for Accurate and Scalable Sales Forecasting," in *IEEE Open Journal of the Computer Society*, vol. 6, pp. 380-391, 2025, doi: 10.1109/OJCS.2025.3538579

[22] G. Sun and S. Deng, “Financial Time Series Forecasting: A comparison between Traditional methods and AI-Driven Techniques,” *Journal of Computer Signal and System Research 2(2):86-93*, vol. 2, no. 2, pp. 86–93, Mar. 2025, doi: 10.71222/339b9812.

[23] Streamlit.app, 2025. https://floraedze-sales-forecasting-project-9wphj6fstwtquwh5sdltsn.streamlit.app/

[24] Streamlit.app, 2025. https://sales-forecast-dashboard-dtkt2ze8ghxhuhbysspn2r.streamlit.app/

[25] Streamlit.app, 2025. https://sales-dashboard-python-vuuhwi9dxakkn9cqrxmehl.streamlit.app/

[26]
H. Herne, W. W. Cooley, and P. R. Lohnes, "Multivariate data analysis.," *Journal of the Royal Statistical Society Series a (General)*, vol. 136, no. 1, p. 101, Jan. 1973, doi: 10.2307/2344428.

[27]
R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. 2013.

[28]
Rahul, "GitHub - Rahul2398/Amazon-Sales-Data-Analysis: In this project work, I managed the end-to-end process and analyzed sales data.," *GitHub*. https://github.com/Rahul2398/Amazon-Sales-Data-Analysis?tab=readme-ov-file

# APPENDIX

## POSTER