Sales Prediction

Submitted by:

Sudeep Bhagat (102217257)

Nitesh Yadav (102217260)

BE Third Year

CSE

Submitted to:

Dr. Anjula Mehto

Assistant Professor

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

November 2024

TABLE OF CONTENTS

S. No	Topic	Page No.
1	Introduction or Project Overview	3
2	Problem Statement	4
3	Overview of the Dataset used	5
4	Project workflow	6
5	Results	8
6	Conclusion	10

Introduction or Project Overview

This project focuses on building a sales prediction model for the Rossmann Store Sales dataset. Predicting sales is critical for businesses to manage inventory, staffing, and marketing strategies effectively. The task involved exploring time-series data, applying feature engineering, and training predictive models using two different approaches:

1. **XGBoost Regressor** - XGBoost leverages tree-based ensemble learning for structured data.

2. **LSTM (Long Short-Term Memory)** - A deep learning model suited for sequential data. LSTM is a neural network designed to capture temporal dependencies in sequential datasets.

Both models were evaluated for accuracy, and their performance was compared. This comparative study aims to identify the most effective model for improving operational efficiency and decision-making in retail businesses.

Problem Statement

The goal of this project is to predict daily sales for Rossmann stores by analyzing historical data. Accurate sales predictions enable businesses to anticipate future demand and prepare accordingly. The prediction model incorporates various factors to improve accuracy, including:

Key Features for Prediction

1. **Temporal Features**:
    - These include attributes such as year, month, day, week of the year, and day of the week, extracted from the date field.
    - Temporal features help identify trends, seasonality, and specific days or months where sales patterns change.
2. **Lagged Sales Data**:
    - Captures dependencies on previous sales data by including lagged features (e.g., sales from the last 3 days or last week).
    - This ensures the model learns patterns from recent sales activity that may influence future performance.
3. **Seasonality Patterns**:
    - Identified using Fourier transformations, which decompose the sales data into frequency components to detect periodic trends (e.g., weekly or monthly cycles).
    - Seasonality is crucial for understanding predictable sales fluctuations (e.g., festive seasons, promotional months).
4. **Store-Related Features**:
    - Attributes specific to each store, such as StoreType, Assortment, **PromoInterval**, and **CompetitionDistance**, are included to account for variations in sales due to store characteristics.

Overview of the Dataset used

The project uses two datasets:

1. **train.csv**:
    - Contains historical daily sales data for each store.
    - Includes features like Sales, Promo, and **DayOfWeek**.
2. **store.csv**:
    - Describes store-specific attributes such as:
        - **StoreType**: Categorizes stores into types (e.g., a, b, c).
        - **CompetitionDistance**: Distance to the nearest competitor.
        - **PromoInterval**: Months when promotional campaigns occur.

Key Features

- **Target Variable**: Sales (numeric value representing daily sales for each store).
- **Categorical Features**:
    - **StoreType, Assortment, PromoInterval, and DayOfWeek**.
- **Numerical Features**:
    - **CompetitionDistance** and sales-related temporal features.

Challenges in Data

- Missing values in features such as **CompetitionDistance** and **PromoInterval**.
- The need for feature extraction from the **Date** column.

Project Workflow

Steps Followed:

Step 1: Data Loading and Merging

- Imported train.csv and store.csv datasets.
- Merged them based on the Store column using a left join to ensure all sales records had store-specific attributes.

Step 2: Data Preprocessing

- Handling Missing Values:
  - Imputed missing values for CompetitionDistance using the median.
  - Logical defaults were used for temporal and categorical fields (e.g., replacing null values in Promo2SinceYear with 0).
- Date Conversion:
  - Extracted useful time-based features (Year, Month, WeekOfYear, DayOfWeek) from the Date column.
- Dropping Irrelevant Columns:
  - Removed columns like Customers (not required for prediction) and StateHoliday (already represented in Promo).

Step 3: Feature Engineering

- Lag Features:
  - Created lagged values of sales to incorporate dependencies on past data.
  - Example: lag_1 captures the sales value from 1 day prior.

- Rolling Means:
    - Generated rolling averages over 5-day windows to smooth fluctuations and capture trends.
- Fourier Transform:
    - Applied Fourier transformation on sales to identify cyclical patterns.
- Label Encoding:
    - Converted categorical variables (StoreType, Assortment) into numerical formats using LabelEncoder.

Step 4: Model Training

- XGBoost:
    - Performed hyperparameter tuning using GridSearchCV to identify the optimal values for learning_rate, max_depth, and other parameters.
    - Trained the model on processed features and evaluated using metrics.
- LSTM:
    - Reshaped data into a 3D format required for sequential learning ([samples, timesteps, features]).
    - Defined an LSTM network with dropout layers to reduce overfitting.
    - Trained the model to learn temporal dependencies in the data.

Step 5: Evaluation

- Models were evaluated on test data using MAE. Predictions were compared visually against actual sales for specific time intervals.

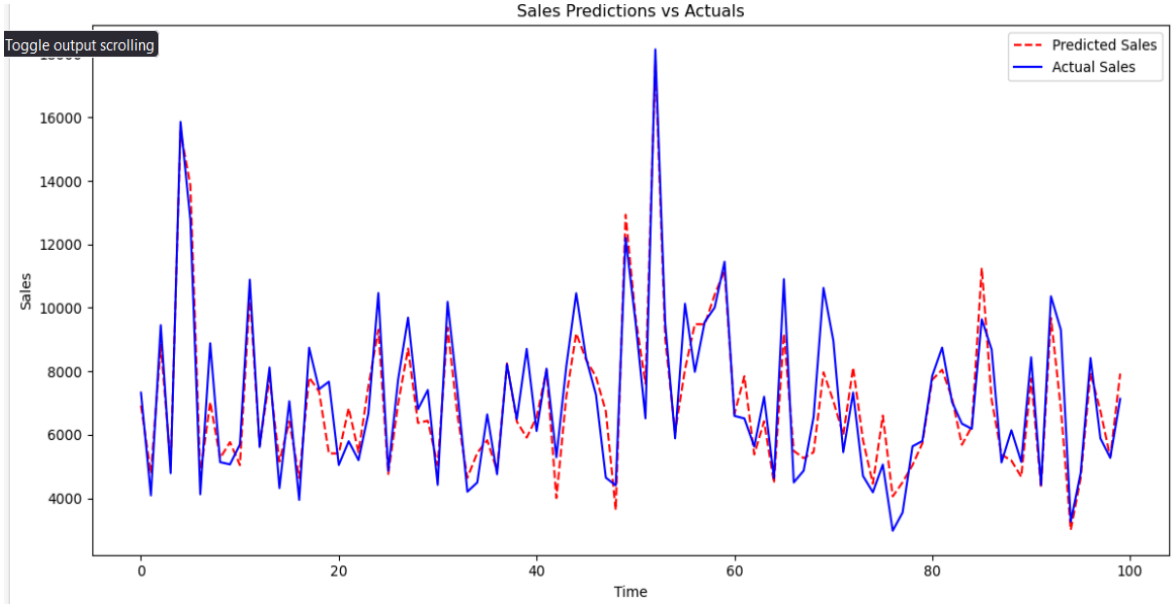
Results

XGBoost Results:

MAE : 444.96

Best Parameters: {colsample_bytree : 1.0, learning_rate : 0.2, max_depth : 7, subsample : 1.0}

- Key Insights:
  - XGBoost performed well, achieving lower error values.
  - It effectively utilized the structured features created during preprocessing.

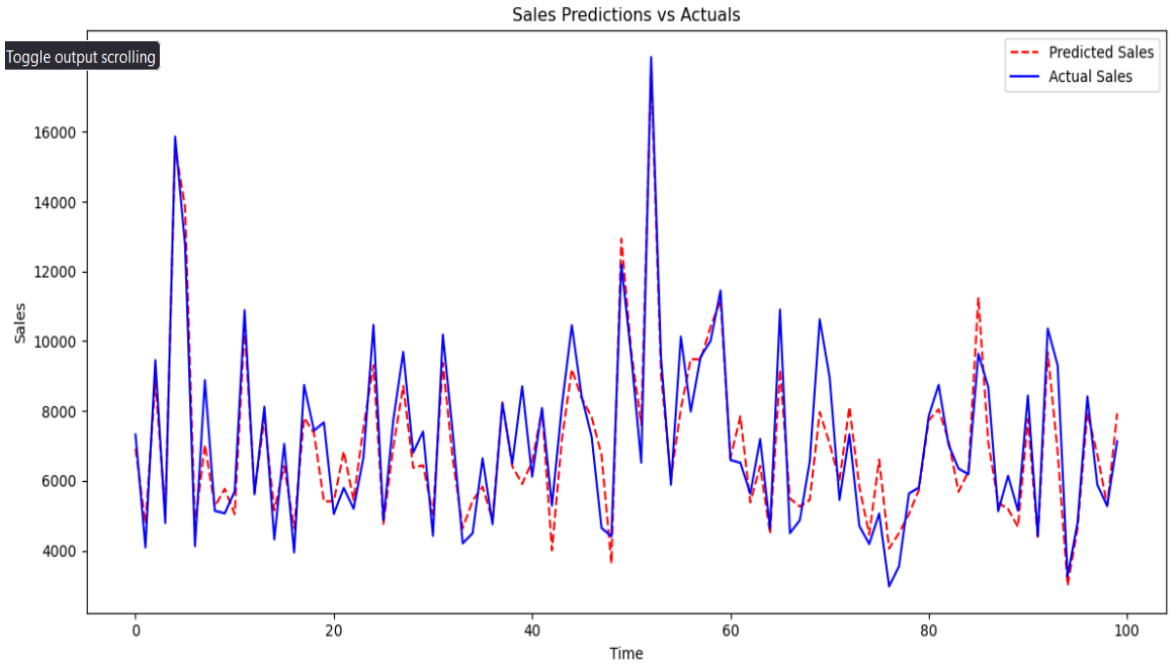


	Sales Predictions	Sales Actuals
0	6922.435059	7326.0
1	4777.191895	4091.0
2	8830.404297	9453.0
3	5061.807129	4791.0

LSTM Results:

MAE : 1143.34

- Key Insights:
  - LSTM was slightly less accurate but better at capturing long-term trends.
  - It required significantly more computational resources than XGBoost.



	Sales Predictions	Sales Actuals
0	6922.435059	7326.0
1	4777.191895	4091.0
2	8830.404297	9453.0
3	5061.807129	4791.0

Conclusion

The project demonstrated:

- XGBoost is more efficient and accurate for structured/tabular data.
- LSTM is a promising model for long-term trend capture but requires extensive tuning.

Future Improvements

1. Model Fusion:
    - Combining XGBoost and LSTM predictions using ensemble techniques.
2. Feature Expansion:
    - Incorporate external factors such as holidays, weather, or regional events.
3. Scalability:
    - Optimize LSTM training for larger datasets with advanced techniques like transfer learning.