Supply Chain Management Demand Forecasting Report

Title: Supply Chain Management Demand Forecasting Report
Author: Akash Kumar
Date: 02/03/2025
Affiliation: Unified Mentor
Date: 02/03/2025
Project Repository: Supply-Chain-Management-Forecasting-Report

Abstract

Demand forecasting is a critical aspect of supply chain management that ensures efficient inventory control, reduces stockouts, and minimizes costs. This project applies machine learning techniques to predict product demand based on historical sales data, promotions, pricing, and other influencing factors. A neural network model with three hidden layers is trained to forecast demand using preprocessed and standardized data. The model's performance is evaluated using Mean Squared Error (MSE), and results demonstrate its capability in providing accurate demand predictions. Additionally, a Streamlit dashboard is developed for real-time visualization, enhancing decision-making in supply chain operations.

Introduction

Supply chain management involves the coordination of various processes, including procurement, production, inventory management, and distribution. Accurate demand forecasting is essential for optimizing these processes, reducing operational costs, and ensuring product availability. Traditional forecasting methods often struggle with complex patterns in demand fluctuations. This project leverages machine learning, specifically a neural network model, to improve forecasting accuracy. The model is trained on historical sales data and various external factors such as promotions and pricing. This report details the data preprocessing steps, model development, evaluation, and deployment. Additionally, a Streamlit dashboard is incorporated to provide interactive visualization for better analysis and decision-making.

Overview

Objective

The objective of this project is to develop a machine learning model for demand forecasting in supply chain management. Demand forecasting plays a crucial role in inventory optimization, supplier selection, and transportation planning, enabling businesses to reduce stockouts and overstocking issues.

Key Components

- Data Preprocessing
- Model Building
- Model Evaluation
- Model Deployment

Data Preparation

Dataset Features

The dataset includes the following columns:

- **Date:** Timestamp of the sales record.
- **ProductID:** Unique identifier for each product.
- **HistoricalSales:** Past sales data.
- **Promotion:** Indicator of whether a promotion was applied.
- **Price:** Price of the product at the time of sale.
- **Weather (if applicable):** Weather conditions that might affect sales.
- **Economic Indicators (if applicable):** Economic trends influencing demand.

Data Preprocessing Steps

1. Handling missing values
2. Feature engineering (extracting Month, Day of Week, Quarter from Date)
3. One-hot encoding categorical variables
4. Standardization and normalization of numerical features
5. Splitting the dataset into training and testing sets

Model Building

Neural Network Architecture

A neural network with three hidden layers was implemented for demand forecasting:

- **Layer 1:** 128 neurons, ReLU activation
- **Layer 2:** 64 neurons, ReLU activation
- **Layer 3:** 32 neurons, ReLU activation
- **Output Layer:** 1 neuron for demand prediction

Training Configuration

- **Optimizer:** Adam
- **Loss Function:** Mean Squared Error (MSE)
- **Epochs:** 50
- **Validation Split:** 20%

Model Evaluation

Performance Metrics

- **Mean Squared Error (MSE) on Test Set:** [MSE_value]

Visualization of Results

- **Scatter Plot:** True vs Predicted Sales
- The model shows a strong correlation between actual and predicted sales values.

Model Deployment

Saving and Loading the Model

- The trained model is saved as `demand_forecasting_model.keras`.
- The model can be loaded for future predictions.

Example Prediction

Using a new data sample, the model predicts expected sales:

```
new_data = np.array([[0.5, 1.2, 0.3, 0, 1, 0, 0, 0, 1]])
new_data_scaled = scaler.transform(new_data)
predicted_sales = loaded_model.predict(new_data_scaled)
print(f'Predicted Sales: {predicted_sales[0][0]}')
```

Conclusion and Future Work

Conclusion

- The implemented neural network provides accurate demand forecasts.
- It helps businesses optimize inventory levels, reduce stockouts, and enhance supply chain efficiency.

Future Improvements

- Incorporate additional features like holiday effects and competitor pricing.
- Experiment with alternative machine learning models (XGBoost, LSTM, etc.).
- Deploy the model as an API for real-time demand forecasting.

References

- Scikit-learn Documentation
- Keras API Guide
- Pandas and NumPy for data preprocessing
- Matplotlib and Seaborn for visualization
- TensorFlow for neural network implementation