Nap – Queens

AKULA MALLIKHARJUNA
akulamallikharjuna2001@gmail.com

Research and Development on Time Series Analysis

Introduction

Time series analysis involves understanding and forecasting data points indexed in time order. Predicting unit sales involves capturing temporal dependencies and trends in the data, which is crucial for making informed business decisions. The choice of model in this project was influenced by extensive research into various time series forecasting methods and machine learning algorithms.

Initial Exploration

1. **Exploratory Data Analysis (EDA):**
    - Plotted the data to visualize trends, seasonality, and cyclic patterns.
    - Examined statistical properties like mean, variance, autocorrelation, and partial autocorrelation functions (ACF, PACF).
2. **Feature Engineering:**
    - Extracted date-related features such as year, month, day, and day of the week.
    - Created lag features and rolling statistics (mean, standard deviation) to capture temporal dependencies.
    - These features help the model understand recent trends and seasonality.

Time Series Forecasting Methods

1. **Traditional Methods:**
    - **ARIMA (AutoRegressive Integrated Moving Average):**
        - Models the data based on its past values and errors.
        - Limitations: Requires stationarity, can be complex to tune, not suitable for non-linear patterns.
    - **Exponential Smoothing:**
        - Captures level, trend, and seasonality.
        - Limitations: Assumes linearity and may not perform well with complex data.
2. **Machine Learning Methods:**
    - **Linear Regression:**
        - Uses lag features to predict future values.
        - Limitations: Fails to capture non-linear patterns.
    - **Decision Trees:**
        - Splits the data based on feature values to make predictions.
        - Limitations: Prone to overfitting, may not capture temporal dependencies well.
3. **Advanced Methods:**

- **Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) Networks:**
  - Designed to capture long-term dependencies in sequential data.
  - Limitations: Computationally expensive, require large datasets, complex to train.

Choosing XGBoost

After comparing various methods, XGBoost was selected for the following reasons:

1. **Handling of Complex Data:**
   - XGBoost can handle non-linear relationships and complex interactions between features.
   - It effectively captures the impact of recent trends and patterns through lag features and rolling statistics.
2. **Robustness and Flexibility:**
   - XGBoost includes regularization parameters to prevent overfitting.
   - It is robust to outliers and noisy data, making it suitable for real-world datasets.
3. **Performance and Efficiency:**
   - XGBoost is optimized for speed and performance, leveraging parallel processing.
   - It provides competitive accuracy and faster training times compared to other advanced methods like RNNs and LSTMs.
4. **Feature Importance and Interpretability:**
   - XGBoost provides insights into feature importance, helping to understand which features contribute most to the prediction.

Implementation

1. **Feature Selection:**
   - Chose features such as year, month, day, day of the week, unit price, and lagged unit sales (lag1, lag7, lag30) along with rolling statistics (mean, std).
2. **Data Preprocessing:**
   - Handled missing values by filling them with zeros.
   - Split the data into training and validation sets to evaluate the model's performance.
3. **Model Training and Evaluation:**
   - Used `XGBRegressor` with default parameters and objective as 'reg

     '.

   - Trained the model on the training set and validated on the validation set.
   - Calculated Mean Squared Error (MSE) to assess the model's accuracy.
4. **Prediction:**
   - Prepared the test data by engineering the same features used in the training phase.
   - Predicted unit sales and generated a submission file with the required format.

Conclusion

The research and experimentation with various time series forecasting methods led to the selection of XGBoost for its robustness, performance, and flexibility. By engineering relevant features and leveraging XGBoost's capabilities, we developed an accurate and efficient model for predicting unit sales. This approach balances the complexity and interpretability, ensuring reliable forecasts for business decision-making.