R&D ON TIME SERIES FORECASTING FOR AMAZON SALES DATA

VINITH J
02-08-2024

---

1. **Introduction:**

   Objective:
   To develop a time series forecasting model to predict the number of units sold for each item ID using dummy sales data from Amazon (USA).

Importance:

- Accurate sales forecasting is crucial for inventory management, marketing strategies, and operational efficiency.
- Time series analysis keeps time intervals in specific time periods into consideration while dealing with data.
- Machine Learning models provide the ability to automatically learn and improve from its experiences.
- Suryoday B. et al. implemented Random Forests and XGBoost classifiers that are used to predict the stock values. The robustness can be evaluated by the parameters- accuracy, precision, recall, and F-score. Exponential smoothing, which is a rule of thumb technique, is used for smoothing the time series. It is founded that the XGBoost classifier works better than the random forest classifier.

---

2. **Dataset Description**

Source: The dataset contains dummy sales data for various items sold by a well-known brand on Amazon (USA).

Columns:

- **date**: The date on which the sales data was recorded, formatted as YYYY-MM-DD.
- **Item Id**: A unique identifier for each item.
- **Item Name**: The name of the item, providing a human-readable reference.
- **anarix_id**: An internal identifier used for tracking or categorizing items within the brand's system.
- **ad_spend**: The amount of money spent on advertising for the item on the given date.
- **units**: The number of units sold for the item on the given date (target variable).
- **orderedrevenueamount**: The total revenue generated from the units sold on the given date.
- **unit_price**: The price per unit of the item.

- **Dataset Statistics**:

The dataset consists of 101,488 entries with the following summary statistics:
ad_spend: Mean = 84.37, Min = 0, Max = 47,934.99, Std Dev = 464.35
units: Mean = 8.51, Min = 0, Max = 9,004, Std Dev = 62.68
unit_price: Mean = 107.26, Min = 0, Max = 21,557.39, Std Dev = 424.51

---

3. Preprocessing

Data Cleaning:

- Handle missing values – for numerical values like ad spend and units filled with 0 in place of missing values and for Item name filled with 'unknown' .
- This ensures that the dataset is more precise and cleaned properly.
- Reason for Not Using Mean to Fill Missing Values for Numerical Data
- In statistical terms, the mean is highly sensitive to outliers. When the dataset contains a significant number of outliers, especially those with extremely high values, the mean can be disproportionately affected. This can lead to a mean value that is not representative of the central tendency of the data.
- In the given dataset, the presence of high maximum values (outliers) skews the mean upwards, making it a less reliable measure of central tendency compared to the median, which is less influenced by outliers. Therefore, using the mean to fill in missing values could introduce bias and distort the data distribution, leading to inaccurate analysis and predictions.
- After cleaning, the train dataset consists of 101,488 entries. The mean advertising spend is $84.37 with a standard deviation of $464.35, and the mean number of units sold is 8.51 with a standard deviation of 62.68.

---

4. EDA- Exploratory Data Analysis

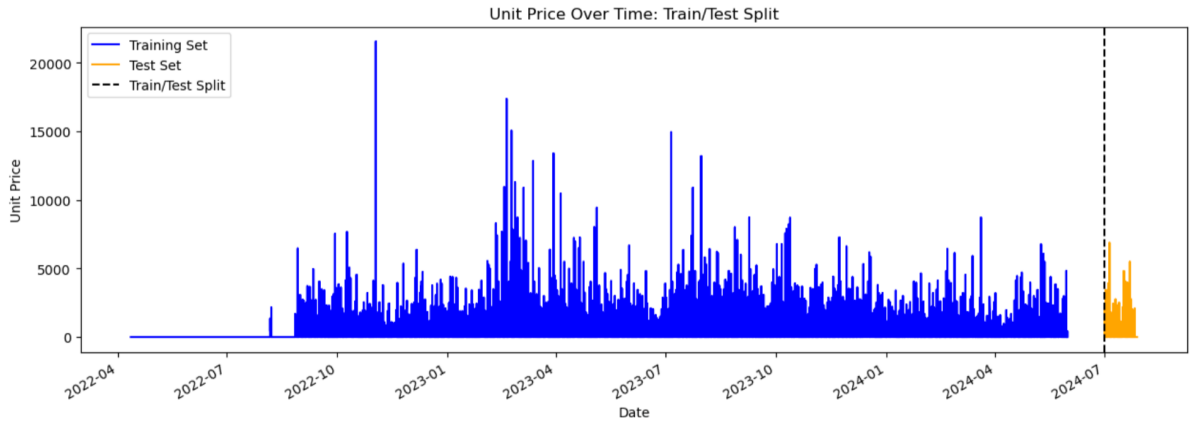
Top 10 Selling Items
Based on the sales data, the top 10 selling items by the number of units sold are:

- **Item ID B09KT5HMNY**: 102,149 units
- **Item ID B0BGDWGRQB**: 99,586 units
- **Item ID B09MR3Y296**: 82,697 units
- **Item ID B09MR4B13C**: 78,548 units
- **Item ID B0BGDX2Z3L**: 33,810 units
- **Item ID B09KTMKDKJ**: 31,790 units
- **Item ID B0BRCYQNSW**: 26,531 units
- **Item ID B09MR3XT5G**: 21,815 units
- **Item ID B0B699PLXD**: 21,622 units

- **Item ID B0BGDZLDX2**: 20,691 units

Top 10 Items by Advertising Spend
The top 10 items by advertising spend are as follows:

- **Item ID B09KT5HMNY**: $851,785.25
- **Item ID B09MR4B13C**: $607,033.59
- **Item ID B09MR3Y296**: $548,484.97
- **Item ID B0BRCYQNSW**: $434,229.75
- **Item ID B09KTF8ZDQ**: $336,946.80
- **Item ID B09KTMKDKJ**: $266,972.17
- **Item ID B0BGDWGRQB**: $262,212.18
- **Item ID B09KDZQJ6P**: $255,192.82
- **Item ID B0B69B8R3C**: $247,132.75
- **Item ID B09MR3XT5G**: $227,750.13



Unit price fluctuates significantly over time, with a clear distinction between training and test sets

---

5. **Feature Engineering:**

The create_features function converts the 'date' column to datetime format and adds new columns for day of the week, month, quarter, year, and a weekend indicator. It is applied to both train_df and test_df. This enhances the dataset with temporal features for improved model performance.

---

6. **Model selection:**

The model used is XGBRegressor.

1. Handling High Variability and Complexity:

- **Dataset Characteristics**: The dataset exhibits high variability in both advertising spend and unit price, with significant outliers and diverse value ranges. XGBoost's robustness to such variability makes it suitable for handling these complexities.
- **Model Choice**: XGBoost is well-suited for datasets with complex relationships and non-linear patterns due to its gradient boosting framework, which iteratively improves model accuracy by focusing on errors from previous iterations.

2. Feature Importance and Interpretability:

- **Feature Engineering**: The dataset includes various features like day of the week, month, and advertising spend. XGBoost can automatically capture and weigh the importance of these features, providing insights into how different variables impact sales.
- **Model Choice**: XGBoost's ability to evaluate feature importance helps in understanding the contribution of different features to the predictions, which is crucial for interpreting the model's performance and making informed decisions.

3. Performance and Efficiency:

- **Training Time**: XGBoost's optimization techniques, such as regularization and parallel processing, lead to faster training times compared to other algorithms. Given the large size of the dataset, efficiency in model training is crucial.
- **Model Choice**: XGBoost's efficiency ensures that the model can be trained and tuned within a reasonable timeframe, even with extensive hyperparameter tuning.

4. Accuracy and Predictive Power:

- **Evaluation Metrics**: The initial performance of XGBoost, with an MSE of 3911.418 before tuning, indicates that the model can capture the data patterns effectively. After hyperparameter tuning, the MSE improved to 1443.7186, demonstrating significant enhancement in predictive accuracy.
- **Model Choice**: XGBoost's ability to improve accuracy through fine-tuning aligns with the goal of minimizing prediction errors and achieving reliable forecasts for unit sales.
- **Initial Model Training**: The XGBoost model was initially trained using default hyperparameters to establish a baseline performance. This provided a starting point for evaluating the model's ability to handle the dataset.

7. **Hyperparameter Tuning**:

- A grid search was performed to identify the optimal hyperparameters, including colsample_bytree, learning_rate, max_depth, min_child_weight, n_estimators, and subsample. This process fine-tuned the model to achieve better performance.

- **Validation and Evaluation**: The model was validated using RMSE as the primary metric. The performance improved from the initial RMSE of 62.67904 to 62.54133 after tuning, and the final validation MSE of 1443.7186 indicated substantial improvement in predictive accuracy.

- **Feature Engineering**: Temporal features such as day of the week, month, and quarter were added to enhance the model's ability to capture seasonal patterns and trends. XGBoost's capability to incorporate and weight these features contributed to its effectiveness.

After hyperparameter tuning, the best parameters for the XGBoost model are:
colsample_bytree: 0.7
learning_rate: 0.01
max_depth: 5
min_child_weight: 1

n_estimators: 200
subsample: 1.0
With these parameters, the validation Mean Squared Error (MSE) improved significantly to 1443.7186, reflecting enhanced model performance.

---

8. REFERENCE :

1. https://www.kaggle.com/code/dansbecker/xgboost
2. Hirabayashi, A., Aranha, C., & Iba, H. (2009, July). Optimization of the trading rule in foreign exchange using genetic algorithm. In Proceedings of the 11th Annual conference on Genetic and evolutionary computation (pp. 1529-1536). ACM.

3. Raghavendra Kumar; Pardeep Kumar; Yugal Kumar , Analysis of Financial Time Series Forecasting using Deep Learning Model, 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 10.1109/Confluence51648.2021.9377158

4. https://xgboost.readthedocs.io/en/latest/prediction.html
5. https://thecleverprogrammer.com/2022/01/17/time-series-analysis-using-python/
6. https://www.kdnuggets.com/2023/08/leveraging-xgboost-timeseries-forecasting.html