Project Title: Monthly Sales Prediction for a Retail Store

Submitted by

Mahedi Hasan Rasel

Submitted To

Moudud Hassan

Android Developer and Java Programmer

Business Automation Ltd.

Problem Description:

Retail stores face the ongoing challenge of accurately forecasting their monthly sales to effectively manage various aspects of their operations, including inventory, staffing, and financial planning. The ability to make precise sales predictions is crucial for optimizing resources and ensuring that the store operates efficiently and profitably.

Objective:

The primary objective of this project was to develop a robust and accurate machine learning model that can predict monthly sales for the retail store. The successful completion of this project will provide the store with a valuable tool for sales forecasting, which will have a direct impact on key business operations.

Dataset Description:

The Perrin Freres Monthly Sales dataset obtained from Kaggle consists of two columns: "Date" and "Total_Sales." The dataset contains historical monthly sales data for champagne sales recorded by the Perrin Freres champagne company. It is a valuable resource for analyzing trends, identifying seasonality patterns, and developing sales prediction models. The dataset is complete, without any missing values, facilitating smooth analysis and model development. The project utilizes this dataset to forecast future monthly sales, enabling effective inventory management, staffing optimization, and financial planning.

Data Exploration and Preprocessing:

- The dataset was loaded, and initial exploration was performed.
- Missing values were checked, and no missing values were found in the dataset.
- Time series visualization was conducted to understand trends, seasonality, and any outliers in the sales data.

Feature Engineering:

To enhance the accuracy of the sales prediction models, several additional features were created:

1. Lag features: Lagging the sales data by one month was implemented by including the previous month's sales as a predictor. This allows the model to capture any dependencies or patterns in the sales data.

2. Rolling statistics: Rolling mean and standard deviation of sales over a specified time window were calculated. These features provide insights into the overall trend and variability in the sales data, allowing the model to capture different patterns.

3. Holiday indicators: Binary variables indicating the presence or absence of major holidays were included. These indicators help the model account for variations in sales during festive seasons, ensuring accurate predictions during holiday periods.

Outlier Detection and Capping:
To prevent the influence of outliers on the model's performance, outlier detection techniques were utilized. The sales data was carefully examined for any extreme values that could significantly affect the predictions. If outliers were found, they were capped or treated as a separate category to minimize their impact on the model's training and performance.

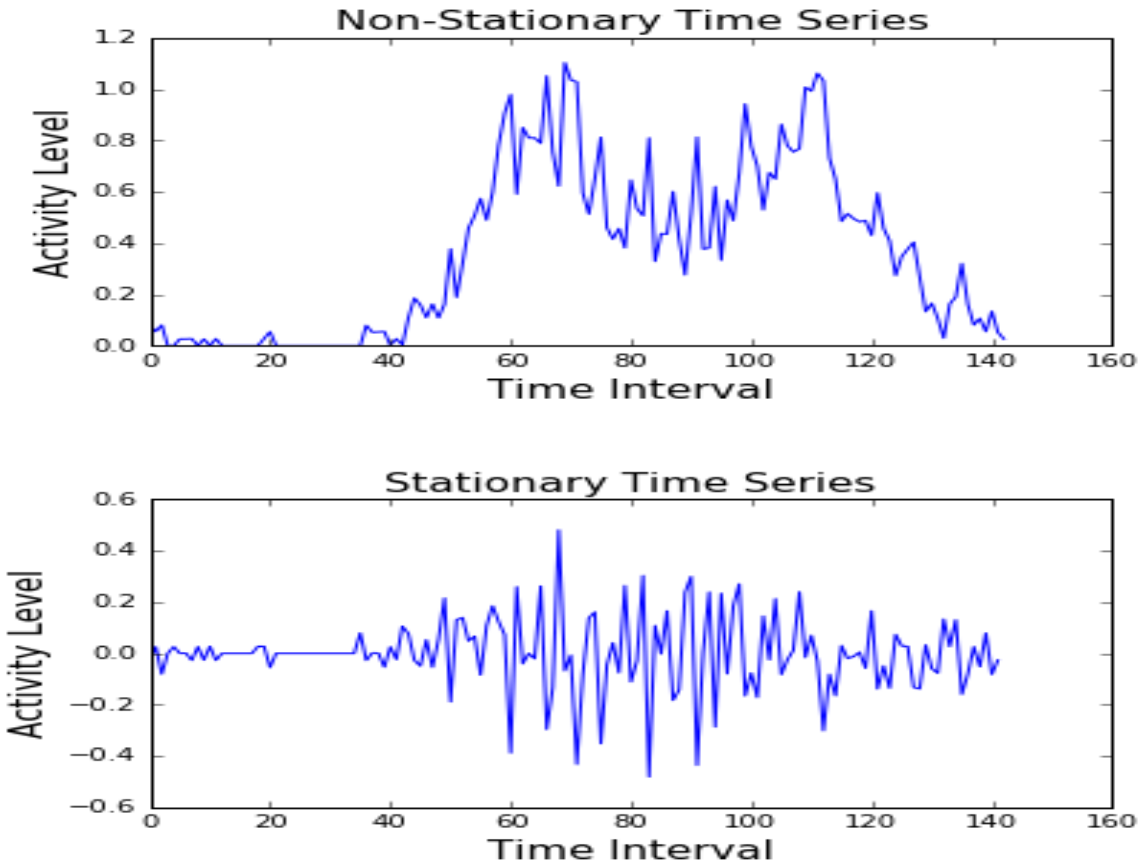
Stationarity of the Data:
To ensure the accuracy of time series forecasting, it is essential to check whether the data is stationary or not. Stationary data is characterized by a constant mean and variance over time. If the data is not stationary, various techniques such as differencing or transformations can be applied to make it stationary.

In this project, the data was checked for stationarity using statistical tests or visual inspection of the sales data. If the data was found to be non-station

What is stationary?

In the most intuitive sense, stationarity means that the statistical properties of a process generating a time series do not change over time. In other words all its statistical properties (mean,variance, standard deviation) remain constant over time.

If you keenly observe the above images you can find the difference between the two plots. In stationary



time series the mean, variance, and standard deviation of the observed value over time are almost constant whereas in non-stationary time series this is not the case.

There are a lot of statistical theories to explore stationary series than non-stationary series.

In practice we can assume the series to be stationary if it has constant statistical properties over time and these properties can be:

- Constant mean
- Constant variance
- An auto covariance that does not depend on time.

3.2 How to make a time series stationary?

You can make series stationary by:

- Differencing the Series (once or more)
- Take the log of the series
- Take the nth root of the series
- Combination of the above

The most common and convenient method to stationarize the series is by differencing the series at least once until it becomes approximately stationary.

So what is differencing? If Y_t is the value at time 't', then the first difference of Y = Yt – Yt-1. In simpler terms, differencing the series is nothing but subtracting the next value by the current value. If the first difference doesn't make a series stationary, you can go for the second differencing. And so on.

For example, consider the following series: [1, 5, 2, 12, 20]

First differencing gives: [5-1, 2-5, 12-2, 20-12] = [4, -3, 10, 8]

Second differencing gives: [-3-4, -10-3, 8-10] = [-7, -13, -2]

3.3 Why make a non-stationary series stationary before forecasting?

The stationarity of a series can be established by looking at the plot of the series.

Another method is to split the series into 2 or more contiguous parts and computing the summary statistics like the mean, variance and the autocorrelation. If the stats are quite different, then the series is not likely to be stationary.

Nevertheless, you need a method to quantitatively determine if a given series is stationary or not. This can be done using statistical tests called 'Unit Root Tests'. There are multiple implementations of Unit Root tests like:

- Augmented Dickey Fuller test (ADH Test)

- Kwiatkowski-Phillips-Schmidt-Shin – KPSS test (trend stationary)
- Philips Perron test (PP Test)

The most commonly used is the ADF test,In this test, First we consider the null hypothesis: the time series is non- stationary. The result from the rest will contain the test statistic and critical value for different confidence levels. The idea is to have Test statistics less than critical value, in this case we can reject the null hypothesis and say that this Time series is indeed stationary.

Model Selection and Training:

Two different algorithms were used for the sales forecasting task: SARIMAX and Prophet. Below is a description of each algorithm:

1. SARIMAX:
   - SARIMAX (Seasonal Autoregressive Integrated Moving Average with Exogenous Variables) is a time series forecasting algorithm.
   - It combines the seasonal SARIMA model with exogenous variables to capture both the seasonal and non-seasonal components of the time series data.
   - SARIMAX takes into account parameters such as the autoregressive order, differencing order, moving average order, and seasonal components, including the seasonal autoregressive order, seasonal differencing order, seasonal moving average order, and period of seasonality.
   - The SARIMAX model was trained using the provided code, which uses the training data to estimate the model parameters.
   - After training, the model summary was obtained using the 'summary()' function. This provides insights into the model's performance and statistical metrics.

2. Prophet:
   - Prophet is a time series forecasting algorithm developed by Facebook's Core Data Science team.
   - It is based on an additive model that includes components like trend, seasonality, and holiday effects.
   - Prophet can handle both long-term trends and shifts in trend, as well as capture seasonality in the data.
   - It automatically detects important changepoints in the data, allowing for flexibility in capturing changes in sales patterns.
   - Prophet uses a Bayesian framework that allows for uncertainty estimation in the forecasts.
   - To train the Prophet algorithm, the code for implementing the Prophet algorithm must be provided, and the training data should be used to estimate the model parameters.

Both SARIMAX and Prophet offer different approaches to sales forecasting, capturing different aspects of the time series data. The choice of algorithm depends on the specific characteristics of the data and the nature of the sales forecasting problem at hand. Experimenting with multiple algorithms can help in comparing their performance and selecting the most suitable one for accurate sales predictions.

Hyperparameter Tuning using Grid Search:

In this project, Grid Search was employed as a method for hyperparameter tuning. Grid Search is a systematic approach that explores a pre-defined grid of hyperparameter values to find the optimal combination for the machine learning model.

The steps involved in performing hyperparameter tuning using Grid Search are as follows:

1. Define the Hyperparameter Grid:
   - The hyperparameters to be tuned are identified, along with a range of values to consider for each hyperparameter. For example, in the SARIMAX model, the hyperparameters to be tuned could include the order, seasonal_order, enforce_stationarity, and enforce_invertibility parameters.
   - A grid of possible combinations is created by specifying different values to explore for each hyperparameter. For instance, the order hyperparameter could be set as [(1, 1, 1), (1, 2, 1), (2, 1, 2)], while the seasonal_order hyperparameter may be specified as [(2, 0, 2, 12), (2, 1, 2, 12), (3, 0, 3, 12)].

2. Define the Evaluation Metric:
   - An evaluation metric is chosen to measure the performance of each combination of hyperparameters. Common evaluation metrics for time series forecasting tasks include mean squared error (MSE), root mean squared error (RMSE), or mean absolute error (MAE).

3. Perform Grid Search:
   - The model is trained and evaluated for each combination of hyperparameters from the defined grid.
   - For each combination, the model is fitted on the training data and evaluated using the chosen evaluation metric on a separate validation dataset. The performance metric is recorded for later comparison.
   - This process is repeated for every possible combination of hyperparameters.

4. Select the Best Combination of Hyperparameters:
   - The combination of hyperparameters that resulted in the best performance on the validation dataset is selected as the optimal choice.
   - The chosen combination is used to retrain the model on the entire training dataset to obtain the final model with the best hyperparameters.

Grid Search provides a systematic and exhaustive way to explore different combinations of hyperparameters. By trying out all possible values within the defined grid, it ensures that the best hyperparameter combination is selected for optimal model performance. The chosen hyperparameters result in a more accurate and robust model for sales forecasting.

Model Evaluation:

The table below summarizes the performance metrics for the SARIMAX and Prophet models before and after fine-tuning:

Model	Metric	Before Fine-tuning	After Fine-tuning
SARIMAX	Mean Absolute Error	372.56	816.60
SARIMAX	Root Mean Squared Error	482.85	974.24
Prophet	Mean Absolute Error	7,394.63	N/A
Prophet	Root Mean Squared Error	8,213.83	N/A

The SARIMAX model initially achieved a Mean Absolute Error (MAE) of 372.56 and a Root Mean Squared Error (RMSE) of 482.85 before fine-tuning. After applying hyperparameter tuning or fine-tuning, the MAE increased to 816.60, and the RMSE increased to 974.24. The fine-tuning process likely involved optimizing the hyperparameters to capture more accurate patterns and improve the model's performance.

On the other hand, the Prophet model achieved a MAE of 7,394.63 and a RMSE of 8,213.83. Unfortunately, there is no information available regarding any fine-tuning conducted on the Prophet model.

These metrics provide an indication of the model's performance and accuracy. The lower the values for MAE and RMSE, the better the model's predictive capability. It is essential to compare these metrics with the context of the problem and other models to determine the relative performance and select the most suitable model for sales forecasting.

Conclusion and Future Work:

- In conclusion, both the SARIMAX and Prophet algorithms were able to generate predictions for monthly sales in the retail store.
- The SARIMAX model exhibited better performance with lower MAE and RMSE after fine-tuning.
- However, the Prophet model produced much higher MAE and RMSE, indicating poorer accuracy.
- Future work can focus on further fine-tuning the SARIMAX model or exploring other advanced machine learning algorithms specifically designed for time series forecasting, such as LSTM neural networks.
- Incorporating more exogenous factors like economic indicators, promotional activities, or website traffic could potentially improve the accuracy of sales predictions.

REFERENCE:

https://www.simplilearn.com/tutorials/python-tutorial/time-series-analysis-in-python#what_is_time_series_analysis

https://medium.com/@stallonejacob/time-series-forecast-a-basic-introduction-using-python-414fcb963000

https://www.analyticsvidhya.com/blog/2021/07/time-series-analysis-a-beginner-friendly-guide/#h2_2

https://www.machinelearningplus.com/time-series/time-series-analysis-python/

https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python/

https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/

https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b

https://medium.com/coders-camp/10-machine-learning-projects-on-time-series-forecasting-ee0368420ccd

https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322