Assignment 2  
ML as a Service

short line

Tahmidul Islam

Student ID: 245787139  
16th October 2023

|  |  |
| --- | --- |
| Github | https://github.com/tahmislam21-uts/adv\_mla\_at2 |
| Heroku | https://aqueous-eyrie-84500-297a02143db1.herokuapp.com/ |

36120 - Advanced Machine Learning Application

Master of Data Science and Innovation

University of Technology of Sydney

Table of Contents

[1. Executive Summary 2](#_Toc148355305)

[2. Business Understanding 3](#_Toc148355306)

[a. Business Use Cases 3](#_Toc148355307)

[3. Data Understanding 4](#_Toc148355308)

[4. Data Preparation 5](#_Toc148355309)

[5. Modeling 7](#_Toc148355310)

[a. XGBoost 7](#_Toc148355311)

[b. Prophet 7](#_Toc148355312)

[6. Evaluation 9](#_Toc148355314)

[a. Evaluation Metrics 9](#_Toc148355315)

[b. Results and Analysis 9](#_Toc148355316)

[c. Business Impact and Benefits 10](#_Toc148355317)

[d. Data Privacy and Ethical Concerns 11](#_Toc148355318)

[7. Deployment 12](#_Toc148355319)

[8. Conclusion 13](#_Toc148355320)

[9. References 14](#_Toc148355321)

# 

# Executive Summary

The main target of this project was to upgrade an American retail company and increase its operational capacity and effectiveness by building strong and un-fragile predictive and forecasting models. By using these models, the project targeted to predict item-specific sales in individual stores for the next 7 days (because it had 10 stores across different states in the US). Using the models, the store would be better able to manage its supply chain, thereby doing better business.

short dash

# Business Understanding

## Business Use Cases

The three different states where the retail store has its 10 stores located are Texas (TX), California (CA) and Wisconsin (WI), where it sells three main products namely foods, household items and hobbies. Now, the company decided to improve its supply chain by making use of machine learning models to make predictions and forecast its sales revenue. This will help them meet customer regional demands, improve goodwill and hopefully lower the cost of operations.

With great usefulness, the project did have several challenges, first of which was the diverse patterns of sales and different ranges of products. Secondly, data when it is generated, are influenced by a lot of external factors like economic and environmental, which may often not get captured in the dataset.

1. Key Objectives

The project is divided into two segments and thus has two main goals. The first is to predict sales revenue given an item, store and date. To achieve this, we plan to develop a predictive model by using historical data from different stores across several years. The model must be robust and accurate. This can help stakeholders improve the way they manage inventory. It will do so by accurately predicting the demands, thereby helping order the right amount of items and prevent over-stocking.

In the second segment, we build a forecasting model that forecasts total revenue across all the stores and items for the next 7 days. The forecasting would be done by a time-series analysis and provide the store (the primary stakeholder) to fine-tune its strategies by predicting demand and revenue trends.

short dash

# Data Understanding

The project uses dataset that contains very useful information to help our models make accurate, robust predictions.

Some of the important features are discussed below:

* Calendar events: event name and type (eg. Eid al Adha, Religious)
* Pricing Information: weekly sale prices
* Store information: Store location, State
* Product Information: Product category, department

There were certain limitations of the dataset like external factors. This includes the economic status of the State, the environmental conditions (e.g. weather conditions that often affect logistics), etc.

short dash

# Data Preparation

To prepare the data, we initiated the process by merging the individual datasets into a unified dataset.

We performed a left join between the calendar data frame and the calendar\_events data frame, using the date column as the key, resulting in a combined data frame named merged\_calendar.

Following this we transformed the sales\_train data frame from a wide format to a long format, introducing a day\_num column that encompassed all unique day numbers, along with an items\_sold column representing the number of items sold.

Then, the sales\_train data frame in its long format was merged with the merged\_calendar data frame, using the day\_num column as the linking key, leading to the creation of a data frame known as sales\_train\_calendar.

Further, we incorporated the items\_weekly\_prices data into the sales\_train\_calendar data frame, thus forming the consolidated dataset.

Following the creation of the unified dataset, a comprehensive data analysis was conducted to identify and address any invalid or missing values. It was observed that only the event\_name and event\_type columns contained missing values, and it was logical because not every day, in a year of 365 days, would have an event.

**Data Preparation for Predictive Modelling:**

In order to prepare the data for predictive modeling, a series of preprocessing and feature engineering steps were implemented:

The date column served as the basis for engineering new features, including year\_month (e.g., 2011-01), day\_name (e.g., Monday), and year (e.g., 2011).

The sales\_revenue target column was generated by calculating the product of items\_sold and sell\_price.

To handle missing values in categorical columns, a new category, MISSING, was introduced. This strategy allowed the model to learn how to manage missing data effectively.

Categorical columns with a high number of distinct categories (item\_id, date, event\_name, year\_month) were subjected to mean target encoding, a technique suited for dealing with large category counts in each column.

Categorical columns with low cardinality (dept\_id, cat\_id, store\_id, state\_id, event\_type, day\_name, year) underwent one-hot encoding.

The categorical columns mentioned in the previous two points were subsequently employed for modeling.

In addition to these data preprocessing and feature engineering steps, the dataset was partitioned into distinct train, validation, and test sets. This division was based on sorting the data by the date column. The first 80% of the data constituted the train set, the next 10% was allocated to the validation set, and the remaining 10% formed the test set. This approach was adopted because the data followed a time-series structure, and it was essential to prevent any form of data leakage between the dataset splits.

**Data Preparation for Forecasting Modelling:**

In preparation for forecast modeling, the following steps were executed:

Within the combined dataset, a new target column, denoted as sales\_revenue, was created by multiplying the values in the items\_sold column with those in the sell\_price column.

Subsequently, the combined dataset underwent a grouping process based on the date column. This entailed the calculation of the total sales revenue across all stores for each specific date.

The resulting data frame designed for forecast modeling was structured with two key columns: date (renamed as ds) and total sales revenue (renamed as y).

Moreover, to facilitate the forecast modeling process, the dataset was divided into distinct train, validation, and test sets. This division was carried out following the sorting of the data by the date column. The initial 80% of the data comprised the train set, the subsequent 10% was earmarked for the validation set, and the final 10% was allocated to the test set. This approach was crucial in ensuring the integrity of the modeling process, especially given the time-series nature of the data.

short dash

# Modeling

The models used were XGBoost (for predictive) and Prophet (for forecasting).

## XGBoost

XGBoost has good capabilities to handle complex datasets. Using its ensemble of weak learners or shallow trees in a sequential manner, it makes predictions by using each learner to learn from its predecessor, which gives this algorithm great predictive power.

Hyper Parameters used:

* n\_estimator or weak learners: 150

Any more than 150 would overfit the training data

* learning\_rate: 0.05

A slow learning rate was used to have proper convergence

* max\_depth: 5

A value less than 5 would miss out the complicated nuances in the data. Value more than 5 would overfit.

* min\_child\_weight: 3

Having few samples would reduce overfitting and give better partitioning.

* gamma: 3
* subsample: 0.7
* colsample\_bytree: 0.8

## Prophet

Prophet was chosen for Time-Series analysis for forecasting. It has the ability to model by incorporating several factors (holiday and special events, in this case) for which it was chosen over other Time-Series Algorithms.

Prophet employs a sophisticated approach to data analysis by breaking down the data into three fundamental components: trend, seasonality, and holiday effects.

The trend component plays a pivotal role in identifying the underlying upward or downward trajectory within the data, granting Prophet the ability to dynamically adapt to shifts over time. This adaptability is essential for accurately capturing changes in the time series.

The seasonality component is designed to capture the recurring patterns present in the data, such as daily, weekly, or monthly cycles. This allows Prophet to account for the cyclic nature of events, which often influences data trends.

Furthermore, the holiday effects component considers the impact of holidays and special events on the time-series data. It acknowledges that certain occasions can have a significant influence on the data's behavior.

## short dash

# Evaluation

## Evaluation Metrics

The two evaluation metrics used were RMSE and MAE to assess performance.

* RMSE (Root Mean Squared Error): this algorithm calculates the distance between target values and predicted values, squares them up (to remove negative values), sums them and finally roots them. A high value shows more error while a low value shows the opposite.
* MAE (Mean Absolute Error): This algorithm does the same but doesn’t square or root the values. It just takes the absolute value of the distance and performs that average of the sum.

## Results and Analysis

Scores for XGBoost (predictive)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Train | Validation | Test |
| RMSE | 7.92 | 7.79 | 7.83 |
| MAE | 4.24 | 4.23 | 4.22 |

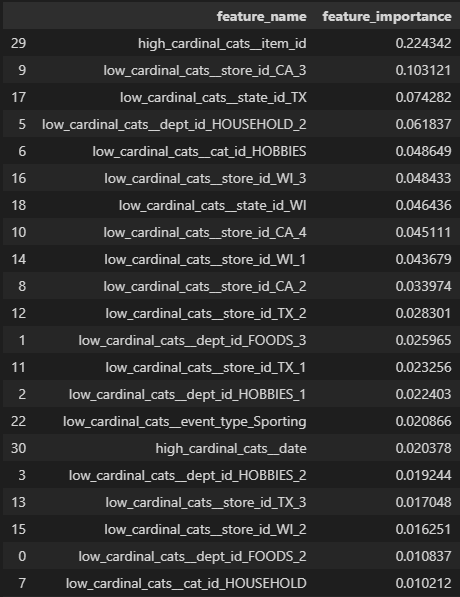
From the RMSE and MAE values, we can see that the error is lower on the Test when compared to Train. However, it is overfitting when compared to the validation score. This the overall performance is not too good nor too bad. However, the errors seem high which might suggest that the models can fit better with the data.

Scores for Prophet (forecasting)

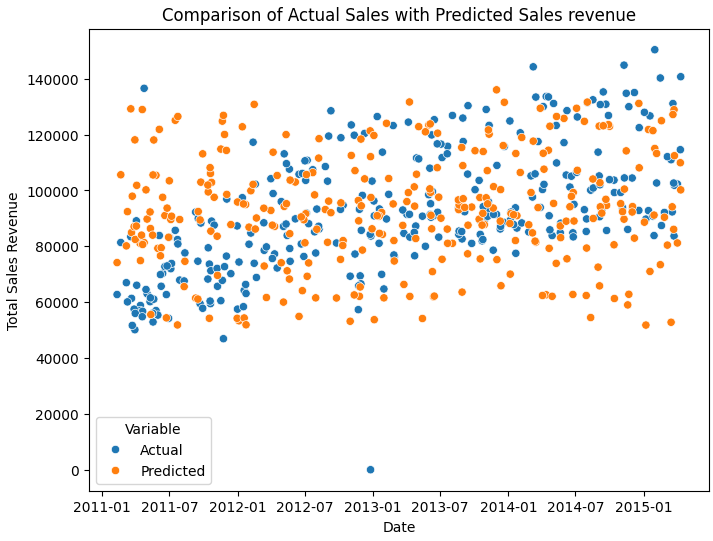
|  |  |  |  |
| --- | --- | --- | --- |
|  | Train | Validation | Test |
| RMSE | 27381.24 | 28335.36 | 28691.14 |

From this result, it can be seen that Prophet is overfitting on the training data because of having high error scores on the test data.

## Business Impact and Benefits



Although the RMSE and MAE show mediocre results, this feature importance table can provide good business insights to the store. It shows that CA\_3, TX and HOUSEHOLD have the top 3 highest feature importance. This means that branch 3 in California plays an important role, branches in Texas are significant Team players and HOUSEHOLD category 2 is also important for the business. They can now focus on these factors to improve on their business by improving them or investing more on them.



The graph above shows the performance of the forecasting model. We know it was overfitting from our previous section, for which it predicted high values at the start and relatively lower values near the end (where we can see isolated orange data points in the middle region). However, it does abide with the Actual sales value in most cases, something which the business can use to approximate their future sales.

## Data Privacy and Ethical Concerns

The dataset has sensitive business information like the sales data for every store and region. An ethical concern might rise because the model may give out biased recommendations for different regions and stores, thus creating possible discriminations.

To remove this concern, the stakeholders need to be explained how the Algorithms work and why biasness is happening in the system. Furthermore, customer data needs to be de-identified so that the model doesn’t do any profiling.

short dash

# Deployment

The deployment involved building an API service with 3 endpoints.

* Root endpoint (/): This endpoint responds with an HTML description of the project objectives.
* Health Check Endpoint (/health): Returns a simple welcome message indicating that the API is operational.
* Sales Prediction Endpoint (/sales/stores/items):
  + Accepts query parameters item\_id, store\_id, and date.
  + Uses the XGBoost to predict sales revenue based on the input parameters.
  + Returns the prediction.
* Sales Forecasting Endpoint (/sales/national):
  + Accepts a date parameter
  + Uses the Prophet model to forecast total sales revenue for the next 7 days starting from the input date.
  + Returns the forecasted sales revenue as a JSON response.

The API service was deployed to Heroku and all necessary dependent libraries were made to be installed.

short dash

# Conclusion

To conclude, this project gives a step-forward to a data-centric retail management system. While the predictive and forecasting models performed mediocre, it did provide certain insights to the business that can greatly improve their service long term.

As future works, the project may be enhanced by incorporating more factors like economy and environment, and the Machine Learning Models may be fine-tuned further for better performance.

short dash

# References

* + - 1. Duca, Angelica Lo. ‘Understanding the Prophet Plot’. Syntax-Error, 3 July 2022, <https://medium.com/syntaxerrorpub/understanding-the-prophet-plot-b5b6856371f4>.
      2. An Introduction to Seaborn — Seaborn 0.13.0 Documentation. https://seaborn.pydata.org/tutorial/introduction.html. Accessed 15 Oct. 2023.

short dash