Assignment 2  
ML as a Service

short line

The Hai Bui

Student ID: 24683423

Oct 4 2024

|  |  |
| --- | --- |
| Github Username | https://github.com/buithehaiuts  buithehaiuts |
| Github Repor | Project Repo: https://github.com/buithehaiuts/at2\_ml\_as\_a\_service\_experiments  API Repo: https://github.com/buithehaiuts/at2\_ml\_as\_a\_service\_api |
| API URL | API: https://at2-ml-as-a-service-api-update-8ac8.onrender.com/ |

36120 - Advanced Machine Learning Application

Master of Data Science and Innovation

University of Technology of Sydney

Table of Contents

[**1. Executive Summary 2**](#_xvtyf97iwmd0)

[**2. Business Understanding 3**](#_if9iwijz2pg3)

[a. Business Use Cases 3](#_bnjfuh5ewogw)

[**3. Data Understanding 4**](#_yg03y3w5hig9)

[**4. Data Preparation 5**](#_tj77rrmixzhi)

[**5. Modeling 6**](#_6vwh14nc44ga)

[a. Approach 1 6](#_uv8sn0ik6017)

[b. Approach 2 6](#_wtwxuxemqtvc)

[c. Approach 3 6](#_nrl8g3ri789x)

[**6. Evaluation 8**](#_mngiyx5tet6w)

[a. Evaluation Metrics 8](#_qx20m6hrk8uo)

[b. Results and Analysis 8](#_25v57zsj5m7g)

[c. Business Impact and Benefits 8](#_hkob76wu4d6q)

[d. Data Privacy and Ethical Concerns 9](#_uf1z6gbsejg6)

[**7. Deployment 10**](#_8rqux13i2o01)

[**8. Conclusion 11**](#_9hr11g79asdx)

[**9. References 12**](#_ma3y8ytjvp1y)

# 

# Executive Summary

This report will introduce an analysis and modelling project for an American Retailer with ten stores across California, Texas, and Wisconsin, focusing on predicting and forecasting sales revenues. The primary objectives are to develop two distinct models for predicting sales revenue and a time series forecasting model for overall sales revenue across all stores for the next seven days.

In this project, I will develop two models for predictive and forecasting purposes. The predictive model equips the retailer with the ability to forecast item-specific demand, while the time-series model offers a broader perspective on overall sales trends. Finally, a streamlit app will be utilized to present the final outcomes.

# Business Understanding

## Business Use Cases

Each store offers various items categorized into hobbies, foods, and household products. By inputting the item ID, store ID, and date into the model, managers can make informed decisions on inventory levels, pricing strategies, and promotions, ensuring that popular items are adequately stocked and minimizing overstock of less popular items.

Move over, by using the forecasting model, the manager can have a broaden view of the sales revenue across all stores and items in the coming days. As a result, it will enhance the decision-making process.

* 1. Key Objectives

The key objectives of this project are to develop machine learning models to predict daily sales revenue for various products across multiple stores, ensuring high levels of accuracy to support inventory management and financial planning. Moreover, the project will be deployed on Streamlit app to illustrate the model performance, as well as project presentation. The app will present model performance and results in a clear and accessible manner, enabling non-technical stakeholders, such as store managers and executives, to interact with the models and gain insights without the need for technical expertise.

# Data Understanding

The dataset utilized for this project is derived from multiple sources, encompassing various CSV files that facilitate a comprehensive analysis of retail sales. Calendar events dataset contains crucial information about significant calendar events, such as holidays and promotional activities, which may influence retail sales. Similarly, the calendar dataset provides a detailed timeline of dates, including additional attributes like day of the week, month, and year, essential for conducting time-series analysis.

The Selling prices dataset includes weekly selling prices for items are captured in the dataset. This information is vital for understanding pricing strategies and their subsequent impact on sales.

# Data Preparation

The data preparation for the analysis involved several crucial steps, including merging datasets, filling missing values, creating additional time-related features, and reshaping the data.

First, the **calendar** and **calendar\_events** datasets were merged on the date column to combine important calendar and event information. Missing values in the **event\_name** and **event\_type** columns were filled with **'no\_event'** to ensure no missing values would affect the model's performance.

A screenshot of a computer code

Description automatically generated

**event\_name and event\_type** columns enhance the dataset by providing contextual information about significant dates, such as holidays and promotional events. This can help the model learn patterns in sales revenue that correlate with specific events, thereby improving predictive accuracy.

Second, sales training and validation are transformed from a wide format to a long format using the pd.melt() function. After that, sales data will be merged with selling prices on the columns **"store\_id", "item\_id"** **and** **"wm\_yr\_wk”** to produce the final datasets (training and validation).

A new column called **“revenue”** was created to quantify the total income generated from sales. This was accomplished by multiplying the **selling price** of each item by the corresponding **sales** for that item on a specific date

A graph with a line going up

Description automatically generated

The above graph illustrates the total revenue grouped by date across months. The graph clearly illustrates that total revenue tends to peak at the beginning of the month, followed by a sharp decline towards the end of the month. This trend may indicate strong sales at the start, potentially driven by new inventory or promotional strategies, while the decrease at the month's end could suggest a need for inventory replenishment or other factors impacting customer demand.

A bar graph with different colored rectangles

Description automatically generated

The total revenue generated over the observed period was categorized into three distinct segments: **Foods**, **Hobbies**, and **Household**. This categorization allowed for a detailed analysis of revenue contributions from each segment.

Food emerged as the dominant revenue category, which contributes the largest share to overall sales. Following food, the Household category generated a significant portion of revenue, indicating a solid demand for household products. By contrast, the Hobbies category represented the smallest revenue segment among the three. This suggests that there may be room for future growth in this area

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | <class 'pandas.core.frame.DataFrame'> | | RangeIndex: 34720691 entries, 0 to 34720690 | | Data columns (total 19 columns): | | # Column Dtype | | 0 id object | | 1 item\_id object | | 2 dept\_id object | | 3 cat\_id object | | 4 store\_id object | | 5 state\_id object | | 6 d object | | 7 sales int64 | | 8 date datetime64[ns] | | 9 wm\_yr\_wk int64 | | 10 wday object | | 11 month int32 | | 12 year int32 | | 13 event\_name\_1 object | | 14 event\_type\_1 object | | 15 event\_name\_2 object | | 16 event\_type\_2 object | | 17 sell\_price float64 | | 18 revenue float64 | | dtypes: datetime64[ns](1), float64(2), int32(2), int64(2), object(12) |   The data includes various data types, including numerical and categorical variables. Column like event name and event type will be useful for forecasting model when it may reflect some seasonal patterns. |

# Modeling

## Predictive model:

A graph with a line graph

Description automatically generated with medium confidence

It is observed from the above graph that Revenue contains many outliers. To enhance the quality of the revenue data in the training dataset, two essential preprocessing techniques were implemented, namely **Winsorization** and **Log Transformation**. As a result, the revenue data now exhibits characteristics of a normal distribution, which is beneficial for many statistical modeling techniques.

A graph with a bar and a line

Description automatically generated

Categories column like item\_id, store\_id, state\_id will be encoded by using LabelEncoder. Below is the correlation matrix of columns in the training set

A screenshot of a graph

Description automatically generated

The train data will be split into training and validation sets using a 65:35 ratio. This division allows for effective model training and evaluation, ensuring that the model can generalize well to unseen data. Next, a regression model using LightGBM regressor will be performed with target variable as revenue.

## Forecasting models:

In this part, I will perform training sales forecasting models using time series analysis. Prophet is well known for its ability to handle various types of seasonal patterns and holiday effects, making it an excellent choice for modeling sales data that often exhibits these characteristics. Developed by Facebook, Prophet is designed to provide reliable forecasts even in the presence of missing data and outliers, allowing for robust predictions.

Prophet will automatically identify the trend and pattern in the seasonality, allowing it to adapt to different business cycles. The model also works well with noisy data, making it suitable for sales data that may experience irregular drops due to seasonal patterns or specific business activities.

By leveraging Prophet's strengths, I aim to develop a robust sales forecasting model that captures essential dynamics in the data, thereby enhancing the performance of traditional predictive machine learning models.

# Evaluation

## Evaluation Metrics

In this analysis, **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE)** were employed as evaluation metrics to assess the performance of the sales predicting and forecasting models.

MSE measures the average of the squares of the errors, provides a clear measure of how close the predictions are to the actual outcomes. This metric may be sensitive to outliers than MAE

MAE, on the other hand, calculates the average absolute differences between predicted and actual values. It treats all errors equally without giving extra weight to larger errors. Therefore, it is more straightforward and easier to communicate results to stakeholders.

## Results and Analysis

* + 1. Predictive model:

|  |  |  |
| --- | --- | --- |
| Metrics | MSE | MAE |
| Baseline model | 0.58 | 0.61 |
| Training | 0.5 | 0.57 |
| Validation | 300 | 8.8 |

The baseline model achieved an MSE of 0.58 and an MAE of 0.61. The training set demonstrated improved performance with an MSE of 0.50 and an MAE of 0.57, indicating that the model effectively learned patterns from the training data.

However, the validation set metrics drastically deteriorated, showing an MSE of 300 and an MAE of 8.8. This substantial drop suggests that the model is overfitting, meaning it has learned the training data too well, including its noise and outliers, without generalizing effectively to unseen data.

It may seem that the regression model does not capture the pattern from the sales pattern, which illustrates the complexity of sales. Each product category and region likely exhibit distinct sales patterns influenced by various factors such as seasonality, demand fluctuations, and regional preferences. This complexity can lead to a model that fails to generalize effectively across the diverse characteristics present in the data. This complexity can lead to a model that fails to generalize effectively across the diverse characteristics present in the data.

An alternative approach for the matter can be a forecasting model. Time series models like Prophet can help deal with patterns that vary across different products, categories, and regions, which will be introduced in the upcoming part.

* + 1. Forecasting model:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | MSE | MAE | R squared |
| Baseline |  | 834,003,433 | 22,668 | -0.97 |
| Prophet | Training | 74,285,063 | 6,202 | 0.80 |
|  | Testing | 162,281 591 | 9, 025 | 0.61 |
| Prophet with monthly seasonality |  |  |  |  |
|  | Training | 30,475,325 | 4,069 | 0.92 |
|  | Testing | 93,182,711 | 7,257 | 0.78 |
| Prophet with calendar event |  |  |  |  |
|  | Training | 48,386,936 | 5,554.8 | 0.87 |
|  | Testing | 106,991,998 | 8,435.1 | 0.76 |

The Prophet model shows a substantial improvement over the baseline in both training and testing datasets. The R-squared value of 0.80 during training indicates that the model explains a significant proportion of variance in the training data, while the testing R-squared value of 0.61 shows a reasonable fit on unseen data.

Incorporating calendar events (such as holidays, promotions, or special sales days) allows the model to account for atypical sales spikes or drops linked to specific dates. Moreover, by adding monthly seasonality, the prophet model becomes more adept at capturing fluctuations in sales that occur regularly within a year. The model's training performance (R-squared) improved to **0.92 (training set)** when calendar events were included, indicating its enhanced ability to explain variability in the sales data.

A graph of a graph

Description automatically generated with medium confidence

It is observable that sales exhibit a consistent upward trend throughout the month, indicating strong and growing demand for the products. The peak in sales at the end of the year may suggest a seasonal demand from customers. The holiday season, including major events like Christmas and New Year, often leads to increased consumer spending as people purchase gifts and prepare for celebrations

## Business Impact and Benefits

The implementation of the final Prophet forecasting model has had a significant impact on various business use cases. By effectively capturing trends, seasonality, and calendar effects, the model enhances the company's ability to predict sales accurately. This predictive capability allows for better inventory management, optimized resource allocation, and informed decision-making regarding marketing strategies

Prophet models provide accurate sale forecasts, which enables businesses to forecast the demand more effectively, thereby enhancing stocks management and preparation. Moreover, with a clear understanding of sales and seasonal pattern, marketing strategies can be more effectively aligned with customer behavior. A promotional campaign at the appropriate time can help business target the desired customers, thereby maximizing their impact and return on investment.

## Data Privacy and Ethical Concerns

The project involves the collection of sales data from various retailers, which may contain sensitive information, such as purchasing habits and demographic details. For example, the information from id or item\_id can be paired with specific products, allowing for tracking of individual customer behaviors. Therefore, it is important to clearly communicate to customers how their data will be used and obtain their consent prior to data collection. Moreover, the data should only be used for appropriate purposes and complied with applicable data protection laws.

# Deployment

Once the model is trained and validated, they will be saved in pickle format. The application will then be splitted into backend and frontend.

**7.1. Fast API Backend**

For the frontend application, the project will utilize the FasAPI for implementation.

A screenshot of a web page

Description automatically generated

The health endpoint will return a welcome message, which indicates that the API is functioning properly

A black rectangular object with white lines

Description automatically generated

/sales/stores/items/ (GET) will return predicted sales for a specific store and items based on an input date

A screenshot of a computer

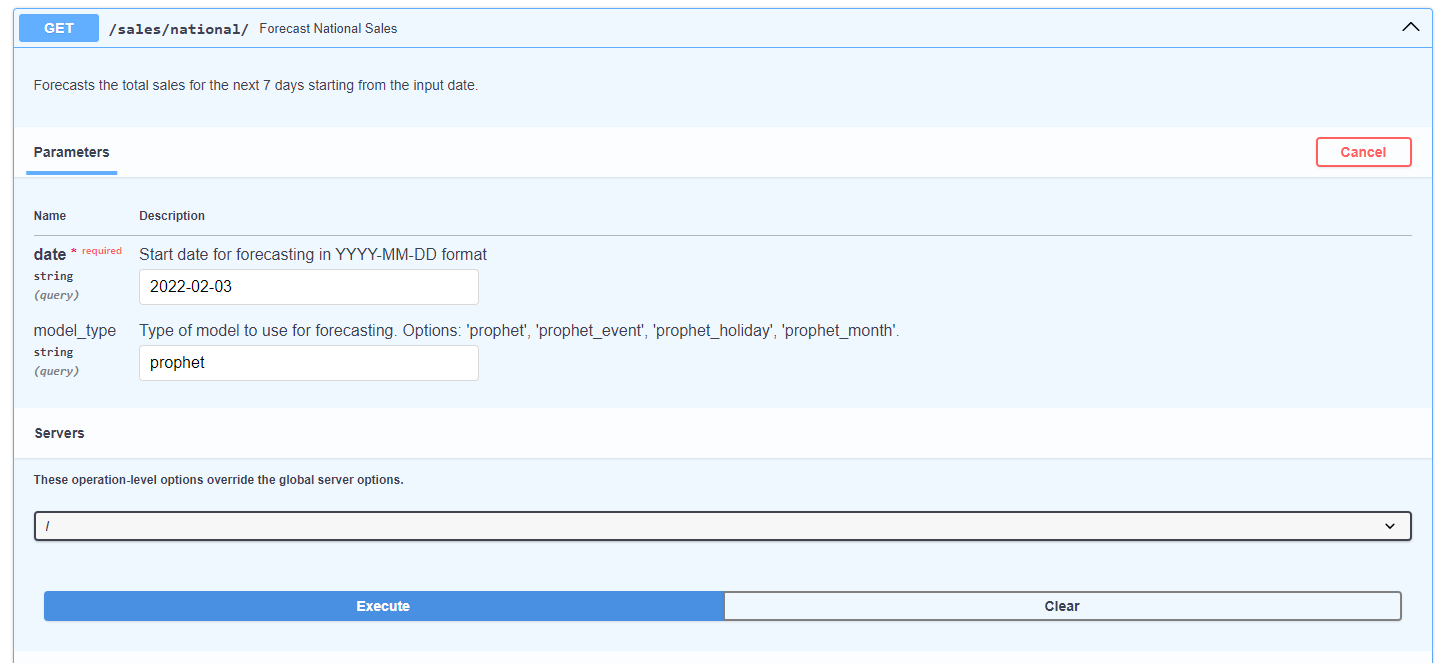
Description automatically generated

User will enter the date and another information about store\_id, item\_id, etc for sales prediction. The responses will be a specific value for the corresponding item

A black rectangular object with a white background

Description automatically generated

For sale forecasting, which will return total sales for the next 7 date, the user will enter the starting date and different models for forecasting, namely  'prophet', 'prophet\_event', 'prophet\_holiday', 'prophet\_month'



The results will be displayed in json format, which include the forecast value and its confidence interval for the next 7days

A black rectangular object with white border

Description automatically generated

**7.2. Streamlit frontend:**

After building API endpoint from FastAPI, a streamlit app will be deployed to present for user friendly interaction.

**7.2.1. API check health tab**

A screenshot of a computer

Description automatically generated

The API Health Check allows users to check the status of the API, confirming whether it is functioning properly. By pressing the “Check API Health Status”, it will display the health status, which confirm that it is functioning properly.

A screenshot of a web page

Description automatically generated

**7.2.2. Sales prediction:**

In sales prediction tab, it will be organized into two subtabs, namely Introduction and Prediction.

A screenshot of a computer

Description automatically generated

In the introduction Subtab, the streamlit app will briefly introduce the functionality of the prediction model and give detailed instruction for users. It also gives some general guidelines for interpretation of model performance and the visualization displayed.

A screenshot of a computer

Description automatically generated

The prediction subtab then displays an interface for user interaction as below.

A screenshot of a computer

Description automatically generated

It displays various options for user interaction. By simply selecting store ID, Item ID, State ID, Category ID and Department ID from the dropout list and entering the date for prediction, and then pressing the “Predict Sales”, button, it will generate the revenue for this item at a specific date.

The date option can be written in “YYYY/MM/DD” format or simply chosen from the calendar.

A screenshot of a computer

Description automatically generated

By simply pressing the “Predict Sales” button, it will automatically generate the predicted result:

A screenshot of a computer

Description automatically generated

7.2.3. Sales Forecasting Tab:

Similar to the structure of prediction tab, the Forecasting Tab has two subtabs, namely Introduction and Forecasting

A screenshot of a sales forecast app

Description automatically generated

The introduction subtab will introduce the overall function of the forecasting tab, which is illustrates as below:

A screenshot of a application

Description automatically generated

Users can choose a start date for the sales prediction using a date picker. This date indicates when the forecast period begins. Then, in a Forecasting Model Type, users select from a dropdown menu to determine which predictive model to apply for forecasting sales. The options available include prophet, prophet\_event, prophet\_holiday, prophet\_month (this function is also detailed on the introduction subtab).

**Prophet** is a powerful forecasting tool developed by Facebook, designed to handle missing data and adapt to shifts in sales trends, making it suitable for various business applications. The **Prophet with Events** model enhances accuracy by incorporating external events—such as marketing campaigns or product launches—that may significantly influence sales. Similarly, the **Prophet with Holidays** variant accounts for holidays that can disrupt normal sales patterns, ensuring that predictions reflect these seasonal fluctuations. Lastly, the **Prophet with Monthly Seasonality** model is tailored to capture monthly trends in sales data, allowing businesses to understand and anticipate sales variations throughout the year. Together, these models provide comprehensive forecasting capabilities that address multiple factors affecting sales performance.

A screenshot of a graph

Description automatically generated



The results will be presented in a prediction table that includes the forecasted sales values along with their corresponding confidence intervals. This table will clearly outline the predicted sales for each time point, highlighting the expected value and the range of uncertainty. Additionally, an illustrated graph will accompany the table, visually representing the forecasted sales and the confidence intervals. User can point to a specific date in the graph to display the corresponding value. This dual presentation allows users to easily interpret the predicted sales trends and assess the reliability of the forecasts

Finally, the about tab provides a comprehensive summary of the project, including key information about its purpose and functionality. It also includes a link to the GitHub repository, allowing users to access the project's source code and documentation. Additionally, the tab features the author's email address for contact, enabling users to reach out for inquiries, feedback, or further information regarding the application.

A screenshot of a computer application

Description automatically generated

# Conclusion

In conclusion, two distinct machine learning models were successfully developed: a predictive model using **LightGBM** for item-specific sales and a time series forecasting model using **Prophet** for overall sales trends. The predictive model, while insightful, faced challenges with overfitting, as evidenced by significant discrepancies between training and validation metrics. The alternative approach of time series models like Prophet successfully addressed this issue by introducing seasonal trends and holiday effects. The superior performance of forecasting model, with a validation set MSE of 7.67 and a **Mean Absolute Error (MAE)** of 2.54, emphasizes the importance of seasonal factor of times series.

By accurately predicting demand and providing forecasts, the models enabled informed decision-making regarding inventory management and pricing strategies. The predictive insights can help reduce overstock and stockouts, ultimately leading to enhanced customer satisfaction and improved profitability.

Moreover, the deployment of a **Streamlit app** allowed non-technical stakeholders to interact with models, specific items and store ID for real time predictions. This interactivity provided stakeholders with valuable insights without required technical knowledge, thereby improving model interpretation and communication across various teams in an organization.

# References

* 1. *Tutorial - User Guide - FastAPI*. (n.d.). Fastapi.tiangolo.com. <https://fastapi.tiangolo.com/tutorial/>
  2. *Create an app - Streamlit Docs*. (2024). Streamlit.io. [https://docs.streamlit.io/get-started/tutorials/create-an-app ‌](https://docs.streamlit.io/get-started/tutorials/create-an-app%20‌)