Assignment 2

Machine Learning as a Service

SimonUTS24661225/ADV_MLA_AT22 (github.com)

Simon Lim
StudentID: 24661225

<10/10/2023>

36120 - Advanced Machine Learning Application
Master of Data Science and Innovation
University of Technology of Sydney

Table of Contents

1. Executive Summary

An American retailer that has 10 stores across 3 different states (California (CA), Texas (TX), Wisconsin (WI)) sells a variety of items from 3 different categories, including hobbies, foods and household. In this regard, they have requested two different trained models that can be further used as a real-world application for business purposes. One model involves a predictive model using a Machine Learning algorithm to accurately predict the sales revenue for a given item in a specific store at a given date. The other model they requested is a forecasting model using a time-series analysis algorithm that can forecast the total sales revenue across all stores and items for the next 7 days. There are five datasets given at the beginning, including training dataset, testing dataset, calendar, calendar event and selling price per week. Those five datasets have been merged on common columns and a process of feature engineering is performed and result in final training dataset and testing dataset. The performance of models is assessed based on RMSE scores on testing dataset, measuring the average difference between predicted values and actual values. This project conducts two different experiments with two different algorithms, including one for predictive model and the other for forecasting model. XGBoost Regressor algorithm is used for a predictive model and Prophet algorithm is used for a forecasting model. Prior to training models, the process of data preparation, cleaning, feature engineering, EDA, data transforming, and data pipeline have been performed. Finally, successful two models with strong performance are deployed into productions using deployment techniques and tools, such as Fastapi, Docker and Heroku and allow the retailer to obtain predicted sales revenue of items.

GitHub: SimonUTS24661225/ADV_MLA_AT22 (github.com)

Heroku:

FastAPI - Swagger UI (calm-mountain-59073-575f6c38303e.herokuapp.com)

calm-mountain-59073-575f6c38303e.herokuapp.com/health/

calm-mountain-59073-575f6c38303e.herokuapp.com/sales/stores/items/

2. Business Understanding

a. Business Use Cases

The American retailer that has 10 stores across 3 different states (California (CA), Texas (TX), Wisconsin (WI)) sells a variety of items from 3 different categories, including hobbies, foods and household. The retailer has a large of amount of data for each item's selling price, sales revenue and quantity of sales records across stores and dates. In this regard, machine learning and time-series models can be potentially helpful in a variety of ways by predicting the sales revenue for a specific item and forecasting the total revenues for next few days. For example, the retailer can optimize inventory management by forecasting the demand for specific items in each store. Additionally, the retailer can implement a pricing strategy based on given predicted sales revenue. Finally, analyzing the predicted sales revenue across different states and stores allow the retailer to identify potential opportunities for market extension.

b. Key Objectives

The two goal of the project is to build potential predictive models using a Machine Learning algorithm to accurately predict the sales revenue for a given item in a specific store at a given date and build a forecasting model using a time-series analysis algorithm to forecast the total sales revenue across all stores and items for the next 7 days. Once two models are successfully established, the next step of goal is to deploy two models into productions and allow access to online so the retailers can use models and predict the sales revenue. The results of predictive model can help the retailers in finding specific items that have high sales revenues over times and the results of forecasting model can help the retailers to identify the total sales revenue for next 7 days.

3. Data Understanding

There are five datasets given at the beginning, including training dataset, testing dataset, calendar, calendar event and selling price per week. Those five datasets have been merged on common columns and a process of feature engineering is performed and result in final training dataset and testing dataset.

Training dataset and testing dataset included 1947 columns, including item ids (i.e., item_id, dept_id, store_id, cat_id and state_id) and the sales quantity in each day with a format "d_". The columns of "d_" will need to be converted into rows, in order to merge with other datasets (i.e., calendar and selling_price_per_week). The sales revenue is a target variable but has not been specified in given datasets. Therefore, the sales revenue column will need to be created in feature engineering part after merging all datasets. It is assumed that many valuable features can be created with many existing columns in datasets. For example, 'date' column can create year and month feature and 'event_name' in calendar can create a new feature regarding the influence of events on sales revenue.

Most importantly, when merging datasets, identifying common columns between datasets would be crucial to smoothly merge datasets and not to lose any data in merging process.

4. Data Preparation/Feature Engineering

There is one each experiment for predictive model and forecasting model and they have different steps of data preparation.

Predictive Model Experiment

1. There were five separate datasets, including training dataset, testing dataset, selling price per week, calendar and calendar of events. The first stage in data preparation was to merge those datasets. Training and testing datasets had many columns representing each day's sales. I switched those columns into rows using "pd.melt()".

2. Calendar and calendar of events were merged on a common column "**date**" and then this combined calendar dataset was merged with training and testing dataset on a common column "**d**" (day).

3. Finally, combined training and testing datasets were merged with selling price per week dataset on common columns **"store_id**" and **"item_id".**

4. There were some null values in **'sell_price', 'event_name'** and '**event_type'** columns. I converted null values to 0, as all values in those columns will be used for feature engineering.

5. Explatory Data Analysis was performed to investigate trends in data.

6.The sales revenue (**'revenues'**) column was created by multiplying '**sales'** by **'sell_price'.** This column will be used as a target variable when training models.

7. The event day **("is_event")** column was created by using "**event_name**" column. If there was any event on a row of data, "**is_event**" was 1 and if there is not event, "**is_event**" was 0.

8.The day of week **("day_of_week")** column was created from Monday to Sunday by using "date" column.

9.The **"year"**, **"month"** and **"day"**, in numeric format, were created by using datetime.

Forecasting Model Experiment

1. There were five separate datasets, including training dataset, testing dataset, selling price per week, calendar and calendar of events. The first stage in data preparation was to merge those datasets. Training and testing datasets were merged first and columns of each day's sales were switched into rows using "**pd.melt()**".

2. Calendar and calendar of events were merged on "**date**" and then this combined calendar dataset was merged with combined training and testing dataset on common column "**d**" (day).

3.Finally, combined training and testing datasets were merged with selling price per week dataset

on common columns, "**store_id**" and "**item_id"**.

4.There were some null values in **'event_name'** columns. I converted null values to 'No Event'. The column '**event_name'** will be used as an external feature in training Prophet forecasting model.
5.The sales revenue (**'revenues'**) column was created by multiplying '**sales'** by **'sell_price'.** This column will be used as a target variable when training models.

6.**'Event_name'** column was used as an external feature, by filling null values with 'No Event'.

7.The sales revenue for each item from different store and date was summed based on the **'date'** column.

6. Modeling

a. Predictive Model Approach

Models and its Rationale

In predictive model experiment, two kinds of XGBoost Regressor models were employed, one with default model and the other with XGBoost Regressor with tuned hyperparameters. The rationale behind choosing XGBoost Regressor is that given that the dataset is large and contains a variety of items across state, store and category, XGBoost Regressor can efficiently handle large datasets. Furthermore, XGBoost is one of the optimal machine learning algorithms, which shows very strong predictive performance. As the model will be deployed for real-world application, the performance of the model is also crucial.

Preprocessing

- Categorical values were converted to numeric values using "OrdinalEncoder", including "store_id", "cat_id", "day_of_week", "dept_id", and "state_id".
- All numeric columns ('sales', 'wm_yr_wk', 'sell_price', 'is_event', 'year', 'month', 'day') were normalized, using "StandardScaler".
- Finally, "ColumnTransformer" technique was used to assemble all transformed data into a preprocessor.
- Data Pipeline was a main preprocessing technique used to transform categorical columns to ordinal numeric columns, scale numeric columns and train XGBoost Regressor algorithm with pre-processed data.

```
preprocessor = ColumnTransformer(
    transformers=[
        ('num_cols', num_transformer, num_cols),
        ('store_id_col', store_id_transformer, ['store_id']),
        ('dept_id_col', depart_id_transformer, ['dept_id']),
        ('cat_id_col', cat_id_transformer, ['cat_id']),
        ('day_col', day_of_week_transformer, ['day_of_week']),
        ('state_id_col', state_id_transformer, ['state_id'])
    ]
)
```

Figure 1. Preprocessing of data columns and assembling by using ColumnTransformer

b. Forecasting Model Approach

In the forecasting model experiment, Prophet is a time-series algorithm that was used in the experiment. Prophet is highly flexible and can handle various time series, including seasonality and special events effects. Also, the Prophet model is simple and straightforward model, especially specialized in deploying forecasting models. Prophet also takes an external feature, such as event or holiday into account. There was an event dataset in this project and hence would be more suitable model if events are included. Only one Prophet model was used to forecast the total sales revenue for the next 7 days.

- U.S country holidays were added into Prophet model using Prophet technique ("prophet.add_country_holidays(country_name = 'US')").
- Prophet model was trained on data from 2011-01-29 to 2016-05-29 and predicted the next 7days and the trends of the total sales revenue.

	ds	y	event_name
0	2011-01-29	81650.61	No Event
1	2011-01-30	78970.57	No Event
2	2011-01-31	57706.91	No Event
3	2011-02-01	60761.20	No Event
4	2011-02-02	46959.95	No Event
...	...	...	...
1940	2016-05-18	116333.11	No Event
1941	2016-05-19	117456.32	No Event
1942	2016-05-20	134000.67	No Event
1943	2016-05-21	162395.65	No Event
1944	2016-05-22	175297.42	No Event

1945 rows × 3 columns

Figure 2. The table of the target variable and features used in Prophet model

7. Evaluation

a. Evaluation Metrics

For performance metrics, four different performance metrics were tested, including RMSE, MAE, MSE and R2.

Predictive Model Experiment.

RMSE is mainly used to assess and compare the performance of trained models. The purpose of the project is to establish a predictive model that can accurately predict the sales revenue. RMSE is sensitive to the difference between predicted and actual values. Also, it is sensitive to large errors and given that the trained model will be used in real-world applications, RMSE is a suitable performance metric in this project.

Forecasting Model Experiment.

Mean Absolute Error (MAE) is mainly used to assess the performance of trained models. Given that the purpose of the experiment is to build a forecasting model that can forecast the total sales revenue for next 7 days, MAE measures the average magnitude of errors in a set of predictions and provides a balanced view of the overall model performance. Therefore, MAE is more suitable as a performance metric for forecasting trends of the predictive total sales revenue.

b. Results and Analysis

Predictive Model Experiment.

As hypothesized, XGBoost Regressor models have shown great performance in predicting the sales revenue of items.

RMSE score of default XGBoost model was 1.58, while RMSE score of tuned XGBoost model was 1.53. XGBoost Regressor model with tuned hyperparameter showed slightly stronger performance than default XGBoost Regressor model.

There was a lack of searching optimal hyperparameters via 'RandomSearchCV' due to a huge size of dataset. Nevertheless, the performance of XGBoost Regressor models were outstanding in predicting the sales revenue and thus can be deployed for real-world application uses.

```
{'RMSE': 1.5266779569705644,
 'MSE': 2.3307455842998164,
 'R2': 0.98174526490954,
 'MAE': 0.16584185171017993}
```

Figure 3. Regression Performance Metric for XGBoost with tuned hyperparameters

	Actual Value (y_test)	Predicted Value (XGBOOST_tuned)	Difference (Best Features)
0	0.00	0.072191	0.072191
1	0.00	-0.073116	0.073116
2	0.00	0.022140	0.022140
3	18.56	17.947886	0.612114
4	8.64	9.025688	0.385688
...	...	...	...
12195995	2.98	3.026613	0.046613
12195996	0.00	-0.006555	0.006555
12195997	7.96	8.013767	0.053767
12195998	0.00	-0.105073	0.105073
12195999	1.00	1.105286	0.105286

12196000 rows × 3 columns

Figure 4. The table of the difference between actual values and predicted values

Forecasting Model Experiment.

MAE score of Prophet model obtained was 19057.38. The trends of total sales revenue tend to increase as dates go by. RMAE and MAE scores were high, indicating that the model is not optimal in terms of performance. However, the scale of predicted values and actual values was very high, the MAE score of 19057.38 was reasonable. The model successfully forecasted the total sales revenue for the next 7 days (e.g., from 2016-05-23 to 2016-05-29).

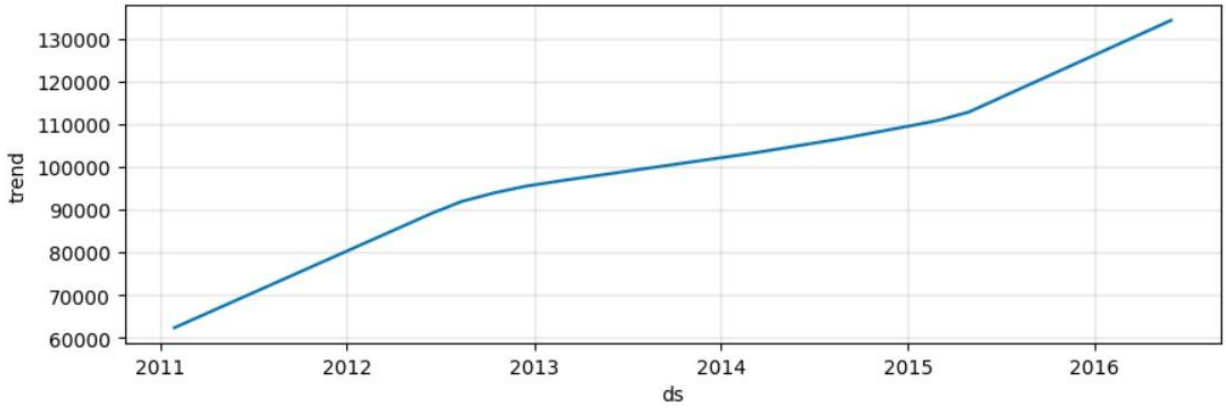

Figure 5. The trend of the total sales revenue across years

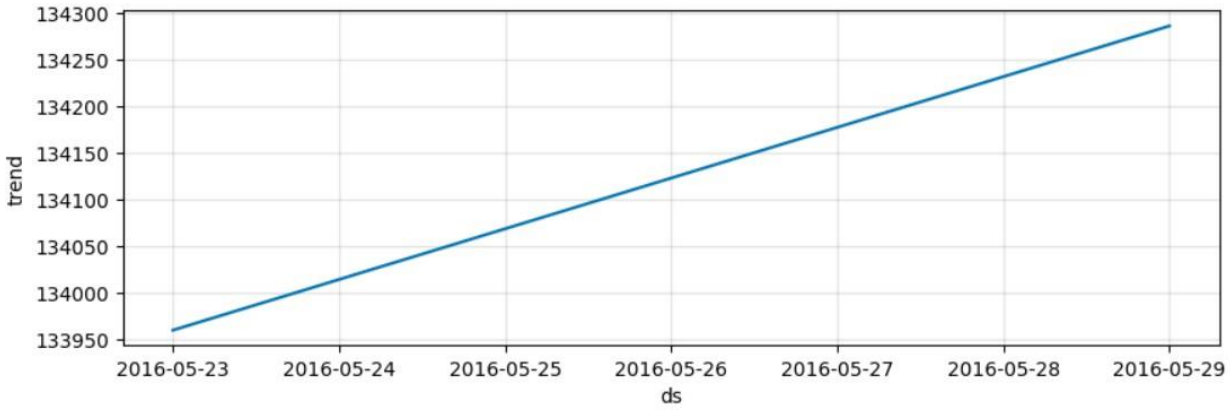

Figure 6. The trend of the total sales revenue for the next 7 days

c. Business Impact and Benefits

In consequence, both predictive and forecasting models obtained reasonable performance scores, indicating that they can correctly predict and forecast required sales revenue. It also means that models can be deployed into productions and the retailers can use models for their desired business purposes.

d. Data Privacy and Ethical Concerns

In terms of data privacy implications of the project, it is important to ensure that information, such as weekly selling price and sales revenue, should be protected and not shared personally. Furthermore, it is also crucial to maintain transparency in the process of building models. Also, when models are used by stakeholders in the future, they should make informed decisions-making based on the predictions of models. In terms of data security, collected and stored data should be protected, and appropriate security measure should be conducted to prevent data breaches and unauthorized access.

8. Deployment

- Github was a great environment for access to users, to save codes, models and other relevant data and keep tracks of models, monitoring and handling models. Also, by setting github private, it also protects both the model and data.
- Also, models' artefacts were saved and kept in separate file (using Joblib) to reload and use the models directly when needed.
- Fastapi was useful in making trained models into productions. It also integrates with machine learning libraries, which simplify the deployment of models and applications.
- Docker works as a bridge between Fastapi and Heroku, as Docker containers are portable and can be deployed across different environments.
- Heroku is a useful machine learning tool that allows users and customers to use trained models online any time.

Heroku Link:

FastAPI - Swagger UI (calm-mountain-59073-575f6c38303e.herokuapp.com) – Production

calm-mountain-59073-575f6c38303e.herokuapp.com -- description

calm-mountain-59073-575f6c38303e.herokuapp.com/health/

calm-mountain-59073-575f6c38303e.herokuapp.com/sales/stores/items/ -- Predictive Model

calm-mountain-59073-575f6c38303e.herokuapp.com/sales/national/ -- Forecasting Model

9. Conclusion

In conclusion, the project aimed at predicting the sales revenue for a given item at a given date and forecasting the total sales revenue for the next 7 days. One predictive model experiment and one forecasting model experiment have been conducted with optimal models and two kinds of models successfully performed well and achieved the objective of the project. The predictive model can be used to predict the sales revenue of given items with specific date and store. Also, the forecasting model can be used to forecast the total sales revenue across all stores for the next 7 days. Finally, by deploying the trained models into production via Heroku, the models can be used in real-life application for stakeholders to plan strategies based on predicted sales revenue.

Some important key learnings and steps that can be considered for future projects.

- Explanatory Data Analysis (EDA) is an essential stage prior to training models, to investigate and obtain particular trends of data.

- Feature engineering is a crucial process in machine learning, which can potentially increase the performance of models.

- It is also crucial to find the optimal hyperparameters of the model for better results.

- If conducting a few different experiments, using functions and methods (i.e., custom packages) can be useful to simplify codes and identify patterns.

- Consider subsampling if the dataset is extremely large.

- Selection of algorithm for training model is also important depending on the context of dataset.

- Deploying trained models, using tools such as Fastapi and Heroku for users to access the models

- Deciding main performance metric in advance is very helpful in building optimal models.

10. References

Gutiérrez, J. (2022, March 24). *How to deploy FastAPI on Heroku using Github.* Medium. https://medium.com/@julgq/how-to-deploy-fastapi-on-heroku-using-github-4363d9ba3d41

Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). (n.d.). Resources.eumetrain.org. https://resources.eumetrain.org/data/4/451/english/msg/ver_cont_var/uos3/uos3_ko1.htm#:~:text=The%20MAE%20is%20a%20linear

Okada, S. (2022, October 31). *How to Deploy Your FastAPI App on Heroku for Free*. Medium. https://towardsdatascience.com/how-to-deploy-your-fastapi-app-on-heroku-for-free-8d4271a4ab9

Note: The CRISP-DM steps (Cross-Industry Standard Process for Data Mining) provide a framework for structuring the project report, but feel free to adapt the template to match the specific requirements and guidelines of your project or organization.