Project Report

COSC2789 - Practical Data Science

03/01/2020

RMIT University Vietnam

Assignment Cover Page

Course Code	COSC2789
Course Name	Practical Data Science
Location/ Campus	RMIT Vietnam Saigon South
Assignment	Group Project
Student names	• Nguyen Dang Huynh Chau (s3777214) • Tran Ngoc Anh Thu (s3879312) • Ho Le Minh Thach (s3877980)
Lectures	• Vo Ngoc Yen Nhi
Group	10
Assignment due date	Jan 14th, 2022
Date of Submission	Jan 14th, 2022
Number of pages including cover pages	33

Table of content

Table of Figures

ABSTRACT

According to Gottlieb in 1976, the relationship between the real estate prices and general economic condition, that linkage have an extensive history which has great influences on the long term of the construction, price growth and economic activity [1]. It is well-established that the sale price fluctuates frequently and is not straightforwardly to be estimated, however, that criteria have a great influence on either the benefits of both the sellers and the clientele.

The objective of this study is to determine the trend of the price in order to have anticipations as well as build a model for predicting the trend of the sale price for housing trading based on different explanatory variables describing aspects of residential houses.

The first part of this report concentrates on the hypothesis formation which is the influences of the number of rooms, floors, length, width, area, location, and time in both urban and suburban on sale price in Hanoi 2020 to understand and predict the trend of the price range. The dataset is retrieved from the sale price of houses in Hanoi from 2019 to 2020 in Kaggle, however, there are only a few data points in 2019 so currently, the dataset has only the price in 2020. These hypothesize is explored and proved by statistic and data exploration

The focus of the second part is on the model training process, which is about the feature engineering, model training, in which the Logistic Regression as a baseline, Random Forest and Random Forest with GridSearchCV are applied, model evaluation, hyperparameter tuning – GridSearchCV, model validation and model persistent.

The results are demonstrated and deployed in Dash and Flask. In the Dash plotty, the results of exploration and the outcome of model training is visualized. In Flask, the function of the application is to predict the sale price in Hanoi city.

INTRODUCTION

The rise of sale prices of houses has been accompanied by a sharp increase in the intention of the stakeholders in the real estate. This correlation has been studied in a remarkable number of research and projects such as Zillow website, many competitions of Kaggle on House Prices to accumulate enough information to predict the house price range. To demonstrate this in the report "Getting Ahead of the market: How big data is transforming real estates" of McKinsey & Company in 2018, the location has affected the sale price [2], however, either factors that have influenced the sale price of Hanoi in 2020 or the model for predicting sale price range of house in Hanoi were not studied and developed properly. Hence, the main target of this project is to research more on that side.

PROJECT MANAGEMENT

1. Project Goal:

The main project goal is to explore the influence of the factors such as the number of rooms, floors, length, width, area, location and time on sale price. Afterwards, the dataset is utilised for the training model to predict the sale price range of houses in Ha Noi.

2. Deliverables:

Realizing the importance of the sale price in real estate, the final product of this project aims to give the clientele not only the more interesting visualization but also the trained model to estimate the highest accuracy. The visualization is built by the Dash plotty, it illustrates the price range and its dependences on many criteria including the number of rooms, floors, length, width, area and location. Moreover, the visualization is not only extremely interactive but also user-friendly, so that the customers can have more options to visualize these plots uniquely and gain more knowledge on the real estate domain. For the predicting model, it is exhibited in the web application for the clientele to select which features they desire to predict. Additionally, this model is relatively adaptive since these criteria are the fundamental for listing a price of a residential house.

3. Program Stack:

To solve the problem, Jupyter Notebook 6.4.5 is applied to process the cleaning, exploration and building model. The notebook is stand out among other software tools for its ability to organize itself into a sequence of cells that can either be the text, graph, or code. Hence, our process is clarified more straightforwardly, and it will be user-friendly to read through the report and understand what we are trying to do in each process as well as the result after we run a cell of code. Furthermore, with the python version 3.8, we include library pandas version 1.1.3, NumPy version 1.18.5 to help us analyze the data better. Thanks to pandas, the table is imported and exported probably under a well-structured data frame. Pandas allow us to do cleaning, visualization step easier with their build-in function. It is also easier to manipulate the table by a data frame structure, which is more advantageous than the build-in from python alone or any basic data structure. Similarly, NumPy allows us to process data faster or prepare for one dimension array.

♦ Using IDE for writing Python Scripts and Web Applications

We use PyCharm as the main IDE to write python scripts both for the dash and model deployment Streamlit. Its advantage is the words suggestion, and format the code in the IPEP standard.

♦ Using Streamlit for Model Deployment

The deployment model needs to serve predictions behind an HTTP endpoint and set up hosting web frameworks, serverless compute, or cloud platform-specific frameworks. Many frameworks are available to build the web app for hosting a web framework to set up a server that serves predictions like Streamlit.

A website calls the API - so we would have a form on the website where the user entered the expected input data posted to our API and would return the predictions that would then display to the users. The incoming HTTP request might first hit an Nginx server that serves as a reverse proxy. It often sits behind a firewall and directs requests to the appropriate back end. Nginx is a popular open-source software for web serving, reverse proxying, caching, load balancing. Nginx might run on top of Gunicorn, or green unicorn, a popular web server for UNIX-based systems. Gunicorn is an HTTP server based on the WSGI standard, Web Server Gateway Interface, a Python standard that determines how a web server communicates with applications. It is simple, lightweight, fast, and works with many web frameworks, including Flask. When deploying a machine learning model to Flask, the Flask application will deserialize the model parameters, perform predictions, and respond to the end-user.

♦ Using Dash plotty for visualization application:

For the visualization, we use Dash version 1.13.1 and Plotly version 5.5.0 to create a locally hosted website. Dash is a library that we need to include in the python script so that we can create an interactive website. It also combines with the Plotly library to create a visualization for the user to interact.

4. Timeline and member contribution:

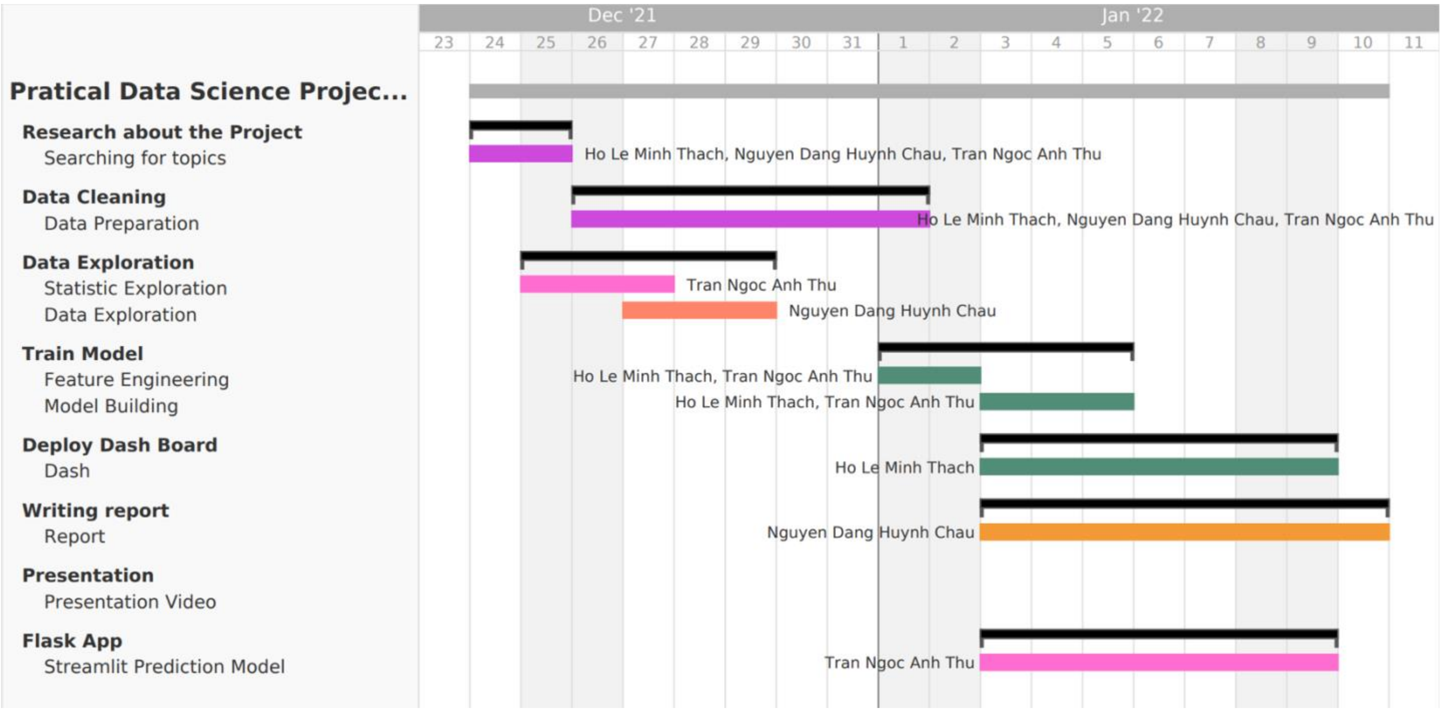


***Figure 1**: Timeline and member contribution*

METHODOLOGY

The main steps in this project are Data Preparation, Exploratory Data Analysis (EDA), Feature Selection, Modeling, and Determine the reasons for misclassification in model.

1. Data Preparation:

To have the highest accurate data, the data preparation step must be done properly. In this process, the data is cleaned with the rename column, changing data types, translation step, make the data homogenous, filling missing value, shorten and simplify categories, whitespaces checks, typos checks, sanity checks, encoding.

a) Rename column:

Since these columns are in Vietnamese so translate into English will be more straightforward in later process. We will also analyse the column meaning:

- The "Ngày" will be the date which will contain all the day, month, year value.
- The “Địa chỉ” will be the address which contains the street, ward, district, city
- The “Quận” is district.
- The “Huyện” is not suppose to be the ward. However, the content they have is actually the ward.
- The “Loại hình nhà ở” is a type of house which depends on the location such as villa, house in the main street, house in the alley, or townhouse.
- The “Giấy tờ pháp lý” is the legal document either already have, or on waiting, or others documents.
- The “Số tầng” is the number of floor.
- The “Số phòng ngủ” is the number of bedroom.
- The “Diện tích” is the area unit in meter square.
- The “Dài” is the length unit in meter.
- The “Rộng” is the width unit in meter.
- The “Giá/m2” is the price in million VND per meter square.

b) Changing the data types:

As we analyzed the Jupyter notebook, we can see that area, length, width, and price columns should be a continuous numerical value. Thus, we need to remove its unit (post-fix) and format it to the correct type by convert the value in those columns into float value. Similarly, with the column district and ward we will remove the prefix because we want to extract only the name of the district and ward. Number of floors and number of bedrooms will also be removed the prefix to extract the categorical value of room or floor from 1 to greater than 10 room or floor.

c) Translating values into English:

After all the columns are in the correct datatype, we will then translate from Vietnamese to English for categorical column that including housing type, legal document, number of rooms, housing type, legal document, and number of floors.

d) Make the data homogenous:

Now the data need to be homogenous by capitalize all the categorical value which is string value so that it will be easier to process the string including any kind of string regex, or replacement. It is also consistence for further encoding process.

e) Filling missing data:

After the data is homogenous, we will do the filling missing value. There are a few approaches that need to be considered for each case including:

- Guess the missingness type
- Drop missing value by deleting rows
- Drop missing value by deleting columns
- Fill the missing value by mean, mode, or median

For the column legal document, we will fill with others because it is safe to assume if a house's legal documents are not listed, it is not either available or waiting. Similarly, we also find out that there are a lot of missing wards. Since the data that has the missing value contains only the street info, we can assume that house is point to a wide range of location. Hence, there is none particular ward because a street can have multiple wards if the street extends across region. Thus, we introduce none to fill the missing value. After filling all the missing value, we do the cleaning extra whitespaces for all the string columns.

After that, we now process the address column where we will eliminate information of the city Hanoi, Ward, and District because it is already presented in the other specify column. We extract the remaining which is the street number and street name into new column street. Now we can drop the column ‘Address’ because all the information is already present in Street, Ward, and District column. With eliminate rows, we noted that the ‘Year’ value is contains around 18-20 rows of 2019 value which is neglectable if compare with the entire data. Hence, to narrow the scope of the project, we will eliminate the row that contains 2019 so that this data will be house price in 2020 only.

f) Sanity check

For the section of interpolation, we have column number of floors, housing type. With the number of floors, we can say that most of the house in Vietnam has 5 floors which is the mode of this column. Thus, we can assume the missing value will be the most popular option in Vietnam which is house with 5 floors. Additionally, we also uniform our result as there is an option call greater than 10. Thus, we will ensure that any house with a floor greater than 10 will be classify as greater than 10 value. Similarly reasoning with filling the missing value for number of floors, we can fill the missing value for number of bedrooms by mode because we can assume that the most popular option in Vietnam is a house with 4 bedrooms which is the mode. For the housing type, we will fill with mode because the missing value is neglectable for the dataset size. With the numerical column such as area, price, length, width, we will fill the missing value with the median by grouping each district because they are heavily influence by the region’s policy and culture, history.

After filling all the missing value, we do the cleaning extra whitespaces for all the string columns. We will strip the left and right if there is any for a string. Then we will do the sanity checks if there is any impossible values and outlier. We can see that the area, length, and width contain great outlier. Thus, we are using boxplot to interpolate those outlier value with mean so that the data for these columns is not too skew to the left or right. Additionally, we also do an impossible check on area where it needs to be less than or equal to the length * width. Additionally, we know that for a particular house in Hanoi to sell, it needs to have length and width not less than 2.4 [4]. Thus, we will interpolate the value less than 2.4 and set it to 2.4 if there is any. We also check for the area that is lower than 30 meter square because government is not allow to sold a house with area that is lower than 30 meter square [4]. We also assume if there is any house that has an area less than 30-meter square, it needs to come with the document available which means at least the government is confirming the house is legal. Thus, we will fill the area value to 30-meter square if there is any house with the area less than that. Lastly, we will create a column price range which is our target column. This column will help us to classify houses in certain ranging for our model prediction.

2. Exploratory Data Analysis:

a) Descriptive statistics for variability:

In order to have a better visualisation, the descriptive statistics for variability is essential since it is a brief coefficients and descriptive summary of the dataset. First of all, calculate the price mean, median, and mode, the result are 102.45500197250817, 90.0, 100.0 respectively. Moreover, the following plot is the histogram plot with mean, median, and mode included demonstrates the mean is larger than the median indicates that the data is skewed to the right. (*)

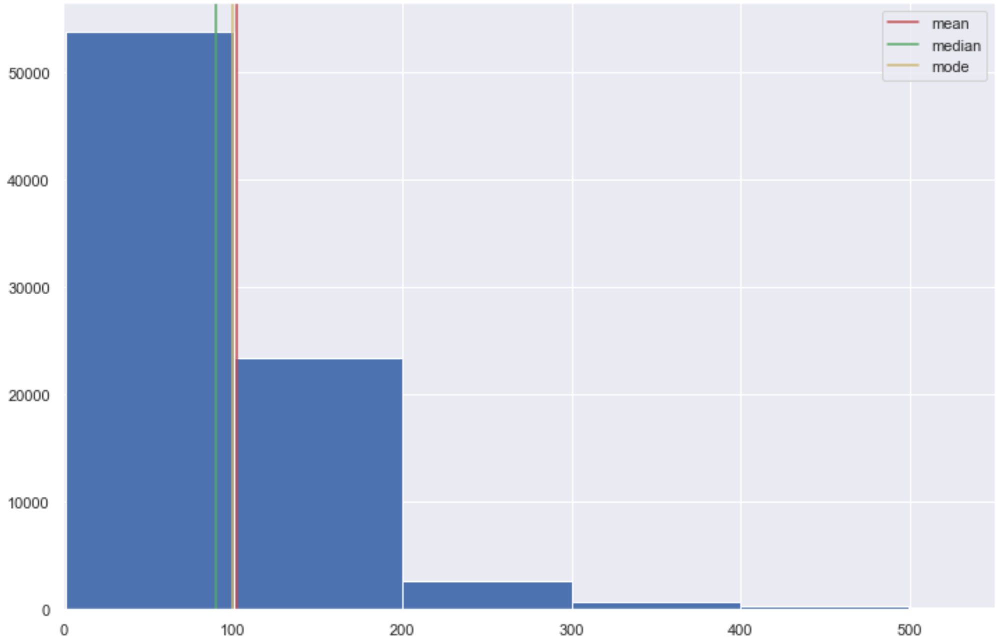


Figure 2: *Descriptive statistics for variability*

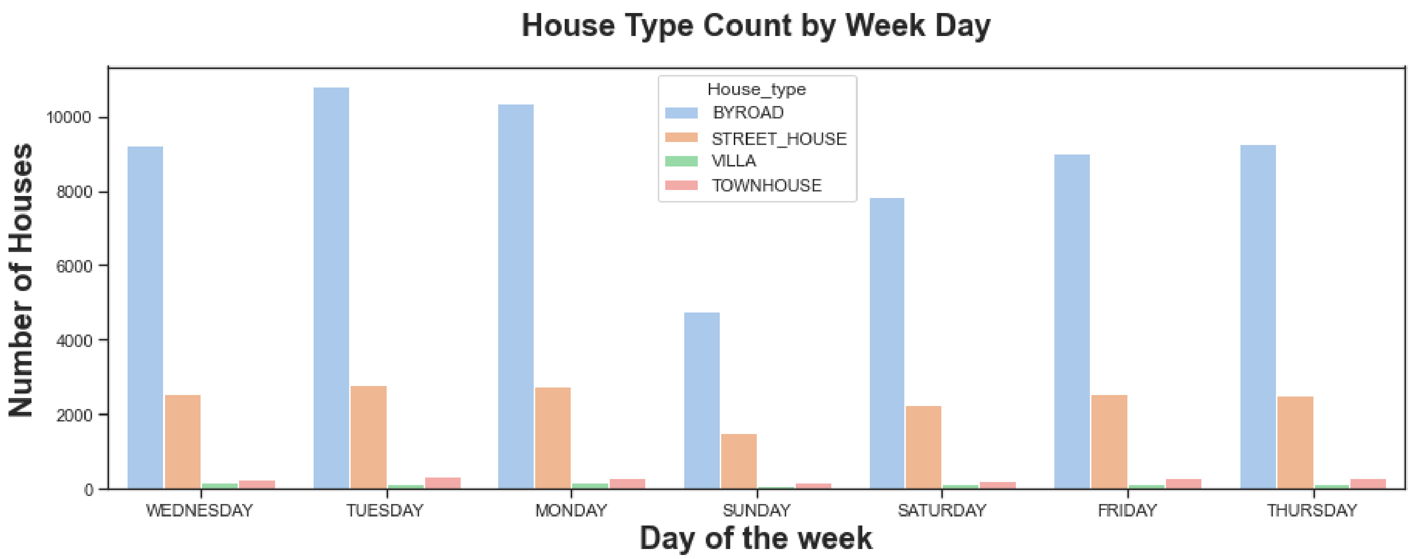
b) State Hypothesis:

In data exploration, the relationship of the target variable with other features. First of all, the hypothesis need to be stated.

- Are house sold more in the week day than in the weekend?
- Is houses price on the weekend is lower than in the week day?
- Are houses sold more in the end of the year?
- Is the houses price higher than in the end of the year?
- Do the plots have higher price have Available Legal Documents, the number of floors higher than 2, the number of bedrooms higher than 2, the Area is higher than 50

metersquare, the house type is Street House or Villa, Locate in the center of Hanoi or urban city?

c) Data Exploration:



<u>Figure 3:</u> *House Transaction in a week*

⇒ **<u>Observation:</u>**

The “House Type Count by Weekday” plot illustrates the houses transaction in a week. The reason that this bar plot is chosen in order to compare the magnitude of the number of house transaction in a week. The plot can be divided into two main parts including the higher part which is from Monday to Tuesday, and the lower part which is from Thursday to Sunday. Based on the plot, the hypothesis which concerns whether houses sold more on the weekday than at the weekend is correct, most of the houses were recorded on Tuesday, while on Sunday, the number of house transaction is low.

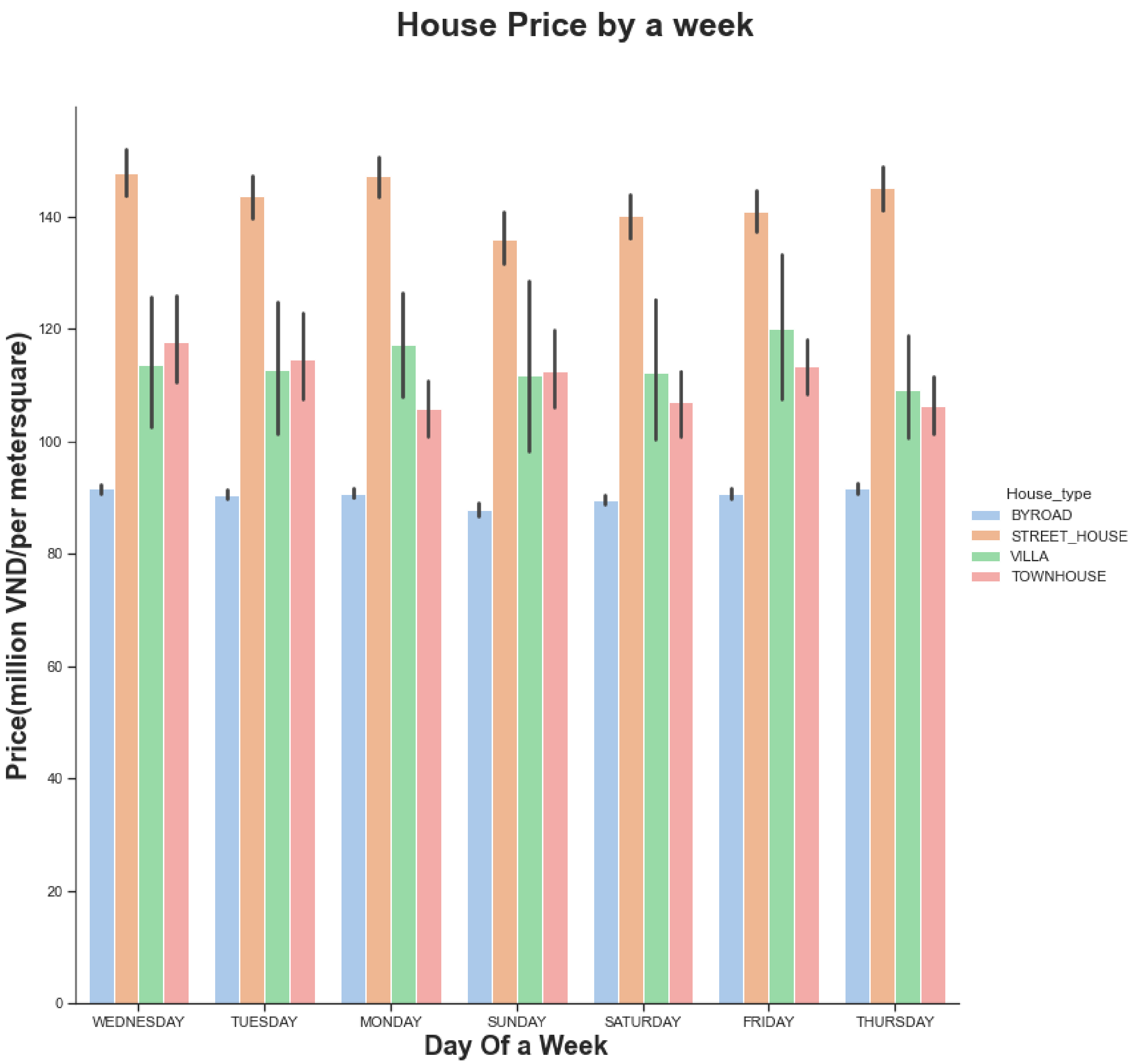


Figure 4: *House Price in a week*

⇒ **Observation:**

The “House Price by a week” plot demonstrates the houses price by a week. The reason that this bar plot is chosen in order to compare the magnitude of the price of house in a week. Based on the plot, the hypothesis which considers whether houses price on the weekend is lower than in the weekday is correct. The house price in the Wednesday is the highest, while in the Sunday the price is the lowest. That can be assumed that the number of house transaction in the Sunday is the lowest so that the price is decreased so that it can attract more customers.

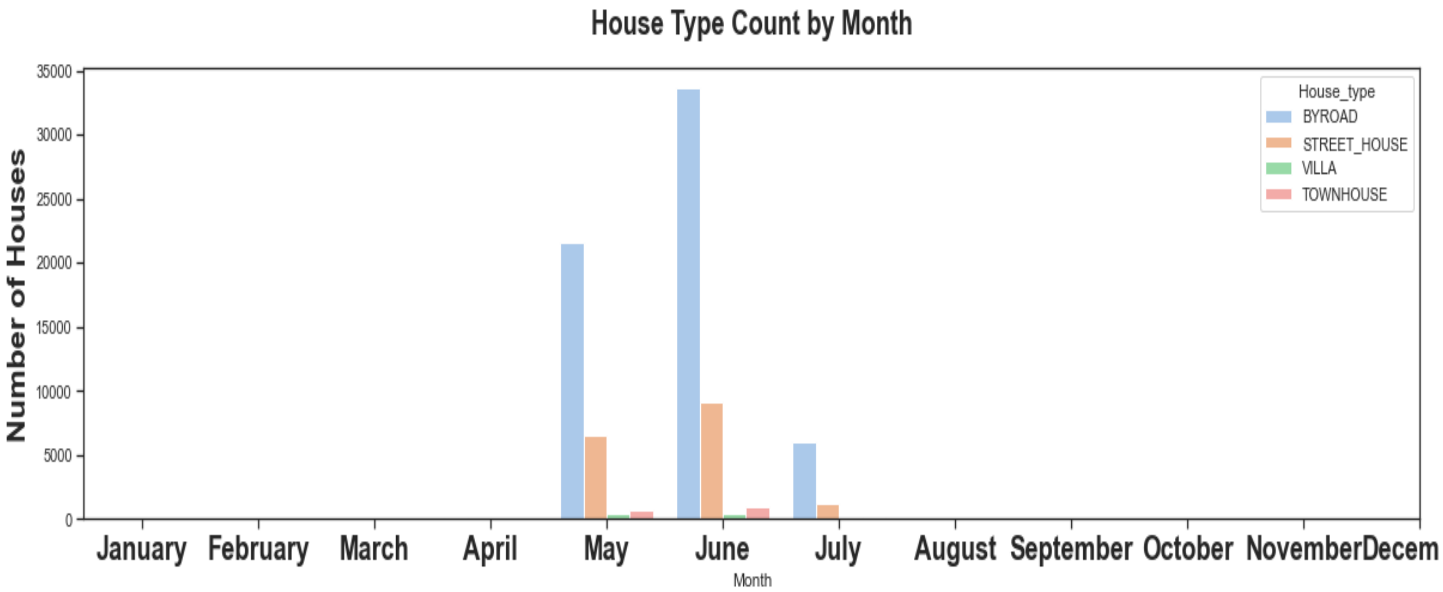


Figure 5: *House Transaction in a Year*

⇒ **Observation:**

The “House Type Count by Weekday” plot illustrates the transaction of the house in a year. The reason that this bar plot is chosen in order to compare the magnitude of the number of house transaction in a year. The plot can be divided into two main parts including the higher part which is from May to July, and the lower part which is in other months. Based on the plot, the hypothesis which concerns whether houses sold more in the end of the year is incorrect, most of the houses were recorded in June, while on other months the number of house transaction is inconsiderable. The reason is that the data set was not crawled from the August, 2020.

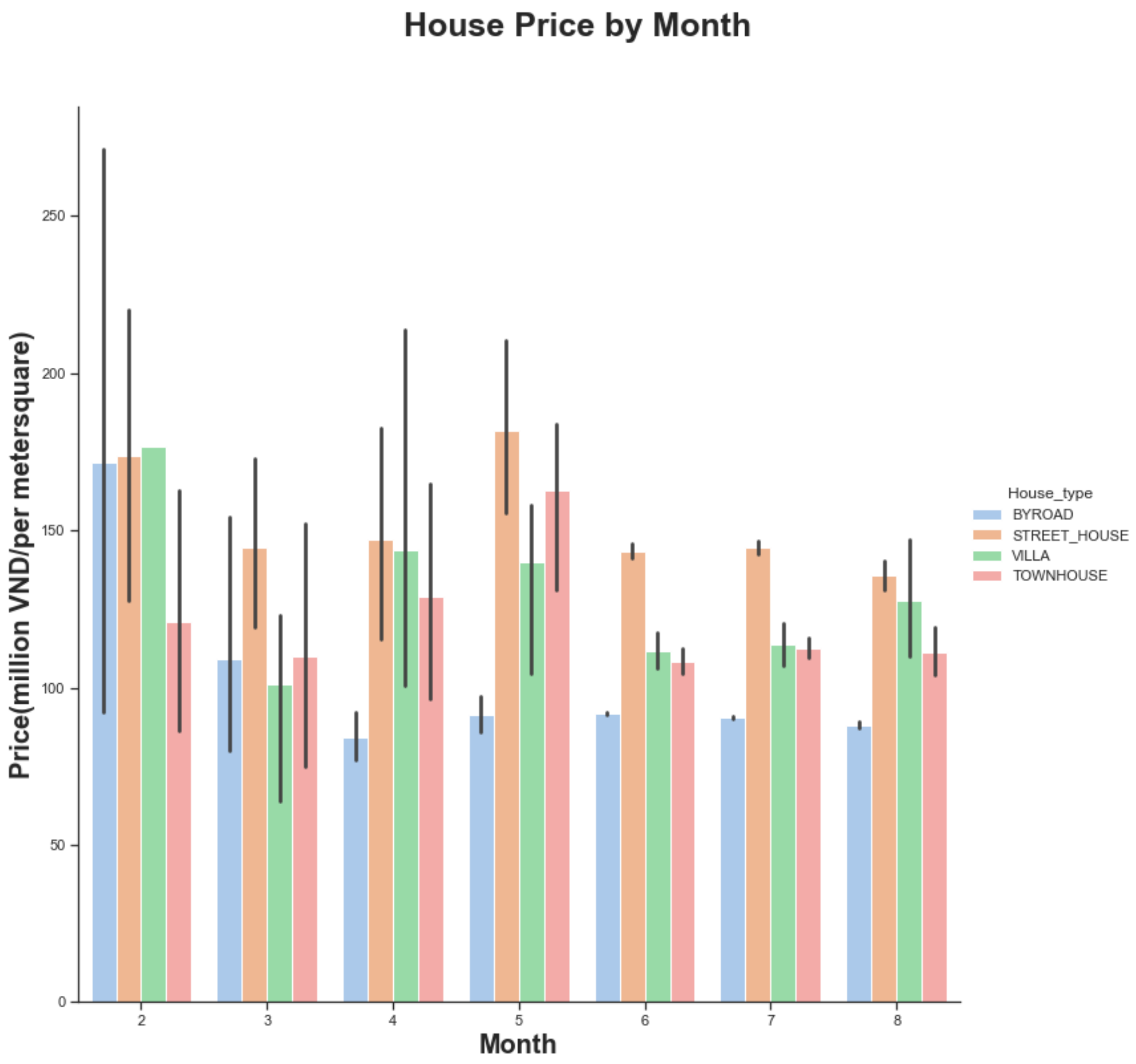


Figure 6: *House Price in a Year*

⇒ **Observation:**

The "House Price by Months" plot indicates the houses price by a year. The reason that this bar plot is chosen in order to compare the magnitude of the price of house in a year. Based on the plot, the hypothesis which considers whether houses price is higher than in the end of the year is incorrect. The house price in the February is the highest, while the March has the lowest price. The reason maybe an interesting topic to discover later since there is no enough information.

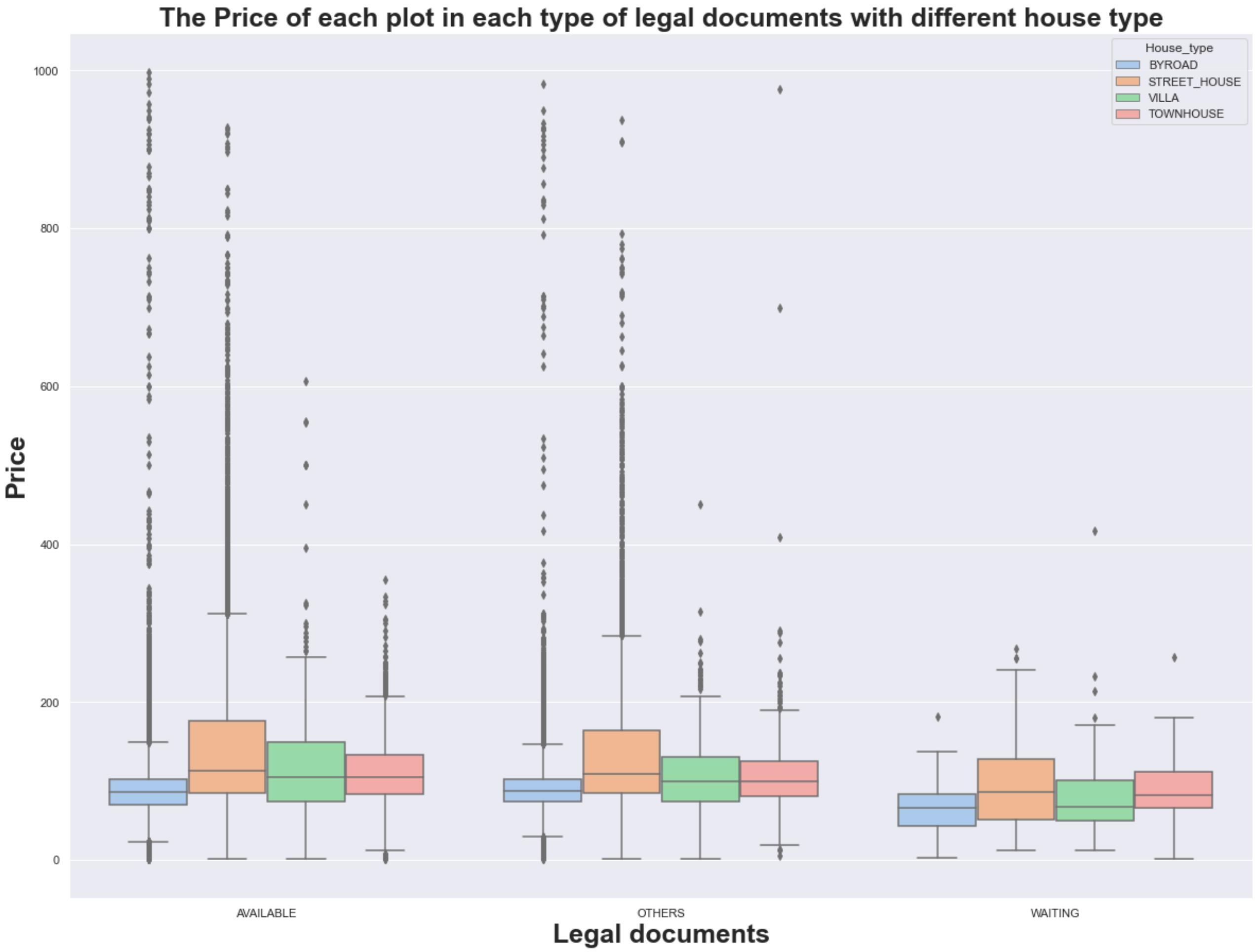


Figure 7: *Price, Legal Document and House Type Relationship plot*

⇒ **Observation:**

The “The Price of each plot in each type of legal documents with different house type” demonstrates the relationship the price and legal documents. The box plot is utilized for demonstrate the magnitude, and the highest frequencies of the price at a specific point. From the plot, the available legal document has the highest price which reached over 1 billion VND per meter square (1000 million).

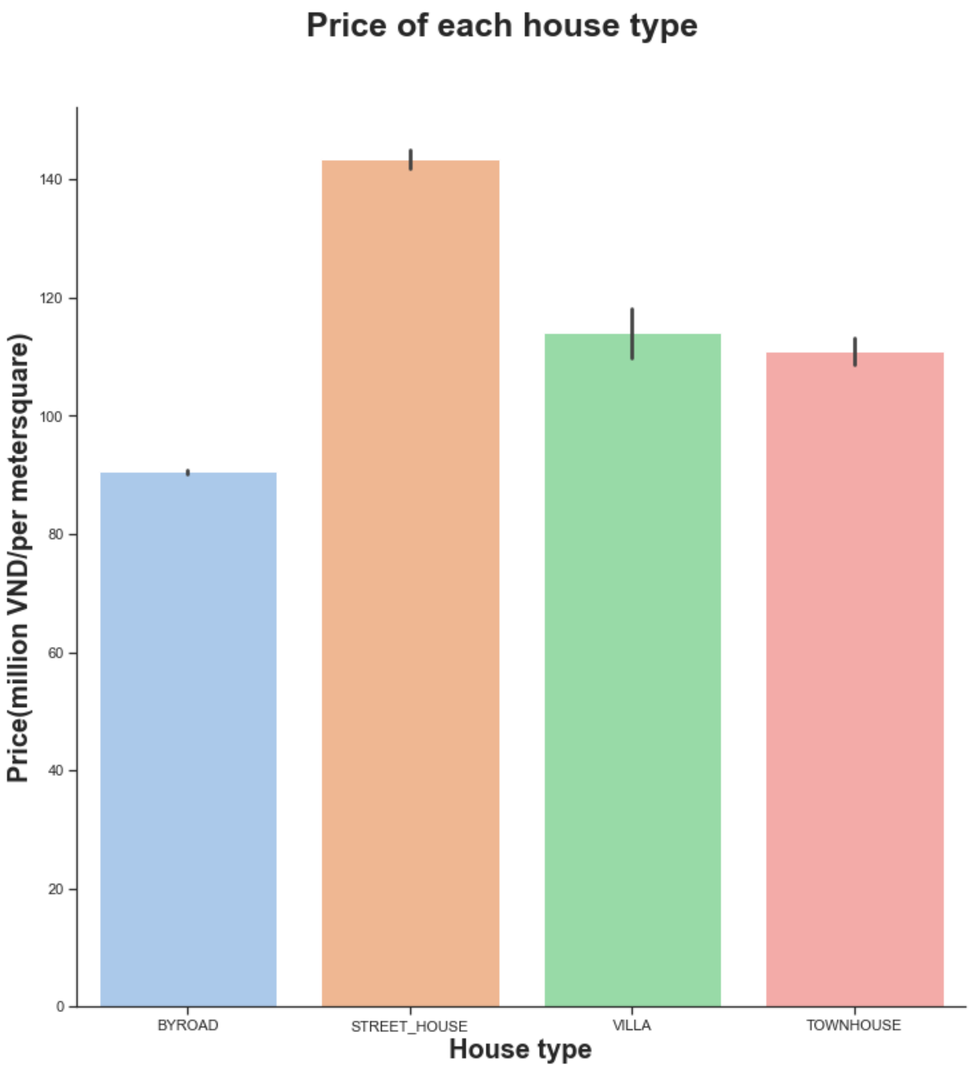


Figure 8: *Price of each house type*

⇒ **Observation:**

The “Price of each house type” plot indicates the price of each house type including byroad, street house, villa, and townhouse. The reason that this bar plot is chosen in order to compare the magnitude of the price of each house type. According to the plots, the price of street house has the highest price, and the byroad has the lowest price

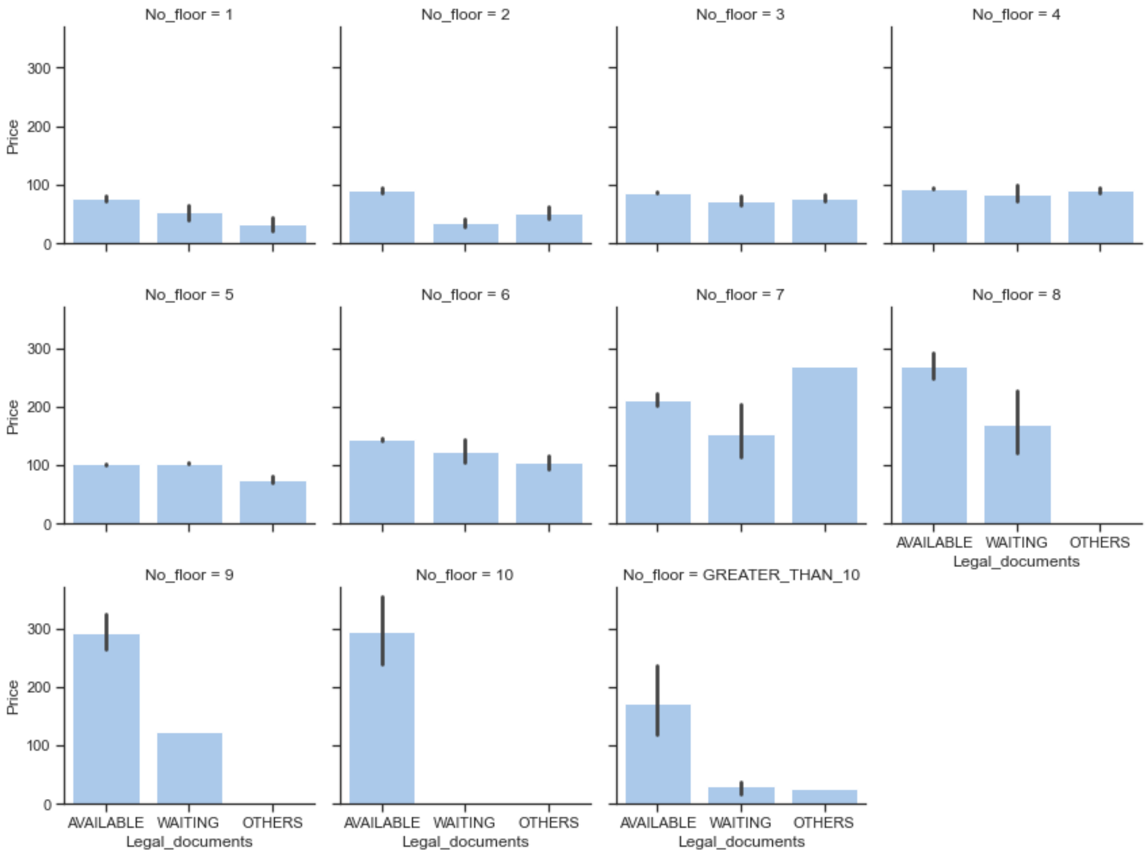


Figure 9: *Price, Legal Document and Number of Floors Relationship*

⇒ **Observation:**

The plot indicates the price of each house having different number of floors. The reason that this bar plot is chosen in order to compare the magnitude of the price of each house having different number of floors. According to the plots, the number of floors that higher than 6 the higher price than those lower than 6 with the price vary from 120 to 200 million VND per meter square. The number of floors 10 has the highest price.

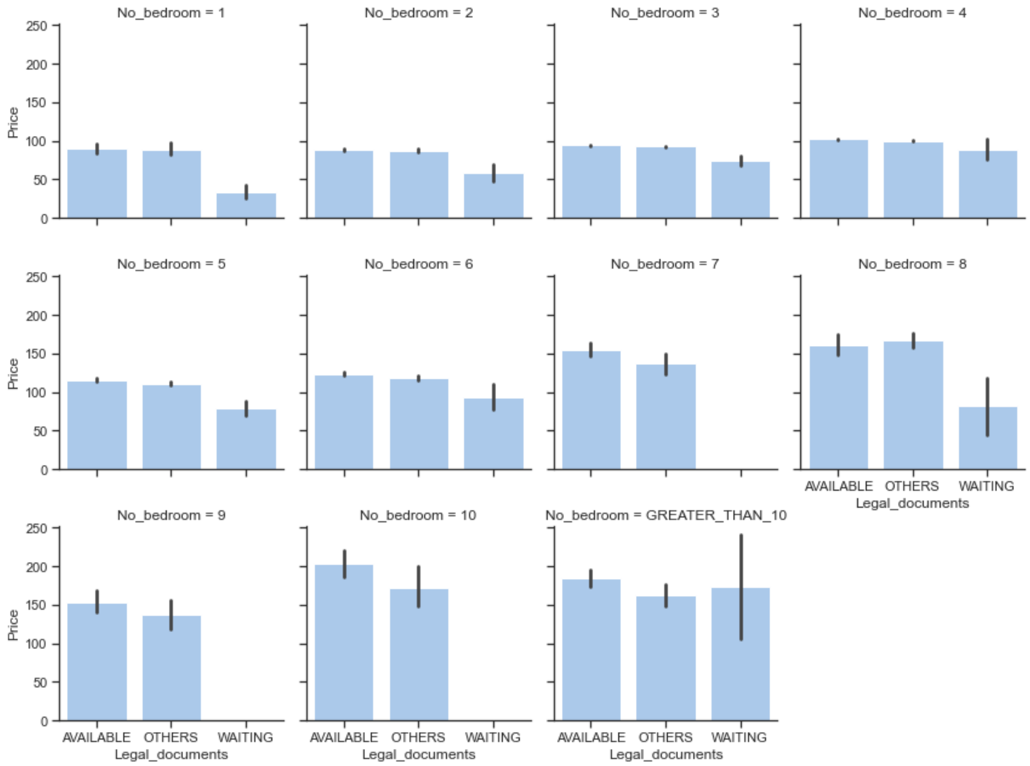


__Figure 10:__ Price, Legal Document and Number of Rooms Relationship

⇒ __**Observation:**__

The plot indicates the price of each house having different number of bedrooms. The reason that this bar plot is chosen in order to compare the magnitude of the price of each house having different number of bedrooms. According to the plots, the number of bedrooms that higher than 6 the higher price than those lower than 6 with the price vary from 120 to 200 million VND per meter square. The number of bedrooms 10 has the highest price.

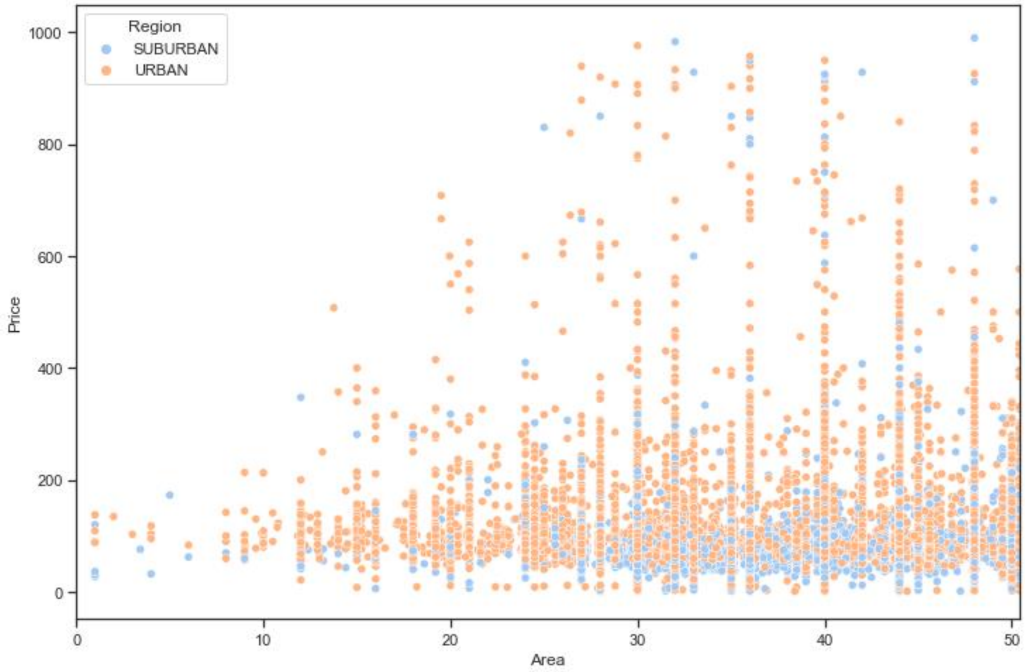


Figure 11: *Price, and Area Relationship*

⇒ **Observation:**

The plot indicates the price of each area vary from 0 to 50-meter square. This plot is chosen since it can indicate the proportional relationship of the price and area feature. From this plot, although it is not a completely proportional relationship, but the general relation is still proportional. Hence, the bigger the area is the higher the price is.

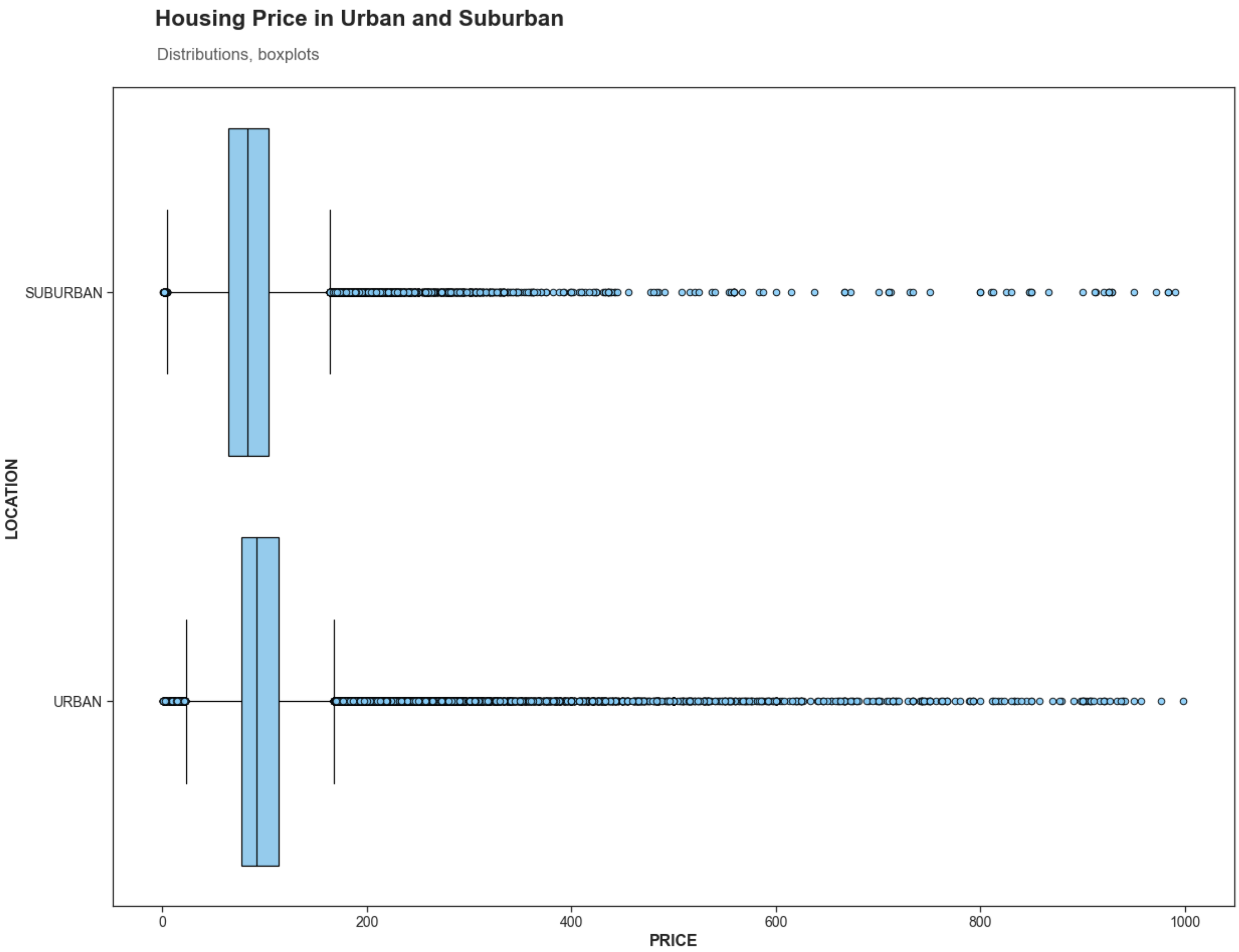


Figure 12: *Housing Price in Urban and Suburban.*

⇒ **Observation:**

The “Housing Price in Urban and Suburban” plot indicates the price of houses in Urban and Suburban. The box plot is utilized for demonstrate the magnitude, and the highest frequencies of the price at a specific point. According to plot, the houses in Urban still have the higher price than houses in Suburban.

d) Summary:

- Most of the houses were recorded on Tuesday.
- The house price in the Wednesday is the highest, while in the Sunday the price is the lowest.
- Surprisingly, significantly higher houses were recorded from May to July.

- The month which has the highest price is February with the highest price is byroad reached over 240 million VND. The lowest price is in April, with the price is lower than 100 million.

- The plots have higher price have Available Legal Documents, the number of floors higher than 2, the number of bedrooms higher than 2, the Area is higher than 50 metersquare, the house type is Street House or Villa, Locate in the center of Hanoi or urban city.

3. Feature Engineering:

a) Select Feature:

Collected features are district, ward, housing types, legal document, number of floors, number of rooms, price (per meter square), day of the week, month, street, price range, region. The area, length, and width are dropped as the price range is predicted depending on the meter square not on the whole area. Moreover, the price range was dealt with the normalization in order to have the equal scale.

b) Encoding:

Before we train the model, we will be encoding all the columns according to its property. For all the category with nominal value, we will use One hot encoding to create columns with value of either 0 or 1. For Categorical that is ordinal, we will use Label encoding to make it into ordered value ranging from 0 to N options.

4. **Modeling**:

The data frame contains 82497 records and 13 columns. There are 82497 training examples in the dataset; this is a good sign since there seems to be large enough data for machine learning. The shape of the dataset tells us that we have 13 attributes. Of the 13 attributes, one is the target variable that the model should predict. This means that the dataset has 12 attributes that can be used to train the future predictive model. In order to train the model properly, there are some factors such as train/test split, model training and data pipeline, classification models for training, hyperparameter tuning – GridSearchCV, evaluation metrics, confusion matrix, model validation and model persistent that we need to concern.

a) Train/Test Split:

We have a medium data size of around 80000 instances; we can split the dataset into 70% train and 30% test without the k-fold Cross-Validation. This sklearn procedure involves randomness, so we set the speed to 42 to ensure reproducibility between runs of the same notebook and between the research and production environment.

b) Model Training and Data Pipeline:

Since training a model duration is tedious, we created a data pipeline for the training models, building a continuous stream of batched data observations to train the model efficiently. Multiple classification modeling is used to create a model of retail house prices.

c) Classification Models:

- **Logistic Regression:**

First, the Logistic Regression as a baseline is used since we desire to determine the potentially unforeseen difficulties in our training model, it is straightforward to apply [5]. Nevertheless, this algorithm sometime ignores some of vital features of the input and limited ability to generate the most efficient results. In the Logistic Regression, it tries to fit an S-curve to estimate the probabilities of outcomes. When we train our logistic regression model, we try to find the best-fit S-curve through all data points.

- **Random Forest with Pipelines:**

The random forest is a machine learning algorithm for solving the classification problems. The reason that it is applied in this project is the accuracy is higher than the decision tree algorithm and assist an efficient way for dealing with the missing values. Nevertheless, it requires more resource owing to the computation and the execution time is high. [6] Moreover, in this project the standard scaler is adopted since there is some numerical values has extremely large scale, so it assists those values to be scaled properly for the machine to learn. Because of the Standard Scaler, the Random Forest must be used with Pipelines.

- **Combining GridSearch + Random Forest with Pipelines:**

scikit-learn is used to search the hyperparameter space to find the best candidate model. Grid search is for hyperparameter tuning in scikit-learn.

Using GridSearchCV is very straightforward. Specifying the trained, the values of the hyperparameters, and scikit-learn will take care of proper evaluation and cross-validation. We observed that grid search works very well but is also computationally expensive because of the number of models that need to be trained. For example, if we had two hyperparameters with three values each multiplied by three, we would need to train and evaluate nine models. So grid search is straightforward to use, but there are a few drawbacks.

- **K-nearest Neighbor with GridSearchCV:**

KNN algorithm assume that the analogue exist in the nearest distance, and calculate the distance between points. [7] It is utilized for this project since it is simple to operate, suitable for classification problem and no model, tune several parameters, or additional assumptions is required. GridSearchCV is used in this project for searching the most optimal parameters over a grid [8]. In this project the best parameter is 5th parameter using uniform distribution.

d) Evaluation Metrics - Accuracy, Precision, and Recall:

Use the built-in classification_report function to print out the metrics for each model quickly, and we can see that the accuracy score is around 68% for logistic, 93% for the random forest, and 35% for the random forest with hyperparameter tuning. The outcome predicted by our random forest model is the one that has the highest accuracy score.

e) Confusion matrix:

Confusion matrix is a performance measurement and applied for describing the performance of classification model. It is a table having four different combinations of both correct and incorrect values. Hence, it is applied for evaluating a model. In this project, it is used to evaluate our model and draw the performance plots.

f) Model Validation:

After training the model for a reasonable length of time, we validate its performance on a piece of the dataset that has been left out. This data must originate from the same underlying distribution as the training dataset, the test set.

g) Model persistent:

Finally, after training and evaluating the model's performance, we store the model algorithms and pipelines using joblib and deploy the model into production. This entails creating a workflow that allows new users to use a pre-trained model to generate predictions quickly.

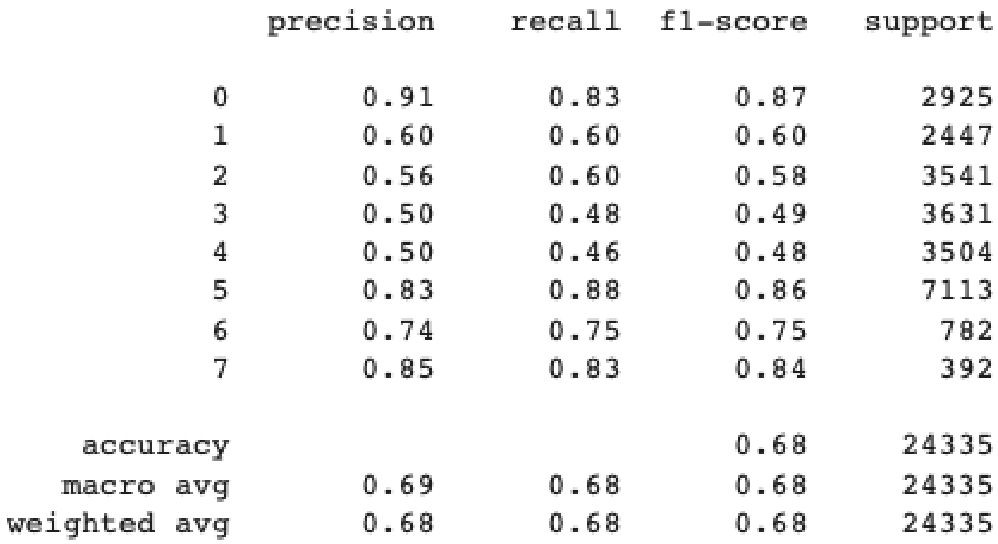
5. Modeling Deployment:

When the model predicts a real-world production environment or even a research environment like what we are doing on Streamlit, saving the model parameters alone, we have also saved out the parameters of the estimator object that we have used to perform preprocessing on the input data. When we deploy this model and use it for prediction, any prediction instance that comes in must be transformed precisely like the training data. In addition to the model and the preprocessing objects, we save any additional information that we might find helpful. Here we have saved the sklearn_version and the accuracy of this model on the test data. Then, we will serialize out to disk using the joblib library. Serializing this to disk involves invoking dumping the pickle file. Once the checkpoint has been saved, we can reload or deserialize the model and the preprocessing object by invoking the load function.

Preprocessing techniques can be complex and may involve several different steps. In that case, it is best to create a scikit-learn pipeline. A scikit-learn pipeline allows the preprocessing we want to perform on the data. Furthermore, the final step of the pipeline is the model that we fit on the data. The best way to serialize the model out to disk for prediction deployments is to create a scikit-learn pipeline that includes all preprocessing and the final model in a single object. We only include the feature scaling step inside the pipeline and directly fit the pipeline on the training data. This will transform the input features into their feature vector representations and then train the classification model on those feature vectors. Once we have a trained model, we can use this pipeline to invoke prediction on the test data directly. The data transformation is taken care of by the pipeline.

RESULTS

After the training model step, the accuracy of the K-nearest neighbors with GridSearchCV has the highest accuracy score. The accuracy score is remarkably high which is around 99 percent, and equal to the micro-F1 score. Although they are convenient for a quick, high-level comparison, their main flaw is that they give equal weight to precision and recall. In the multi-class case, different prediction errors have different implications. They are predicting X as Y is likely to have a different cost than predicting Z as W, and so on. The standard F1 scores do not take any domain knowledge into account. The F-1 score accuracy is used to determine the per-class scores of a multi-class classification issue. Hence, it is the best score for this problem because each independent class will have its own contribution to the price range in the real-world problem. A score that determines the per-class scores will give this a more realistic aspect when we try to compare between models.

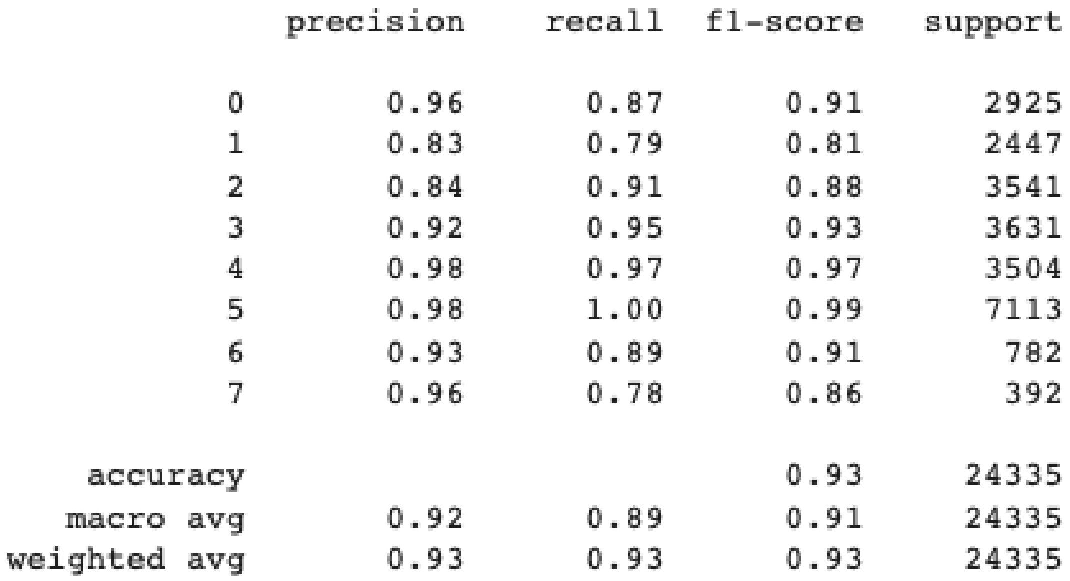


```
              precision    recall  f1-score   support

           0       0.91      0.83      0.87      2925
           1       0.60      0.60      0.60      2447
           2       0.56      0.60      0.58      3541
           3       0.50      0.48      0.49      3631
           4       0.50      0.46      0.48      3504
           5       0.83      0.88      0.86      7113
           6       0.74      0.75      0.75       782
           7       0.85      0.83      0.84       392

    accuracy                           0.68     24335
   macro avg       0.69      0.68      0.68     24335
weighted avg       0.68      0.68      0.68     24335
```

Figure 13: *F1 score of the Logistic Regression*

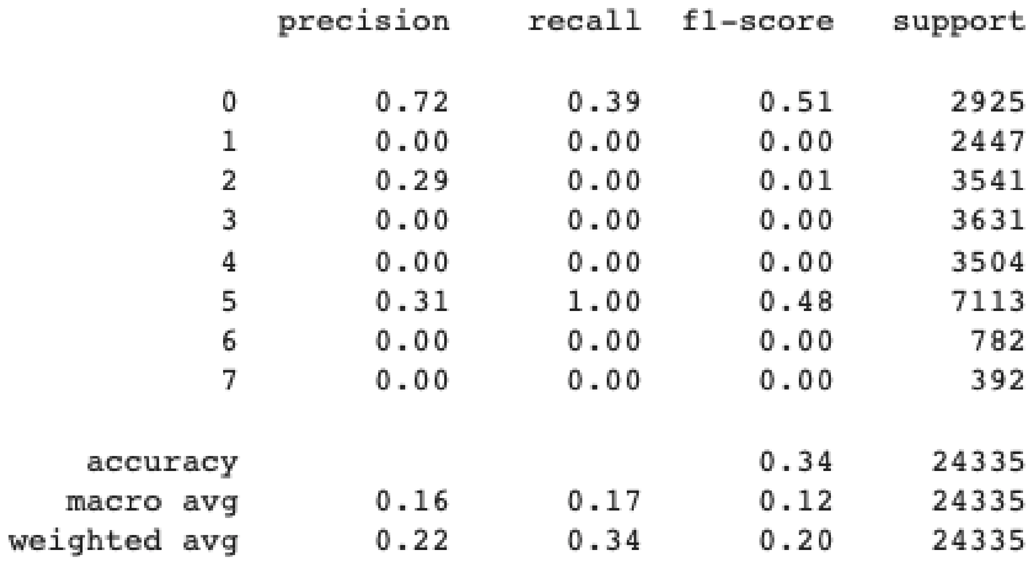


```
              precision    recall  f1-score   support

           0       0.96      0.87      0.91      2925
           1       0.83      0.79      0.81      2447
           2       0.84      0.91      0.88      3541
           3       0.92      0.95      0.93      3631
           4       0.98      0.97      0.97      3504
           5       0.98      1.00      0.99      7113
           6       0.93      0.89      0.91       782
           7       0.96      0.78      0.86       392

    accuracy                           0.93     24335
   macro avg       0.92      0.89      0.91     24335
weighted avg       0.93      0.93      0.93     24335
```

Figure 14: *F1 score of the Random Forest with Pipeline*

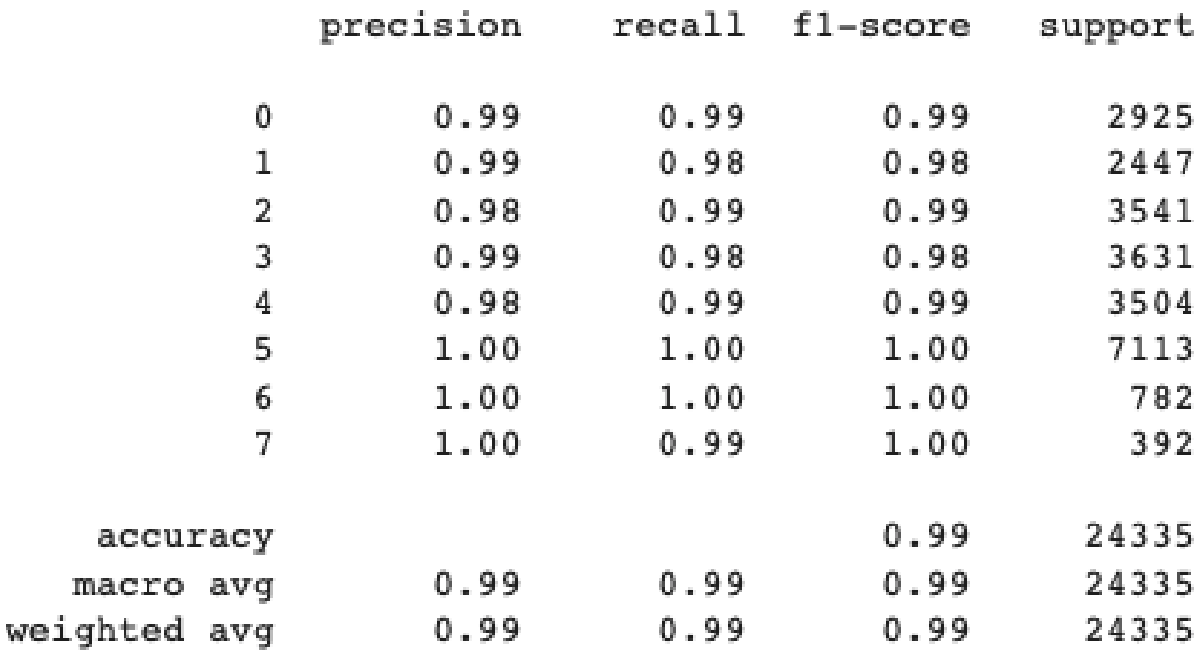


```
              precision    recall  f1-score   support

           0       0.72      0.39      0.51      2925
           1       0.00      0.00      0.00      2447
           2       0.29      0.00      0.01      3541
           3       0.00      0.00      0.00      3631
           4       0.00      0.00      0.00      3504
           5       0.31      1.00      0.48      7113
           6       0.00      0.00      0.00       782
           7       0.00      0.00      0.00       392

    accuracy                           0.34     24335
   macro avg       0.16      0.17      0.12     24335
weighted avg       0.22      0.34      0.20     24335
```

Figure 15: *F1 score of the Combination of GridSearchCV and Random Forest with Pipeline*

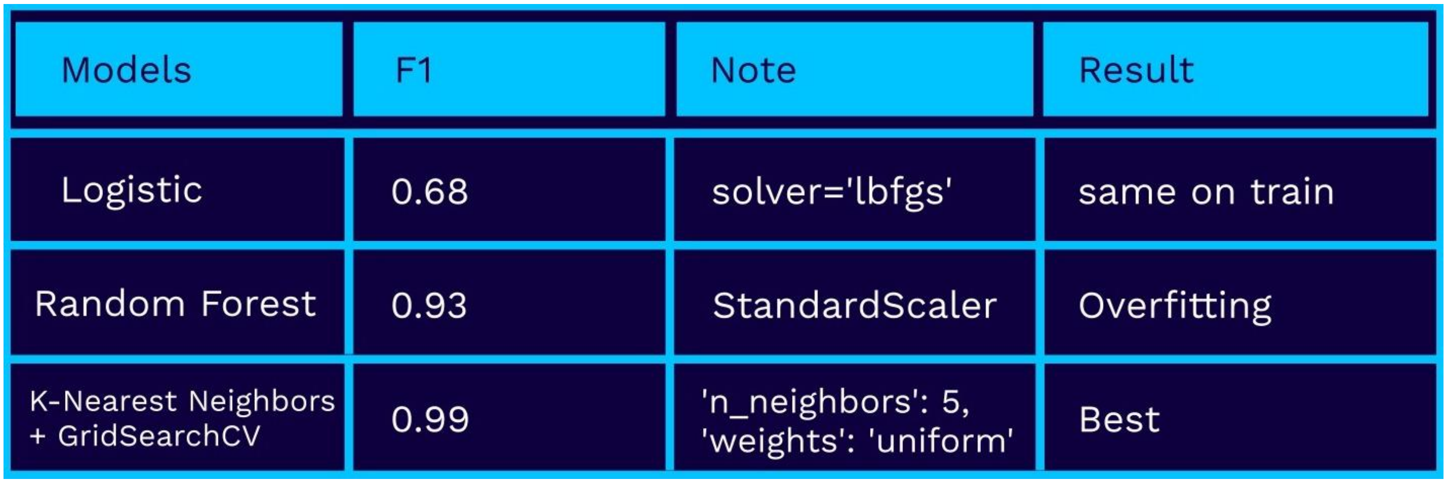


```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      2925
           1       0.99      0.98      0.98      2447
           2       0.98      0.99      0.99      3541
           3       0.99      0.98      0.98      3631
           4       0.98      0.99      0.99      3504
           5       1.00      1.00      1.00      7113
           6       1.00      1.00      1.00       782
           7       1.00      0.99      1.00       392

    accuracy                           0.99     24335
   macro avg       0.99      0.99      0.99     24335
weighted avg       0.99      0.99      0.99     24335
```

Figure 16: *K-nearest Neighbor with GridSearchCV*



Models	F1	Note	Result
Logistic	0.68	solver='lbfgs'	same on train
Random Forest	0.93	StandardScaler	Overfitting
K-Nearest Neighbors + GridSearchCV	0.99	'n_neighbors': 5, 'weights': 'uniform'	Best

Figure 17: *Evaluation Matrix Summary*

DISCUSSION

1. Hyperparameter Tuning – GridSearchCV:

The cost and complexity of grid search can proliferate. So if we train on a cloud platform, it can quickly become costly. Also, grid search does not differentiate between important and trivial hyperparameters, and it treats all hyperparameters the same way. An alternative to grid search for hyperparameter tuning is a random search of the hyperparameter space, which we would consider in future improvement.

2. Maintainability Machine Learning Models into Production

We only deploy our machine learning algorithms in the research environment using Flask and Dash. The project does not include deployment lifecycle, CI/CD, differential testing, reproducibility, and the best coding techniques and factors to consider when putting a model into the fully integrated production environment. We also do not have the flexibility disposal for deploying our models, and we are in a better position to steer the deployment of the models in whatever path that best fits the project goals is a complete predictor web app to the end-user.

We need a solution to migrate from a Jupyter notebook to a fully deployed machine learning model, considering CI/CD, and deploying to cloud platforms and infrastructure. We also need to consider model monitoring, advanced deployment, scheduled workflows, and various testing paradigms such as shadow deployments.

CONCLUSION

As foremetioned, the real estate prices has played an essential role, it is neccessary to study and understand in order to develop a training model to predict the house price range. By analysing and exploring the Vietnamese Housing Dataset in Hanoi which is retrieved from the Kaggle, there are some interesting insight which was explored.

We ingested, prepared, and explored the Vietnam Housing Dataset (Hanoi) from Kaggle to serialize our model parameters to disk. We explored three different techniques for doing so. We were serializing scikit-learn models using pickle. Then, we discussed how it was essential to include the model preprocessing steps as a part of model serialization. We then put all of this together. It was preprocessing steps, model fitting into a single scikit-learn pipeline, and building a classification model to perform multi-classification on housing prices. We saw how to deploy an exploratory process on Dash and the classification model to a Streamlit web application and use it for predictions. We will continue working with the same classification model and deploy it to a serverless environment for future improvement.

Although, there are Logistic Regression, Random Forest with Pipeline, Combining GridSearch and Random Forest with Pipeline, and K-nearst Neighbors with GridSearchCV, the K-nearst Neighbors with GridSearchCV is still the most optimised model.

REFERENCES:

[1] W. C. Apgar, with R. Peng and J. Olson. Recent Trends in Real Rents. Working Paper W87-5, Joint Center for Housing Studies of Harvard University, 1987.

[2]"Getting ahead of the market: How big data is transforming real estate", McKinsey & Company, 2022. [Online]. Available: https://www.mckinsey.com/industries/real-estate/our-insights/getting-ahead-of-the-market-how-big-data-is-transforming-real-estate. [Accessed: 10- Jan- 2022]

[3]"Điều kiện tách thửa tại Hà Nội năm 2021 mới nhất", danviet.vn, 2022. [Online]. Available: https://danviet.vn/dieu-kien-tach-thua-tai-ha-noi-nam-2021-moi-nhat-20210329062551O3.htm. [Accessed: 10- Jan- 2022]

[4]"Quy định về mặt sàn với lộ giới để được phép xây dựng số tầng?", *luatminhkhue*, 2022. [Online]. Available: https://luatminhkhue.vn/quy-dinh-ve-mat-san-voi-lo-gioi-de-duoc-phep-xay-dung-so-tang-.aspx. [Accessed: 10- Jan- 2022]

[5]"Always start with a stupid model, no exceptions.", Medium, 2022. [Online]. Available: https://blog.insightdatascience.com/always-start-with-a-stupid-model-no-exceptions-3a22314b9aaa. [Accessed: 10- Jan- 2022]

[6]"Introduction to Random Forest in Machine Learning", Engineering Education (EngEd) Program | Section, 2022. [Online]. Available: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/. [Accessed: 10- Jan- 2022]

[7]"Machine Learning Basics with the K-Nearest Neighbors Algorithm", *Medium*, 2022. [Online]. Available: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761. [Accessed: 10- Jan- 2022].

[8]"Grid Search for model tuning", Medium, 2022. [Online]. Available: https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e. [Accessed: 10- Jan-2022].

APPENDIX:

Link github of our project: https://github.com/tnathu-ai/COSC2789_Group_Project