# GUIDELINES

A PREPRINT

May 12, 2024

**ABSTRACT**

The final project for discussion sections in this quarter will be graded by the integrity of the report, code and prediction file. This Guideline is a rubric and brief instruction for the project. You may check the previous discussion file for detailed information.

## 1 Datasets

There are four datasets provided.

1. housing prediction data(response is Price)

2. job fraud prediction data(response is fraudulent(binary))

3. uber travel time prediction(response is duration)

4. uber fare prediction(response is fare amount)

You may also select other datasets you are interested in as long as you can use the data to finish all the sections needed.

## 2 Data Exploration and Cleaning

Once you get the dataset, the first step after reading it into environment will always be data exploration.

```
data=pd.read_csv(filelocation)
```

You may check the format of each columns including the response and predictors. Some datasets has the numeric response(in float form) then leads to the regression problem while other datasets could have binary response which leads to the classification problem. To get an over view of the data, you can use the describe function in pandas package to check out all the numerical values.

```
data.describe()
```

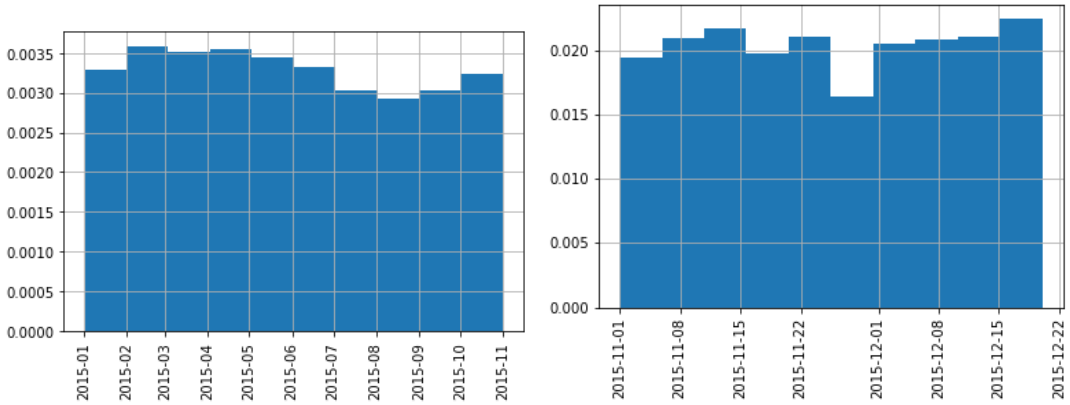
You will get a summary table like the example below:

| | start_lng | start_lat | end_lng | end_lat | duration |
|---|---|---|---|---|---|
| mean | -73.973910 | 40.750574 | -73.973641 | 40.751329 | 837.098389 |
| std | 0.037917 | 0.027880 | 0.035833 | 0.031879 | 708.095112 |
| min | -74.514618 | 40.368916 | -74.517853 | 40.368874 | 1.000000 |
| 25 | -73.992119 | 40.736870 | -73.991478 | 40.735409 | 399.000000 |
| 50 | -73.981995 | 40.753445 | -73.980156 | 40.753956 | 662.000000 |
| 75 | -73.967789 | 40.767799 | -73.963570 | 40.768894 | 1070.000000 |
| max | -73.414452 | 41.031418 | -73.414352 | 41.031509 | 43178.000000 |

You can also use the pandas package to check the unique values for the categorical variable.

You can also use the plot function in pandas to generate histogram, scatterplot, barplot and etc.

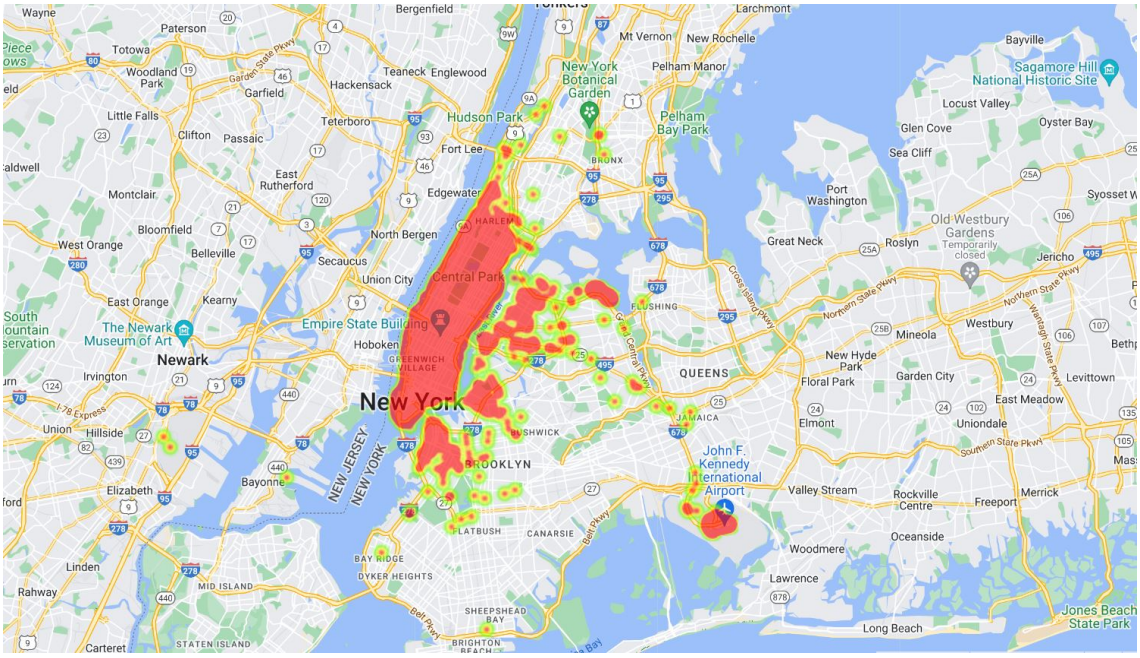
Figure 1: (a) histogram of start time for training data (b) histogram of start time for testing data




(a) (b)

There are also outside API tool can help illustrate the dataset. Below is an example using google map API to help understand the distribution of locations in the uber data. It might be interesting to learn this but not necessary for this report.

```
gplot = gmplot.GoogleMapPlotter(data_train.start_lat.mean(), data_train.start_lng.
                                                    mean(), 6)
gplot.heatmap(data_train[0:30000].start_lat, data_train[0:30000].start_lng)
gplot.apikey = 'My google api key'
gplot.draw('\heatmap.html')
```




This heatmap shows that the riding data is within the New York Metropolitan Area.

The next step is data cleaning. It is essential as the data may or may not be in a well organized way. Some records may be missing(nan value) and some records may be not reasonable( for example, in the uber data, the travel time is over 6 hours within a small change in geolocation).

```
data.dropna()
data_train=data_train.loc[data_train.duration<=20000]
```

## 3 Feature Engineering

After cleaning the data, sometimes, the datasets are still not suitable to be fed into the model. This could be because the variable is not in numeric format(most of the machine learning models can only handle numeric values) or the variable is not a good predictor of the response. Sometimes, we can draw more information from the variables (for example, get the day, week, month from the time variable).

### 3.1 Feature Extraction

Part of the feature engineering can be extracting features from the existing variables.

For example, when a variable has a great proportion missing values, we can define a new variable to indicate if one row misses the record for this specific variable.

If we have the latitude and longitude of the start and end locations, we can extract euclidean distance between the two locations by the haversine equation:

$$d = 2Rarcsin(\sqrt{\sin^2(\frac{\phi_2 - \phi_1}{2}) + \cos\phi_1 \cos\phi_2 \sin^2(\frac{\lambda_2 - \lambda_1}{2})}) \tag{1}$$

If we have the variable time stamps, the date of the year(**date** in the form of yyyy-mm-dd), the hour of the day(**hour** 0-23), and the day of the week(**weekday**1-7) can be extracted from it. Moreover, if you have external data, you can get more information like the weather for that specific time.

### 3.2 Feature Encoding

Once all the feature needed for training is prepared, there is one more step before modeling. We need to encode all the categorical variables into numeric values.

Sometimes, one variable only has several unique values, then you may use unique function in numpy to encode the values into numeric values.

```
data[col]=np.unique(data[col],return_inverse=True)[1]
```

Sometimes, one variable can have a huge vocabulary of categorical values, then you may use the build in hashing method(encode the word into vectors) to encode the variable.

```
from sklearn.feature_extraction import FeatureHasher
```

You also don't need to select all the variables. For example, you don't have to include the id in the model training. You can use the drop function to drop the variables you don't need.

```
data.drop(columns=col_list)
```

## 4 Model selection and Training

As the training data is with response which is numeric, we can begin to select a machine learning model and train the model in our datasets.

### 4.1 Model selection

Based on the data you chose, you need to select a proper model to handle the data. For example, a regression model for numeric response and a classification model for binary response.

### 4.2 Model Training and Evaluation

After picking a model, you may split the data into predictors and the response.

```
X = data_train.drop(columns=['duration'])
Y = data_train.duration
```

It does not make a lot of sense if you train the model on a dataset and evaluate the performance on the same data because the model can always overfit the data given enough complexity and we need a model with more ability in generalization. So you will need to split the data into training and testing set and only evaluate the model performance in testing set. You can also use the cross-validation technique to help you split the training and testing data.

```
X_train,X_test,Y_train,Y_test = train_test_split (X ,Y , test_size = 0.2 )
```

For the model evaluation, you will need a metric the measure the success of the model. You can use Mean Squared error for the numeric response. For the classification problem, it will be a little bit tricky. You can always use the accuracy but sometimes, the data is imbalanced and then accuracy might not be a good metric. You can check the precision and recall instead.

And, sometimes, the model has the hyperparameters, it is also helpful to do some hyperparameter tuning.(It is nice to have but not required).

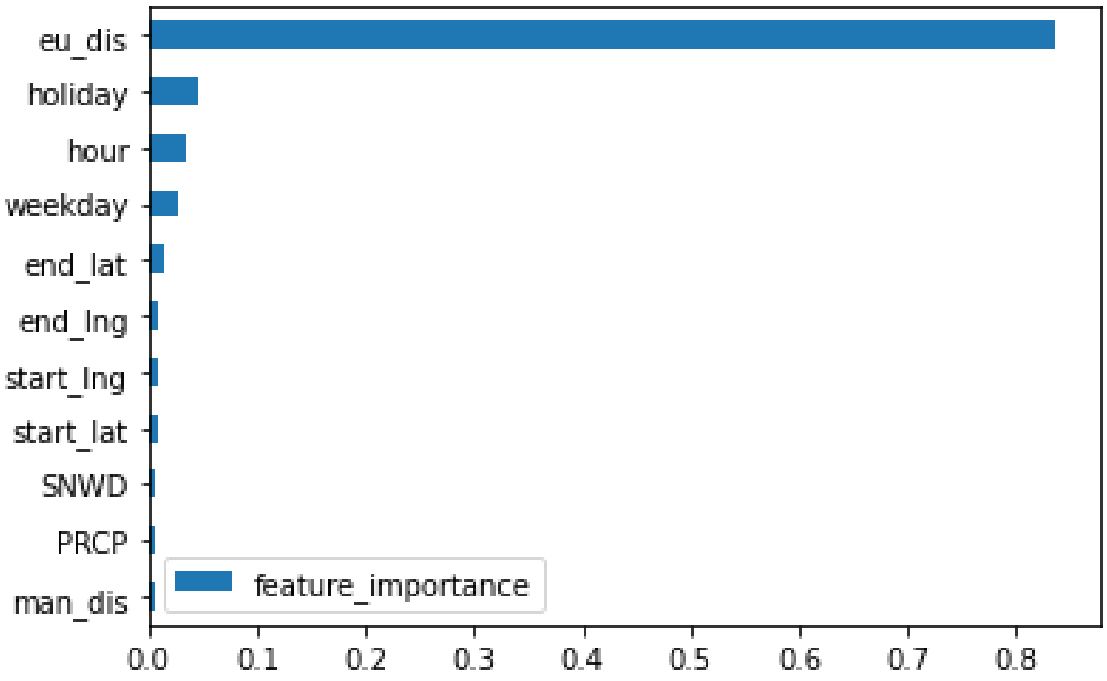
```
max_depth=[8,9,10,11,12]
learning_rate=[0.04,0.045,0.05,0.055]
for md in max_depth:
    for lr in learning_rate:
        xgb = XGBRegressor(max_depth=md, learning_rate=lr, n_estimators=500,
                                                reg_lambda=0.5,tree_method="hist"
                                                , device="cuda")
        xgb.fit(X_train,Y_train)
        Y_pred=xgb.predict(X_val)
        mse=mean_squared_error(Y_pred,Y_val)
        print('max_depth : {}, learning_rate : {}, MSE : {}'.format(md, lr,np.sqrt
                                                (mse)))
```

### 4.3 Feature importance

It is also important to check the feature importance of the prediction model.

For some models like Xgboost, there is built-in function to check the contribution of each variable.

```
plot_to_show = pd.DataFrame(xgb.feature_importances_,columns = ['
                                        feature_importance'],index = X_train.
                                        columns).sort_values(['feature_importance
                                        '], ascending=True)
plot_to_show.plot(kind = 'barh')
```




For models like linear regression, you can use statistical inference instead. By calculating the p value of each coefficient, you can get the significance of each variable.

## 5 Rubric

The grading will be based on the following parts. The grading will not be very strict. You will get most of the points as long as you include all the must-have parts. You don't have to get an accurate model and this is not a kaggle competition. However, you need to comment your code so I can understand each of steps you take.

### 5.1 data exploration and cleaning(5 points)

1. Successfully read the data into the environment (1pt)
2. Get the summary information of the data (1pt)
3. Draw plots to help understand the data (1pt)
4. Data cleaning (2pts)

### 5.2 feature engineering(5 points)

1. Feature extraction (3pts)

2. Feature encoding (2pts)

### 5.3 modeling and evaluation(5 points)

1. Split the predictors and response (1pt)
2. Model selection (1pt)
3. Model evaluation (2pts)
4. Feature importance (1pt)

### 5.4 prediction(5 points)

For this part, you will need to use the model you trained to predict the response in the test data(You will need to first separate the data into training and testing data) and save it as a csv file.(5 pts)

## 6 Submission

This project can be a group assignment. You can either choose to work on your own(you will get 3 points bonus for that when you lose any points in any sections) or work in a group with a maximum total 3 members.

A valid submission need to include the code, output file and a report.

There are two acceptable combination of files:

1. A jupyter notebook file with all the needed comments(explanation of each step) with a csv file as output.
2. A py file as your code and an attached pdf report to explain each step you take and a csv file as output.

**All the submitted results should be reproducible with your code provided.**

**All the submission file needed to be compressed in one zip file with your name and student id with all members in the team.**