INF2179 Final Project Report

New York City Taxi Trip Duration Prediction

Group 8: Jia Jia Ji, Yingying Zhou, Zhengzheng Wang

1. Introduction to data

Our dataset is the New York City taxi trip record data, which contains the trip characteristics and durations of yellow taxis in 2016 obtained from Kaggle (an open-source data repository website). The dataset is sampled from the 2016 NYC Yellow Cab trip record in Big Query on Google Cloud Platform, and the original dataset was collected and published by NYC Taxi & Limousine Commission (TLC). We have a dataset with 1,458,644 observations. This Kaggle competition is designed to predict trip durations (our target variable) in the test data based on the best machine learning model trained by the trip features and durations in the training set.

2. Motivation

According to the 2016 annual report by the New York TLC Taxi and Limousine Commision, the taxi service is in need of improvement of delay occurrences due to frequent road congestion, and certainly there is high demand for taxis due to high traffic volume. Meanwhile at the same time the traditional taxi industry is facing fierce competition from share-ridings such as Uber and Lyft. Its market share is being cannibalized by the new entrants although the whole market size is expanding due to the new business mode from share-riding. The high market demand, increasing competition and the problem of trip delays make the duration prediction a prominent business problem for the taxi operators in New York.

3. Dataset description

There are 11 variables in the dataset. The column *'id'* serves as an index to uniquely identify each trip. The numeric variable *'trip_duration'* records the duration of each trip in seconds as the target variable. It has integer values that range from 1 to 3526282 (the longest trip duration) with an average of 959. Figure 14(b) shows that the *trip_durations* is normally distributed after log transformation. Besides, there are 9 features in the data:

- *'vendor_id'* is used to identify which technology provider records this trip. It has 2 values '1' and '2' to indicate the corresponding vendor type. Figure 15 shows that vendor 2 carries more trip records (780,302 trips) than vendor 1 (678,342 trips).

- The numeric variable '*passenger_count*' represents the number of passengers entered by the driver in a trip, ranging from 0 to 9 in integer. Figure 16 shows that most trips have only 1 or 2 passengers (1,033,540 trips for 1 passenger, and 210,318 trips for 2 passengers).
- '*pickup_datetime*' and '*dropoff_datetime*' respectively indicate the time that a taxi picks up and drops off passengers on a trip. They are object type variables in the data, but we convert them to datetime format to easily extract the data-related attributes, such as day of the week and hour of the day. Figure 23 shows that the trip counts are evenly distributed between January and June in 2016, except that there is a large drop of trip counts around January 20.
- '*Pickup_longtidude*', '*pickup_latitude*', '*dropoff_longtidude*', and '*pickup_latitude*' record the longitudinal and latitudinal coordinates of pickup and dropoff location in a trip respectively. They are numeric variables with floats. We do feature engineering by computing the distance between the pickup and dropoff location for each trip, and the distance ranges from 0 to 1241.30 with an average of 3.44 km. Figure 17 shows that most trips have distances around 0 to 8 km (we omit some large extreme values (outliers) in the boxplot to better capture the spread of distances).
- The nominal variable '*store_and_fwd_flag*' indicates that if a taxi cannot connect to the server, whether the trip information is saved in the taxi memory before the vendor records it. It has 2 unique levels: 'N' with 1450599 trips and 'Y' with 8045 trips.

4. Exploratory Data Analysis

The scatter plot (Fig. 1) identifies an interesting pattern between the day_of_week and the total number of trips provided by two technology vendors. Generally, the number of pickups of vendor 2 is much higher than that of vendor 1. For both vendors, the number of pickups is the lowest on Monday, and it gradually increases during weekdays, peaks on Friday, and drops largely on Sunday.

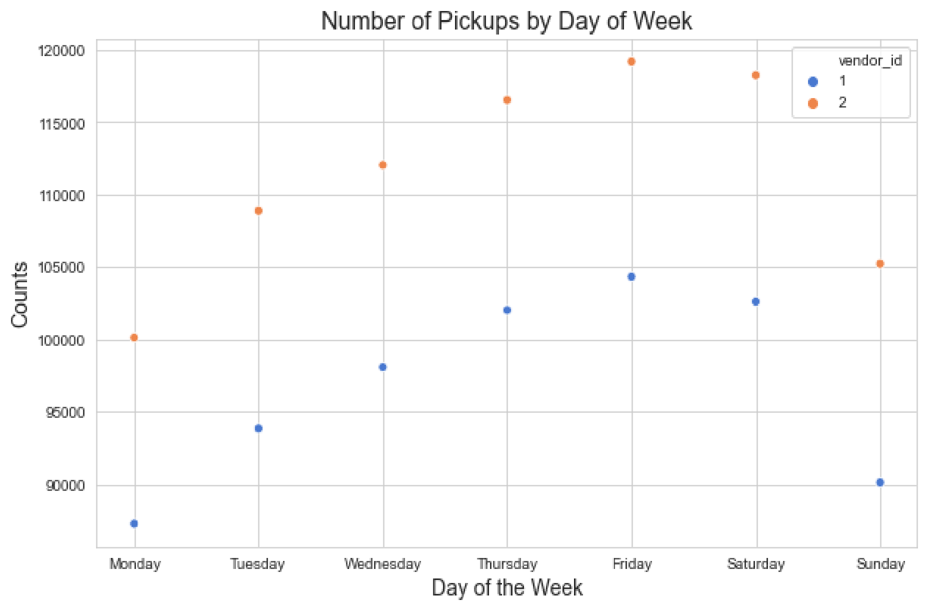


Figure 1. The scatter plot of pickup_counts vs. day_of_week for two vendors

In addition, the scatter plot in Figure 2 illustrates how the number of pickups changes within one day. For both vendors, most pickups happen during 7 am to 11 pm, and the most frequent pickups occur around 7-10 pm. Vendor 1 has the least number of pickups at around 5 am, while the most frequent pickups happen at around 7 pm; for vendor 2, the number of pickups is also lowest at around 5 am, it peaks at around 6 pm.

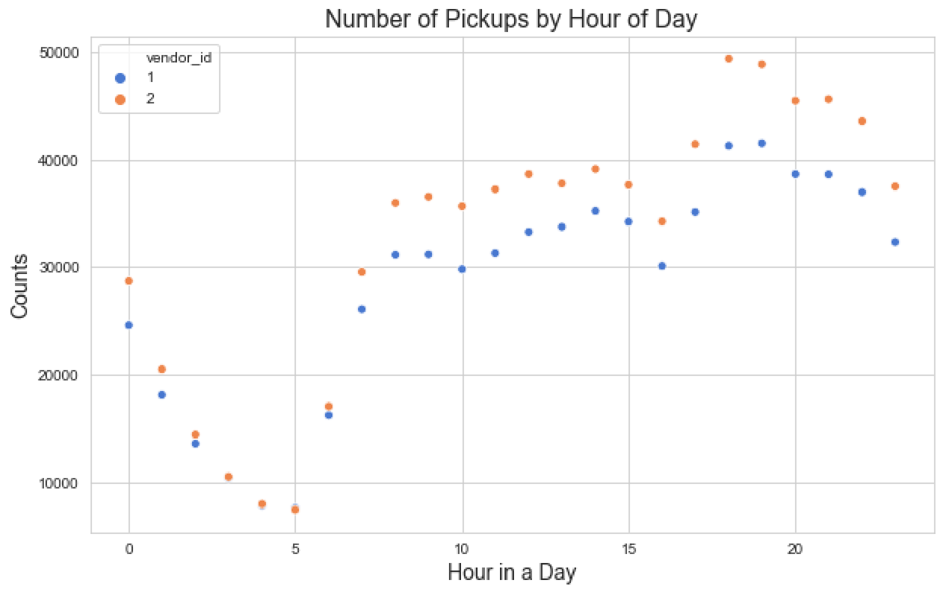


Figure 2. The scatter plot of pickup_counts vs. day_of_hour for two vendors

Another interesting finding is how the average trip duration varies by the days in a week for two vendors, as shown by the line plot in Figure 3, We can see that the average trip duration is the longest on Tuesday and the shortest on Sunday for vendor 1; while for vendor 2, the average trip duration is the longest on Thursday and the shortest on Monday.

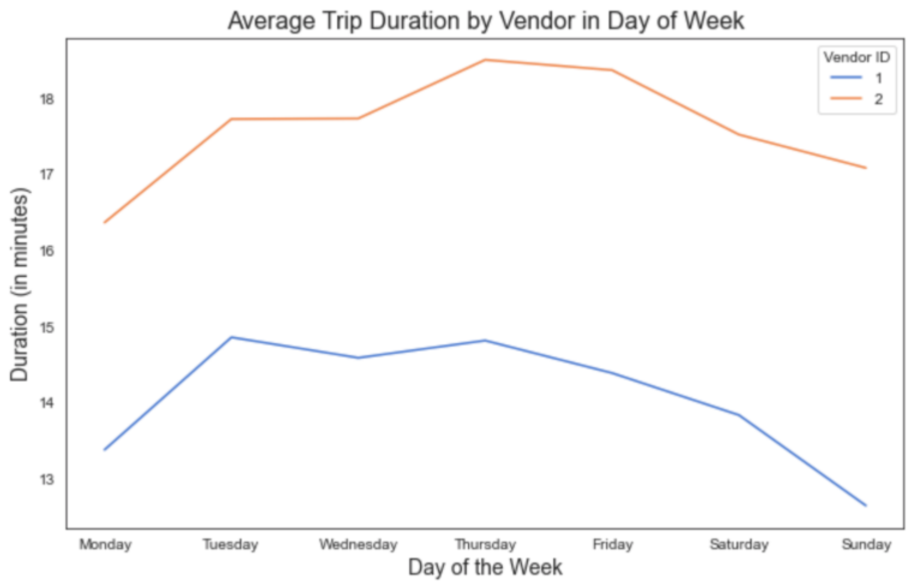


Figure 3.The line plot of mean_trip_duration vs. day_of_week for two vendors

In addition, Figure 4 shows the distribution of the average trip duration during different hours in one day. For both vendors, the average trip duration is the longest at around 3 pm, meanwhile the average trip duration is the shortest at about 5 pm for vendor 1 and at about 6 pm for vendor 2.

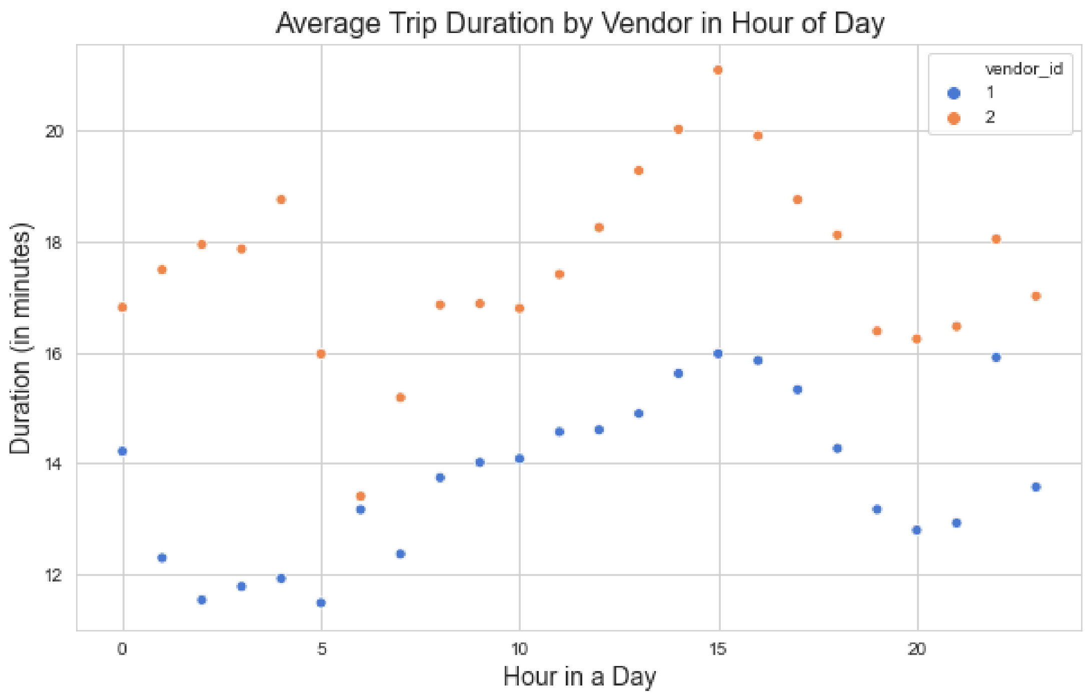


Figure 4. The scatter plot of mean_trip_duration vs. day_of_hour for two vendors

Then, we plot the line plot (Fig. 5) to explore how the average trip duration changes by date. On average, trip duration is higher for vendor 2 than for vendor 1, and most trips have similar durations during the data record period for both vendors. For vendor 1, the average duration climbs to a peak around early January and in mid-February; as for vendor 2, the average duration slightly increases around late January and in the beginning of June.

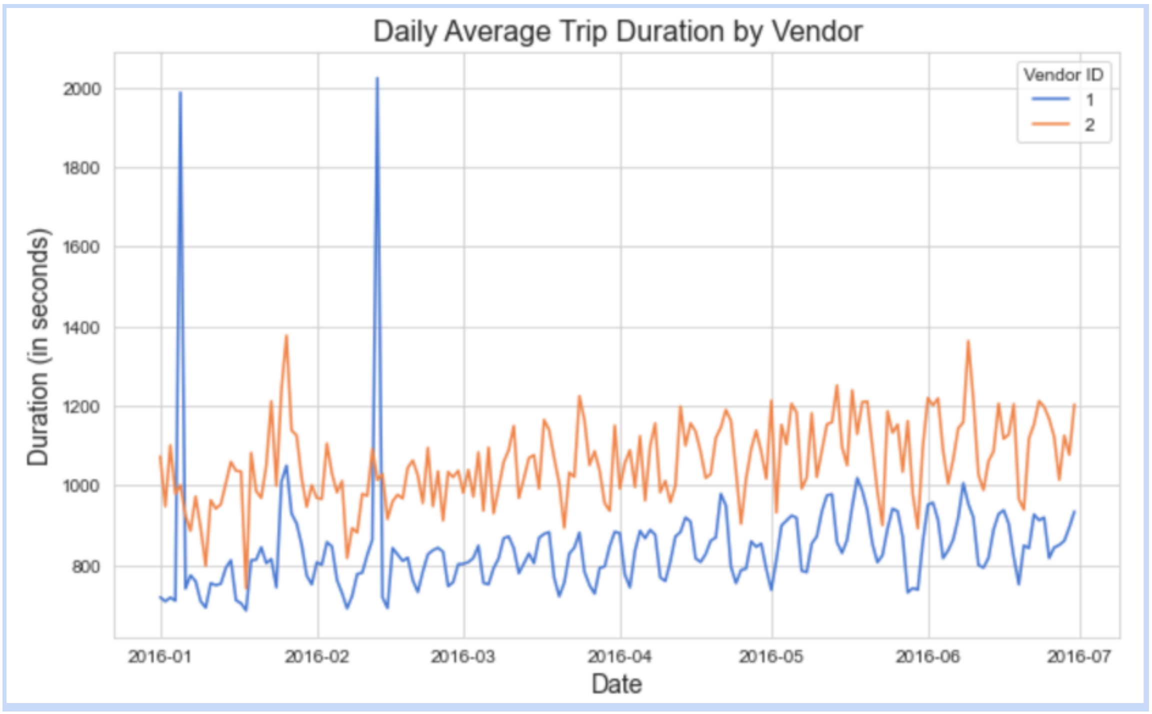


Figure 5. The line chart of daily average trip duration by date

5. Project Questions and Purpose

We conduct exploratory data analysis and have gained insights on patterns between taxi rides and feature variables. The main purpose of our project is to predict taxi trip durations in New York City with different regression models, including two linear models of linear regression and linear Support Vector Regression (SVR), and two non-linear models of Random Forest regression and XGboost. The best model would be selected based on the prediction accuracy on the evaluation set. Hyperparameter tuning and feature engineering will be performed on the best model to further boost its predictive performance.

6. Feature Engineering

Data is processed to generate new features by extracting useful information and to transform variables into a required format for model input.

Trip records where the passenger count equals 0 and the duration equals 1 are excluded from our analysis as those observations are not considered as revenue-generating taxi operations. Time information such as *day of week*, *hour in a day*, and *month* are extracted from the *pickup_datetime* column. A new feature, *distance*, is generated from the sphere distance of the trip pickup and dropoff geospatial coordinates using the haversine formula. To shorten the model computation runtime caused by large data size, 10% of the total data is randomly subsampled as our modelling database, which is further split into training and test set in proportion of 70:30. The modelling training set has 102,105 observations and the test set has 43,760 rows. The data split is done in temporal order to capture the time component of taxi business operation evolutions over time.

Dummy encoding is performed on the categorical variables for training and test sets separately, with one level dropped for each of them to prevent multicollinearity. The numeric variable, *distance*, is standardized to be approximately N(0,1) distribution to rescale its influence on the regression coefficient interpretation.

Our variable of interest, trip duration in seconds, is log transformed. Since the original trip duration distribution is right skewed with a large number of occurrences in low values as shown in Fig. 13(a), logarithm can normalize the variable distribution. To make the error measurement interpretable, the obtained error will be taken by exponents to convert back to the unit in seconds.

The modelling database has 37 features including the target variable of trip duration. Random forest regression model is used to select more insightful features based on the ranking of regression variance. As illustrated by the feature importance plot (Figure 6), the top 20 features are: trip distance, vendor type, passenger number (whether single, couple, or 5), and trip hour (whether early morning/afternoon/night, etc). The identified important features are fed into the machine learning models below.

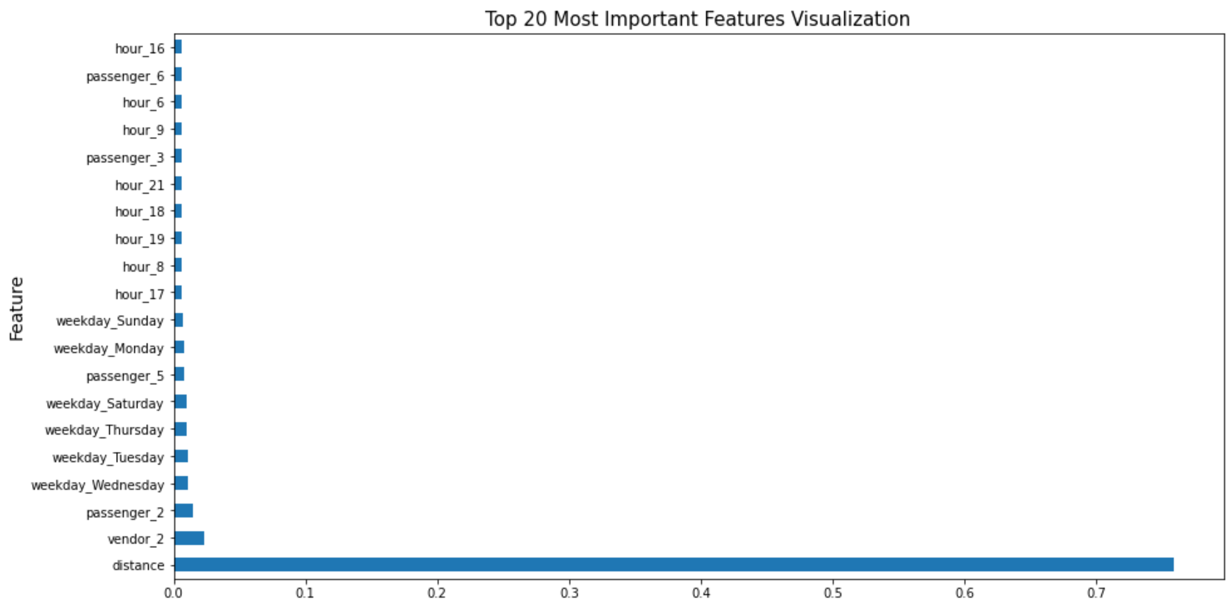


Figure 6. Top 20 feature importance plot by Random Forest Regression

7. Modeling Design

Since the target variable log(trip_duration) is continuous, we use 4 regression models, including Linear Regression, Support Vector Regression (SVR), Random Forest Regression, and Extreme Gradient Boosting Regression (XGBoost Regression), to make predictions on the trip duration based on the selected 20 features.

7.1 Linear regression

We start with the baseline model: Multiple Linear Regression that uses OLS (Ordinary Least Squares) methods, to capture the linear relationship between the target variable and features. There are 4 main assumptions of linear model:

1. The linear relationship between target variable and independent variables is appropriate.
2. Data points are independent of each other.
3. Errors are normally distributed with mean 0 and constant variance.
4. No multicollinearity among features.

7.1.1 Linear Regression Model with All Training Data

We fit the multiple linear regression using the training data and predict on the test set. We also check the assumptions to see if they are satisfied and if this model is appropriate. Firstly, we claim that there exist linear patterns in the data based on the exploratory analysis. Secondly, all the observations in the data take place at different locations and various time points, so the assumption of independence is satisfied. Then, to check the normality of errors, we plot the distribution of errors in Figure 20(a), and find that the errors roughly follow the normal distribution but are slightly left-skewed; meanwhile, the QQ plot of errors in Figure 20(b) shows that the middle points are on/near the QQ line, but there are large deviations around the tails. Therefore, the assumption of normality does not hold, and we need to remove the extreme values of trip durations to eliminate the impact of outliers on the linear model and see if this model can be improved.

7.1.2 Linear Regression Model with Outliers Removed

We compute the first quantile, third quantile and interquartile range (IQR = 25% - 75%) of log-transformed duration, and consider the values larger than Q3+1.5 IQR or smaller than Q1-1.5 IQR as outliers. We remove these extreme values and fit the model again.

Similarly, we check the assumptions of this model. We already check that linearity and independence hold. Then, for the normality assumption, the histogram of errors in Figure 21(a) shows that the errors are normally distributed, and the QQ plot of errors in Figure 21(b) shows that most points are on the QQ line, accordingly, we can assume that errors follow the normal distribution. Besides, for the constant variance assumption, we plot the plot of errors and the scatter plot of fitted values and residuals in Figure 21(c) and Figure 21(d) respectively, and find that there is no apparent pattern in these plots, so this assumption holds. Finally, we plot the correlation matrix among features in Figure 22 and find that most correlations are very small, so the assumption of no multicollinearity is satisfied.

7.2 Linear Support Vector Regression

Another linear machine learning model we have experimented with is the Linear Support Vector Regressor (SVR). Linear kernel is used so that a linear hyperplane is fit to the data while allowing for some error within defined range, which is in contrast to the OLS methods. The objective function of SVR is to minimize the L2 norm of the coefficient vector instead of the squared error as in OLS. The related error term for SVR, epsilon, can be tuned in its constraints to gain desired model accuracy. Another hyperparameter we have tuned is C. As C increases, our tolerance for points outside of epsilon also increases.

To find the best values of C and epsilon, we implemented grid search and 5-fold cross validation over the two hyperparameters. The best SVR can be used in further model performance analysis with the final hyperparameters, C=0.1, epsilon=0.5. The best cross-validated score of the best parameters is 30.65%.

7.3 Random Forest Regression

As the linear models fail to deliver satisfactory performance in duration predictions, we switch to non-linear models to allow for more flexibility in fitting data. One non-linear regression model we have conducted is the Random Forest Regressor, which builds multiple decision trees and merges them together to get a more accurate prediction. The objective of adjusting hyperparameters in random forest is either making the model faster or increasing the predictive power of the model. The main hyperparameters we have tuned include n_estimators, max_depth, and min_samples_split. The hyperparameter n_estimators is the number of trees in the forest. Generally, when the number of trees increases, the predictions are more stable and the model performance is better, while the computation will be slowed down. The hyperparameter max_depth represents the maximum number of levels in each decision tree. The deeper the tree is, the more splits it has and more information about data is captured. The third tuned hyperparameter min_samples_split refers to the minimum number of samples required to split an internal node before it is split. When this parameter increases, the trees in the forest become more constrained since more samples have to be considered at each node.

After implementing grid search and 5-fold cross validation to tune the hyperparameters, the result of the best hyperparameters is, max_depth=10, min_samples_split=20, n_estimators=100. The mean cross-validated score of the best hyperparameters of random forest regressor is 51.86%. We will use the model with the best hyperparameters in further model performance analysis.

7.4 XGBoost Regression

Extreme Gradient Boosting is a type of ensemble model and it uses an optimized distributed gradient boosting algorithm to minimize the training loss (including regularization) and provides solutions in a fast and accurate way.

7.4.1 XGBoost Regression with 20 Selected Features

We set the booster type as the default ‘gbtree’ to make this a tree-based algorithm, so that the feature importance can be visualized. We set the objective as ‘reg:squarederror’ for the purpose of regression, then we fit the model with the other default hyper-parameter values.

7.4.2 XGBoost Regression with All Features

To see if the XGBoost regression can be further improved, all 37 features are included in the model and the model is re-fitted. Also, we tune the hyper-parameters using grid search with five-fold cross validation for a more stable model performance. We choose the best values for the following hyper-parameters:

- Learning rate: Controls how fast a model learns; we set it as 0.1.
- Reg_alpha: L1 regularization term, we set it as 0.25.
- Max_depth: Maximum depth of a tree; we set it as 3.
- N_estimators: Number of trees, we set it as 150.

We fit the model again with the best hyper-parameter set, and compare the error for the test set with that for the other models to see if this gives a better fit.

8. Model Analysis

Besides, since the target variable is log transformed, we take the exponential on RMSE (actually RMSLE here due to the log transformation) to convert the unit measurement of error back to seconds for an intuitive performance interpretation and model comparison.

8.1 Linear Regression

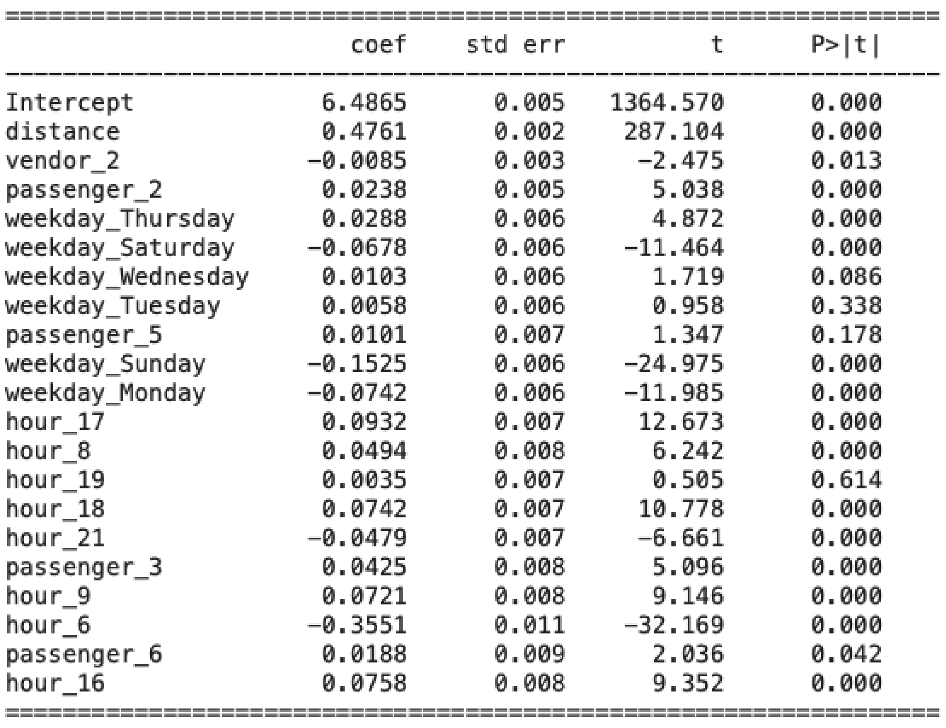
8.1.1 Linear Regression Model with All Training Data

This test error and training error in seconds for this model are 1.898 and 1.843 respectively, which are similar, so there is no overfitting problem. However, since the errors are not normally distributed, this is an invalid model and we will not interpret the results of this model.

8.1.2 Linear Regression Model with Outliers Removed

This model gives a high error in seconds of 1.896 and a low R2 score of 0.397, which further indicates a poor performance of the linear model. To make the model more robust, we also do the 5-Fold cross validation. The mean accuracy score is 0.454 with a standard deviation of 0.006.

Then, we compute the intercept and estimated coefficients of the fitted linear model in Table 1. The intercept is 6.49, which is the trip duration (in log form) when all features have values of 0. And for instance, the estimated coefficient of distance is 0.48, which shows that the duration will increase by 1.616 seconds (exp(0.48) = 1.616) as distance increases by 1 km. Also, the p-values for features are very small (<0.05), except that *'weekday_Wednesday', 'weekday_Tuesday', 'passenger_5'* and *'hour_19'* have relatively larger p-values, and this indicates that these features probably do not significantly affect the trip duration.



```
==================================================================
                         coef    std err          t      P>|t|
------------------------------------------------------------------
Intercept              6.4865      0.005   1364.570      0.000
distance               0.4761      0.002    287.104      0.000
vendor_2              -0.0085      0.003     -2.475      0.013
passenger_2            0.0238      0.005      5.038      0.000
weekday_Thursday       0.0288      0.006      4.872      0.000
weekday_Saturday      -0.0678      0.006    -11.464      0.000
weekday_Wednesday      0.0103      0.006      1.719      0.086
weekday_Tuesday        0.0058      0.006      0.958      0.338
passenger_5            0.0101      0.007      1.347      0.178
weekday_Sunday        -0.1525      0.006    -24.975      0.000
weekday_Monday        -0.0742      0.006    -11.985      0.000
hour_17                0.0932      0.007     12.673      0.000
hour_8                 0.0494      0.008      6.242      0.000
hour_19                0.0035      0.007      0.505      0.614
hour_18                0.0742      0.007     10.778      0.000
hour_21               -0.0479      0.007     -6.661      0.000
passenger_3            0.0425      0.008      5.096      0.000
hour_9                 0.0721      0.008      9.146      0.000
hour_6                -0.3551      0.011    -32.169      0.000
passenger_6            0.0188      0.009      2.036      0.042
hour_16                0.0758      0.008      9.352      0.000
==================================================================
```

Table 1: The estimated coefficients & p-values of the fitted linear model without outlier

To visualize the model performance, we plot the distributions of predicted values and actual values in Figure 7. We find that the predicted values are right-skewed and there are much more trips with duration (in log form) between 6 and 6.5, while the actual values are normally distributed. Therefore, their distributions are totally different, and the linear model does not fit well and we need to try other regression models to see if they can give a better fit.

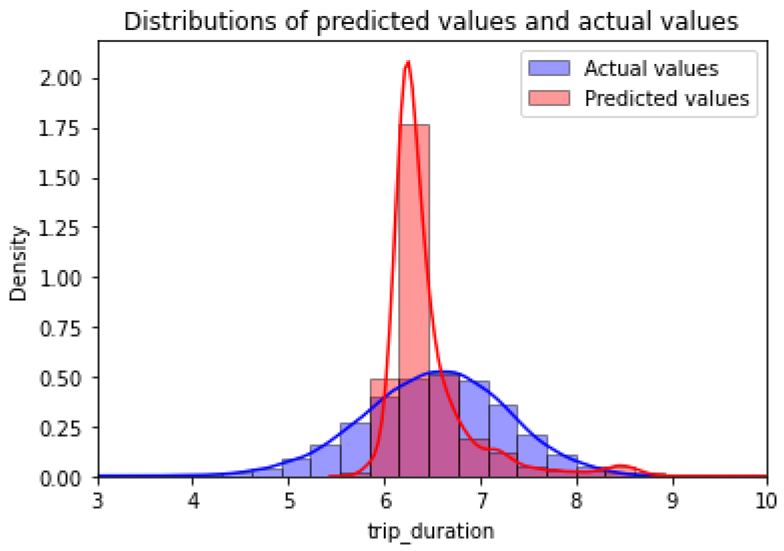


Figure 7. Linear Regression: Density plot of predicted values Vs.actual values

8.2 Linear Support Vector Regression

We use the error in seconds as the measurement to evaluate the model performance, which is defined as the exponential function of the Root Mean Squared Error (RMSE). After tuning the hyperparameters, the best SVR model is built with final parameters, C=0.1, epsilon=0.5. The training error is 1.8437, which is a little lower than the test error at 1.8988.

The SVR model performance is shown in the density plot of actual values vs. predicted values (Fig. 8). The actual values of trip_duration are normally distributed, while the distribution of predicted values is slightly right skewed. Besides, the density of predicted values of trip_duration are much higher than that of actual values when log(trip_duration) is between 6 and 7, which further shows poor fitting.

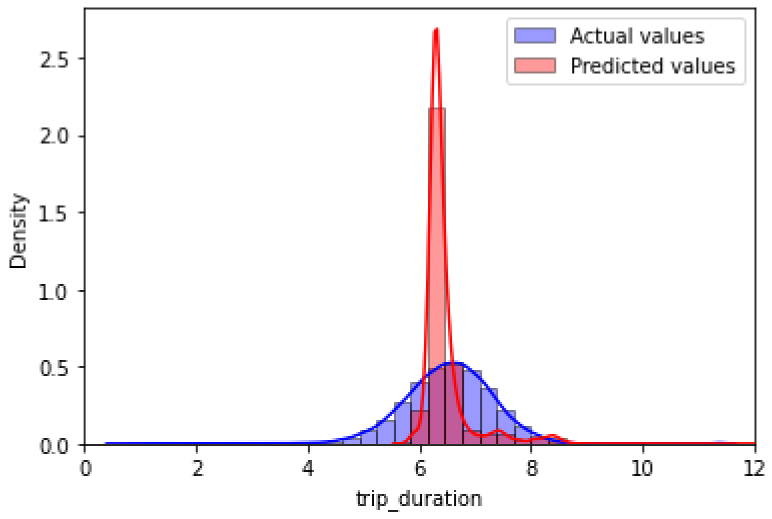


Figure 8. SVR: density plot of predicted values Vs.actual values

8.3 Random Forest Regression

After tuning the hyperparameters including n_estimators, max_depth and min_samples_split, we fit the final model to the training data and test data. The values of RMSE on training data and test data are 0.4666 and 0.5274 separately. The error on training data is 1.5945, smaller than error on test data at 1.6946. Since the values of training error and test error are close, there is no model overfitting with the random forest regressor.

From Figure 9 below, we can directly see that the density distribution curves of actual values and predicted values of log(trip_duration) are more similar, compared with those of linear models we have implemented above. The random forest regressor fits better than linear regressors and is less prone to overfitting than the decision tree model.

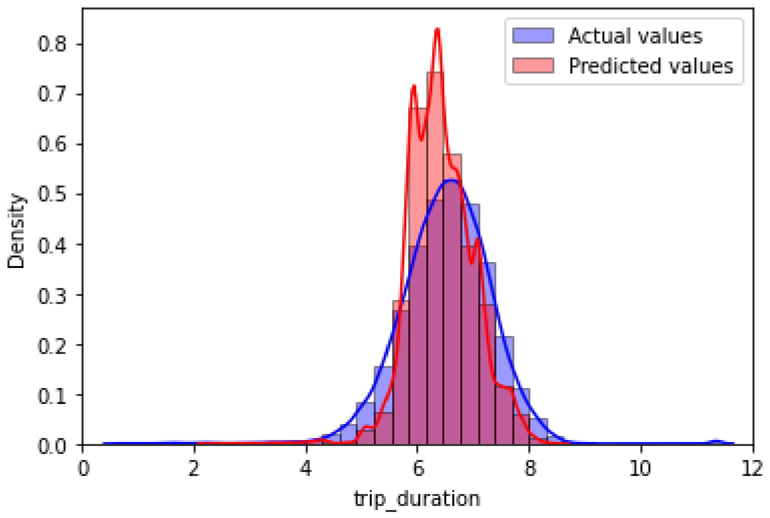


Figure 9. Random Forest Regressor: Density plot of predicted values vs.actual values

8.4 XGBoost Regression

8.4.1 XGBoost Regression with 20 Selected Features

The error in seconds for the test set is 1.686, which is much lower than that for the other models. And, the error in seconds for the training set (1.610) is similar to that for the test set, so no overfitting exists in the model.

8.4.2 XGBoost Regression with All Features

After tuning the hyper-parameters, the model gives a best accuracy score of 0.654. The test error in seconds is 1.657, which is the lowest among all regression models (including the XGBoost model with selected features), so this full model performs better.

To visualize the model performance, we plot a bar chart of the feature importance. As shown in Figure 10, the importance of *’distance’* in the model outweighs the others, *‘vendor_2’* and *‘hour_2’* are also among the most important features, while *‘passenger_6’*, *‘flag_Y’* and *‘passenger_4’* are the least important features. Besides, as Figure 11 shows, both the predicted values and actual values are normally distributed and most of their distributions overlap, which also reveals that this model gives the highest accuracy and performs the best.

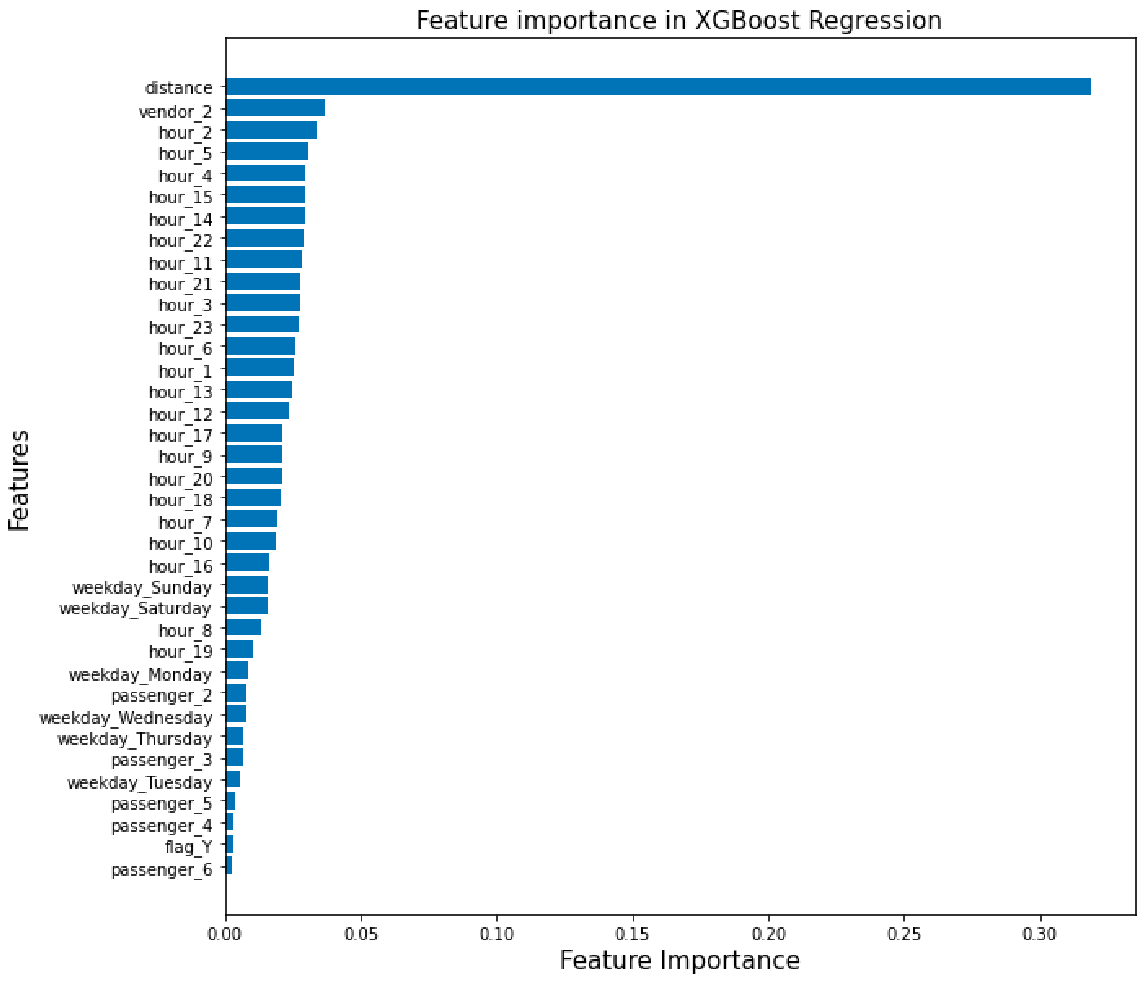


Figure 10. Feature importance plot in XGBoost Regression

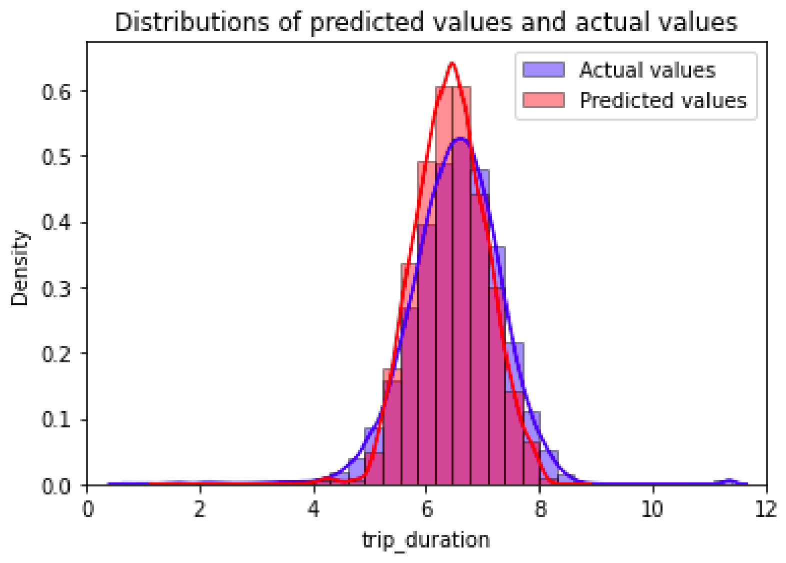


Figure 11. The density plot of predicted values Vs.actual values for XGBoost Regression

Moreover, to know which feature contributes to the highest error rate and to further refine our modes in the next steps, we plot the error in seconds for the values in each feature. For the numeric feature *'distance'*, we divide the values into 4 categories based on the quantiles. In Figure 12, we can see that the feature 'distance' has the smallest error, and *'hour_17', 'hour_16', 'hour_18'* and *'vendor_2'* have the highest errors. This shows that XGBoost works well with the feature *'distance'*, but it cannot predict the durations well for the trips that happen during 4pm-6pm or are recorded by vendor 2, so we may consider using other regression models to predict these kinds of trips to improve the overall prediction accuracy.

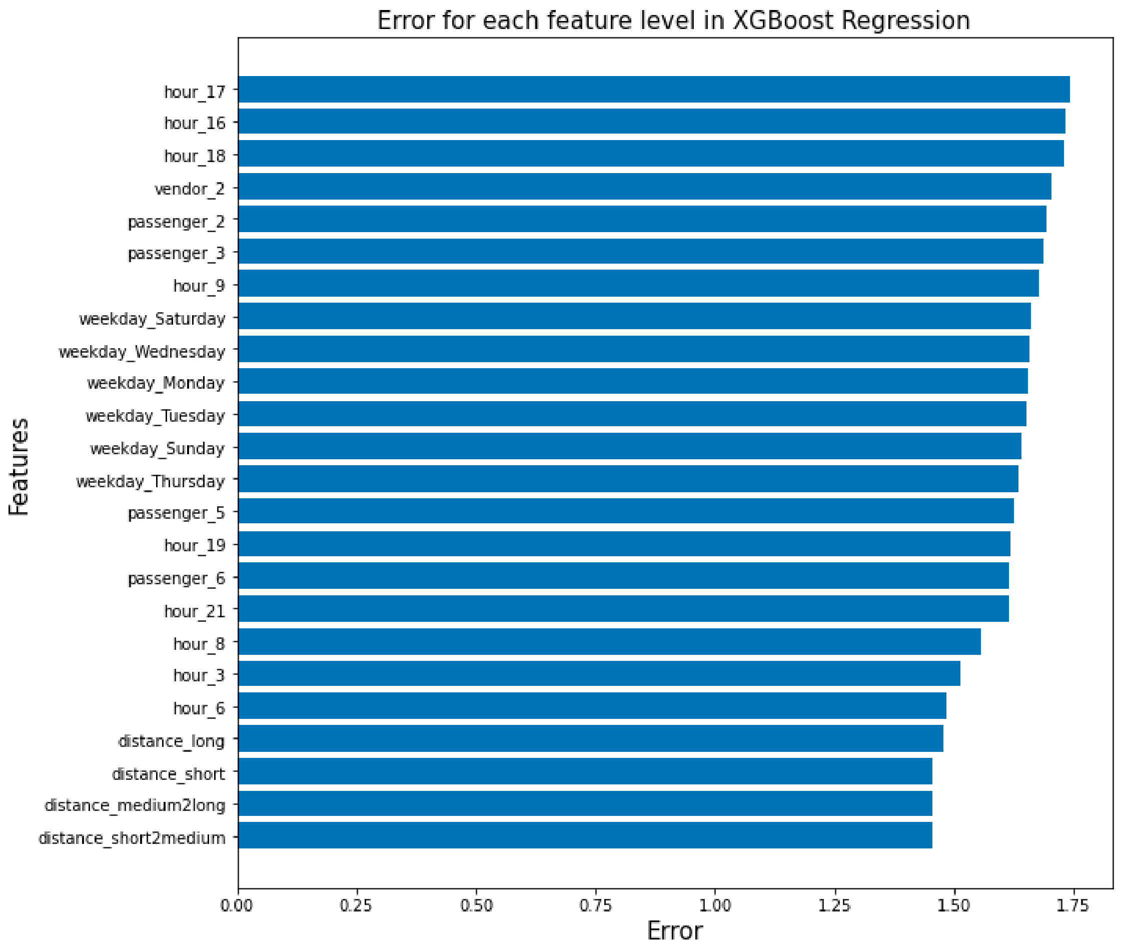


Figure 12: Error for each feature value in XGBoost Regression

9. Discussion

9.1 Model Comparison

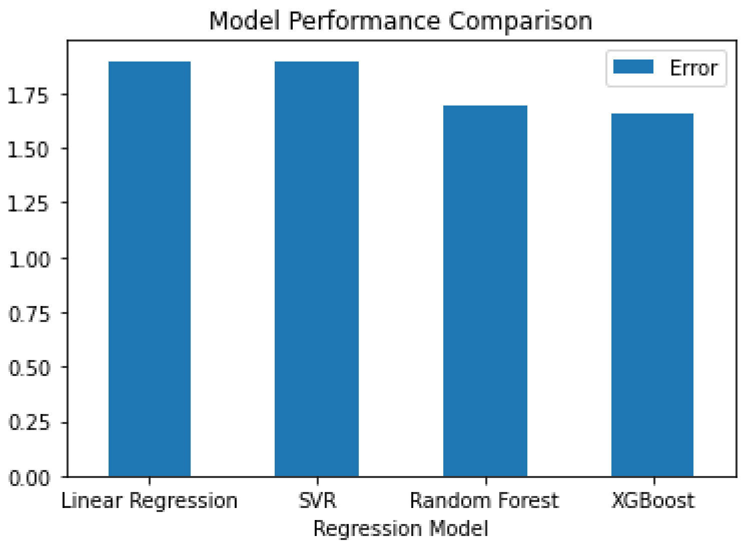


Figure 13. Error comparison plot for four models

From the model performance comparison plot in Figure 13, we can see that nonlinear models have much lower error in seconds than the linear models, and in particular, XGBoost has the lowest error in seconds on the test set at 1.791.

The linear regression is parisonmous and straightforward. In general, it is able to delineate linear relationships well. However, it fails to incorporate the intrinsic non-linearity of our taxi trip data and this is probably why it underperforms in our case.

Although Random Forest Regressor has higher accuracy compared with linear regression models, one of its disadvantages is that the computation is complex, resulting in a relatively longer runtime.

XGBoost regression has quick runtime, outperforms the other models in our data and it scales well on large dataset as well as on small data size (because we only took 10% of the entire data). Also, the feature importance can be visualized and is interpretable, in contrast to most black-box machine learning models. But it can only deal with numerical values, so we need to do one-hot encoding to convert the categorical features to numerical during data preprocessing. And, the model performance heavily depends on careful hyperparameter tuning.

9.2 Business Implication

This project can generate business returns to the stakeholders and users. Its model predicted trip durations can be cross-compared with historic average data for delay classification. And subsequently, the taxi company can inform passengers of potential delays in advance of their trip. Additionally, based on the delay analytics data, companies can identify patterns and anomalies in geographic and time dimensions to re-route for optimal journeys so as to increase operational efficiency and business revenue.

10. Self Assessment

Group 8 has team members Jia Jia Ji, Yingying Zhou, and Zhengzheng Wang. Project tasks are equally assigned to each member and we collaborate closely with each other in informing our individual work progress and communicating potential problems encountered. Weekly meetings are held to update our project workflow and schedule our next-stage goals and timelines. Project coding and documentation collaboration is realized through online sharing applications such as Google docs, colab, and slides. Each of us is responsible for documenting and presenting our own deliverables in the report-writing, slides-making and presentation.

Jia Jia Ji is mainly responsible for the Dataset Description part, the Linear Regression model and XGBoost Regression parts, including the model design, hyper-parameter tuning, model performance analysis and strengths & limitations. Also, she is actively engaged in the group meetings about exploring data, model analysis and presentation preparation. She attends the in-class presentations and Q&A sessions.

Yingying Zhou is mainly responsible for Exploratory Data Analysis in feature visualizations and data preprocessing in terms of feature engineering, transformation, and feature selection with Random Forest regression model. She also plays a proactive role in facilitating the overall teamwork engagement in meetings, contributing to model designs and business ideations, and presenting during the in-class live presentation and Q&A sessions.

Zhengzheng Wang is mainly responsible for project purposes, Exploratory Data Analysis in feature visualizations, and implementation and analysis of Random Forest Regression model as well as SVR model, including the model design, hyperparameter tuning, and model performance analysis. She also engaged actively in the group meetings, presentation preparation, presentation video editing and Q&A sessions.

Reference

1. Kaggle 2017, New York City Taxi Trip Duration, NYC Taxi and Limousine Commission (TLC), https://www.kaggle.com/c/nyc-taxi-trip-duration/data.
2. New York City Taxi and Limousine Commission 2016 Annual Report, https://www1.nyc.gov/assets/tlc/downloads/pdf/annual_report_2016.pdf
3. Hyperparameter Tuning the Random Forest in Python. (2018). Towardsdatascience. https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
4. An Introduction to Support Vector Regression (SVR). (2020). Towardsdatascience. https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2

Appendices:

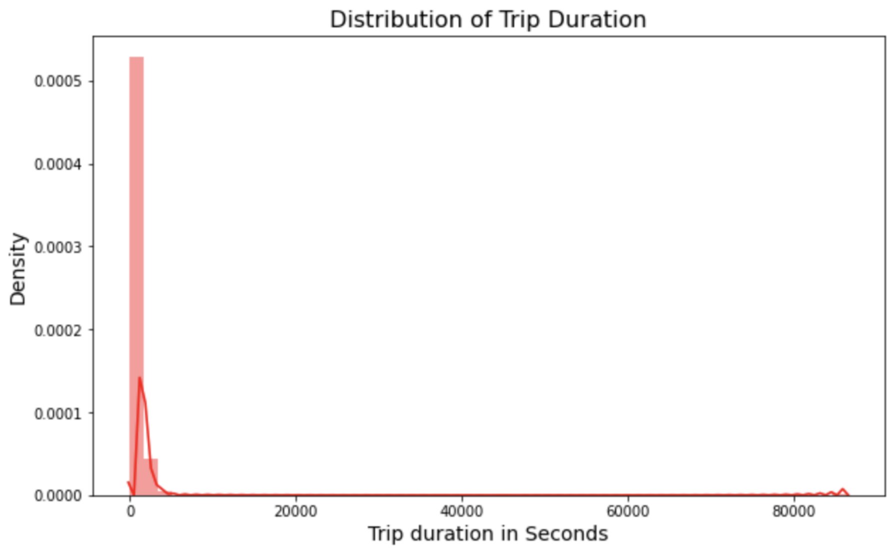


Figure 14(a): The histogram of *trip_duration*

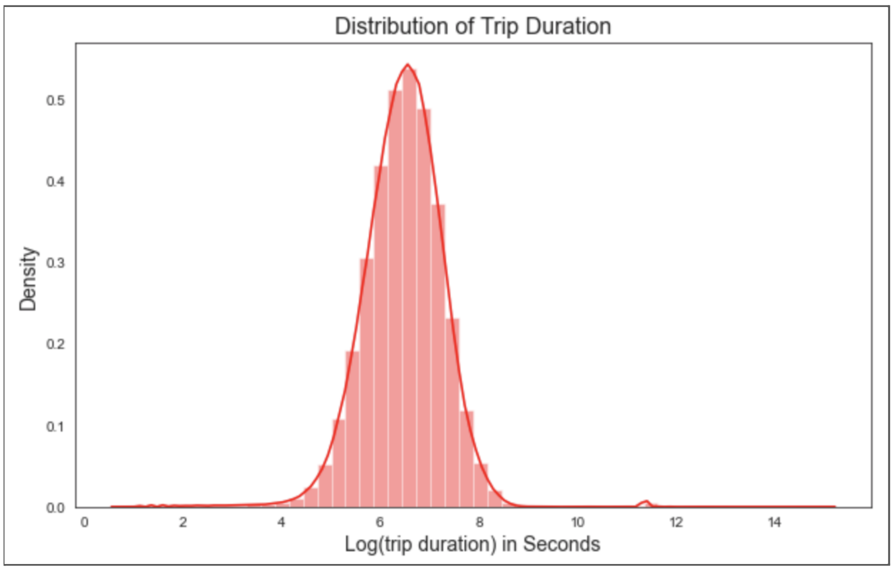


Figure 14(b): The histogram of log(*trip_duration*)

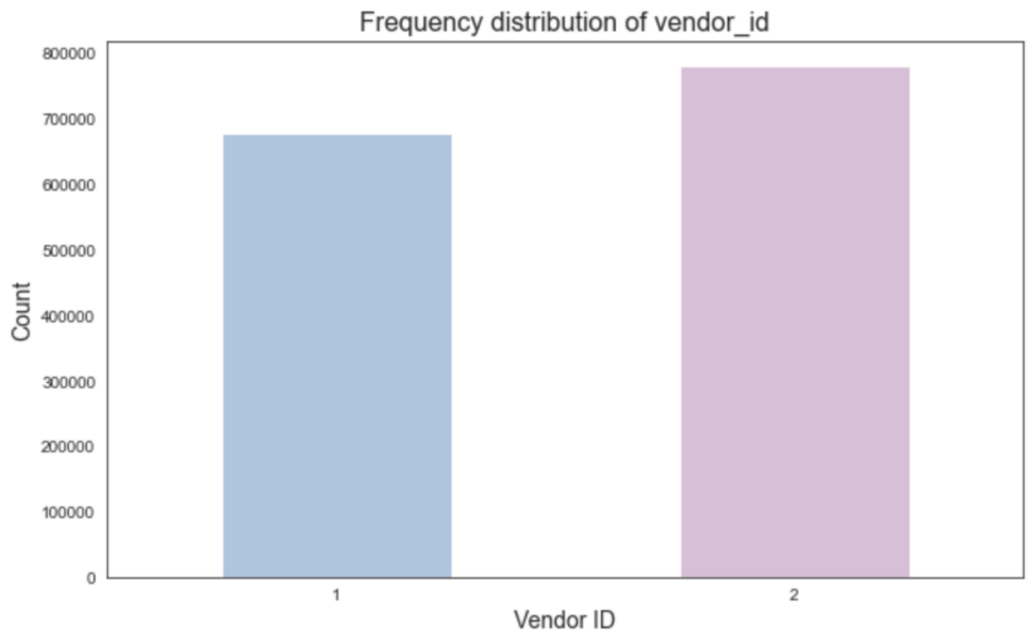


Figure 15: The frequency distribution bar chart of *vendor_id*

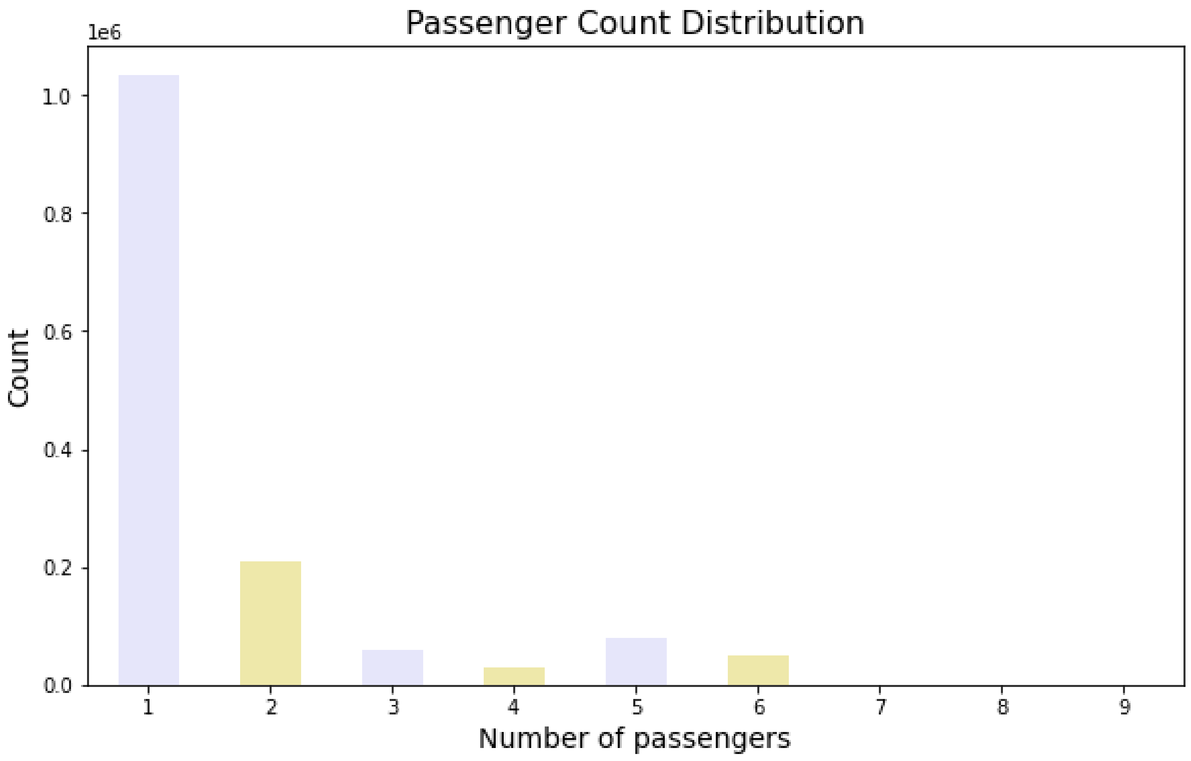


Figure 16: The frequency distribution bar chart of *passenger_count*

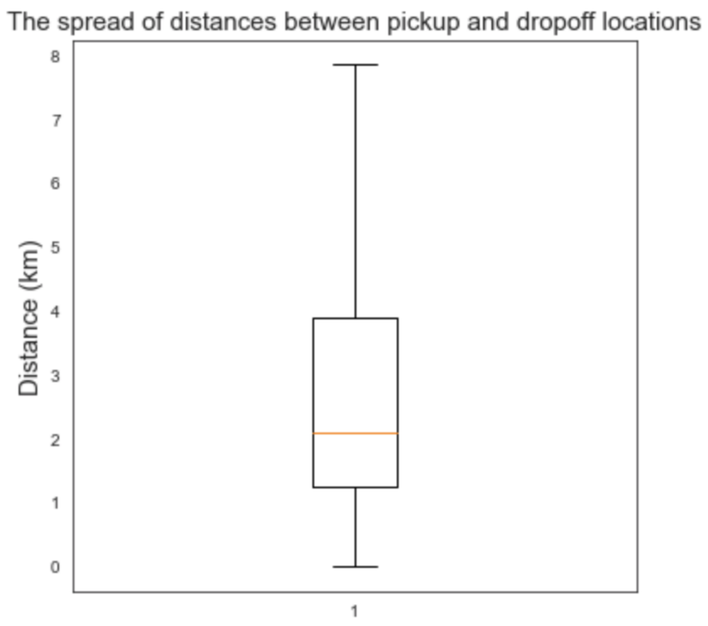


Figure 17: The boxplot of distances between pickup and dropoff locations

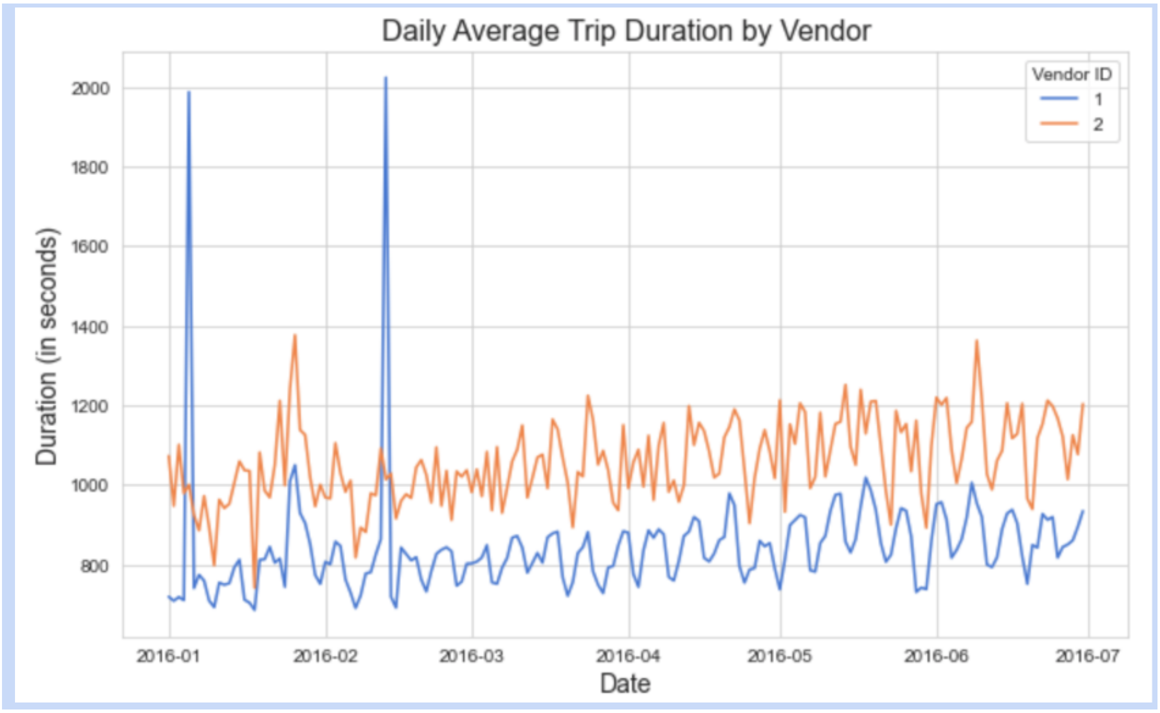
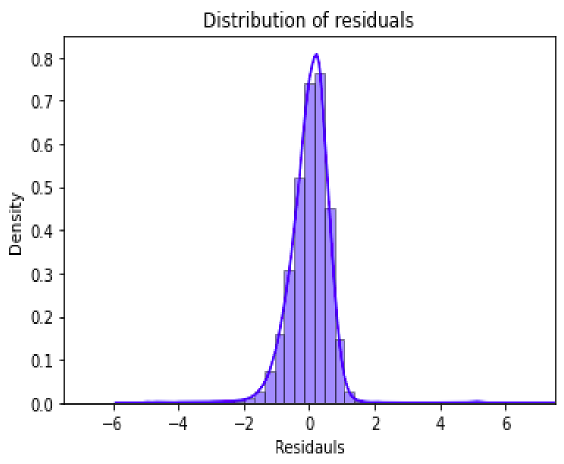
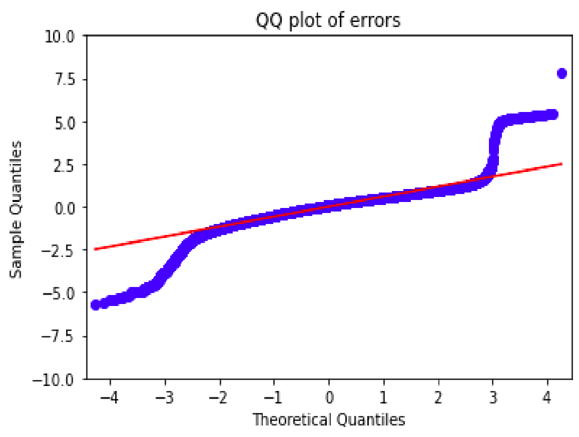


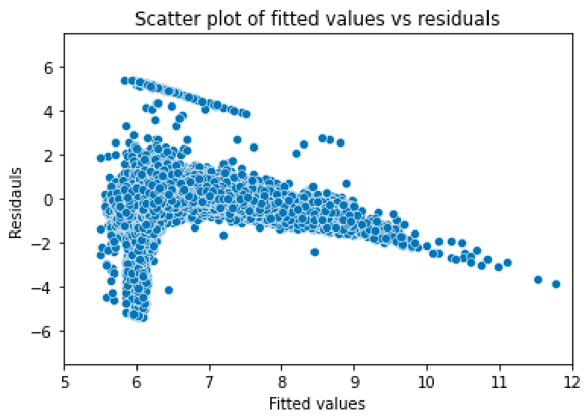
Figure 19: The line plot of average_trip_duration vs. pickup_date for two vendors



(a) The histogram of residuals

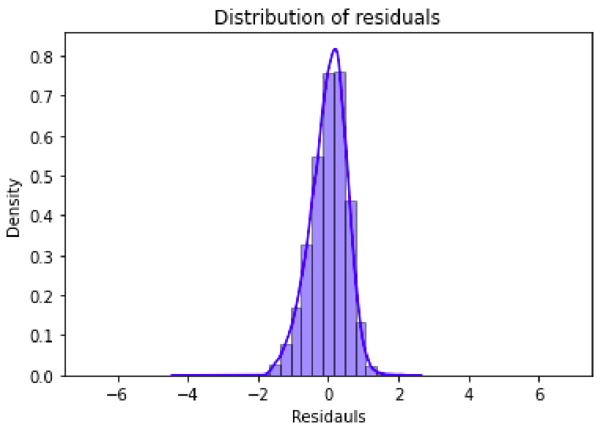


(b) The QQ plot of residuals

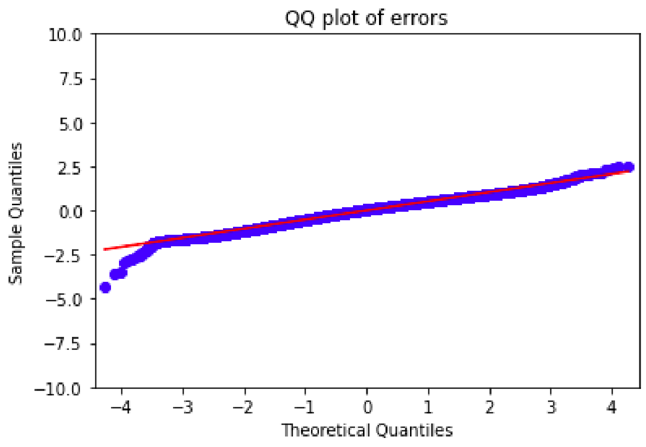


(c) The scatter plot between fitted values and residuals

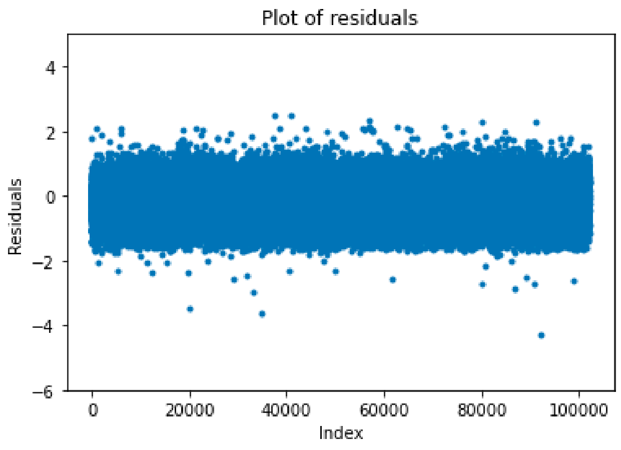
Figure 20: Model assumptions of linear model with all training data



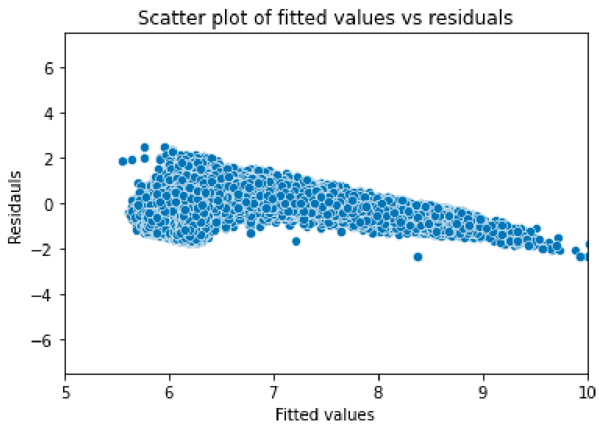
(a) The histogram of residuals



(b) The QQ plot of residuals



(c) The plot of residuals



(d) The scatter plot between fitted values vs residuals

Figure 21: Model assumptions of linear model with outliers removed

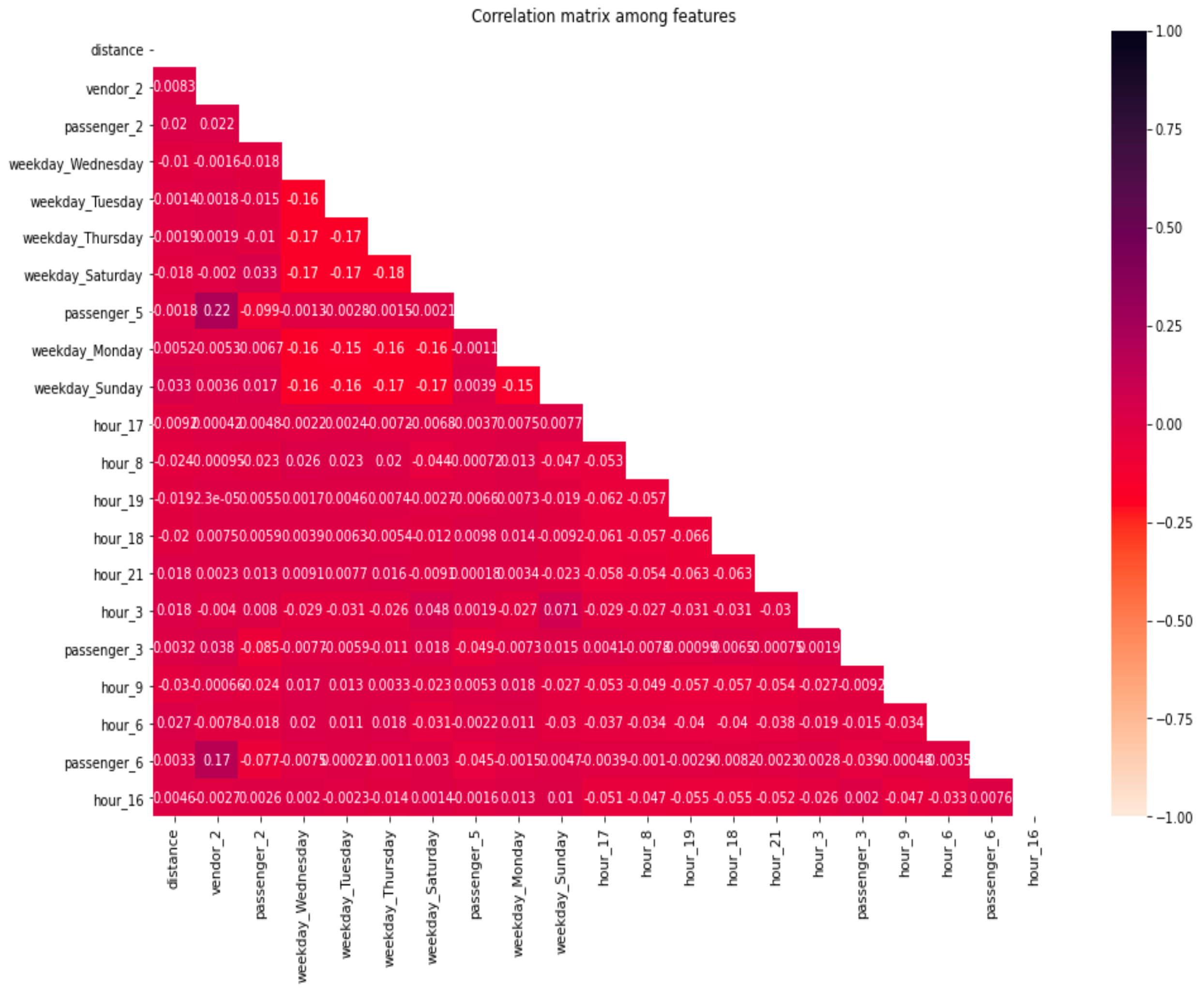


Figure 22: Correlation matrix among features

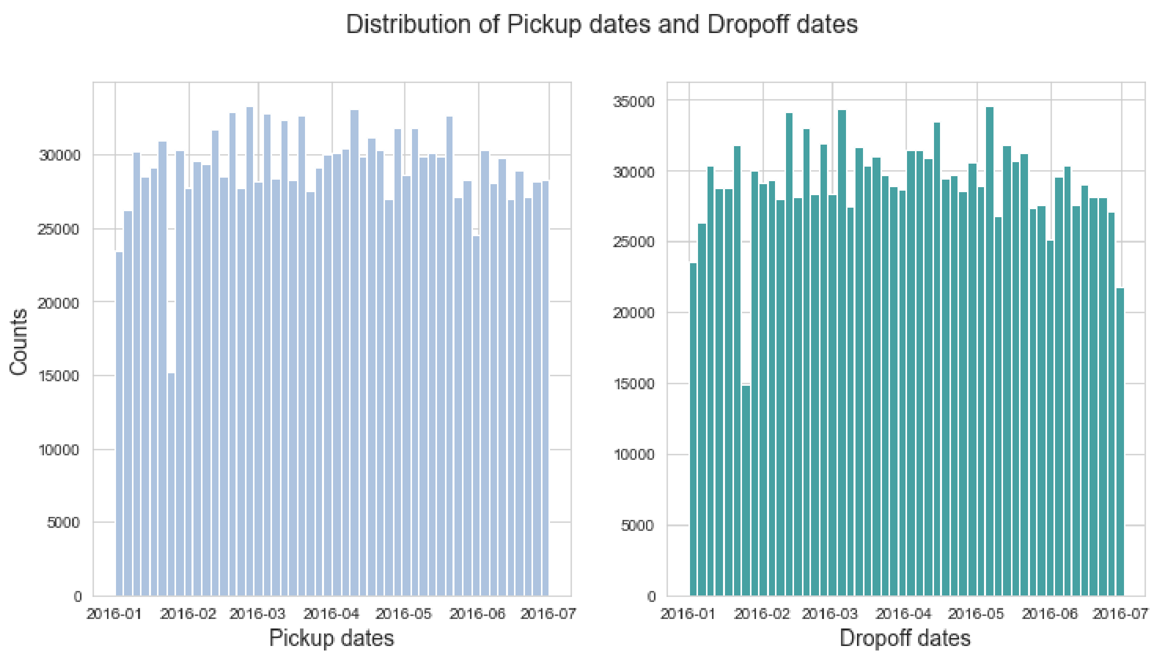


Figure 23: Distributions of pickup date and dropoff date