New York City Taxi Trip Duration

Tianjiao Wang

Beijing Technology and Business University

2023-12-02

Overview

Introduction

Introduction

The project will build a model that predicts the total ride duration of taxi trips in New York City. The primary dataset is one released by the NYC Taxi and Limousine Commission, which includes pickup time, geo-coordinates, number of passengers, and several other variables. Accordingly, this project problem is taxi trips duration, which is a outlier detection.

- Data loading and overview
- Data cleaning
- Features engineering
- Model selection
- Hyperparameters tuning
- Training and predictions

This project can predict the duration of each trip in the test set , after model selecting, and Hyperparameters tuning.

Data loading and overview

- Data loading - First 5 lines
  - Data overview: I take a overview of the type and amount and other information of df and test data.

Colonne	Description
id	a unique identifier for each trip
vendor_id	a code indicating the provider associated with the trip record
pickup_datetime	date and time when the meter was engaged
dropoff_datetime	date and time when the meter was disengaged
passenger_count	the number of passengers in the vehicle (driver entered value)
pickup_longitude	the longitude where the meter was engaged
pickup_latitude	the latitude where the meter was engaged
dropoff_longitude	the longitude where the meter was disengaged
dropoff_latitude	the latitude where the meter was disengaged
store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server (Y=store and forward; N=not a store and forward trip)
trip_duration	duration of the trip in seconds

Figure 1: overview of the data

Data Cleaning

Data cleaning

- I do the data clearning to check the duplicated and missing values and deal with outliers.

```
id                    0
vendor_id             0
pickup_datetime       0
dropoff_datetime      0
passenger_count       0
pickup_longitude      0
pickup_latitude       0
dropoff_longitude     0
dropoff_latitude      0
store_and_fwd_flag    0
trip_duration         0
dtype: int64
```

Figure 2: No duplicated or missing values

Visualize outliers

- There are outliers. I can't find a proper interpretation and it will probably damage our model, so I choose to get rid of them.

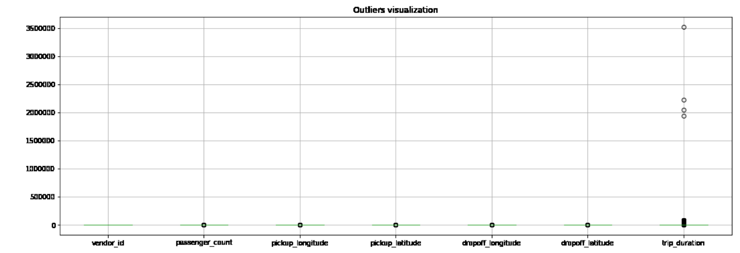

Figure 3: boxplot for trip-duration

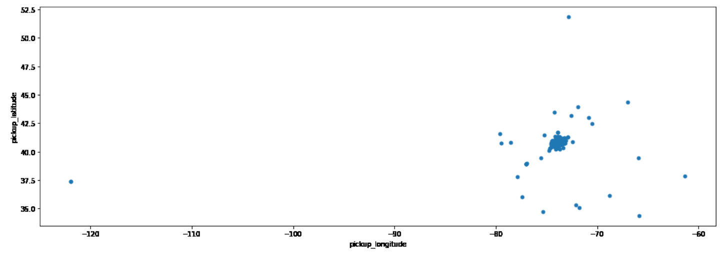

Figure 4: pickup-longitude

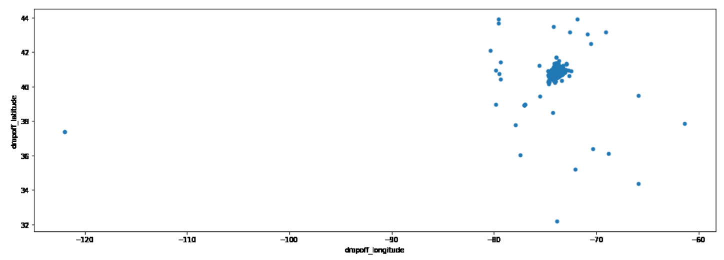

Figure 5: dropoff-longitude

Outlying Degree Scoring

- In this step, I only keep trips that lasted less than 5900 seconds, and only keep trips with passengers, and remove position outliers(pickup-longitude > -100, pickup-latitude < 50; dropoff-longitude < -70 and dropoff-longitude > -80, dropoff-latitude < 50).

Features engineering

Target

- We take a look of the distribution of trip-duration value.

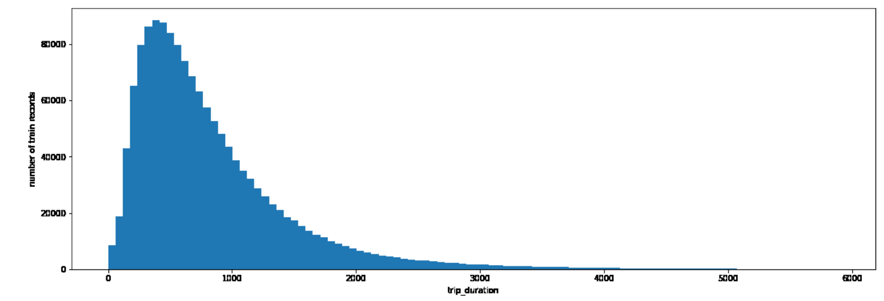

Figure 6: trip-duration

- The distribution is right-skewed so we can consider a log-transformation of trip-duration column.

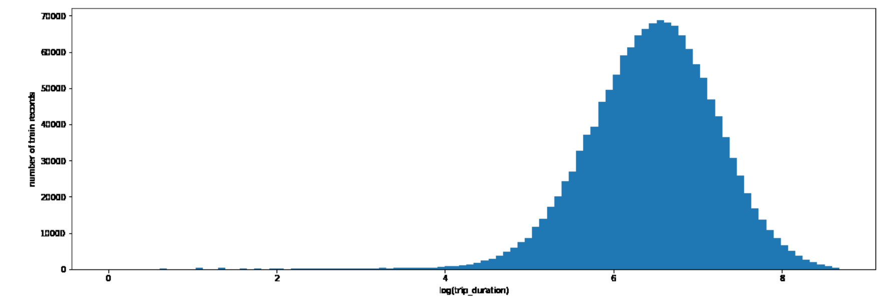

Figure 7: log-transformation of trip-duration column

Deal with data

- Deal with categorical features
    - One-hot encoding binary categorical features
- Deal with dates
    - Datetyping the dates
    - Date features creations and deletions
- Distance and speed creations
    - Function aiming at calculating distances from coordinates
    - Add distance feature
    - Function aiming at calculating the direction
    - Add direction feature
    - Visualize distance outliers
    - Remove distance outliers
    - Create speed feature
    - Visualize speed feature

Figure 8: boxplot for distance

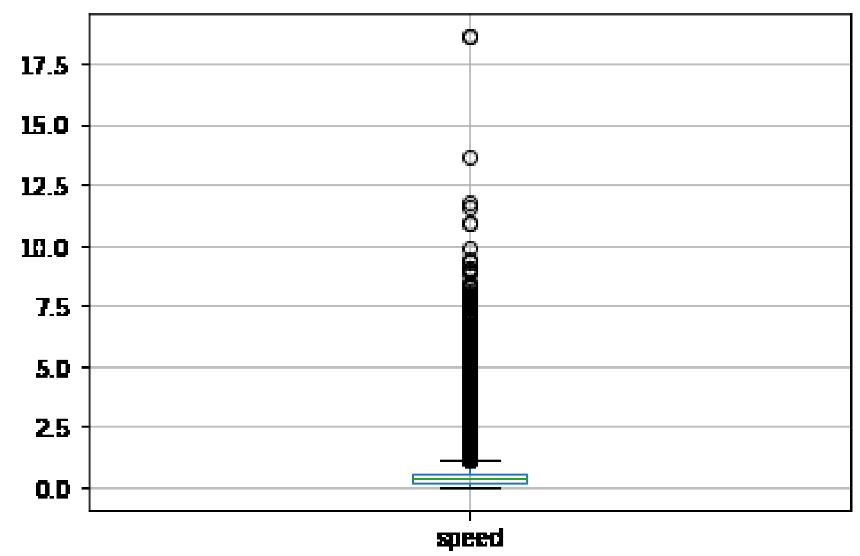

Figure 9: boxplot for speed

- Correlations between variables

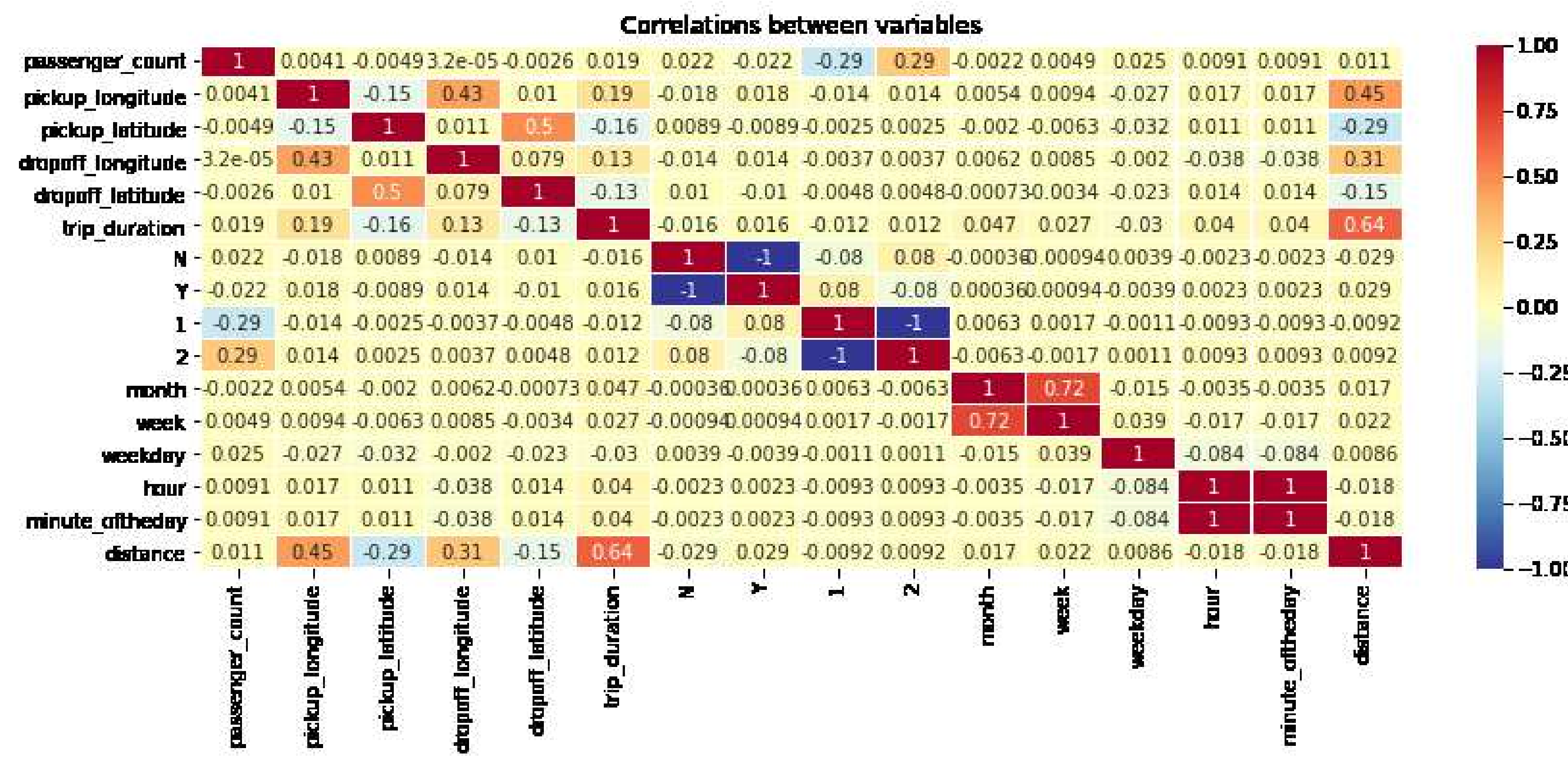

Figure 10: correlations between variables

Model selection

Model selection

- Split
  - Split the labeled data frame into two sets: features and target
  - Split the labeled data frame into two sets to train then test the models
- Metrics
  - For this specific problematic, we'll measure the error using the RMSE (Root Mean Squared Error).
- Models
  - Try GradientBoosting
  - Try RandomForest
  - Try LightGBM

LightGBM is blazingly fast compared to RandomForest and classic GradientBoosting, while fitting better. It is our clear winner.

- Cross-validation

Our LightGBM model is stable.

Hyperparameters tuning

Hyperparameters tuning

- Hyperparameters tuning using RandomizedSearchCV
- Test the following parameters

Training and predictions

Training and predictions

- Training on all labeled data using the best parameters in hyperparameters tuning
- Training on all labeled data using the best parameters (sklearn API version)
- Training on all labeled data using the best parameters
  - CPU times: user 9min 22s, sys: 8.74 s, total: 9min 31s
    Wall time: 4min 50s
- Make predictions on test data frame
- Create a data frame designed a submission on Kaggle
- Create a csv out of the submission data frame

	id	trip_duration
0	id3004672	716.070826
1	id3505355	672.125770
2	id1217141	455.368356
3	id2150126	938.637832
4	id1598245	354.432595

Figure 11: predict-result

Conclusion

Conclusion

- The dataset provided has very low missing values although observations provided cover only two vendors (two taxi companies) and also the data provided is across a single year and only six months of the year (fall data is missing)

- We can see how the taxis in a city like New York is so much location and time based and it's usage is more or less predictable on the basis of these factors (among others).

Questions?

Contact Information

Tianjiao Wang

Business School

Beijing Technology and Business University, China

WANGTIANJIAOJ@GMAIL.COM

BEIJING TECHNOLOGY AND BUSINESS UNIVERSITY