

New York City Cab Trip Duration

GUIDED BY

Prof. G.Srinivasaraghavan

SUBMITTED BY

Harshit Joshi : MT2016059

Nitesh Kataria : MT2016097

Palak Langer : MT2016100



December 2017

TABLE OF CONTENTS

Abstract.....	3
Chapter 1: Introduction.....	4
1.1 Problem Statement	
Chapter 2: Dataset and Evaluation Metric.....	5
2.1 Dataset	
2.2 Evaluation Metric	
Chapter 3: Data Visualization.....	7
Chapter 4: Model.....	12
4.1 Regression Model	
4.2 Feature Engineering	
4.3 Cluster Visualization	
4.4 Getting Important features	
4.5 Hyperparameter tuning	
Chapter 5: Final Result.....	18
References	

ABSTRACT

This report presents an evaluation study of various models and features to predict the duration of a taxi trip in New York City. We investigate the optimal model and hyperparameters for this problem. We show that Decision Trees based models, especially Gradient Boosting are effective in learning the predictive patterns from the given features. We examine the effective ways to represent the features and also study the importance of various features in our predictive models. We achieved lowest RMSLE using the Gradient Boosting model. Looking at results, we can claim that though the model was not completely successful in predicting the exact duration, it can still be used as an effective tool to give an approximate estimate.

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

To build a model that predicts the total ride duration of taxi trips in New York City. Your 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.

CHAPTER 2

DATASET AND EVALUATION METRIC

2.1 Dataset

Primary :

- Input.csv : NYC Taxi and Limousine Commission dataset
published on kaggle

The dataset was taken from 2016 NYC Yellow Cab Trip Record Data. It has following columns:

- *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* - 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

External :

- Weather.csv : retrieved from National weather Service Forecast Office
Link :<http://w2.weather.gov/climate/xmacis.php?wfo=okx>
- fastest_routes.csv : OSRM api was used to extract information about the fastest routes for each data point.

The Open Source Routing Machine or OSRM is a C++ implementation of a high-performance routing engine for shortest path.

2.2 Evaluation Metric

The evaluation metric for this competition is [Root Mean Squared Logarithmic Error](#).

The RMSLE is calculated as

$$\epsilon = \sqrt{\frac{1}{n} \sum_{i=1}^n (\log(p_i + 1) - \log(a_i + 1))^2}$$

Where:

ϵ is the RMSLE value (score)

n is the total number of observations in the (public/private) data set,

p_i is your prediction of trip duration, and

a_i is the actual trip duration for i .

$\log(x)$ is the natural logarithm of x

Since RMSLE is used as evaluation metric, we took log of time-duration and then we were able to use RMSE as default metric, later on before submission time-duration values were inverse log transformed for submission.

CHAPTER 3

DATA VISUALIZATION

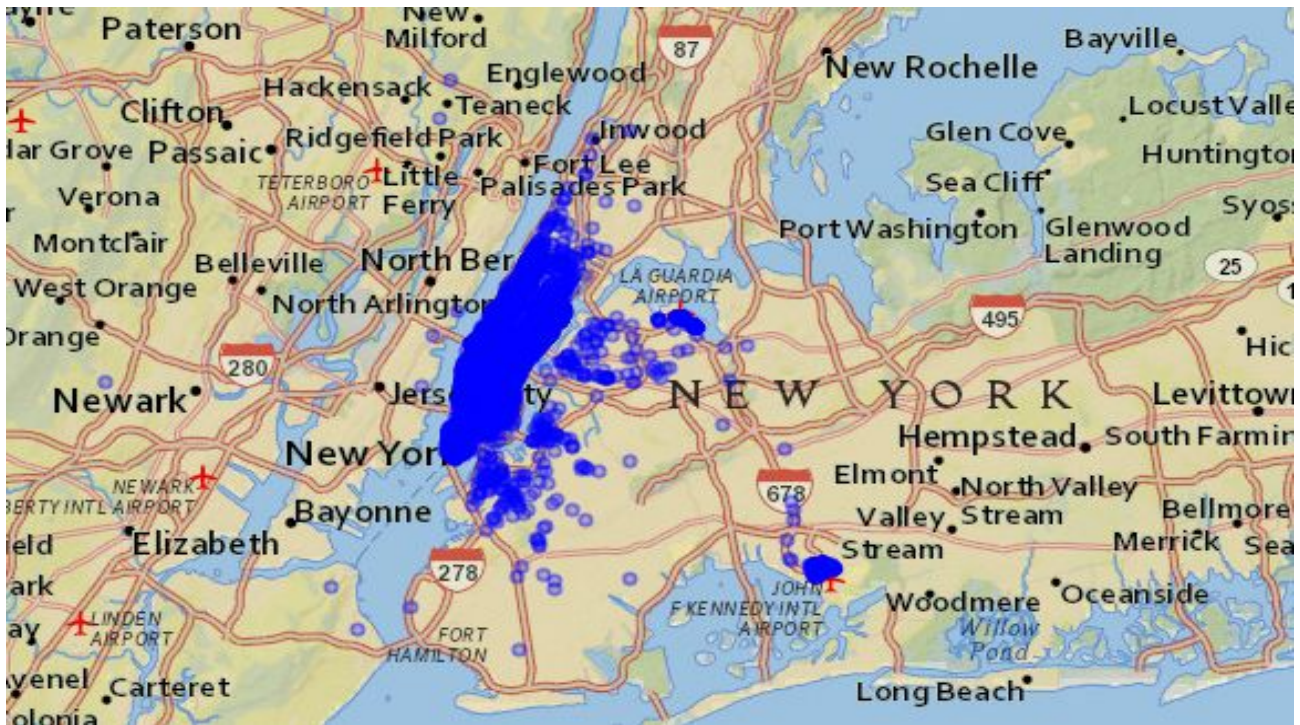


Fig 1. Clusters of pickup and dropoff points

Almost all of our trips were in fact taking place in Manhattan only. Another notable hot-spot is JFK airport towards the south-east of the city.

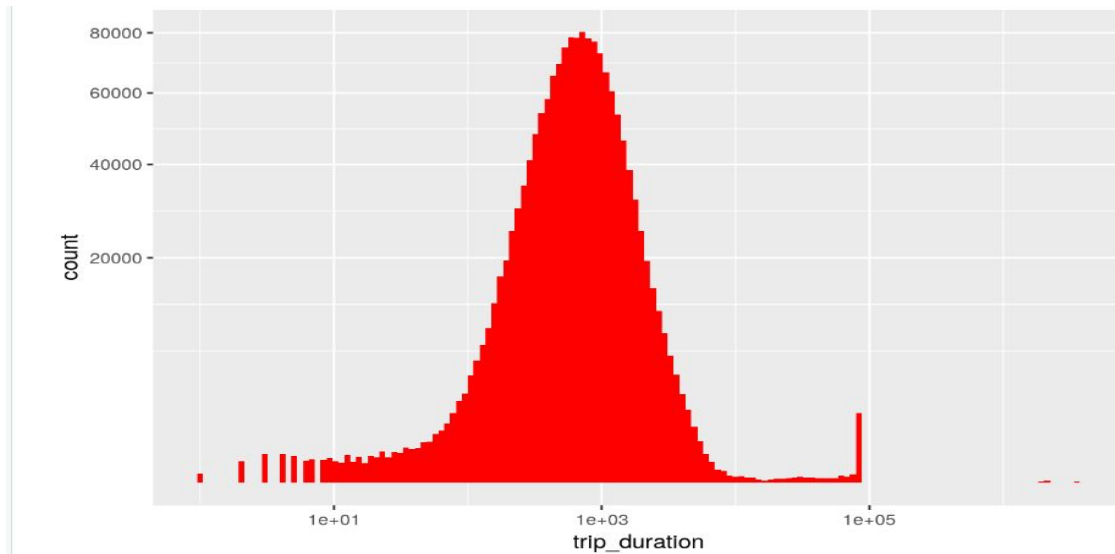


Fig 2. Trip Duration v/s no. of trips

- Majority of rides follow a rather smooth distribution that looks almost log-normal with a peak just short of 1000 seconds, i.e. about 27 minutes.
- There are several suspiciously short rides with less than 10 seconds duration. Additionally, there is a strange delta-shaped peak of *trip_duration* just before the $1e5$ seconds mark and even a few way above it.

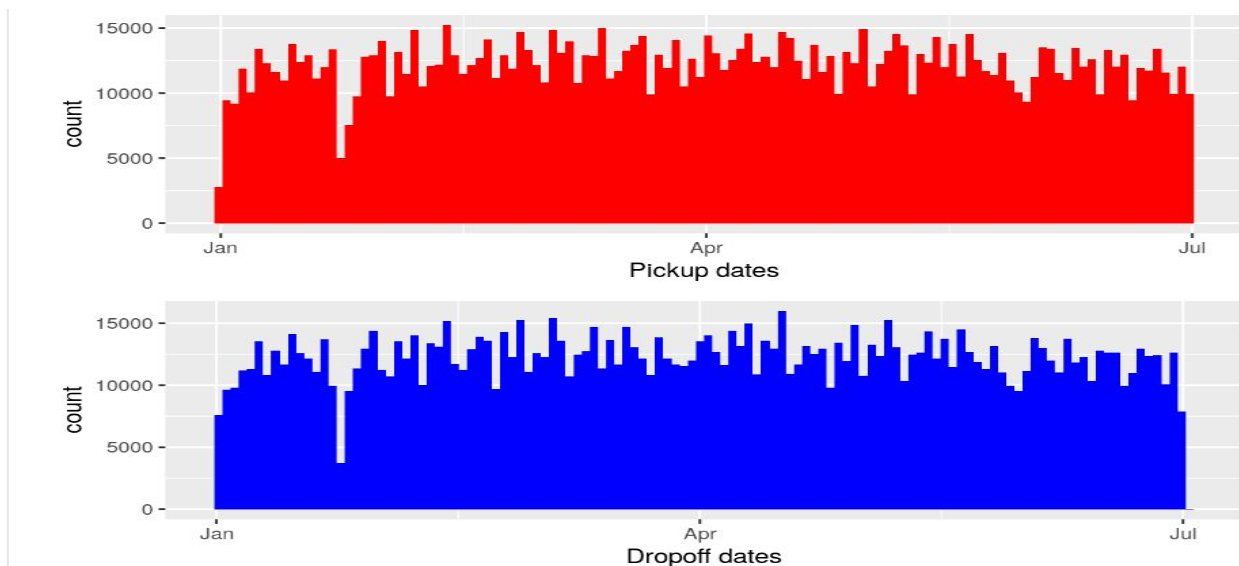


Fig 3. Date v/s No. of trips

Fairly homogenous, interesting drop between Jan and Feb could be due to some harsh weather or a storm.

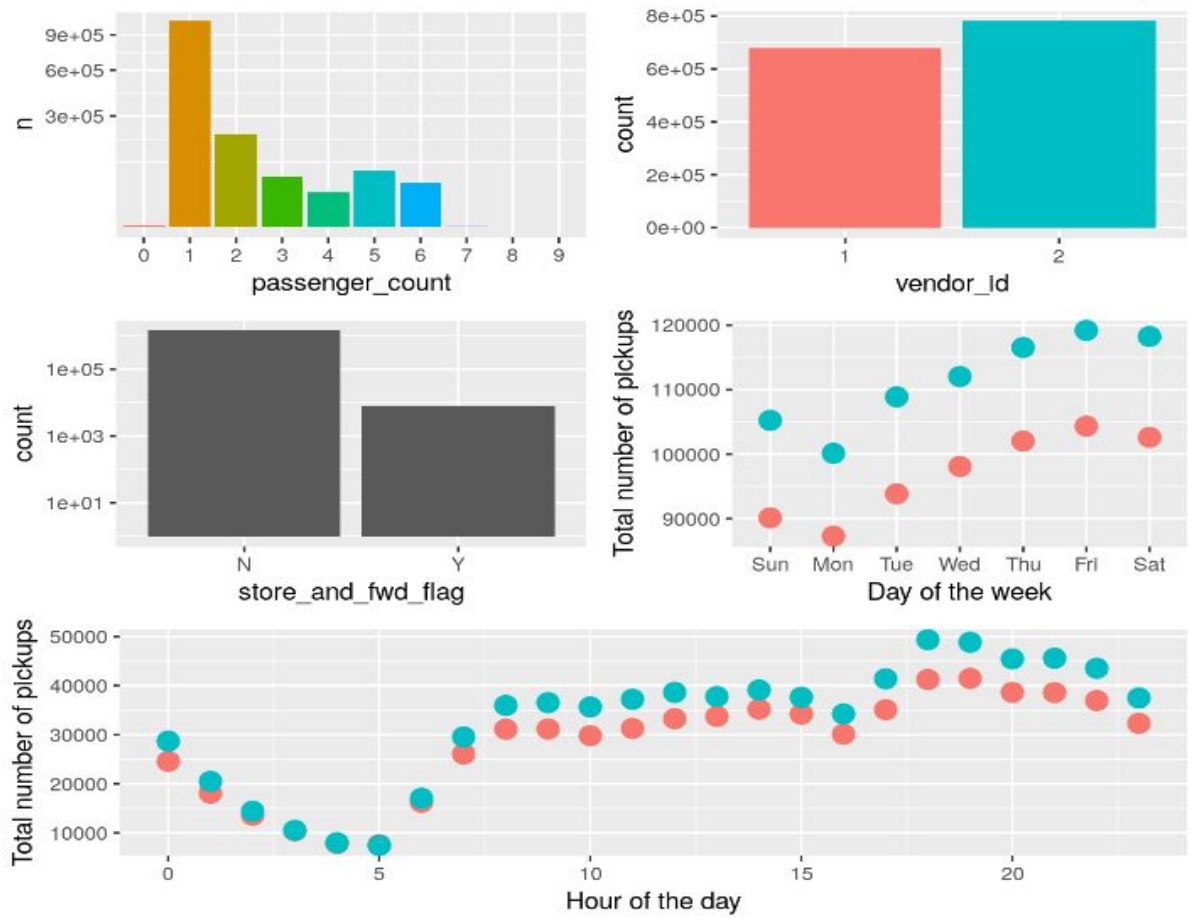


Fig 4. Visualizing different features

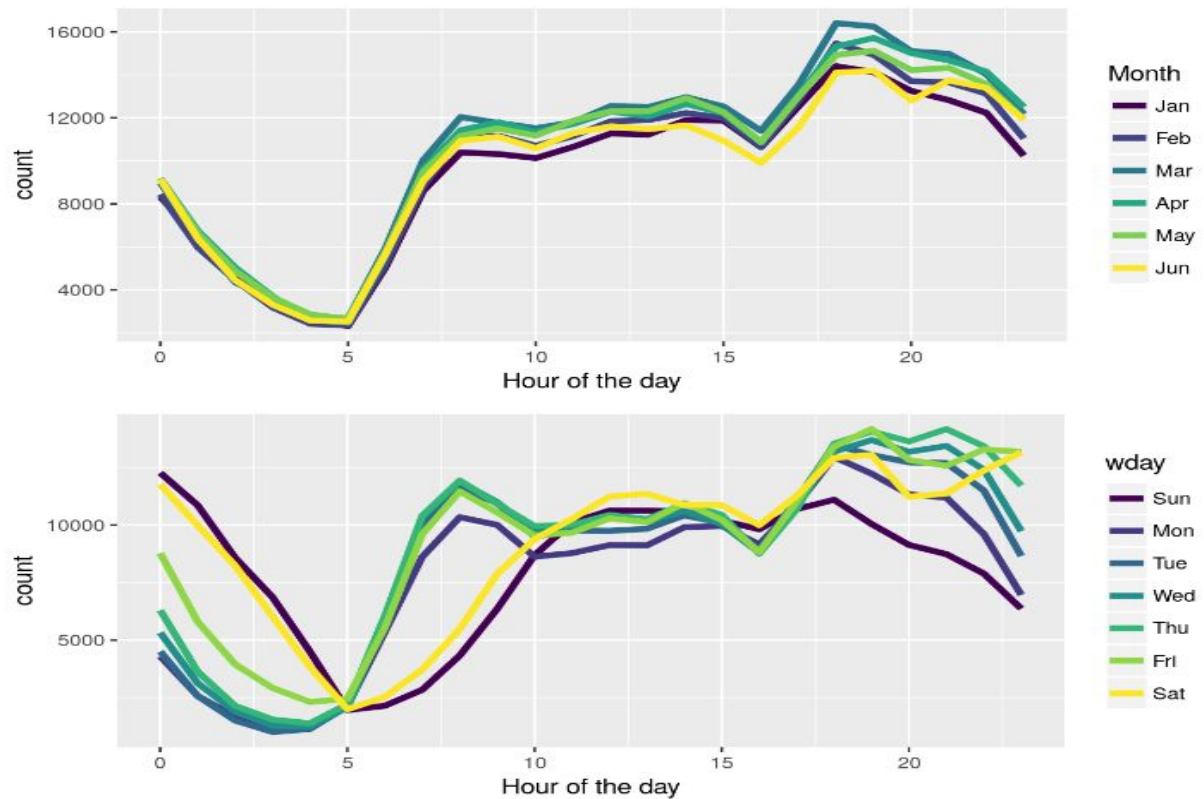


Fig 5. Visualizing Hour of the day feature

- January and June have fewer trips, whereas March and April are busier months. This tendency is observed for both *vendor_ids*.
- The weekend (Sat and Sun, plus Fri to an extent) have higher trip numbers during the early morning hours but lower ones in the morning between 5 and 10, which can most likely be attributed to the contrast between NYC business days and weekend night life. In addition, trip numbers drop on a Sunday evening/night.

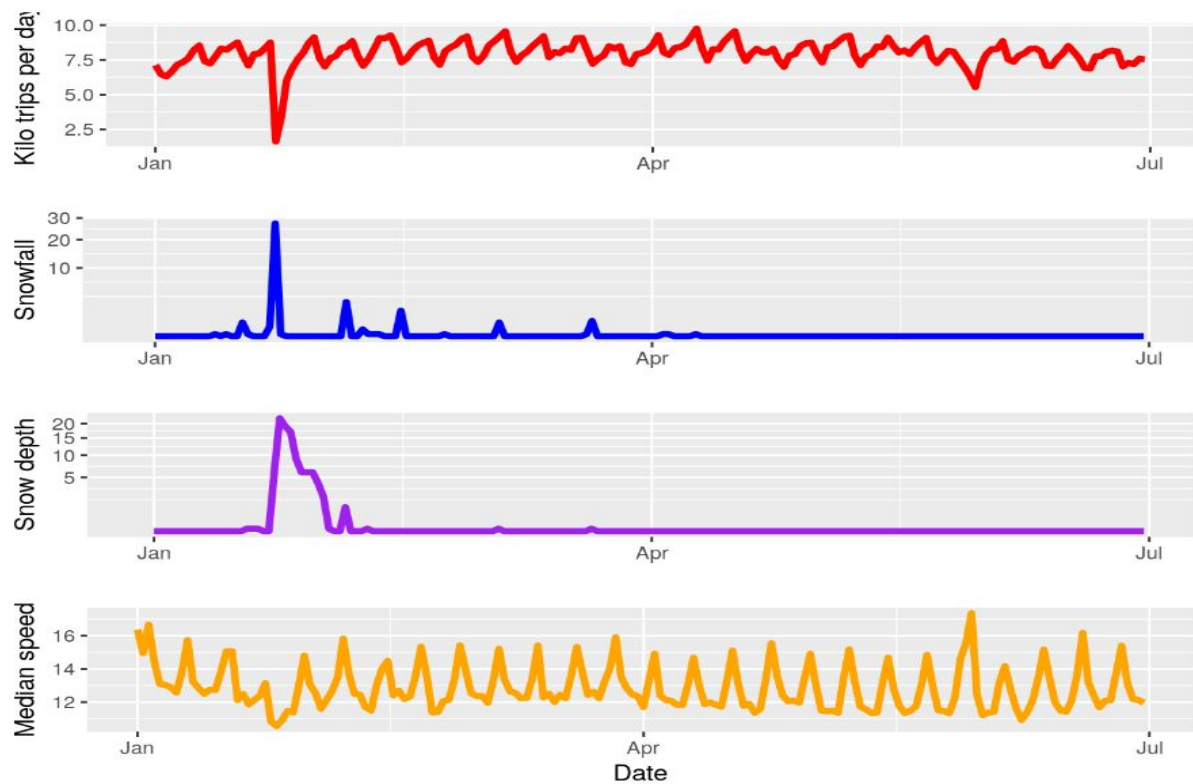


Fig 6. Visualizing effect of weather

The dip in trip volume corresponds to the largest (and first?) snow fall of the winter in NYC (Jan 23rd). In fact, NYC was hit by a blizzard and experienced record-breaking snowfall. The impact on the traffic patterns was enormous, and the median speed slowed down notably.

CHAPTER 4

MODEL

4.1 Regression Model

- Linear regression and ridge regression
- Input features - 'wday', 'yday', 'pu_hour', 'passenger_count', 'pickup_latitude', 'pickup_longitude', 'vendor_id'.
- Linear Regression RMSLE - 0.816
- Ridge regression RMSLE - 0.808
- **The score was not good so we merged our data with external Data and did a lot of feature engineering.**

4.2 Feature Engineering

1. **PCA Features :** 2d-2d transform

We use PCA to transform longitude and latitude coordinates. In this case it is not about dimension reduction since we transform 2D-> 2D. The rotation could help for decision tree splits.

2. **Date-Time Features :**

1. Pickup_weekday
2. Pickup_dayofyear
3. Pickup_hour
4. Pickup_minute

3. **Distance Features**

1. pickup_latitude
2. dropoff_longitude
3. **Harversine distance**
4. Direction : **Bearing**
5. center_latitude [given as - (pickup_latitude+dropoff_latitude) /2]

6. center_longitude

Note : the latitude and longitude were finally passed to model via clustering algo.

4. OSRM Features

1. total_distance
2. Total_travel_time
3. number_of_steps

5. Weather Features

1. average temperature
2. precipitation
3. snow fall
4. snow depth

6. Speed Features

1. Avg Speed pickup_cluster
2. Avg Speed dropoff_cluster
3. Avg Speed centre_cluster

7. Clustering Features

1. Pickup_cluster
2. Dropoff_cluster

4.3 Cluster Visualization

- K-mean Clustering
- Clustering on latitude,longitude
- 100 cluster.

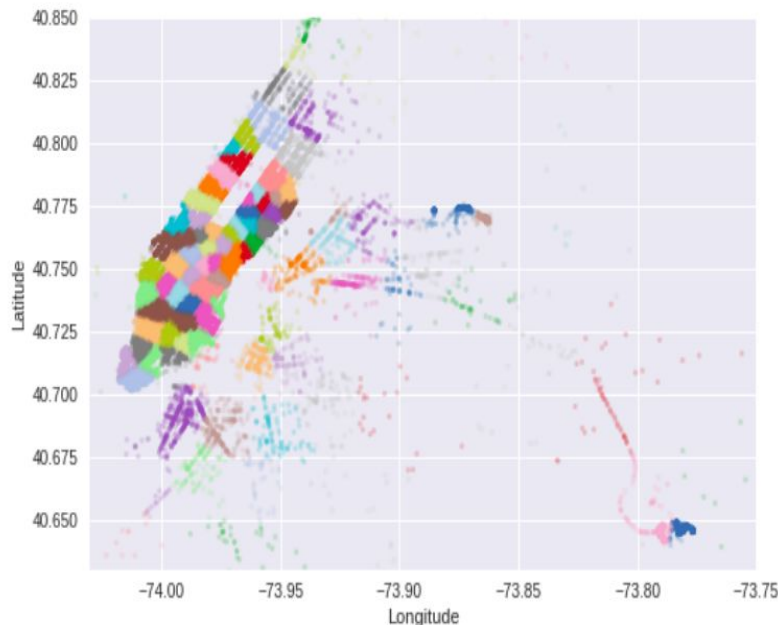


Fig 7. Visualizing Cluster on locations

4.4 Getting Important Features

1. Xgboost get_fscore()

Xgboost models has feature importance score which is often used. Please note that it *does not necessary means that the feature is really important*, higher score means that the *feature was used in more tree splits*.

2. RMSE without feature :

We tried a simple backward feature elimination round. *For each feature a new model was trained without that feature*. It helps to understand what are the essential features which we should not remove.

The following plot shows the feature elimination result and the xgboost importance. A few observations:

- We had quite a few location related feature with high feature importance score although removing just one of them does not really increase the error.
- Vendor_id is the second least used feature according to feature importance plot but removing it increases the rmse significantly. It is binary value, even a few decision trees using it captures its information.
- Direction is important from both aspect. It does not have many correlated feature and removing it would hurt the model.
- There are lots of different features which could be removed without really breaking a leg.

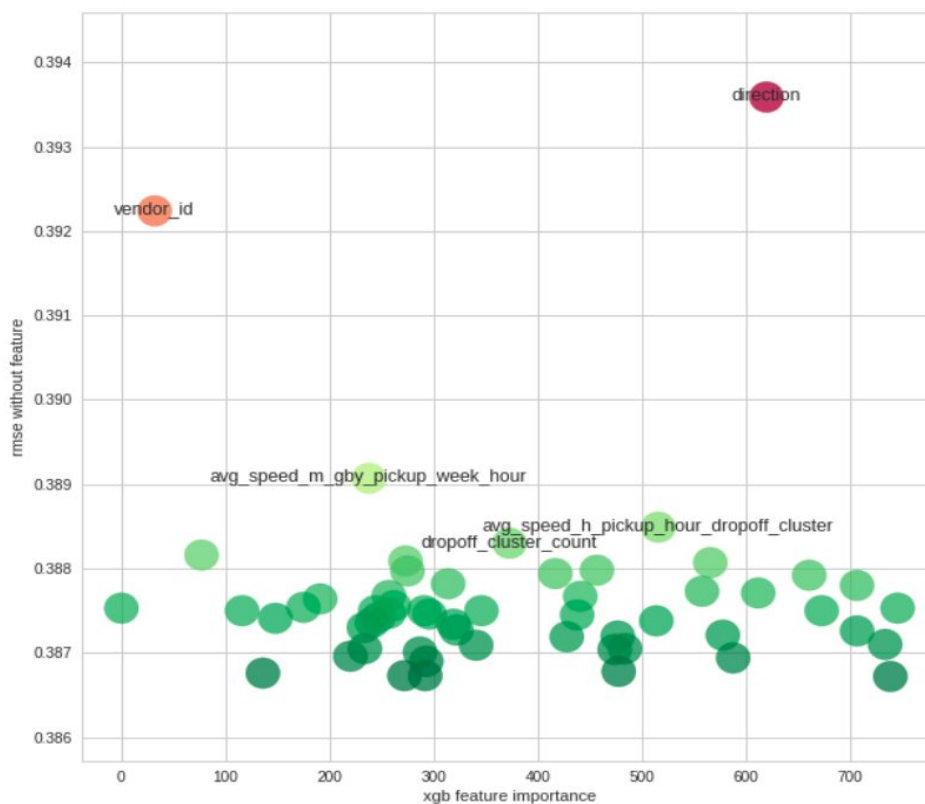


Fig 8. Visualizing important features

	feature_name	importance	rmse_wo_feature
37	avg_speed_h_pickup_cluster_dropoff_cluster	746.0	0.38753
50	pca_manhattan	739.0	0.38672
28	pickup_pca1	734.0	0.38710
49	total_distance	718.0	0.00000
18	pickup_longitude	707.0	0.38726
7	pickup_latitude	707.0	0.38780
3	distance_haversine	673.0	0.38750
51	dropoff_latitude	661.0	0.38792
0	total_travel_time	658.0	0.00000
13	direction	620.0	0.39359
30	log_trip_duration_gby_pickup_dt_bin	612.0	0.38771
24	center_longitude	588.0	0.38694
42	cnt_pickup_cluster_dropoff_cluster	578.0	0.38721
6	count_60min	566.0	0.38807
44	dropoff_longitude	558.0	0.38773
27	avg_speed_h_pickup_hour_dropoff_cluster	516.0	0.38849
56	dropoff_pca1	514.0	0.38738
33	pickup_dt	484.0	0.38705
16	distance_dummy_manhattan	478.0	0.38678
19	avg_speed_h_center_lat_bin_center_long_bin	477.0	0.38720
25	avg_speed_h_pickup_hour_center_lat_bin_center_...	474.0	0.38704
54	cnt_pickup_hour_pickup_cluster	457.0	0.38798
47	dropoff_pca0	441.0	0.38767
11	pickup_pca0	438.0	0.38745

Fig 9. List of Important Features in decreasing order of importance

4.5 Hyperparameter Tuning

Trying different values for the following parameters

eta [default=0.3]

- Analogous to learning rate in GBM
- Makes the model more robust by shrinking the weights on each step
- Typical final values to be used: 0.01-0.2

max_depth [default=6]

- The maximum depth of a tree, same as GBM.
- Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.
- Typical values: 3-10

subsample [default=1]

- Same as the subsample of GBM. Denotes the fraction of observations to be randomly samples for each tree.

lambda [default=1]

- L2 regularization term on weights (analogous to Ridge regression)

Tuning these using -
sklearn CV options (RandomizedSearchCV or GridSearchCV)

CHAPTER 5

FINAL RESULT

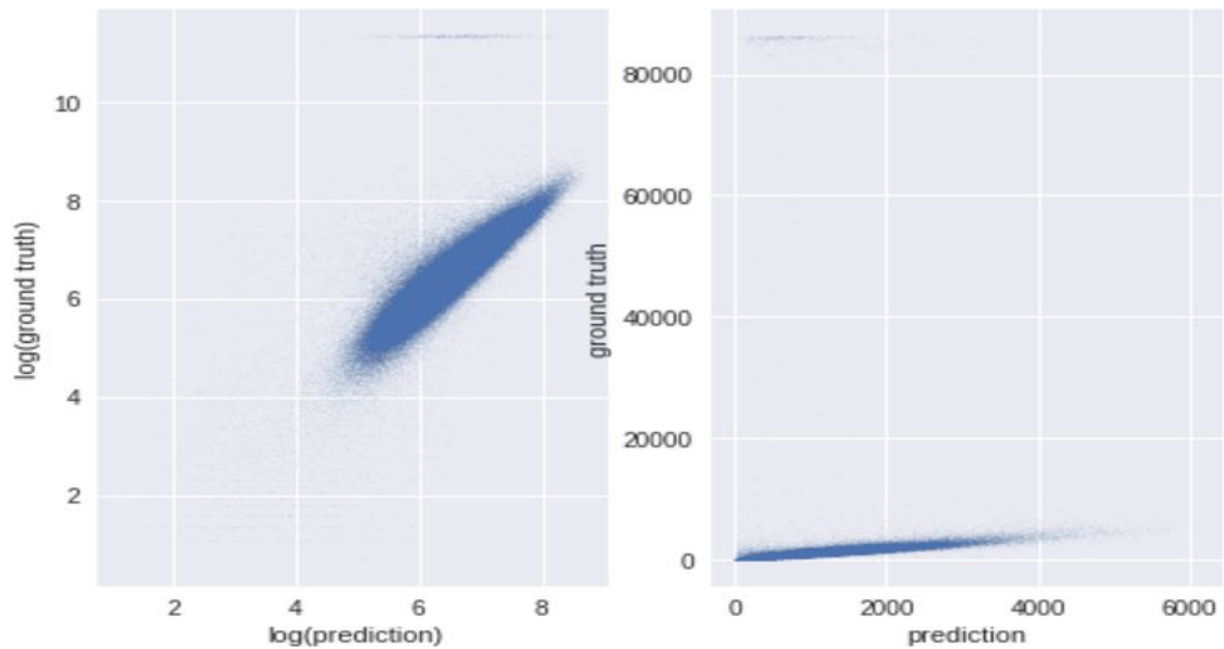



Fig 9. Visualizing final prediction v/s ground truth

92

▼ 6


inSANE_SVD



0.37449

Your Best Entry ↑

Your submission scored 0.37449, which is an improvement of your previous score of 0.61129. Great job!

 Tweet this!

REFERENCES

- [1] Urban link travel time estimation using large-scale taxi data with partial information. Xianyuan Zhana, Samiul Hasanb, Satish V. Ukkusuria, Camille Kamgac. ScienceDirect 2004.
- [2] Visual Exploration of Big Spatio-Temporal Urban Data: A Study of New York City Taxi Trips. Nivan Ferreira ; Jorge Poco ; Huy T. Vo ; Juliana Freire ; Cláudio T. Silva. IEEE Transactions on Visualization and Computer Graphics (Volume: 19, Issue: 12, Dec. 2013)
- [3] Fare and Duration Prediction: A Study of New York City Taxi Rides Christophoros Antoniades, Delara Fadavi, Antoine Foba Amon Jr. December 16, 2016
- [4] Travel time prediction under heterogeneous traffic conditions using global positioning system data from buses. Vanajakshi, L., S. C. Subramanian, and R. Sivanandan. IET intelligent transport systems 3.1 (2009): 1-9.
- [5] Easytracker: automatic transit tracking, mapping, and arrival time prediction using smartphones. Biagioni, James, et al. Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems. ACM, 2011.
- [6] Travel-time prediction with support vector regression. Wu, Chun-Hsin, Jan-Ming Ho, and Der-Tsai Lee. IEEE transactions on intelligent transportation systems 5.4 (2004): 276-281.
- [7] Source of dataset: <https://www.kaggle.com/nyctaxi/yellow-taxis>

