Machine Learning Engineer Nanodegree

Capstone Project

Samit Singh
October 5th, 2017

I. Definitions

Project Overview

The project aims to solve a problem from Kaggle Competition: New York City Taxi Trip Duration. (https://www.kaggle.com/c/nyc-taxi-trip-duration)

Here, I am building a model that can predict the total ride duration of taxi trips in New York City. An algorithm that can accurately predict the total ride duration will help passengers to better plan their schedule and reach their destination on time. Such an algorithm can also be used by ride hailing services like Uber, Lyft etc. in efficiently utilizing their resources and ultimately providing better service to the passengers. For example knowing when a taxi driver would be ending their current ride, would help these services to identify which driver to assign to each pickup request.

Datasets and Inputs

Here I am using the train and test data provided by the Kaggle Competition. The train data has the following variables which I think directly relates to the problem:

- Pickup date and time: On some dates the traffic will be more than usual and will affect the ride duration. Also time of travel will can also affect the ride duration of trips.
- Pickup and drop-off location: Ride duration directly correlates to the distance to be travelled. Based on pickup and drop-off coordinates given in the dataset, areas (neighborhoods) can be identified in New York City. Some areas have may have narrow and crowded roads, other may have wide highways. All this will affect the total ride duration.
- Other variable in the train data include id, vendor_id, passenger_count.

The training set contains 1458644 trip records.
The testing set contains 625134 trip records.

The dataset can be accessed from here: https://www.kaggle.com/c/nyc-taxi-trip-duration/data

Problem Statement

This is a regression problem which tries to predict the total ride duration of taxi trips in New York City based on pickup time, geo-coordinates, number of passengers, and several other variables.

I am following the following workflow in order to solve the problem

- Exploratory data analysis – Analyzing the given data set, doing feature engineering and coming up with new more relevant features.
- Acquiring additional data – If possible I would explore additional data from public sources which may help the model in better predicting the ride duration.
- Using different regression algorithms – Next step would be to implement and test different algorithms and select the one which will give the best result based on the evaluation metrics.
- Comparing results with benchmark model – The results will then be compared with the benchmark model to see if our model is any good. If not, the process above will be reiterated by coming up with better features.

Metrics

The evaluation metric for this competition is Root Mean Squared Logarithmic Error.

The RMSLE is calculated as

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

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 ii.
log(x) is the natural logarithm of x

RMSLE can be used when we want to penalize under estimates more than over estimates. For this problem, underestimating the trip duration should be penalized more as people generally dislike getting late more but would not mind arriving earlier than the total estimated duration.

II. Analysis

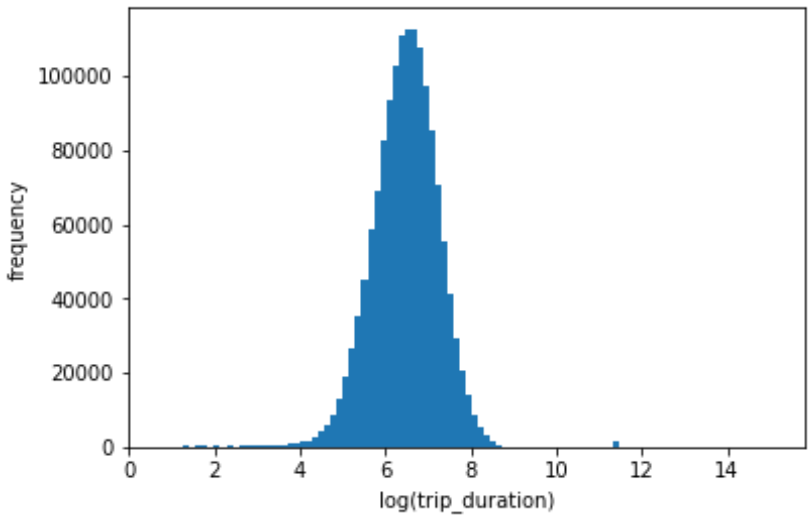
Data Exploration

We start by first loading the dataset. On exploring the dataset we find that

- The train dataset contains 1458644 rows and 11 columns.
- There are no missing values in the dataset.
- Outliers:
    - Maximum trip duration is approximately 980 hours, minimum trip duration is 1 second.
    - All data that fall outside the boundary of New York.
    - Also the maximum no. of passenger in the vehicle comes out to be 9, which can be considered as an outlier.
- The target variable ride duration is skewed. We use logarithmic transformation.
- Categorical variables for example vendor id, store_and_fwd_flag will have to be one hot encoded.

Exploratory Visualization

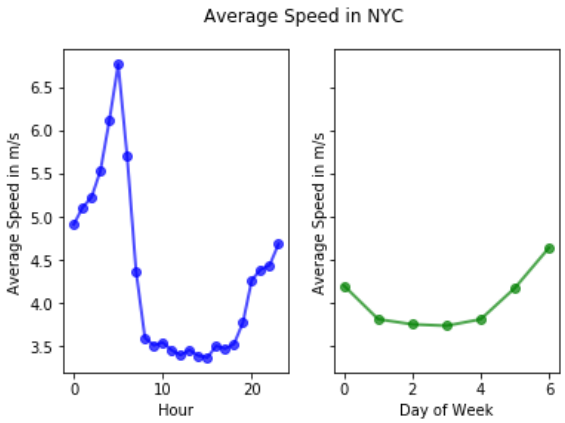
Since there are a few outliers that, we will first log transform the 'trip_duration' variable and see if the distribution is normal or skewed.



We see that our target variable is normally distributed, with only a few outliers. Most trips are between e^4 and e^8 seconds i.e. most trips are between 1 min and 60 mins long.

Average speed visualization

Here I plotted the graphs for Hour of the day Vs. Average speed and Week of the day Vs. Average speed.



We see that the average speed of taxis depend on the hour of pickup and also on the day of the week. As expected the speed is the slowest during the time period 09:00 hrs to 19:00 hrs. Similarly weekdays are the slowest and there is some increase in average speeds during the weekends.

Algorithms and Techniques

- *Since this is a regression problem, I intend to use the decision tree regressor and random forest regressor for the problem and see which performs better.*
    - *In a regression tree, since the target variable is a real valued number, we fit a regression model to the target variable using each of the independent variables. Then for each independent variable, the data is split at several split points. We calculate Sum of Squared Error (SSE) at each split point between the predicted value and the actual values. The variable resulting in minimum SSE is selected for the node. Then this process is recursively continued till the entire data is covered.*
    - *Similarly Random forests when used for regression are an ensemble of different regression trees and are used for nonlinear multiple regression. Each leaf contains a distribution for the continuous output variable/s. (source)*
- *For decision tree regressor, I will use the GridSearchCV to identify the optimum max depth that would give the best results. The max_depth parameter will range from 1 to 10. Next I would implement the Random Forest Regressor algorithm with a maximum depth of 10.*
- *In order to use these algorithms all the categorical variable will first be one hot encoded, and outliers will be removed in the data preprocessing step. Then we will divided the training set into train and test sets and evaluate our results against our chosen metrics (RMSLE error) on the test set.*

Benchmark

A good benchmark model would be one that is solely based on the distance of the trip divided by the average speed of taxis in city like New York (not taking into account other variables like traffic, road conditions, location etc.) Here we will assume an average speed of 11.5 mph (18.50 kilometer per hour). (source: http://animalnewyork.com/2014/data-reveals-worst-times-traffic-nyc)

Using this benchmark model on the test data we are getting an RMSLE error of 0.7551

III. Methodology

Data Preprocessing

- There are some outliers in the trip duration variable. For example the 987 hours trip durations and 1 second trip duration. So I will exclude the data the lies beyond 2 standard deviations from the mean.
- Removing all data with coordinates that fall outside the boundary of New York.
- Removing outlier passenger_count data where the count of passenger more than 6

Additional dataset added:

- Fastest route data - Using OSRM the fastest routes for each data point was extracted. It contains features like starting_street, end_street, total_distance, total_travel_time, number_of_steps (one step consists of some driving and an action the taxi needs to perform. It can be something like a turn or going on to a highway)
  (Source: https://www.kaggle.com/oscarleo/new-york-city-taxi-with-osrm)

Feature Extraction

- The total trip duration may depend on the hour of the day, month as well as day of the week. So we will extract these features from the train and test data.
- We can determine the distance and direction of each trip based on the pickup and drop-off coordinates. Taken from this post, we will extract these features: haversine distance, manhattan distance and directionality.

Handling pickup and drop-off coordinates data

One way to handle the pickup and drop-off coordinates in our train and test data is form clusters of these locations. We can use k-means clustering algorithm to form clusters and then use one hot encoding technique. Here we will be forming 50 clusters.

One hot encoding

We will be using one hot encoding for handling our categorical variables namely vendor id, passenger count, store_and_fwd_flag, month, DayOfMonth, Hour, DayOfWeek, pickup_cluster, dropoff_cluster.

Implementation

- *After data preprocessing, our final training data set consists of 1437127 rows and 157 columns.*
- *We now split the train_final set further into train and test set, this will enable us to tweak model parameters to increase accuracy and help us to avoid overfitting.*
    - *Before splitting our train_final we will change the form to X matrix (features) and Y labels (duration), as this form is the required format for inputting into our sklearn algorithm.*

```
# using trip_duration as Y labels


duration = train_final['trip_duration']

features = train_final.drop('trip_duration', axis=1)
```

    - *After the split: training set consists of 1149701 data points and test set consists of 287426 data points.*

- *We then create a performance matrix function that calculates and returns the performance score between true and predicted values based on the metric chosen. Here the chosen metric is RMSLE error.*
- *Now we have the data in the desirable format and we can use different sklearn algorithms. I have tried decision tree regressor and random forest regressors.*

Once the task of data preprocessing is completed and the input data is obtained in X matrix (features) and Y labels (duration) form, the process of applying sklearn regression algorithms is pretty straightforward. We only need to modify our scoring function for this task and write a function to implement RMSLE scores.

Refinement

Initial solution:

Using the benchmark model described above we are getting an RMSLE error of 0.7551 on the test data

Intermediate solutions:

Using Decision Tree Regressor we get an RMSLE error of 0.6674. Tuning max_depth parameter for decision tree regressor did not show any improvements, so I tried using Random Forest Regressor and obtained the following results.

Algorithm	Hyper-parameter	RMSLE Error
Decision Tree Regressor	max_depth = 1	0.6674
Random Forest Regressor	max_depth = 1	0.6672
Random Forest Regressor	max_depth = 2	0.5526
Random Forest Regressor	max_depth = 5	0.4798
Random Forest Regressor	max_depth = 10	0.4440
Random Forest Regressor	max_depth = 15	0.4201

Final solution:

Finally on using Random Forest Regressor with max_depth of 15 we get an RMSLE error of 0.4201 on the test data.

IV. Results

Model Evaluation and Validation

- *The final model a Random Forest Regressor, gives a RMSLE error of which is reasonable and aligns with the solution expectations. The final parameter of the model uses a max_depth of 15 and seems appropriate.*
- The model generalizes well on unseen data as the test error comes out to be 0.4201
- The model has been trained on a large dataset of input values (*1149701 data points*) *and small changes in training data or input space should not greatly affect the results.*
- *Yes the results from the model can be trusted, as the model performs significantly better than the benchmark model.*

Justification

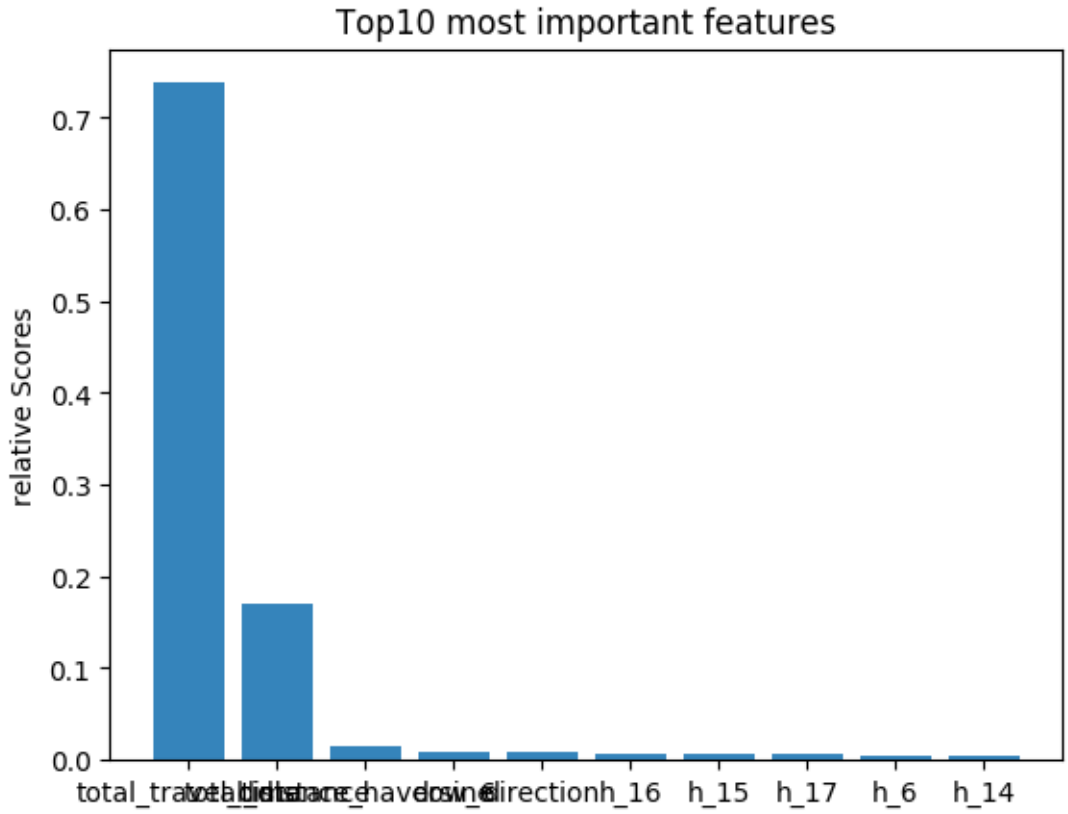
- Our benchmark model obtained a test error of 0.7551 whereas our final model has an error of 0.4201 on the test dataset. So, yes the final results found are stronger than the benchmark results.

- In my humble opinion the final model is significantly better than our benchmark model and the solution can be considered to be significant enough to have solved the problem.

V. Conclusion

Free-Form Visualization

Here I am plotting the feature importance of the final model.



As can be seen from the plot above most important feature that decides the ride duration time are:

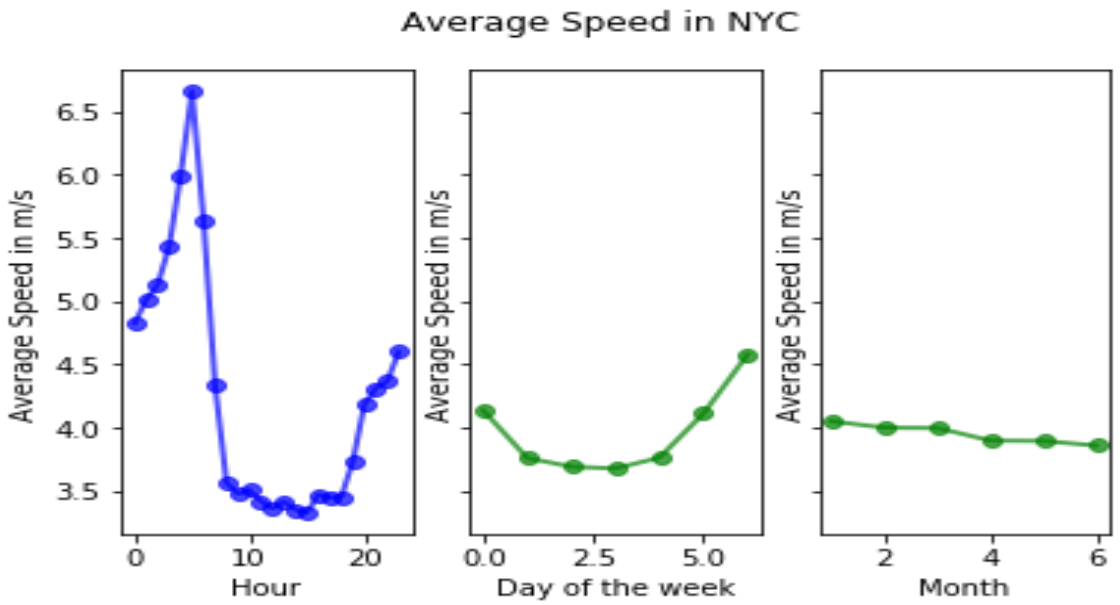
total_travel_time – The total travel time that we extracted from the OSRM fastest route data is the most important factor in determining the ride duration.
total_distance – The total distance between pick up and drop location is the second mo st significant feature
distance_haversine – Similarly the haversine distance we calculated is a significant featu re that determines the total ride duration

dow_6, direction, h_16, h_15, h_17, h_6, h_14 – dow_6 i.e. the day of the week (Saturda y), direction of travel and the different hours of day are other important features.

Reflection

- *Summary of entire process used for the project:*
  - *Loading and exploring datasets.*
  - *Data cleaning: Identifying and removing the outliers.*
  - *Feature Extraction: Extracting most useful features from the data e.g. extracting hour, month, day of the week, distance and directionality features from the data.*
  - *Handling the pickup and drop-off coordinates in our train and test data by forming clusters of these locations using k-means clustering algorithm.*
  - *Visualizing the target variable to see how it is distributed. Visualizing correlations between various variable in the input dataset.*
  - *Augmenting the given dataset by adding useful information from additional dataset that can help in building better models.*
  - *Data Preprocessing: Next we use one hot encoding for handling our categorical variables. We then drop the categorical variables and use the one hot encoded features.*
  - *We then split our final training data set further into train and test set, this enables us to tweak model parameters to increase accuracy and help us to avoid overfitting.*
  - *We then establish a performance metric to evaluate our model against. Here we will be using root mean squared logarithmic error.*
  - *Next we implement different models and evaluate the results.*
  - *We choose a model, refine it by testing out different hyperparameters and finally compare it with our benchmark model.*

- *Interesting aspects of the project*
  I plotted the average speed of taxis in NYC versus the month graph and the result was interesting. The average trip speed versus the month follows an unexpected trend. During winters the average speed is more and decreases gradually as summer approaches. There could be various reasons for this, one possible explanation is that during winter months there is less traffic on roads as a results the vehicles can attain higher speeds.



- *Difficult aspects of the project*

  The large dataset was certainly a difficult part of the project. I could not load the data set on my local machine. So eventually I used the service on Floyhub.com in order to complete the project.

- *Yes the final model and solution fit my expectation for the problem and can be used in general setting to solve these kinds of problems.*

Improvement

- For further improvement we could:
  - Try using deep learning model to solve the problem
  - Gather additional datasets and add to our model like weather data, accidents data, crowded hubs in NYC data etc.
  - Increase the maximum depth for our Random Forest Regressor algorithm.

Yes, I do believe that many improvement can be made and even better solution exists.
