# Machine Learning Engineer Nanodegree
# (Capstone Project)

**Himanshu Shekhar**

1

## 1. Definition

### 1.1. Project Overview

I will be working on Kaggle competition "New York City Taxi Trip Duration" where in I will build a model that predicts the total ride duration of taxi trips in New York City. This problem statement is relevant to transportation industry and can be applied in different contexts like ride sharing services, food/grocery delivery. Being able to predict ride duration accurately is extremely important for both the entities (service provider customers) involved in the transaction. From service provider standpoint, it's important as ride duration would dictate pricing that could be charged to a customer. In addition, predictability (duration) also provides good estimate of the number of drivers who would be present in certain region at any given time which is important so that customers could be matched with drivers/service provider in an efficient manner. From customer standpoint, predicting duration could help them make a decision as to when is the optimal time to start their commute.

With the advent of technology based cab services, this problem has become particularly important for the drivers and the customers. A good prediction mechanism can be instrumental for drivers in optimizing their returns, while also saving the customers from the uncertainties attached to a trip.
*Data Source:* https://www.kaggle.com/c/nyc-taxi-trip-duration/data

### 1.2. Problem Statement

Objective is to predict the time duration (in secs./mins.) of a New York taxi ride as a function of independent attributes like pick up and drop off location, time, volume etc. I will study the impact of various features and also attempt to find the best possible ways to leverage those features. I will explore Ensemble **(decision tree/random forest)** models to do the prediction.

### 1.3. Metrics

I will use Root Mean Square Error (RMSE) as a metric to measure model's accuracy. To validate the usefulness of model, RMSE values for baseline model and the actual model developed would be compared on the test set. It has the advantage of being convex and physically interpretable (it has the same unit as time for duration prediction).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2}$$

where $y_i$ is the actual value and $\hat{y_i}$ is the predicted value

## 2. Analysis

### 2.1. Data Exploration

The data provides the details of taxi rides in the New York City from Jan 2016 to June 2016 (contains 1,458,644 trip records). This data is provided by the NYC Taxi and Limousine Commission (downloaded from Kaggle). Each trip record fields:

- 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
- trip duration - duration of the trip in seconds

**Tabela 1. Sample observations:**

| | Field | sample1 | sample2 |
|---|---|---|---|
| 1 | id | id2875421 | id2377394 |
| 2 | vendor id | 2 | 1 |
| 3 | pickup datetime | 2016-03-14 17:24:55 | 2016-06-12 00:43:35 |
| 4 | dropoff datetime | 2016-03-14 17:32:30 | 2016-06-12 00:54:38 |
| 5 | passenger count | 1 | 1 |
| 6 | pickup longitude | -73.982155 | -73.980415 |
| 7 | pickup latitude | 40.767937 | 40.738564 |
| 8 | dropoff longitude | -73.964630 | -73.999481 |
| 9 | dropoff latitude | 40.765602 | 40.731152 |
| 10 | store and fwd flag | N | N |
| 11 | trip duration | 445 | 663 |

- From a preliminary analysis, it is evident that the target variable has some unusually high values so I will get rid of those to make the model less vulnerable to these outliers. Also, trip duration is highly skewed (right tailed) so I will apply a logarithmic transformation on this feature to reduce the range of values.
- The passenger count variable has a minimum value of 0 passengers, which is not relevant in the context of this usecase. These observations are most likely errors and will need to removed from the dataset.
- Based on different coordinate estimates of New York City, the latitude and longitude ranges are as follows: Latitude is between 40.7128 and 40.748817 whereas Longitude is between - 74.0059 and - 73.968285. The statistical summary of pick-up and drop-off coordinates show max and min observations that fall outside of the NYC city coordinate range. I will exclude these data points as this analysis is limited to New York City.

## 2.2. Exploratory Visualization

- **Distance:** I computed the haversine distance which determines the great-circle distance between two points on a sphere given their longitudes latitudes. Below plot demonstrates a positive correlation between trip distance and duration

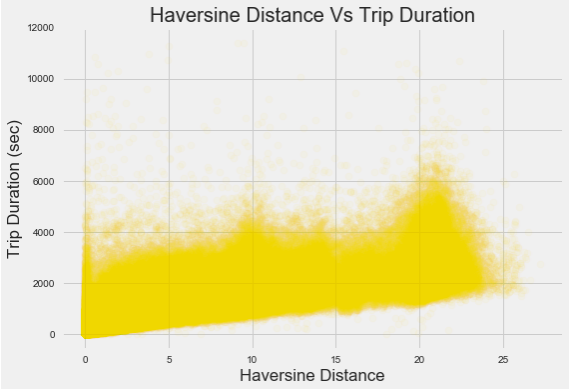



**Figura 1.**

- **Time of day / Day of week:** The time of the day is an important factor to determine the duration a trip can take. The same trip during rush hours can take much longer compared to non-rush hours. The day of the week can have a significant impact on the predictions because we expect the weekdays to be more congested than the weekends especially during the day time. We can clearly observe from below heatmap the peak hours:
  - Monday - Friday : 8:00am - 6:00pm which is the ususal office hours.
  - Thursday, Friday, Saturday Nights: 6:00 pm through midnight.
  - Early Saturday Sunday Mornings: 12:00 am - 1:00 am
  - Sunday Afternoons: 2:00 pm and 4:00 pm

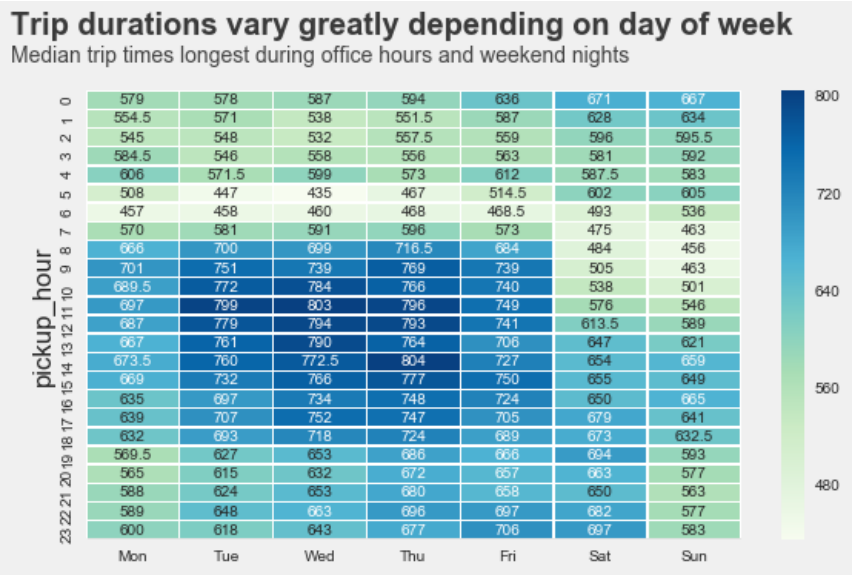



**Figura 2.**

- **Passenger count:** The passenger count variable is the number of passengers in the vehicle as inputed by the driver. My assumption was that trips with more passengers are inherently longer due to more stops. However, as we can observe in below figure, median trip duration does not vary much as passenger count increases.

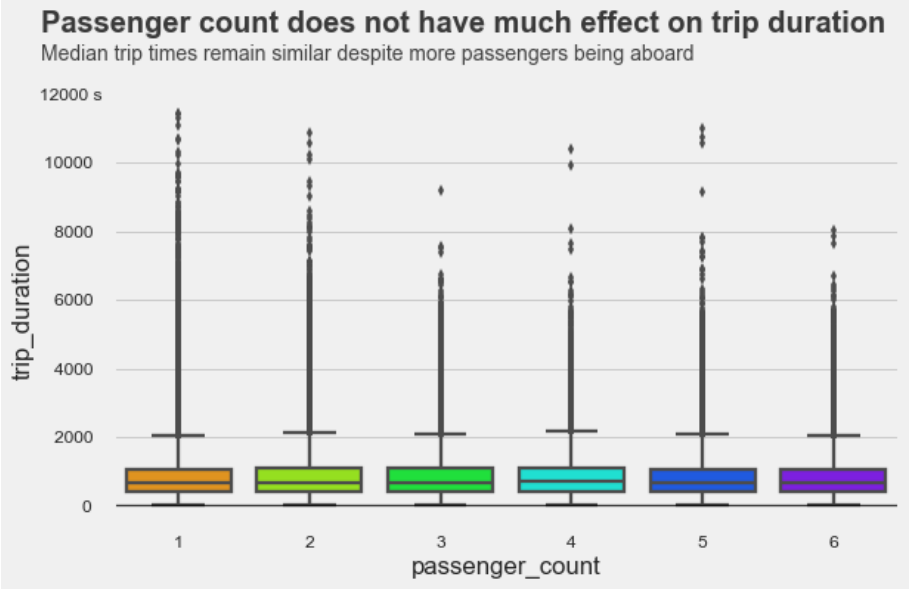



**Figura 3.**

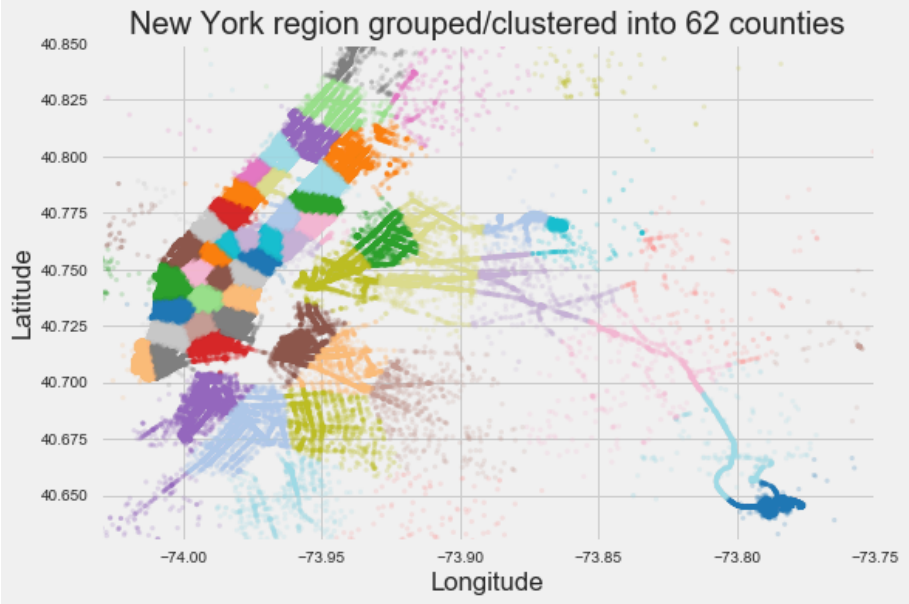



**Figura 4.**

- **Location:** An important aspect of this prediction task is the pick-up and drop-off locations. As discussed above, the distance factor is indeed import to determine the duration but it is also important to consider the exact route the taxi is taking. Though we might not have the exact routes of the trip but the pick-up and drop-off coordinates can act as a proxy for this feature. An important aspect of this problem

was to use the pickup and drop-off coordinates effectively. Directly feeding them to the model would have made less sense as the coordinates could have potentially infinite possible values and with the limited number of cases given in the training set, it would have been difficult to extract relevant patterns. To tackle this problem, I used kmeans clustering to create 62 clusters (see fig. 4 above) using the pick-up and drop-off coordinates.

### 2.3. Algorithms and Techniques

**K Means**

I would use K-means algorithm to cluster the pick up / drop off longitude and latitude. The k-means algorithm divides a set of N samples X into K disjoint clusters C, each described by the mean $\mu_j$ of the samples in the cluster. The means are commonly called the cluster "centroids"; note that they are not, in general, points from X, although they live in the same space. The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum of squared criterion:

$$\sum_{i=0}^{n} min_{\mu j \in C}(||x_j - \mu_i||)^2$$

Inertia, or the within-cluster sum of squares criterion, can be recognized as a measure of how internally coherent clusters are.

**Random Forest**

As traffic is clustered and aggregated more densely to different locations at different times, the location of the ride will clearly have an affect on the trip duration. Although there is no straightforward way of considering all locations between the start and end points of a ride, the pickup and dropoff locations are available in the dataset and can be used to model some of the effect of traffic and conjunctions. As traffic is clearly not varying solely based on the magnitude of the coordinates, the linear models fail to account for the nonlinear effect the locations have on traffic and hence trip duration. An algorithm that can better account for these nonlinearities is the random forest. The random forest algorithm aggregates many decision trees built on bootstrapped samples of the training data in order to reduce the high variance of a single decision tree and improve prediction accuracy [1][2]. Each of these decision trees aims to divide the predictor space, i.e. the set of all possible values for the features $x_1, x_2, ..., x_n$ , in J distinct and non-overlapping regions $R_1, R_2, ..., R_J$ . The predictor space is divided into high-dimensional rectangles, with the goal to find rectangles $R_1, R_2, ..., R_J$ that minimize the RSS,

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y^{(i)} - \hat{y_{Rj}})^2$$

where $\hat{y_{Rj}}$ is the mean response for the training observations within the jth rectangle. When building each tree, a top-down approach is taken. Beginning with all points in the same region, the algorithm successively splits the predictor space into two halves, stopping when there are no more than five points in a region. At each split, a prediction $x_j$ and

cutpoint s are chosen such that splitting the predictor space into the regions $\{x|x_j < s\}$ and $\{x|x_j >= s\}$ leads to the biggest reduction in RSS. Defining the pair of halves as $R_1(j,s)$ and $R_2(j,s)$ , at each split we seek to find $j$ and $s$ that minimize the equation

$$\sum_{i:x^{(i)} \in R_1(j,s)} (y^{(i)} - \hat{y_{Rj}})^2 + \sum_{i:x^{(i)} \in R_2(j,s)} (y^{(i)} - \hat{y_{Rj}})^2$$

Once the regions are defined, a prediction by a single tree is made by averaging the responses of the training observations in the region to which the test observation belongs. In the random forest, a large number of trees are fit, each using a bootstrap sample from the training data, and a prediction of a new observation is made using the mean of the predictions by all the trees. At each split, only m of the total n predictors are randomly chosen to be considered. This approach is taken to decorrelate the trees, as considering all predictors might yield very similar trees when one or a few predictors are particularly strong. As averaging many uncorrelated trees leads to a larger reduction in variance, this approach often yields better prediction results.

### 2.4. Benchmark

As a baseline prediction, I ran linear regression to do the prediction. The linear regression model finds the set of coefficients that minimize the sum of squared errors

$$y^{(i)} = \theta_0 + \sum_j \theta_j x_j^{(i)}$$

The RMSE for the Baseline model on the test set was obtained as 0.48 for trip duration.

## 3. Methodology

### 3.1. Data preprocessing

- **Trip Duration:** We can clearly observe(fig 5) that trip duration is highly skewed to the right. So I applied log transformation to trip duration which will normalize its distribution (right graph) and reduce the influence of the high observations in the right tail.
- **Outlier treatment/abnormal observations:**
  - Removed observations which had 0 passengers.
  - Capped the observations within 2 standard deviation of trip duration.
  - Restricted observations based on coordinate estimates of New York City. Latitude is between 40.7128 and 40.748817 whereas Longitude is between - 74.0059 and - 73.968285

### 3.2. Implementation

- Here is the list of **libraries** which I used for implementation- numPy, pandas, matplotlib, scikit-learn, datetime, calendar, time, scipy, math, seaborn, plotly
- Representing the pick-up and drop-off coordinate data is an important aspect for prediction as it captures the location dynamics. I opted for **creating clusters** on the given pick-up and dropoff coordinates. To represent the pickup and dropoff location, the latitude, longitude pairs were partitioned into clusters. The choice for the number of clusters used for this task was done in a simplistic manner i.e. NY region can be sub-divided into 62 counties so I used that as a frame of reference when choosing the number of clusters.

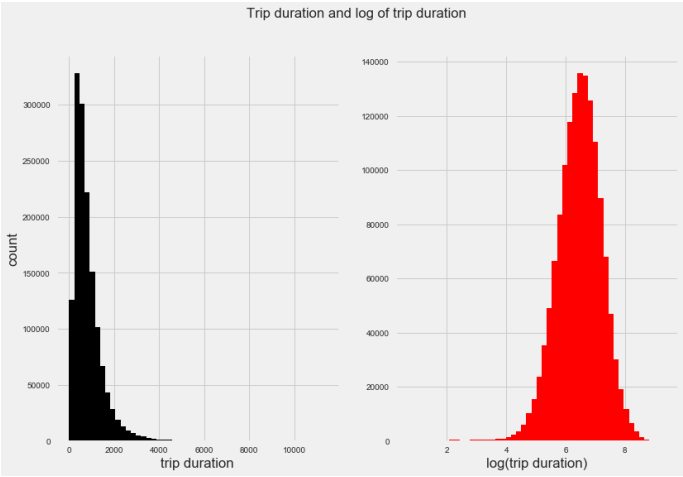



**Figura 5.**

- **Extracted features** like pickup weekday, pickup hour, pickup month from pickup datetime. As observed in exploratory visualization section, trip duration can vary depending upon the day/time so including these independent features into the model would help us explain the target variable.
- As an initial pass **(unoptimized model)**, I created the regressor object with RandomForestRegressor with default values of the parameters and fit it on the training data
- Subsequently, I iterated with these **parameters** (n estimators, max features, min samples leaf, min samples split) to fine tune the model and obtain better output
- This is the first time I'm using seaborn library so had to do research to make the codeset right for appropriate visualizations.

### 3.3. Refinement

- As mentioned in earlier section, in order to improve the accuracy of the unoptimized model, I will tweak below listed parameters[3]:
  - **n estimators:** This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower (default = 10)
  - **max features:** These are the maximum number of features Random Forest is allowed to try in individual tree (default = 'auto' i.e. use all features)
  - **min samples leaf:** The minimum number of samples required to be at a leaf node (default =1)
  - **min samples split:** The minimum number of samples required to split an internal node (default = 2)

We know that the validation RMSE will tend to reduce as we increase the parameter value of n estimators. However, we need to choose a reasonable value of n estimators beyond which the validation RMSE does not significantly reduce. To see this, observe in below table that RMSE value decreases from 0.3863 to 0.377 as we increased the value of estimators from 10 to 150. However, there is only a marginal improvement in RMSE when we change the estimator value from 100 to 150 so I chose to stop at this point.

**Tabela 2. Below table captures various iterations I ran on the model for different parameter values:**

| | n estimators | max features | min samples leaf | min samples split | RMSE |
|---|---|---|---|---|---|
| 1 | 10 | auto | 1 | 2 | 0.3863 |
| 2 | 10 | auto | 50 | 75 | 0.3788 |
| 3 | 10 | sqrt | 50 | 75 | 0.3864 |
| 4 | 50 | auto | 50 | 75 | 0.3773 |
| 5 | 100 | auto | 50 | 75 | 0.3771 |
| 6 | 150 | auto | 50 | 75 | 0.377 |
| 7 | 50 | sqrt | 50 | 75 | 0.3838 |
| 8 | 100 | sqrt | 50 | 75 | 0.3838 |
| 9 | 150 | sqrt | 50 | 75 | 0.3836 |

# 4. Results

## 4.1. Model Evaluation and Validation

During development, a validation set was used to evaluate the model. The final hyper-parameters were chosen because they performed the best among the tried combinations. Here is the final list of parameters for the chosen model

- n estimators: 150
- max features: 'auto'
- min samples leaf: 50
- min samples split: 75

In order to test the robustness of the model, I applied K fold cross validation strategy on the model. In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k minus 1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. With K=3 in this case, I obtained mean RMSE of 0.379 which is consistent with the score(0.377) I obtained earlier.
Most important features observed for Duration prediction are distance, pickup hour, center latitude, center longitude and pickup weekday.

**Tabela 3. From Random forest, I obtained the importance of features as follows:**

| | Feature | Random Forest Feature Importance |
|---|---|---|
| 1 | Haversine distance | 0.8228 |
| 2 | Pickup hour | 0.0575 |
| 3 | Center latitude | 0.0416 |
| 4 | Center longitude | 0.0291 |
| 5 | Pickup weekday | 0.0215 |
| 6 | Dropoff cluster | 0.0087 |
| 7 | Vendor ID | .0073 |

### 4.2. Justification

The random forest model outperform all other models used, as it manages to model the nonlinearities of traffic and location effect. Although the model accounts for the effect of pickup and dropoff locations, it has no way of modeling the effects of the locations along the route. A ride between two locations with high traffic can still be relatively fast if it goes through high-speed areas with little or no traffic. To account for the route, I computed the center latitude and longitude and fed that as independent factor in the model. Considering what is accounted for in the models, they are believed to predict duration relatively precisely.

**Tabela 4. Model comparison:**

| | Performance Metric | Benchmark Model | Unoptimized Model | Optimized Model |
|---|---|---|---|---|
| 1 | RMSE | 0.56 | 0.3863 | 0.3770 |

Performance metric of optimized model is significantly better than the benchmark model.

## 5. Conclusion

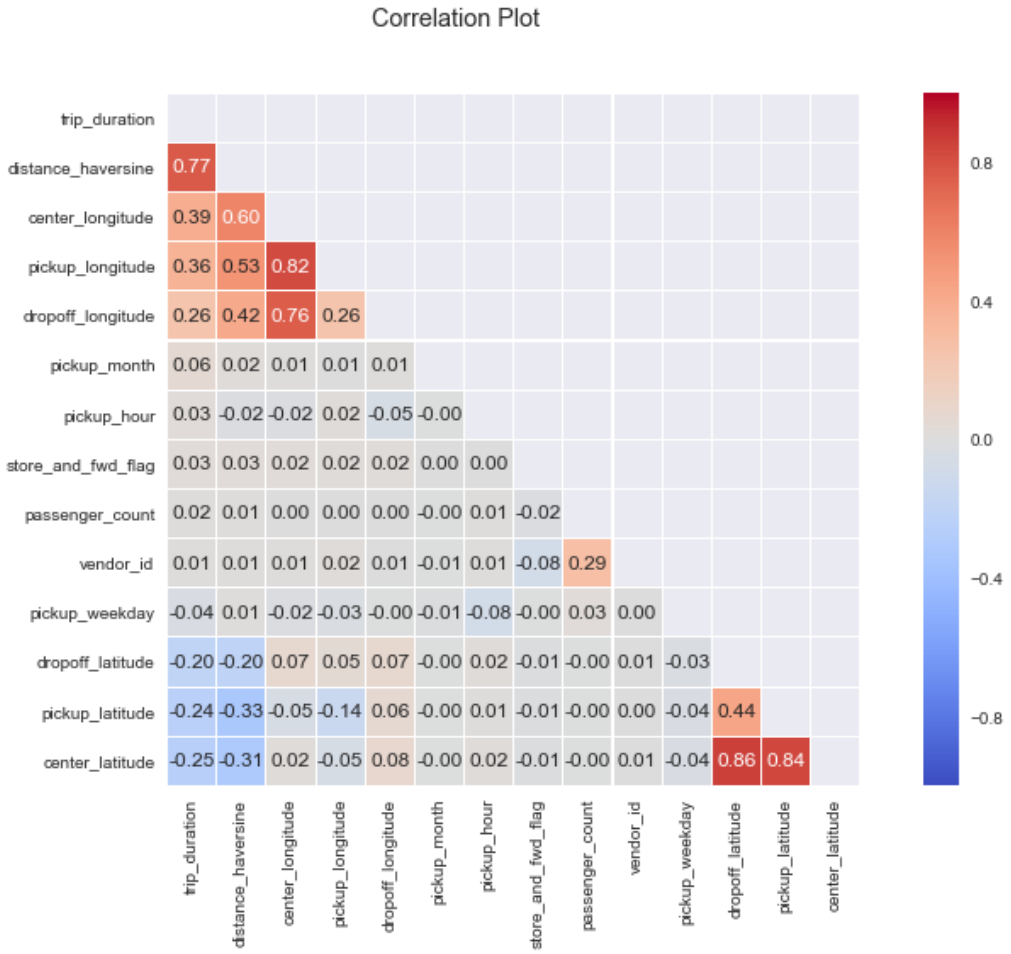
### 5.1. Free-Form Visualization

**Relationship between variables**

Below figure has the correlation heatmap which explains the relationship between all the features. Variables that are highly correlated are likely to convey redudant information and can be removed from the dataset. Reducing the data's dimentionality in this way will make the data easier to work with and allow for better model results. Not suprisingly, the correlation coefficient of the coordinate features: pickup latitude, pickup longitude, dropoff latitude, dropoff longitude,center longitude, center latitude indicates a linear relationship exists between them.

There is a weak positive correlation between the longitude variables and trip duration. There is also a weak negative correlation between the latitude variables and trip duration. Only feature with which trip duration is highly positively correlated is haversine distance which is also very intuitive. Otherwise, there doesn't appear to be much a of a

linear relationship between our target variable and the remaining features. This also explains why linear regression has a sub optimal performance when compared to Random Forest.




## 5.2. Reflection

The process used for this project can be summarized using the following steps:

- An initial problem and relevant, public datasets were found (Kaggle)
- The data was downloaded and preprocessed for feature engineering
- A benchmark (linear regression model) was created for prediction
- The optimized model was trained with RandomForestRegressor. It was trained using the data (multiple times, until a good set of parameters were found)
- Implemented k-fold cross validation strategy to test the robustness of the model

One of the most interesting aspect of the project was feature engineering . Challenge was to make effective use of the location co-ordinates without diluting its value. As explained in earlier section, I used k-means clustering to effectively capture the effect of pick up drop off coordinate. Also to capture on route dynamics, I computed center longitude latitude and included them as independent factor in the model (assumption here is that taxi would pass through this point to reach their final destination).

### 5.3. Improvement

Below are some ideas to improve upon the performance of the model which can essentially be categorized into:

- **Feature Generation**
  - Making use of **weather information** could help us explain the trip duration as snowfall, rainfall among others dictate the speed at which a vehicle is moving.
  - Similarly, having access to fastest route, second fastest route and path could help us explain the trip duration. This dataset can be generated from **Open Source Routing Machine**.
  - **Coordinate Transformation :** To further model the effect of the pickup and dropoff locations we could transform the coordinates [4]. Most of the streets and avenues in Manhattan are aligned in a grid structure. With the hypothesis that the avenue or street could explain some of the effect of the location, transforming the coordinates so that the splits in the random forest algorithm will be made aligned and perpendicularly to the avenues and streets, could potentially yield better predictions.
- **Algorithm**
  - We could develop a model based on neural nets / Multilayer perceptrons. This might not necessarily improve upon the performance but it's definitely worth giving it a shot given the problem statement.

## 6. References


1 James, G, D Witten, T Hastie, and R Tibshirani.*An introduction to statistical learning. Vol. 6.* , New York, Springer., 2013.

2 Friedman, J, T Hastie, and R Tibshirani.*The elements of statistical learning. Vol. 1.* , Berlin, Springer, 2001.

3 http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

4 Blaser, Rico, and Piotr Fryzlewicz. "Random rotation ensembles."*J Mach Learning Res 2* (2015): 1-15.

5 *Data source:* https://www.kaggle.com/c/nyc-taxi-trip-duration/data