**TAXI DEMAND PREDICTION**

*Project report submitted*

*in partial fulfilment of the requirement*

*for the degree of*

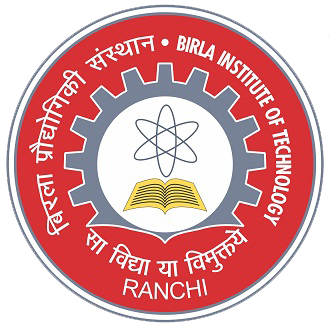
**Bachelor of Engineering**

By

**PRASHANT SHANDILYA (BE/6087/14)**

**ATUL VERMA (BE/6124/14)**

**MOHIT KUMAR (BE/6138/14)**



**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**

**BIRLA INSTITUTE OF TECHNOLOGY MESRA**

**(May, 2018)**

**CERTIFICATE**

It is certified that the work contained in the project report titled “**TAXI DEMAND PREDICTION** ” by “ **PRASHANT SHANDILYA, ATUL VERMA and MOHIT KUMAR**” has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree.

**--------------------------------------**

**Internal Guide**

**-----------------------------------                                ----------------------------------**

**I/C, Department of CSE                                External Examiner**

**DECLARATION**

We declare that this written submission represents my ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. we also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission.

we understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**( PRASHANT SHANDILYA )**

**( BE/6087/14 )**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**( ATUL VERMA )**

**( BE/6124/14 )**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**( MOHIT KUMAR )**

**( BE/6138/14 )**

Date: \_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of any project would be incomplete without mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

We would like to express our deep sense of gratitude to our **Director Prof. R. C. Jha,** and project mentor **Prof. Swapan Kumar Borat,** for his continuous effort in creating a competitive environment in our college and encouraging throughout this course.

We would like to convey our appreciations for giving us the occasion to go aboard on this topic and for his continued encouragement throughout this project work.

We must not forget to relay our gratitude towards all the faculty members, every staff member of the department of Computer Science Engineering and our friends for helping us directly or indirectly in the completion of this project work successfully.

**Table of Content**

**Abstract…………………………………………………………………………………………...**

**1. Introduction………………....……………………………………………………….……..**

1.1 Problem Statement

1.2 Scope of the project

1.3 Detailed Description

1.4 Tools and Resources

**2. Literature Review……...………………………………………………………………….**

2.1 Literature Survey

2.2 Roadmap of Proposed Taxi Prediction

**3. Methodology.…………………………………………………………...………………..**

3.1 Data collection

3.2 Data Cleaning

3.2.1 Latitude and Longitude

3.2.2 Trip Duration

3.2.2 Speed

3.2.4 Trip Distance

3.2.5 Total Fare

3.3 Data Preparation

3.3.1 Clustering

3.3.2 Time binning

3.3.3 Smoothing

3.4 Models

3.4.1 Baseline model

3.4.2 Regression models

3.4.2.1 Linear Regression

3.4.2.2 Random Forest Regressor

3.4.3 Xgboost Regressor

3.5 Results

**4. Snapshots…………………………………………………………………………………**

**5. Conclusion……………………………………………………………………………….**

**6. References……………………………………………………………………………….**

**List of figures**

1. Work flow of the project
2. Outliers of pickup latitude and longitude**…………………………………………**
3. Outliers of drop off latitude and longitude**……………………………………….**
4. Kmeans clustering example**………………………………………………………**
5. Visualization of clusters in the map**………………………………………………**
6. Plotting of cluster centres**.......................................................................................**
7. Features in the training dataset
8. Linear regression output
9. Random forest regressor output
10. Xgboost regressor output
11. Snapshots of the software(GUI)

**List of tables**

1. Features in the dataset.

**Abstract**

According to some survey conducted it has been observed that about 200 million taxi rides occur in New York City each year. This fact shows that the people of New York use the services of taxi in a frequency much higher than that of other cities. Exploiting an understanding of the location and time in which people use the taxi service can greatly help the taxi drivers to maximize their pickups. Thus, they will be automatically tempted to move to locations with greater number of pickups for a given time interval. It will actually prove to be very profitable for taxi drivers as they would be acquainted with the availability of pickups in their nearby locations or places where they could reach in search of passengers for the time being. Our project analyzes the NYC yellow taxi trip data for obtaining essential features such as location and time and makes various models such as random forest, linear regression and baseline. We also list the error of all these models using mean square percentage error as the error metric. Finally, we develop the GUI based software for showing the results For the GUI we have used tkinter.

.

**INTRODUCTION**

**Introduction**

**1.1Problem Statement** For a given region (latitude and longitude) and time instant predict the number of pickups.

**1.2Scope of our work**

The first step in implementing any program is to understand the strategic and tactical approach of a system. Therefore, it is helpful to know the scope of the said project which are:

* It is useful for taxi drivers as they are able to visualize which regions will have more pickups in consideration with the time.
* It could also provide valuable insights to city planners and taxi dispatchers.

**1.3Overview**

Taxi Demand Prediction aims at predicting the taxi ridership which would bring valuable insights to city taxi drivers and also to the city planners in answering queries such as where and how to position the cabs keeping in view the needs in the specified time so that both the taxi drivers and the people using taxi get their satisfaction. This project emphasizes on predicting the number of pickups in a given region and at a specified time interval. The regions are visualized into different clusters (using k means clustering algorithm) seeing the density of pickups using the latitude and longitude of a place and the entire time is converted into 10 minute time bins. For making time bins the time from the data is first converted into Unix time stamp (it is the number of seconds passed since January 1 1970). For accomplishing this project we have taken the data of Yellow taxi of New York City which is freely available on the New York Government website.

The Data set of Yellow taxi includes the following:

* pick-up and drop-off dates/times,
* pick-up and drop-off locations,
* trip distances,
* driver-reported passenger counts
* fare
* Speed.

*Input*: Date, 10 minute time window, latitude and longitude within the boundary of New York city.e.g. 25 March 2016 from 5:00pm to 5:10pm at coordinates (40. 75, -73.97).

*Output*: the number of pickups eg.125.

**1.4 Tools & Resources   
  
 Tools :**

1. Jupyter Notebook:The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code equations, visualizations and narrative texts. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning and much more.
2. Tkinter is a standard GUI (graphical user interface) package. Tkinter is a thin object-oriented layer on the top of python world.

**System Requirements :**

1. Microsoft Windows 10/8/7/Vista/2003/XP (incl.64-bit)
2. 4 GB RAM minimum
3. 8 GB RAM recommended
4. 1024x768 minimum screen resolution
5. Python 2.4 or higher, Jython, PyPy or IronPython

**LITERATURE REVIEW**

**2. Literature Review**

**2.1 Literature Survey**

With the rise of intelligent system and availability of data, taxi pick-up demand prediction has recently come into attention of many scholars. It is explicit that taxi demand of each zone changes from one time interval to another in real time. Hence, to capture this time variation in taxi demand, figuring out the correlation among the taxi data, and having real time prediction, time series model can be a strong statistical tool. For a univariate data, a well-known family of time series called ARIMA (Autoregressive Integrated Moving Average) can be beneficial, and it has been applied to many transportation related problems (Moghimi et al., 2017).

Some of the primary researches about taxi demand were to find factors influencing taxi demand. Schaller (1999) developed a citywide empirical time series regression model on NYC taxi to understand the relationship between taxicab revenue per mile and economic activity in the city, taxi supply, taxi fare, and bus fare. Afterward, Schaller (2005) tried to figure out the relationships between taxi demand and factors including city size, availability and cost of privately owned autos, use of complements to taxicabs, cost of taxi usage, taxi service quality, presence of competing modes, senior and disabled population.

Subsequently with the emergence of GPS technology, extensive researches about spatial information have been applied in transportation-related problems. GPS based system is also utilized on taxis of New York City to track them and to analyze taxi ridership with such data source. Yang and Gonzales (2017) processed the GPS taxi data of New York City and used negative binomial method to capture the variation of taxi pick-up demand. Six explanatory variables were used in their study including population, education, median age, median income per capita, employment by industry sector, and transit accessibility. Correa et al. (2017) performed empirical analysis to explore the spatial dependence between Uber and taxi pick-up data. Results from Moran’s I tests confirmed the significantly spatial correlation of both taxi and Uber demand.

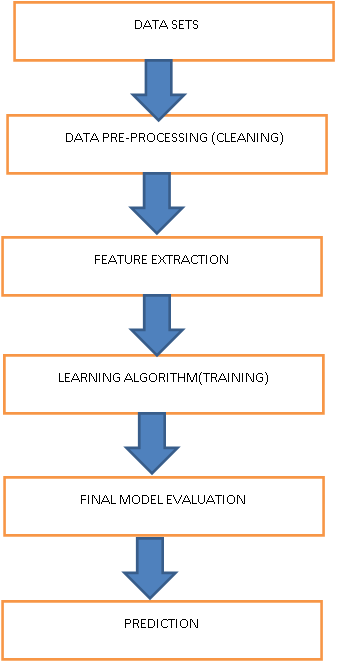
Moreira-Matias et al. (2013) proposed a methodology to predict short-term taxi demand at 30 min time intervals. Their methodology is an ensemble of three predictive models including Time Varying Poisson Model, Weighted Time Varying Poisson Model, and Auto Regressive Integrated Moving Average Model. They found that their proposed model outperformed to all three models if run individually. In the recent study done by Qian et al. (2017), a Gaussian Conditional Random Field (GCRF) model is presented to predict a short-term taxi demand. The proposed models together with 2 other algorithms (ARIMA and ANN) were run in 4 different scenarios to evaluate its performance. The results reported that the proposed model outperformed the two other algorithms with Mean absolute percentage error (MAPE) close to 0.1. In this paper, the case study was the same as used in Qian et al. (2017) study.

**2.2 Roadmap of Proposed Taxi Demand**

For accomplishing this task we first collected the data from the New York website. Now this dataset had many outliers or illegitimate values which needed to be removed hence, we first cleaned the data. For cleaning we analyzed each of the features one at a time and using statistical approaches such as box plot or percentile analysis we have removed the outliers.

After cleaning we have to divide the New York City into various regions. We have the latitude and longitude values of each point of the dataset. Hence, we divided the entire New York City into various clusters. Each of these clusters have a center point (unique latitude and longitude) and the clusters are visualized in such a manner that pickup density remains uniform so that our prediction becomes better.

As the clustering part is over, the entire time has to be divided into different time bins. For our analysis we have chosen 10 min time intervals. In order to create these time bins we first converted the pickup times of all the values into unix time stamp. The unix time stamp basically means the number of seconds it has passed from January 1 1970 to the current time. The advantage of using unix time stamp is that it is easy to divide the time into bins. Thus the entire pickup time was converted into 10 minute bins.

After the data preparation part is over we have to make various models for the project. We first started with simple baseline models such as simple moving averages, weighted moving averages and exponential weighted moving averages. In these models we have used two techniques for predicting; one using previous year values of same time region and time and another using previous time bin values of the same year. We also made various other models such linear regression, random forest and xgboost. For these last three models we first divided the data into training (70%) and test data (30%). We then calculated the error % of the predicted output and the trained output.

**Fig2.1 showing workflow of the project**

**METHODOLOGY**

**3. Methodology**

**3.1 Data Collection**

We began this project by first getting the dataset. The dataset was taken from the New York taxi and Limousines Commission.We have collected all yellow taxi trips data from jan-2016 to march-2016.

**3.1.1 Features in the dataset**

|  |  |
| --- | --- |
| **Field name** | **Description** |
| vendorID | A code indicating the TPEP provider that provided the record. |
| tpep\_pickup\_datetime | The date and time when the meter was engaged |
| tpep\_dropoff\_datetime | The date and time when the meter was engaged. |
| Passenger\_count | The number of passengers in the vehicle. This is a driver-entered value |
| Trip\_distance | The elapsed trip distance in miles reported by the taximeter. |
| Pickup\_longitude | Longitude where the meter was engaged. |
| Pickup\_latitude | Latitude where the meter was engaged. |
| RateCodeID | The final rate code in effect at the end of the trip. |
| Store\_and\_fwd\_flag | This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server.  Y= store and forward trip  N= not a store and forward trip |
| Dropofflongitude | Longitude where the meter was disengaged |
| Dropoff latitude | Latitude where the meter was disengaged |
| Payment\_type | A numeric code signifying how the passenger paid for trip. |
| Fare\_amount | The time-and-distance fare calculated by the meter. |
| Extra | Miscellaneous extras and surcharges. Currently, this only includes. the 0.50and0.50and1 rush hour and overnight charges. |
| MTA\_tax | 0.50 MTA tax that is automatically triggered based on the metered rate in use. |
| Improvement\_surcharge | 0.30 improvement surcharge assessed trips at the flag drop. the improvement surcharge began being levied in 2015. |
| Tip\_amount | This field is automatically populated for credit card tips.Cash tips are not included. |
| Tolls\_amount | Total amount of all tolls paid in trip. |
| Total\_amount | The total amount charged to passengers. Does not include cash tips. |
|  |  |

.

**Table 3.1 showing features in the dataset.**

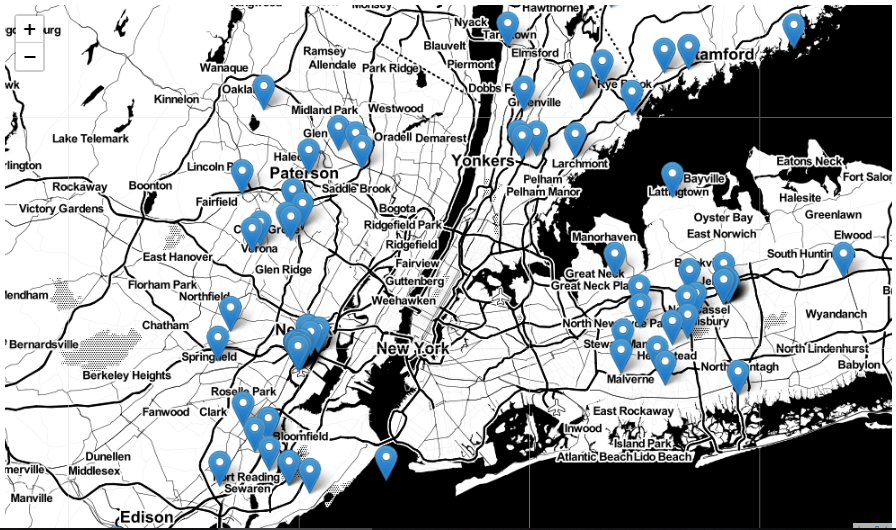
**3.2 Data Cleaning**

In this section we will be doing univariate analysis and removing outlier/illegitimate values which may be caused due to some error. For this we analyze the various features of our data set one at a time and visualize their proper values. The features are as follows:

**3.2.1 Latitude and Longitude**

We plotted the pickup latitude and longitude and drop-off latitude and longitude of the data points to visualize whether a pickup has happened outside of New York or any drop outside the boundaries of New York. Clearly all these data points are outliers for our analysis hence, these shall be removed.

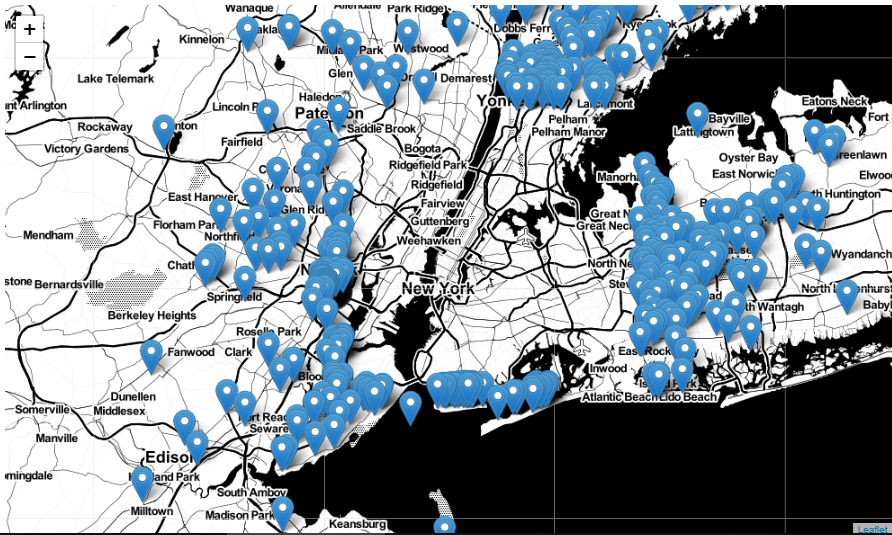
**Pickup Position (latitude and longitude)-**



**Fig 3.2.1(a)**

**Observation:-** As you can see above that there are some points just outside the boundary but there are a few that are in either South america, Mexico or Canada

**DropoffPosition (latitude and longitude)-**

****

**Fig 3.2.1(b)**

**Observation:-** The observations here are similar to those obtained while analysing pickup latitude and longitude as in fig 3.2.1(a).

**3.2.2 Trip duration**

The trip duration is defined as the difference between drop-off time and pickup time. According to New York Taxi and Limousine Commission, within a 24 hour window , a trip of more than 12 hours is not allowed. Thus if any trip which lasts for more than 12 hours(720 minutes as we will be using time in minutes for our analysis) are clearly unacceptable values for our analysis. We converted the time into Unix time stamp for better visualization. We then plotted a box plot of trip time and we found there are many outliers. We then also performed percentile analysis to estimate the values till the 99 percentile. Finally, we took all the trip values between 1 minute and 720 minutes (according to taxi rules) for the legitimate values.

**3.2.3 Speed**

The speed in our analysis is defined as 60\*(trip distance / trip time). Here it is multiplied by 60 as we have converted the time into unix time stamp. So to convert into minutes we multiplied it by 60.We plotted a box plot of speed for visualizing the outlier/illegitimate values and we found many such points. Further we performed percentile analysis to get valuable insights for our analysis and at 99.9 percentile we found that it is acceptable. After that we calculated the average speed which came out to be 12.45 miles/hour. This meant a cab driver could travel 2 miles in about 10 minutes.

**3.2.4 Trip distance**

Trip distance is the difference between the point of drop off and point of pickup. For analyzing the trip distance feature again we plotted a box plot and there were several values of trip distance which are not legitimate hence these were removed. Also we performed percentile analysis till the 99 percentile value and found that the last acceptable trip distance value was 23 miles.

.**3.2.5 Total fare**

It is the cost charged for the trip from the passenger. To visualize this feature we have plotted a box plot and found many outlier values. These values must be the ones either going outside of New York or coming outside of New York. In both these cases, these are not required for our analysis and hence these have to be removed. Again we looked deeply till the 99.9 percentile value of fare but we could not get a threshold for our analysis. So we decided to perform graphical analysis. We plotted the last 50 values (excluding its last two values) and using the elbow method we found the threshold.

**3.3 Data Preparation**

In this section we prepare our data into our usable input form. The Data preparation step includes the following:

**3.3.1 Clustering**

For dividing the city of New York into regions of our interests we have to make various clusters. For accomplishing this task we use the k-means algorithm. The k means algorithm is explained as:

K means clustering is used where we have un labeled data (data with not defined groups or categories). The motive of this algorithm is to find groups in the provided data and the number of groups is represented by a variable K. The k means algorithm works in a iterative manner and assigns each data point to each of the groups (K) on the basis of the features being provided. The algorithm is as:

Step 1: choose the number of clusters K

Step2: select at random K points the centroids (not necessarily from the data set)

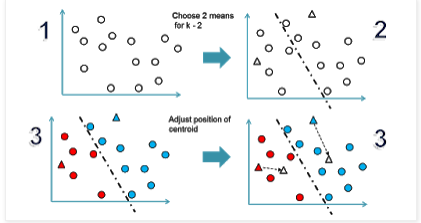
Step 3: assign each data point to the closest centroid 🡪 that forms K clusters.

Step4: compute and place the new centroid of each cluster.

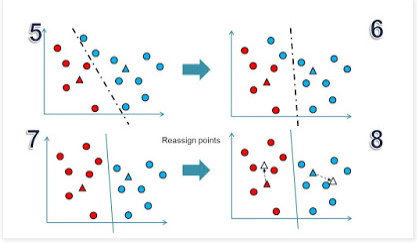
Step5: reassign each data point to the new closest centroid.

If any reassignment takes place, go to step 4, otherwise finish.

The diagrammatic visualization of k means algorithm is presented below (Fig 3.3(a) to Fig3.3(c)

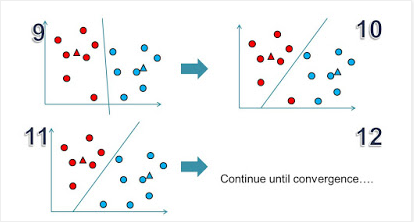


**Fig 3.3.1(a)**

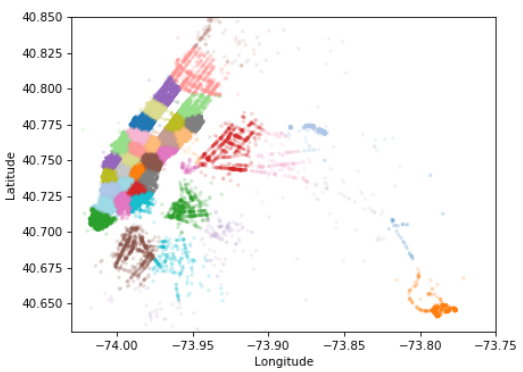


**Fig 3.3.1(b)**

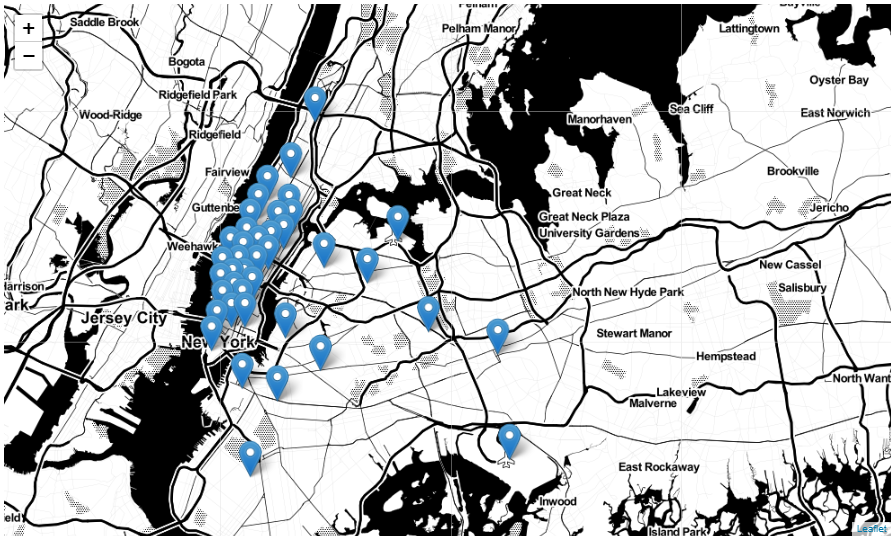
**Fig 3.3.1(c)**

From our analysis of average speed we found that on an average 12.45 miles can be covered in 1 hour, so in ten minutes a taxi driver can roughly cover 2 miles. Thus from this we can draw an important conclusion of maximum inter cluster distance for choosing the right number of clusters. Also it is undesirable to have too tiny clusters as it won’t prove to be effective from a driver’s perspective, hence taking into consideration we choose minimum inter cluster distance to be more than 0.5 miles. On these two constraints, we check for various values of K and for K=40 both these constraints are satisfied. So the optimal number of clusters chosen is 40.

.**Fig 3.3.1(d) below shows the clusters.**



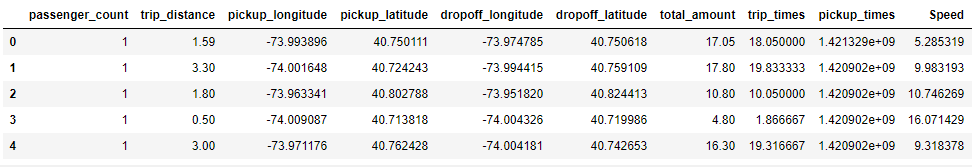
**For each cluster there is unique cluster center coordinate. Each position is the unique center of all 40 clusters.**



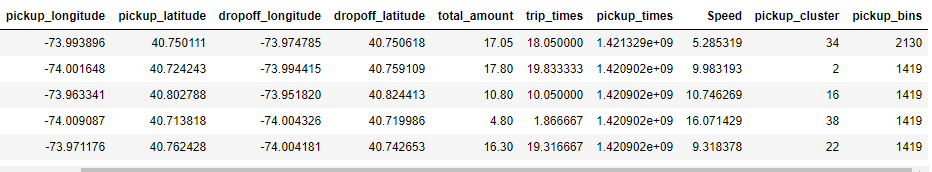
**Fig 3.3.1(e) plot of the cluster centres.**

**3.3.2 Time binning**

* In this section we divide the entire clustered data into sizes of 10 min bins.
* For this we convert the time in pickup time into Unix time stamp.
* Unix time basically means the number of seconds it has passed from jan 1 1970 from 00hrs.
* Representing time in unix time stamp makes it easy to break time into intervals.



**Fig 3.3.2(a)**



**Fig 3.3.2(b)**

* In this figure the last two columns show the cluster in which pickup has happened and the time bin in which this pickup falls

**3.3.3. Smoothing**

* In our prepared data with cluster id and time bin there are quite a few intervals in which there are zero pickups.
* These data points are not very useful and also they lead to a problem in analyzing the ratio feature in baseline models.
* So for countering this problem we performed smoothing as discussed. Suppose we have pickups in 3 consecutive time bins of t,t+10 and t+20 as 100,0 and 50 respectively. Now in pickup bin 2 (t+10) there are zero pickups hence it is not useful for us, so we take some values from previous and next bins to make the pickup curve smooth keeping in mind that the sum of all the pickups remains constant. In this case we take values from previous bin and so the 3 bins have 50,50 and 50 pickup values respectively.

We can come across three different situations where smoothing of data is needed.

These are as follows:

* + Case 1:(values missing at the start)   
    Ex1: \_ \_ \_ x =>ceil(x/4), ceil(x/4), ceil(x/4), ceil(x/4)   
    Ex2: \_ \_ x => ceil(x/3), ceil(x/3), ceil(x/3)
  + Case 2:(values missing in middle)   
    Ex1: x \_ \_ y => ceil((x+y)/4), ceil((x+y)/4), ceil((x+y)/4), ceil((x+y)/4)   
    Ex2: x \_ \_ \_ y => ceil((x+y)/5), ceil((x+y)/5), ceil((x+y)/5), ceil((x+y)/5), ceil((x+y)/5)
  + Case 3:(values missing at the end)   
    Ex1: x \_ \_ \_ => ceil(x/4), ceil(x/4), ceil(x/4), ceil(x/4)   
    Ex2: x \_ => ceil(x/2), ceil(x/2)

**3.4 Models**

Now we turn our attention towards the task of modeling in order to forecast the pickup density.

**3.4.1 Baseline model**

Here we are using exponential weighted moving average with two variations:

Using the previous year’s value of the same time bin and cluster id.

Using previous known values of the same year to predict future values.

**Exponential Weighted Moving Averages**

Through weighted averaged we have satisfied the analogy of giving higher weights to the latest value and decreasing weights to the subsequent ones but we still do not know which is the correct weighting scheme as there are infinitely many possibilities in which we can assign weights in a non-increasing order and tune the hyper parameter window-size. To simplify this process we use Exponential Moving Averages which is a more logical way towards assigning weights and at the same time also using an optimal window-size.

In exponential moving averages we use a single hyper parameter alpha (α) which is a value between 0 & 1 and based on the value of the hyper parameter alpha the weights and the window sizes are configured.

for e.g. If α=0.9α=0.9 then the number of days on which the value of the current iteration is based is~1/(1−α)=101/(1−α)=10 i.e. we consider values 10 days prior before we predict the value for the current iteration. Also the weights are assigned using 2/(N+1)=0.182/(N+1)=0.18 ,where N = number of prior values being considered, hence from this it is implied that the first or latest value is assigned a weight of 0.18 which keeps exponentially decreasing for the subsequent values.

R′t = α x Rt−1 + (1−α) x R′t−1 (using ratios)

P′t = α x Pt−1 + (1−α) x P′t−1 (using previous values)

**3.4.2 Regression models**

Before we start predictions using these regression models we take 3 months of pickup data into account and split it such that for every region we have 70% data in train and 30% in test, ordered date-wise for every region. The various regression models are as follows:

**3.4.2.1 Linear Regression**

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response ( or dependent variable) and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. For more than one explanatory variable, the process is called multiple linear regression. (This term is distinct from multivariate linear regression, where multiple correlated dependent variables are predicted, rather than a single scalar variable.)

In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models. Most commonly, the conditional mean of the response given the values of the explanatory variables (or predictors) is assumed to be an affine function of those values; less commonly, the conditional median or some other quantile is used. Like all forms of regression analysis linear regression focuses on the conditional probability distribution of the response given the values of the predictors, rather than on the joint probability distribution of all of these variables, which is the domain of multivariate analysis.

A simple regression problem (a single x and a single y), the form of the model would be:

y′=b+w1x1where:

y′ is the predicted label (a desired output).

b is the bias (the y-intercept), sometimes referred to as w0.

w1 is the weight of feature 1. Weight is the same concept as the "slope" m in the traditional equation of a line.

x1 is a feature (a known input).

Although the above modelmakes use of only one feature, a more sophisticated model might rely on multiple features, each having a separate weight (w1, w2, etc.). For example, a model that relies on three features might look as follows:

y′=b+w1x1+w2x2+w3x3

**3.4.2.2 Random forest Regressor**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees  at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set.

The pseudocode for random forest algorithm can split into two stages.

Random forest creation pseudocode.

Pseudocode to perform prediction from the created random forest classifier.

First, let’s begin with random forest creation pseudocode

Random Forest pseudocode:

* Randomly select “k” features from total “m” features.

Where k << m

* Among the “k” features, calculate the node “d” using the best split point.
* Split the node into daughter nodes using the best split.
* Repeat 1 to 3 steps until “l” number of nodes has been reached.
* Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

The beginning of random forest algorithm starts with randomly selecting “k” features out of total “m” features. In the image, you can observe that we are randomly taking features and observations.

In the next stage, we are using the randomly selected “k” features to find the root node by using the [best split](https://dataaspirant.com/2017/01/30/how-decision-tree-algorithm-works/) approach.

The next stage, We will be calculating the daughter nodes using the same best split approach. Will the first 3 stages until we form the tree with a root node and having the target as the leaf node.

Finally, we repeat 1 to 4 stages to create **“n”** randomly created trees. This randomly created trees forms the random forest.

Random forest prediction pseudocode:

To perform prediction using the trained random forest algorithm uses the below pseudocode.

* Takes the **test features** and use the rules of each randomly created decision tree to predict the oucome and stores the predicted outcome (target)
* Calculate the **votes** for each predicted target.
* Consider the **high voted** predicted target as the **final prediction** from the random forest algorithm.

**3.4.2.3 Xgboostregressor**

XGBoost (Extreme Gradient Boosting) is an optimized distributed gradient boosting library. The two reasons to use XGBoost are:

* Execution Speed.
* Model Performance.

**3.5 Results**

For measuring the performance of the models we have used mean absolute percentage error and mean square error.

**Baseline model**

Exponential Moving Averages (Ratios) –

MAPE: 0.177835501949

MSE: 378.34610215053766

Exponential Moving Averages (2016 Values) –

MAPE: 0.135091526367

MSE: 159.73614471326164

MAPE – Mean absolute percentage error

MSE – Mean square error

**Regression models**

Before we start predictions using the tree based regression models we take 3 months of 2016 pickup data and split it such that for every region we have 70% data in train and 30% in test, ordered date-wise for every region

Preparing data to be split into train and test, The below prepares data in cumulative form which will be later split into test and train-

Number of 10min indices for jan 2016 = 24\*31\*60/10 = 4464

Number of 10min indices for feb 2016 = 24\*29\*60/10 = 4176

Number of 10min indices for march 2016 = 24\*31\*60/10 = 4464

regions\_cum: it will contain 40 lists, each list will contain 4464+4176+4464 values which represents the number of pickups

Size of train data – 9169

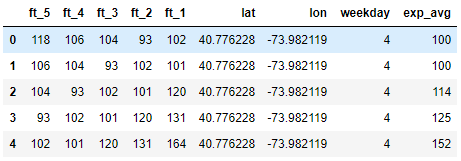
Size of test data- 3929

**Features**

Number of data clusters 40 Number of data points in trian data 9169 Each data point contains 5 features

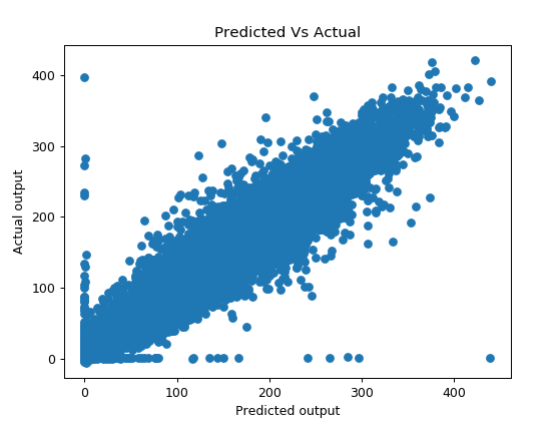
Number of data clusters 40 Number of data points in test data 3930 Each data point contains 5 features

These 5 feature are previous pick up value at that position. We have total 8 features including exp\_avg that we have calculated in base line model.



**Fig 3.4**

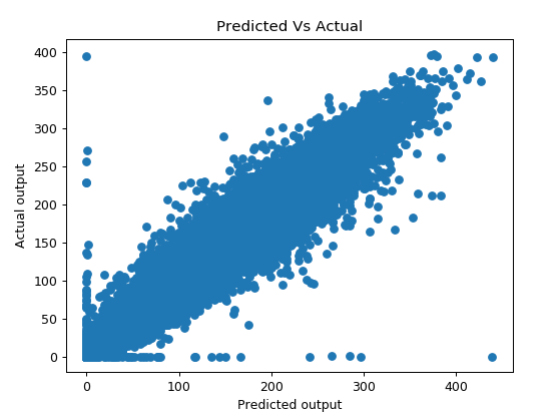
The Output is shown below between actual output and predicted output for **linear regression**.



**Fig 3.4.2(a)**

**Random forest regressor**

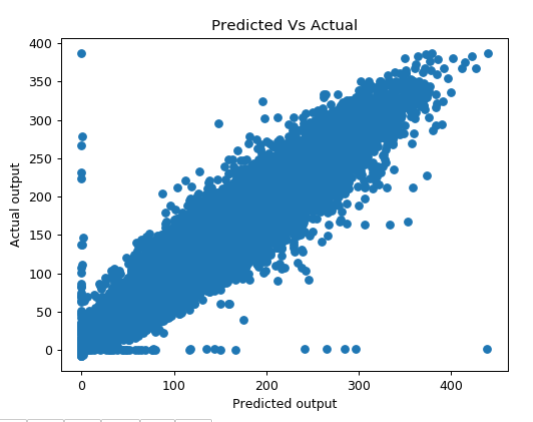
Size of train data – 9169, Size of test data- 3929



**Fig 3.4.2(b)**

**Xgboostregressor**

Size of train data – 9169, Size of test data- 3929



**Fig 3.4.2(c)**

### Calculating the error metric values for various models

We have calculated error metric using MAPE (Mean absolute percentage error).

**1. Baseline Model**

Train: 0.140052758787 Test: 0.136531257048

**2. Linear Regression**

Train: 0.13331572016 Test: 0.129120299401

**3. Random Forest Regression**

Train: 0.0917619544199 Test: 0.127244647137

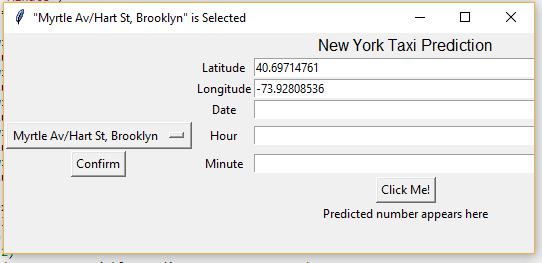
**4. XgBoostREgression**

Train: 0.129387355679 Test: 0.126861699078

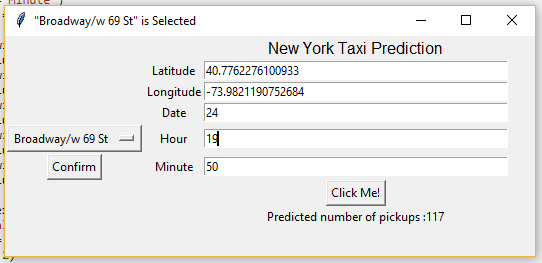
**GUI**

By using Tkinter, we have made GUI for predicting the no of pickups in a given region at given time bin.

Here you have to select any of the 40 clusters in new York city, or you can give any Latitude and Longitude in range of New York city.

****

After selecting the Lat and Lon, you have select date (as March is default month), and hour and minute and then Click Me!.

****

As the output says, In Broadway/w 69, the number of predicted pickup is 117 on 24th March at 19:50.

**Conclusion**  
In this project we have predicted the pickup values for a given latitude and longitude of New York City within a10 minute time window.

The exponential moving average also gave about 87% accuracy hence, we also took this value and added as a feature for further prediction using the regression models.

Among the regression models (linear regression, random forest and xgboost) all gave quite good results but the xgboost model gave the least difference between train output and test output. The accuracy of all these models are close to 88% but the most accurate is the xgboost in our case.

Seeing the output of pickups in different clusters and in different time instants it has been observed that the pickup density is high in the morning (especially when there is office hours) it then somewhat drops in the afternoon, it again starts to increase somewhat in the evening and at night again the pickups increase and continue till late night.

Though this is the general trend in majority of the clusters there are some clusters where the pickups don’t increase very much. The reason might be that they may be too far from the locations of offices and restaurants. The other reason could be that these locations are not very densely populated as compared to the others.

**REFERENCES**

**References**

[1] Todd W. Schneider “Analyzing 1.1 Billion NYC Taxi and Uber Trips, with a Vengeance” <http://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/>

[2] Yunrou Gong, Bin Fang, shuo zhang and Jingyu Zhang, Predict New York City Taxi Demand, [http://blog.nycdatascience.com/student-works/predict-new-york-city-t axi-demand/](http://blog.nycdatascience.com/student-works/predict-new-york-city-t%20axi-demand/)

[3] <http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml>

[4] <https://www.flickr.com/places/info/2459115>

[5] <https://www.unixtimestamp.com/>

[6] <https://docs.python.org/3/tutorial/>

[7] <https://www.youtube.com/watch?v=2uCXIbkbDSE>

[8 ]<https://www.w3schools.com/python/default.asp>

[9 ] <https://www.python-course.eu/python_tkinter.php>

[10 ] [www.tutorialspoint.com/python/python\_gui\_programming.htm](http://www.tutorialspoint.com/python/python_gui_programming.htm)