Journal of Computer Allied Intelligence
ISSN: 2584-2676
Vol.02 (01), Feb 2024, pp.26-45

Research Article

Data Analysis and Fair Price Prediction Using Machine Learning Algorithms

K. VinayKumar[1], Santosh N.C [2,*] and Narasimha reddy soor [3]

[1,2] Assistant Professor, Department of CSE, Kakatiya Institute of Technology and Science, Warangal,India.
[3]Associate Professor, Department of CSN, Kakatiya Institute of Technology and Science, Warangal, India
[*]Corresponding Author: Santosh N.C. Email: *ncs.cse@kitsw.ac.in*
Received: 31/01/2024; Accepted: 22/02/2024.
DOI: https://doi.org/10.69996/jcai.2024004

Abstract: Data Analysis is the main important subject in recent times as the ongoing demand for it is growing accordingly with the huge amounts of data we get from many sources, All the huge data we get should be properly analyzed so that the information will be used accordingly to its needs. In this paper, the main objective is to analyses the data that is taken from uber related data from a csv file which is already available in the outside world. In addition to the analyzing of data we are also including two features to our project, from which the first one is fare price detection and the second one goes with optimal allotment of a cab using appropriate machine learning algorithms. we used k- means clustering, Db Scan for optimal allotment of a cab. We used Linear regression, Logistic Regression, Random Forest algorithms, Decision Tree Algorithm for fair price detection. We are also finding the best algorithm for increasing the accuracy of selection using unsupervised algorithms which is the best moto of our project. The additional feature we are wanting to add in to this mechanism of analyzing the data is to use an android app developed by us in order to receive the required data from the users and perform various actions on it to obtain a best result for segmented two operations.

Keywords: Data analysis; uber data; fare price detection; machine learning algorithms; accuracy.

1 Introduction

First knowing the importance of analyzing, visualization is very well required in order to perform any actions on data that is taken from .csv files or any other file formats. In one word we can say that the main idea is to give reliable and accurate data to the end users after complete analyzation [4].

Data Visualization: Better Understanding of data is the main purpose of data visualization and techniques. So as we are having bulk amount of data from Uber [1] which is in the form of tables, which is very difficult to understand and read as so we use some tools like bar graphs , histograms, split graphs to understand the data exactly.

Uber related Data is taken from uber because it has given its New York data from the year 2018 between the months of November and December with no cost [1]. Uber provide best services to the customers with best satisfaction and optimal in providing the requirements (basic needs) effectively.

2 Literature Survey

Generally when it comes to data analysis every one checks for an existing data that is present on various platforms and do perform any operations to get their results from it after reading and understanding the data[4] , But it will be difficult to understand what exactly the already existing data contain for which we have to perform many operations like data virtualization for better understanding and data cleaning for reducing the noisy data, All these are costly and time taking. Hence we have developed an application which takes the only necessary data for detecting fare price[12] and optimal cab positioning this reduces the unwanted data and increases the understand ability of data that we receive.

Barlow, H. B.,[13] Explained the details about the Unsupervised learning eliminates the need for labelled data and manual handcrafted feature engineering enabling general, more flexible and automated ML methods which highly needed for the development of highly positive results with great accuracy and improvements.

From our findings [15] Unsupervised algorithms like K-means, Clustering will be the better to directly apply to the classification or regression problem and find the unknown values easily. So including such algorithms will make better results as well. These methods also do not require more number of parameters or labels of data to process, and also only depends on the unique training mechanism. Possibilities that can't be found out using the human can be easily determined by these Unsupervised algorithms

Rokach,L,.&Maimon,O.,[14]Details about clustering methods that are required for data mining[17] .which begins with comparing two objects about the similarity and processing for the next. Data mining is an important thing that one should more concentrate on when it comes to considering huge datasets. All the methods that Rokach and Maim on explained about brings a self-contained review of the concepts and the mathematics underlying clustering techniques.

The Linear Regression algorithm is used in fair price detection. But when it is compared with other algorithms like random forest the accuracy is increased similarly for optimal cab assigning we used k-means clustering and DB-scan algorithms K-Means was found to work effectively as DB-Scan is not working well in conditions like more rowed data which exceeds 5000 rows. Hence K-means has chosen.

For the fair price detection we have used dataset, downloaded from Kaggle and we performed the data cleaning and data visualization techniques and we applied Linear regression, Decision Tree and Random Forest algorithm. There are some parameters like weather, type of vehicle, day or night which affects the fair price. if you give these details then a fair price is generated and out of all used algorithms the Random Forest algorithm gave the best efficiency so we are using that algorithm to detect fair price based on source and destination

3 Existing System

Visualising the uber related data from kaggle using pictorial representations and sketches like bar graphs, plots, in the same way considering the analysis of the respective information by taking multivariate, univariate and bivariate analysis is seen in the already existing system. But the proper understanding of the Data set and ruling out the possible faulty data is not clear for users understanding.

Analyzing the uber price and position of the cab using machine learning algorithms that are not accurate to consider so better algorithms must be taken and should increase the efficiency of

the system. Implementing other unsupervised algorithms will build the profound positive results for the increment of the performance.

4 Proposed System

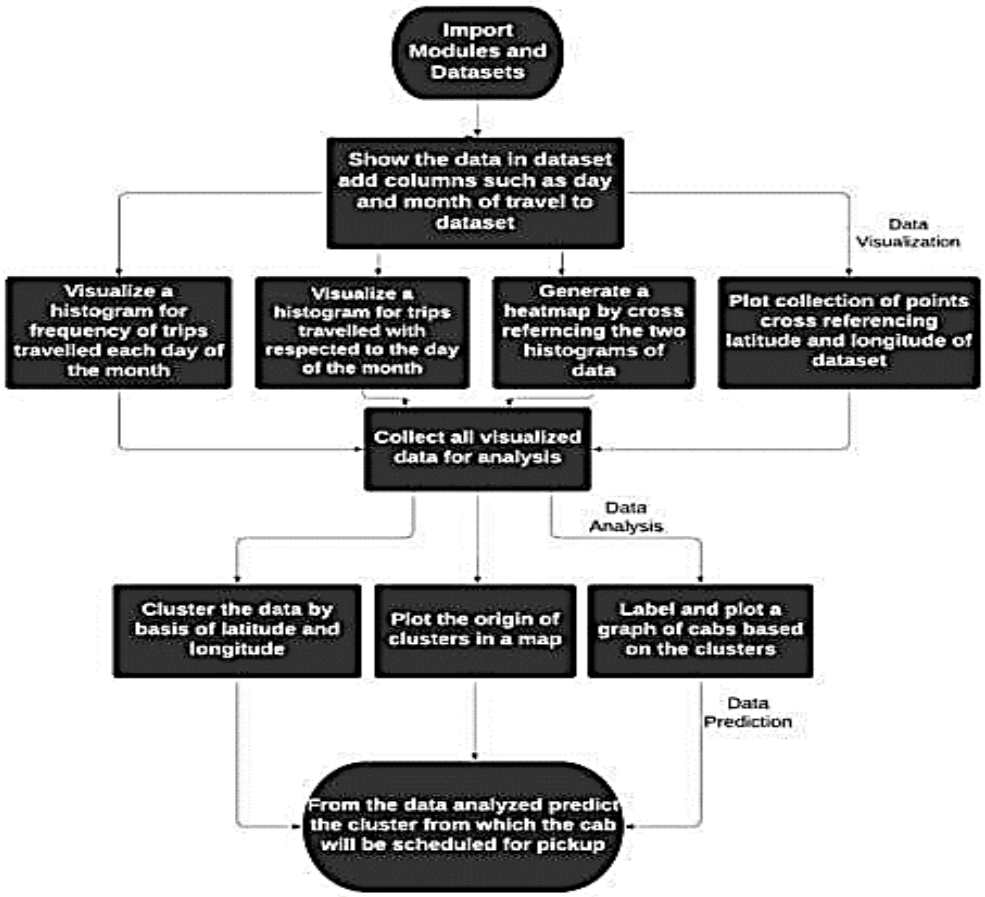


Figure 1: Flow diagram of proposed system.

The main task we are considering is predicting to which cluster of centroids does the new source comes and we are showing the mechanism of cab assigning, for this we have used the k-means clustering and Db-scan algorithms. We have used the folium package to plot the coordinates in the map and we have applied the k-means clustering to the dataset which we have downloaded from the Kaggle and later we have prepared the dataset of our own, where we designed an app where it collects the live location of the user and the data is stored in our google firebase and we have converted the file into j-son format and we collected the latitudes and longitudes and converted into a python Data frame as a dataset. if you give a new location it predicts into which cluster the given location comes.

For the fair price detection we have used dataset, downloaded from Kaggle and we performed the data cleaning and data visualization techniques and we applied Linear regression,

Decision Tree and Random Forest algorithm. There are some parameters like weather, type of vehicle, day or night which affects the fair price. if you give these details then a fair price is generated and out of all used algorithms the Random Forest algorithm gave the best efficiency so we are using that algorithm to detect fair price based on source and destination.

Android App for live data Tracking: Our team has developed an innovative app that allows users to share their live location with ease. When you open the app if you are a new user then it asks for registration [18][19]. Then the app asks for the user's permission to access their location data and then stores their latitude and longitude coordinates. This information is securely stored in Google Firebase, a cloud-based database that ensures the user's data is safe and easily accessible.

The primary purpose of this app is to enable clustering of user data [20][21]. By collecting and analyzing this location data, we can create clusters of users based on their geographical proximity. This clustering mechanism helps us to provide customized services and recommendations to users based on their location and preferences.

The app has been designed with a user-friendly interface that makes it easy for users to share their location and view their data on a map. Overall, our app provides a convenient and secure way for users to share their live location data and benefits both users and service providers by allowing for customized services based on location data.

Here we are illustrating the mechanism of how a cab is assigned in the popular ride-sharing app Uber. When a user enters their location in the Uber app, the system uses a similar clustering mechanism to identify the closest available driver. This helps to ensure that users receive the fastest and most efficient service possible.

Here the data of users stored in firebase is converted into json format and the unnecessary information is deleted and only the username, latitudes and longitudes are retained and passed as parameters for the k-means clustering algorithm.

Hence a model is trained and it is tested. When a new users' details are given as input then he is assigned to a new cluster.

There are some parameters [2] like weather, type of vehicle, day or night which affects the fair price. if you give these details then a fair price is generated and out of all used algorithms the Random Forest algorithm gave the best efficiency so we are using that algorithm to detect fair price based on source and destination.

First know what are all the contents we need so they are

- Dataset contains locations and car rides, latitudes and longitudes of location [3][6].
- Live data app is a j-son file which contains information about the live location of the app users.
- Uber cab optimal positioning is a Jupiter notebook file which contains code in it.

Figure 2: Contents of system for fair price detection.

- Rideshare kaggle is the dataset which contains information about the car rides and it contains different price affecting columns like weather ,type of vehicle, type of sharing vehicle etc.[6]
- Uber Data Analysis is a Jupiter notebook file which contains code in it.

5 Algorithms

5.1 K-Means Clustering Algorithm:

- K-Means clustering algorithm is used to assign the cab for the customer. It shows the optimal positioning of cabs if we provide the source.
- Each cluster so far formed contains a centroid, which is why it is also known as centroid based algorithm.
- It is mainly used to reduce the sum of the distance between the datapoint and the centroid.
- It takes the un labelled dataset as input and forms the clusters by following iteration method until and unless it finds the best clusters.
- The clusters so formed are belonging to set which contains the similarities.

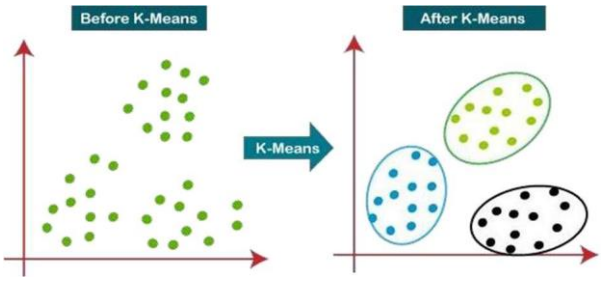


Figure 3: K-Means clustering for assigning a cab

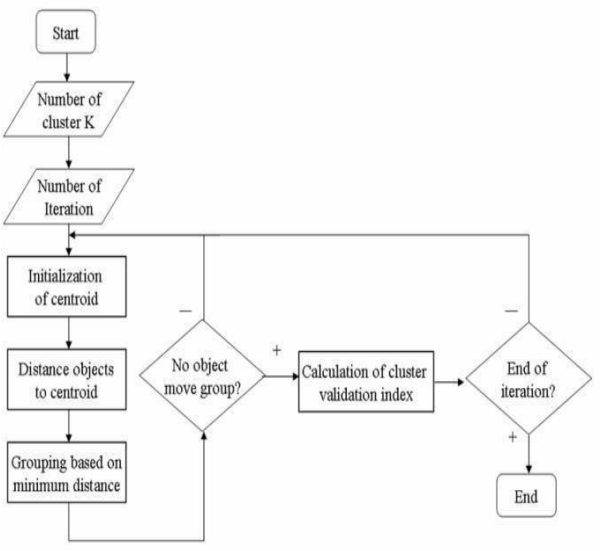


Figure 4: Algorithm of K-Means clustering

Algorithms For Fair Price Detection:

5.2 Linear Regression:

- Linear regression is an algorithm which is used to find the relationship between the dependent variables and the independent variables.
- This algorithm helps in predicting the value of a variable with the help of another variable.
- Here the independent variable is the one which is used to predict the other variable, and the dependent variable is something with the help of other variables its value is predicted.
- The dependent variable is called as and the independent variable is also termed as
- It is shown as the line in the format y= ax+b , here a,b,x are the independent variables and the y is the dependent variable.

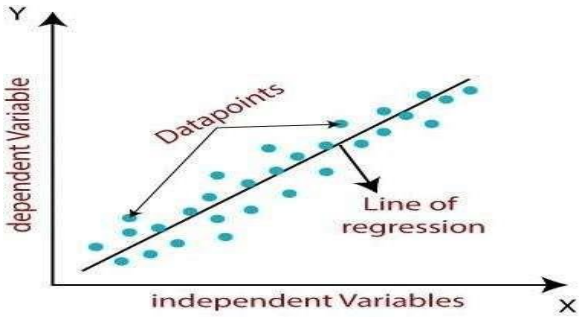


Figure 5: Linear regression graph representation.

5.3 Decision Tree Algorithm:

Decision tree is a supervised machine learning algorithm.[11] [16]

It is same like tree structure which is used for classification and regression. It has three main parts:

1) Internal nodes
2) branches
3) leaf node

Internal nodes denotes the dataset's features, branches represents the decision which are made and finally the outcome is denoted by the leaf node of the decision tree.

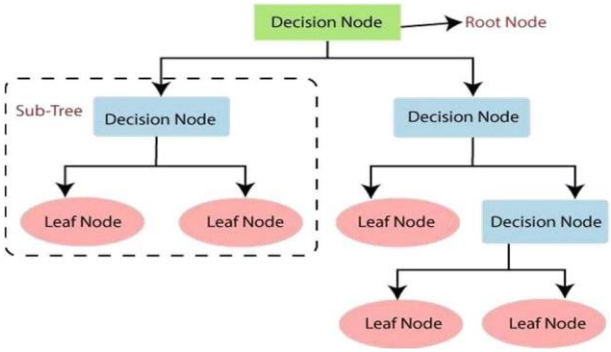


Figure 6: Decision tree representation.

5.4 Random Forest Algorithm:

Random forest is a supervised machine learning algorithm which is used for the purpose of classification of data [11].

It is like the combination of the decision tree algorithms.

In the dataset is divided into some random number of sub datasets and the decision trees are formed for all the sub datasets.

And finally, the average of all the decision trees are made and the conclusions are drawn from that.

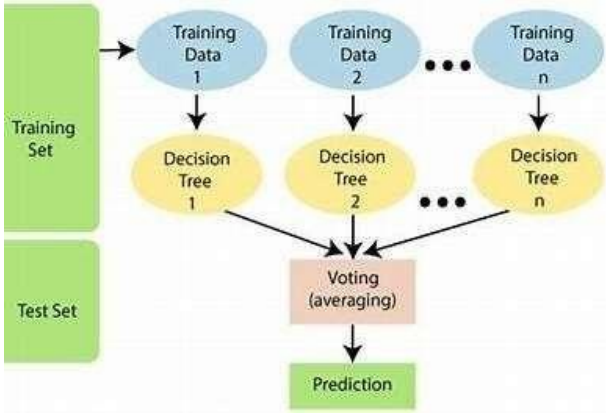


Figure 7: Random forest tree representation

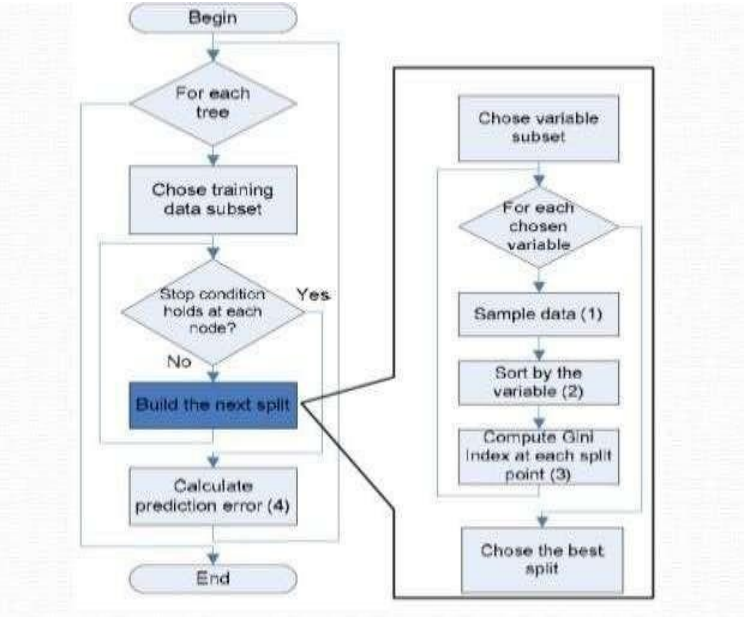


Figure 8: Algorithm of random forest

Allowing system operators to divide traffic, control flows for optimum efficiency, and start testing the new settings and services. Benefits of OpenFlow

- Programmability.

6 Code Implementation Optimal Cab Positioning

Importing required libraries:

Input:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Pandas are used for the analysis [4], seaborn are used for the graphical visualization and representation of data, matplotlib is used for arrays and linear algebra.

Creating data frames:

Input:

```
apr14 = pd.read_csv('Dataset/uber-raw-data-apr14.csv')
may14 = pd.read_csv('Dataset/uber-raw-data-may14.csv')
jun14 = pd.read_csv('Dataset/uber-raw-data-jun14.csv')
jul14 = pd.read_csv('Dataset/uber-raw-data-jul14.csv')
aug14 = pd.read_csv('Dataset/uber-raw-data-aug14.csv')
sep14 = pd.read_csv('Dataset/uber-raw-data-sep14.csv')

merged_df = pd.concat([apr14, may14, jun14, jul14, aug14, sep14])
merged_df.head()
```

Here the dataset is collected from kagel which is already existing dataset of the new York city of the months November and December which is downloaded in the form of csv files(comma separated values).

Output:

	Date/Time	Lat	Lon	Base
0	4/1/2014 0:11:00	40.7690	-73.9549	B02512
1	4/1/2014 0:17:00	40.7267	-74.0345	B02512
2	4/1/2014 0:21:00	40.7316	-73.9873	B02512
3	4/1/2014 0:28:00	40.7588	-73.9776	B02512
4	4/1/2014 0:33:00	40.7594	-73.9722	B02512

Rideshare histogram

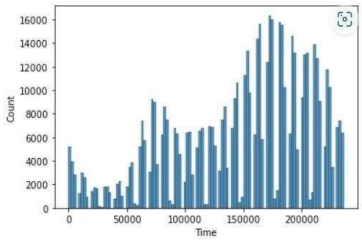
The above data segregates the data collected from kaggle into the month and time, latitude and longitudes divisions for further data analysis. The actual collected includes the data of the

uber in the months of November and December in the city of New York. The histogram [6] [7] represents the time and count of the rides. The time morning or evening time of the day and weather conditions like suny, rainy, cloudy. Based on the weather cabs availability from the particular location will be more or less.

Output:

```
sns.histplot(current_df['Time'])
```

```
<AxesSubplot:xlabel='Time', ylabel='Count'>
```



Filtering Morning and Evening Rides

The data is filtered into morning and evening rides [6] [8] for making clustering process easy. The actual data collected is the combination of the rides of all the parts of the day in the New York City. By dividing them into morning and evening data makes the data analysis part easy.

Input:

```
morning_df_idx = (current_df['Time'] > 50000) & (current_df['Time'] < 110000)
morning_df = current_df[morning_df_idx]
evening_df_idx = (current_df['Time'] > 150000) & (current_df['Time'] < 220000)
evening_df = current_df[evening_df_idx]

morning_df
```

Output:

	Date/Time	Lat	Lon	Base	Time
30	2014-04-01 05:08:00	40.7141	-74.0094	B02512	50800

```
from sklearn.cluster import KMeans
import numpy as np
```

Finding clusters:

```
n_clusters = 6
model = KMeans(n_clusters=n_clusters, init='random', max_iter=3
model.fit(morning_df[['Lat','Lon']])
```

564031	2014-04-30 10:57:00	40.7710	-73.8659	B02764	105700
564032	2014-04-30 10:58:00	40.7300	-73.9867	B02764	105800

111422 rows × 5 columns

Importing K-Means Input:

Output:

```
array([[ 40.66319511, -73.77366135],
       [ 40.73293011, -73.99564334],
       [ 40.69828782, -74.20574989],
       [ 40.77335945, -73.96742616],
       [ 40.68789427, -73.96537129],
       [ 40.79334938, -73.86509224]])
```

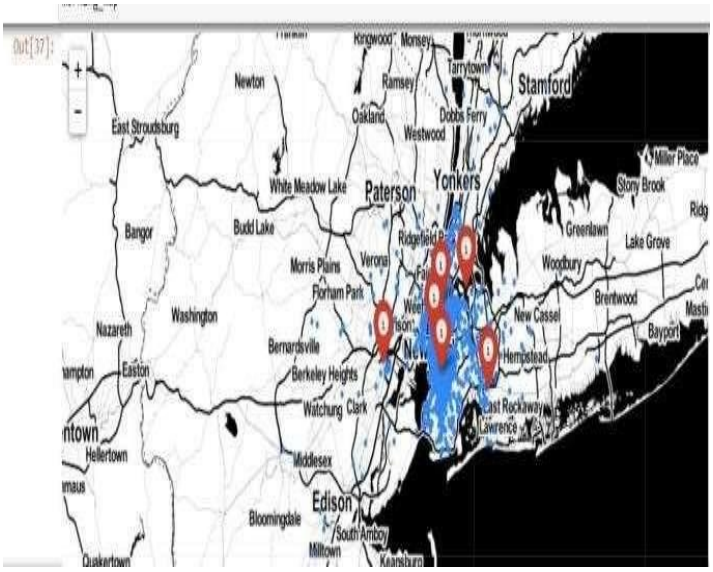
The output indicates the centriods of the clusters so formed based on the longitude and latitude locations. These clusters are having the centriods based on which the current location of the user is used to locate the user in that particular cluster by considering the latitudes and longitudes mapping of the user geographically.

Representing the Centroids of clusters with folium:

Input:

```
for i, coordinate in enumerate(morning_centroids):
    folium.Marker(tuple(coordinate), popup='Centroid {}'.format(i+1), icon=folium.Icon(color
='red')).add_to(morning_map)
morning_map
```

Output:

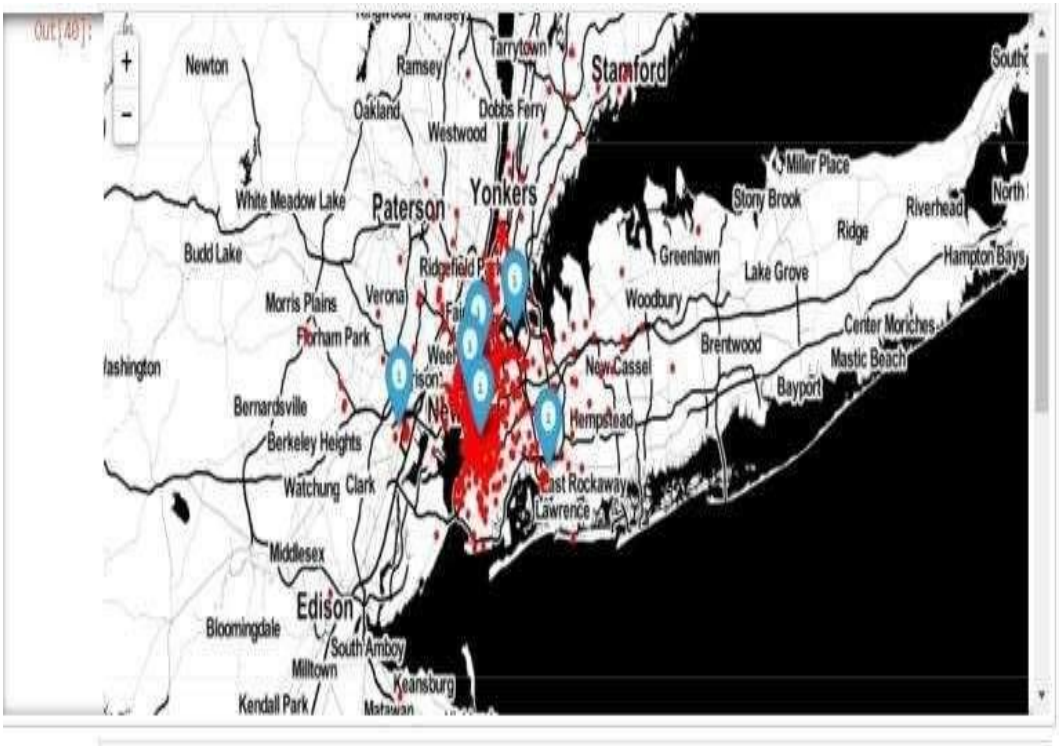


For evening:

Input:

```
n_clusters = 6
model = KMeans(n_clusters=n_clusters, init='random', max_iter=300)
model.fit(evening_df[['Lat','Lon']])

output: KMeans(init='random', n_clusters=6)

evening_centroids = model.cluster_centers_
evening_centroids

Output:
array([[ 40.79451853, -73.86956433],
       [ 40.76288665, -73.97458095],
       [ 40.68838102, -73.9681331 ],
       [ 40.69961741, -74.20066416],
       [ 40.73069857, -73.99943104],
       [ 40.6571523 , -73.77434702]])

for i, coordinate in enumerate(evening_centroids):
    folium.Marker(tuple(coordinate), popup='Centroid {}'.format(i+1), icon=folium.Icon(color
='blue')).add_to(evening_map)
evening_map
```

Output:



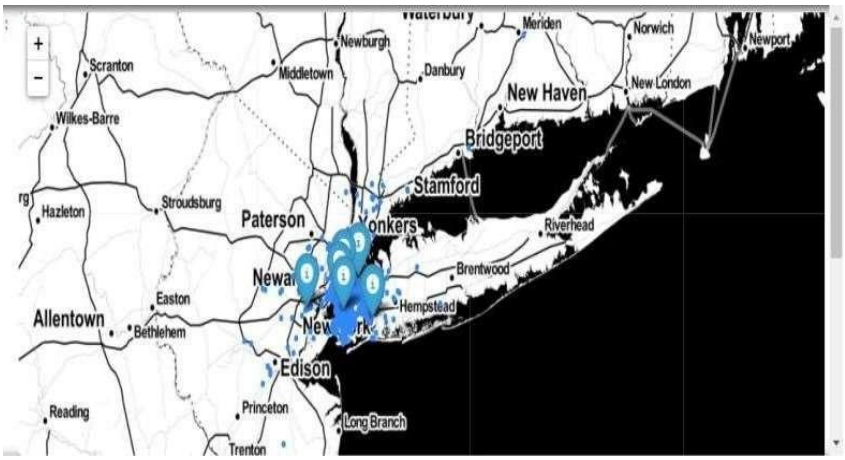
For morning:

Input:

```
n_clusters = 6
model = KMeans(n_clusters=n_clusters, init='random', max_iter=300)
model.fit(morning_df[['Lat','Lon']])

Output: KMeans(init='random', n_clusters=6)

morning_centroids = model.cluster_centers_
morning_centroids

output: array([[ 40.66307319, -73.77356222],
       [ 40.68789652, -73.96537623],
       [ 40.73294324, -73.99563532],
       [ 40.77339578, -73.96740612],
       [ 40.79307386, -73.86486688],
       [ 40.69828782, -74.20574989]])

for i, coordinate in enumerate(morning_centroids):
    folium.Marker(tuple(coordinate), popup='Centroid {}'.format(i+1), icon=folium.Icon(color
='blue')).add_to(morning_map)
morning_map
```

K-MEANS clustering is applied and formed the clusters which are shown as location pop ups in the graph above. The formed indicates the cab availability from the location where the parts of the city are divided into clusters for easy location of them on the graph.

Output:



In the next phase testing all the implemented procedures and get the results as expected with predicted values, here in the bellow code using Random forest the dataset is divided into different parts equally and on each and every sub part decision tree algorithm is applied and the results are added or the average of it is considered to make the final prediction accurately. K-means Clustering is applied to the app data.

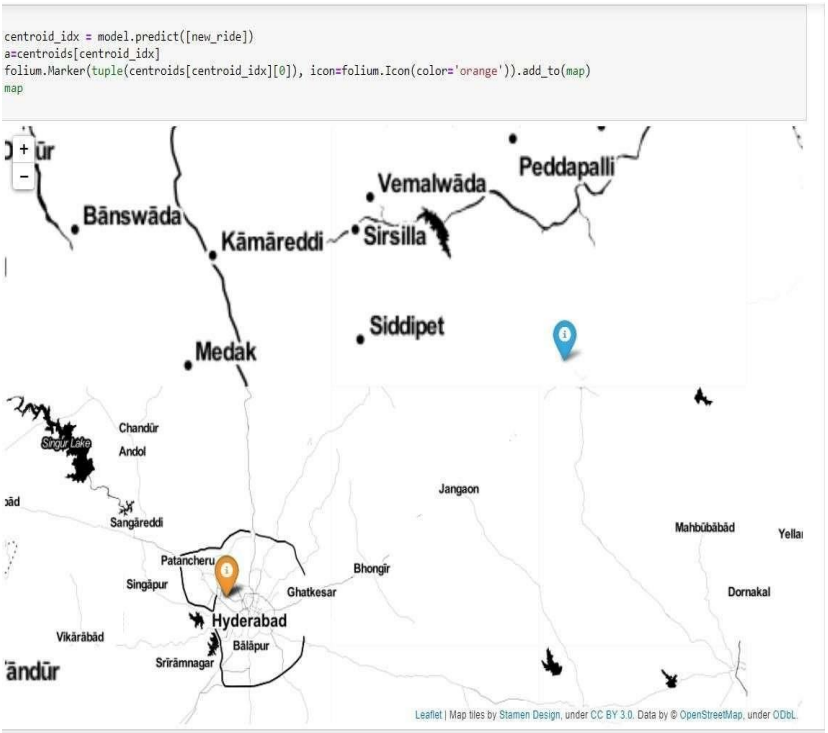


Figure 9: K-means clustering for new data input point.

Here in the above figure the new data point is denoted with green marker and the predicted cluster is denoted by orange cluster and the remaining clusters are denoted by blue markers.

Dbscan Algorithm:

Input:

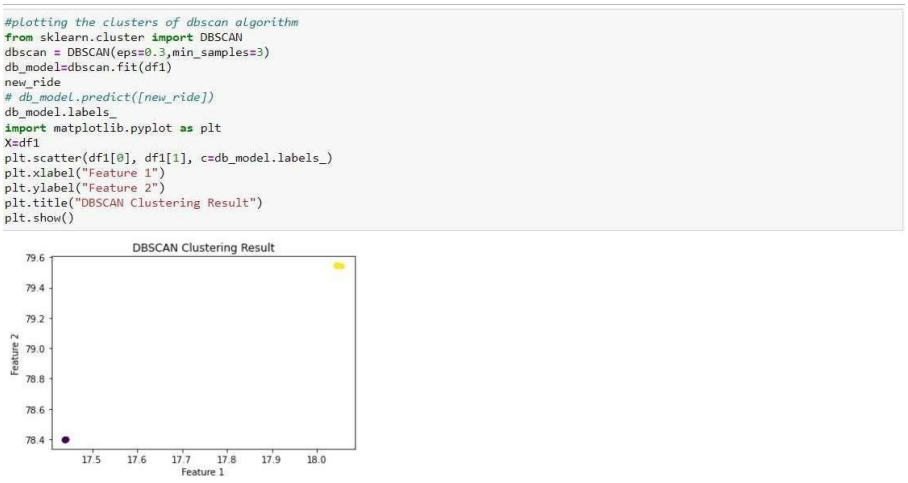


Figure 10: Db-scan algorithm code and clustering result.

In the above Db-scan clusters are plotted using PYPLOT library. Here there are two clusters plotted.

```
#predict method in dbscan
def db_predict(db, x):
    dists = np.sqrt(np.sum((db.components_ - x)**2, axis=1))
    i = np.argmin(dists)
    return db.labels_[db.core_sample_indices_[i]] if dists[i] < db.eps else -1

X_test = [new_ride]
for i in range(len(X_test)):
    print('test point: {}, predicted label: {}'.format(X_test[i],
                                                       db_predict(db_model, X_test[i])))
```

```
test point: (17.4418461, 78.3970421), predicted label: 0
```

```
#centroids points of cluster in dbscan
import numpy as np
labels = db_model.labels_
n_clusters = len(set(labels)) - (1 if -1 in labels else 0)  # Number of clusters, ignoring noise points
centroids = []
for i in range(n_clusters):
    cluster_points = X[labels == i]
    centroid = np.mean(cluster_points, axis=0)
    centroids.append(centroid)

print("Centroids of each cluster:", centroids)
```

```
Centroids of each cluster: [0    17.440847
1    78.395898
dtype: float64, 0    18.050146
1    79.542034
dtype: float64]
```

Figure 11: Cluster centroid points using dbscan.

In the above the cluster is predicted for the new datapoint and the cluster centroid centers are found.

Accuracy Using dbscan algorithm:

Input and Output for checking accuracy using dbscan algorithm:

```
#accuracy of dbscan algorithm
from sklearn.metrics import silhouette_score
X=df1
labels=db_model.labels_
silhouette_coefficient = silhouette_score(X, labels)
```

```
silhouette_coefficient
```

```
0.9968661692282582
```

The Db-scan algorithm is imported and model is fitted with the input data and accuracy is calculated.

Fair Price Detection:

Linear Regression:

Input and Output of the Linear Regression:

```
#predict method in dbscan
def db_predict(db, x):
    dists = np.sqrt(np.sum((db.components_ - x)**2, axis=1))
    i = np.argmin(dists)
    return db.labels_[db.core_sample_indices_[i]] if dists[i] < db.eps else -1

X_test = [new_ride]
for i in range(len(X_test)):
    print('test point: {}, predicted label: {}'.format(X_test[i],
                                                       db_predict(db_model, X_test[i])))
```

```
test point: (17.4418461, 78.3970421), predicted label: 0
```

```
#centroids points of cluster in dbscan
import numpy as np
labels = db_model.labels_
n_clusters = len(set(labels)) - (1 if -1 in labels else 0)  # Number of clusters, ignoring noise points
centroids = []
for i in range(n_clusters):
    cluster_points = X[labels == i]
    centroid = np.mean(cluster_points, axis=0)
    centroids.append(centroid)

print("Centroids of each cluster:", centroids)
```

```
Centroids of each cluster: [0    17.440847
1    78.395898
dtype: float64, 0    18.050146
1    79.542034
dtype: float64]
```

```
linear = LinearRegression()
linear.fit(xx_train, yy_train)
```

```
LinearRegression()
```

```
linear.score(xx_test, yy_test)
```

```
0.41980809487436843
```

Linear Regression model is imported and fitted with data and accuracy is evaluated.

Random Forest:

Input and Out of the Random forest:

```
random = RandomForestRegressor(n_estimators = 100, random_state = 0)
random.fit(xx_train , yy_train)
random.score(xx_test, yy_test)
```

```
0.957307485990282
```

Random Forest model is imported and fitted with data and accuracy is evaluated.

Logistic regression:

Input and Out of the Logistic regression:

```
from sklearn.linear_model import LogisticRegression
xx_train, xx_test, yy_train, yy_test = train_test_split(new_uber, y, test_size=0.3, random_state=42)
logistic = LogisticRegression()
logistic.fit(xx_train, yy_train)
logistic.score(xx_test, yy_test)
```

```
0.21066666666666667
```

Logistic Regression model is imported and fitted with data and accuracy is evaluated.

Decision Tree:

Input and Out of the Decision Tree:

Decision Tree algorithm is imported and fitted with input data and accuracy is determined.

```
decision = DecisionTreeRegressor(random_state=0)
decision.fit(xx_train, yy_train)
print(decision.score(xx_test, yy_test))
```

```
Fitting estimator with 56 features.
Fitting estimator with 55 features.
Fitting estimator with 54 features.
Fitting estimator with 53 features.
Fitting estimator with 52 features.
Fitting estimator with 51 features.
Fitting estimator with 50 features.
Fitting estimator with 49 features.
Fitting estimator with 48 features.
Fitting estimator with 47 features.
Fitting estimator with 46 features.
Fitting estimator with 45 features.
Fitting estimator with 44 features.
Fitting estimator with 43 features.
Fitting estimator with 42 features.
Fitting estimator with 41 features.
0.956147160063091
```

price prediction:

Input and Out of the Price prediction:

```
# loading dependency
import joblib
# saving our model # model - model , filename-model_jlib
joblib.dump(decision , 'weights_saved_file')
```

```
['weights_saved_file']
```

```
# opening the file- model_jlib
m_jlib = joblib.load('weights_saved_file')
final_list=[0,4,3,3,2,1,0,3]
# check prediction
a=m_jlib.predict([final_list]) # similar
a[0]
```

```
47.5
```

In the above the weights of decision tree algorithm is stored in a joblib file and the the file is loaded and opened and the it predicts for the new input datapoint.

Parameters Required for the cab positioning:

```
source_list = ['back-bay', 'beacon-hill', 'boston-university', 'fenway', 'financial-district', 'haymarket-square', 'north-end','north-station', 'northeastern-university', 'south-station', 'theatre-district', 'west-end']
destination_list = ['back-bay', 'beacon-hill', 'boston-university', 'fenway', 'financial-district', 'haymarket-square', 'north-end','north-station', 'northeastern-university', 'south-station', 'theatre-district', 'west-end']
month_list= ['november', 'december']
WeatherIcon_list = ['clear-day', 'clear-night', 'cloudy', 'fog', 'partly-cloudy-day', 'partly-cloudy-night', 'rain']
cabtypename_list = ['black', 'black-suv', 'lux', 'lux-black', 'lux-black-xl', 'lyft', 'lyft-xl', 'shared', 'taxi', 'uber-pool', 'uberx', 'uber-xl','wav']
Surge_multiplier_list = [0, 1, 2, 3, 4]
Uvindex_list = [0, 1]
Product_id_list = ['lyft', 'lyft_line', 'lyft_lux', 'lyft_luxsuv', 'lyft_plus', 'lyft_premier']
```

Figure 12: The affecting factors for pair price are shown above.

7 Evaluation Details

The sklearn MSE (Mean squared Error), MAE (Mean Absolute Error), RMAE Values are calculated. Mean squared error (MSE) measures the amount of error in statistical models. It assesses the average squared difference between the observed and predicted values. When a model has no error, the MSE equals zero. As model error increases, its value increases. The mean squared error is also known as the mean squared deviation (MSD).

For example, in regression, the mean squared error represents the average squared residual.If you give new location then it is plotted on the map and the nearest cluster centroid point is shown.

```
new_ride = (17.972366, 79.503448)
folium.Marker(tuple(new_ride), popup='New Rider', icon=folium.Icon(color='green')).add_t
(map)
map
```

```
centroid_idx = model.predict([new_ride])
a=centroids[centroid_idx]
folium.Marker(tuple(centroids[centroid_idx][0]), icon=folium.Icon(color='orange')).add_to(map)
map
```

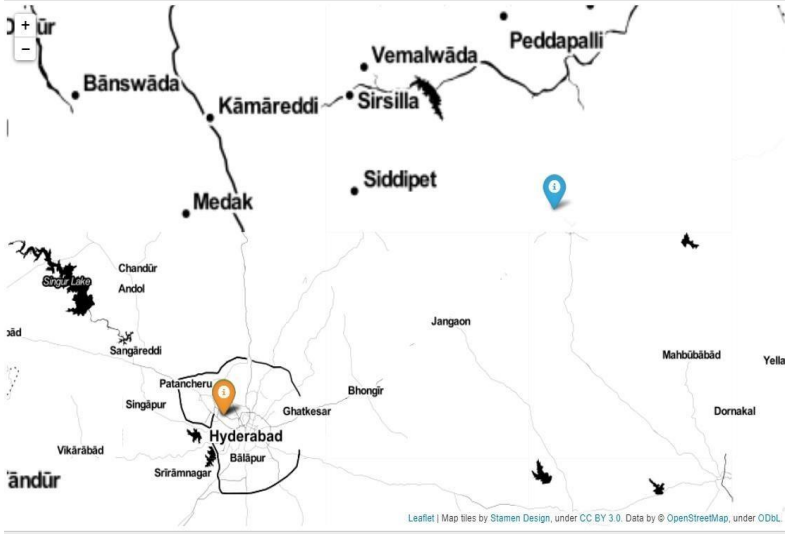


Figure 13: Showing nearest cluster of given location.

Here above the orange colored is the nearest cluster of the input data point, the input data point is represented by green marker and all other clusters are represented by blue marker.

Next, we have to detect the fair price and if you enter the type of vehicle and weather conditions then fair price is detected.

Input:

```
# loading dependency
import joblib
# saving our model # model - model , filename-model_jlib
joblib.dump(decision , 'weights_saved_file')
```

```
['weights_saved_file']
```

```
# opening the file- model_jlib
m_jlib = joblib.load('weights_saved_file')
final_list=[0,4,3,3,2,1,0,3]
# check prediction
a=m_jlib.predict([final_list]) # similar
a[0]
```

```
47.5
```

Figure 14: Fair price detection.

By considering the parameters such as type of car, fuel cost, distance of travelling, time of the day either morning or evening, uv-index, etc the fair price of the cab is predicted. These factors are the independent variables and the price is the dependent variable. Price affects as if any changes are made in the independent variables.

8 Future Work

The Future scope suggest that the cab will be assigned by using the complex machine learning algorithm, as of now the cab is just positioned according to its nearest cluster. If the size of non- linear separable dataset increases then Db-scan would best suits and even in the complex scenario then here the data points are not separated by hyperplane then more advanced techniques like kernel methods or neural networks may be required to perform effective clustering or classification.

As of now only some main factors are used in fair price detection and in future you can add other factors which affects the fair price and even you can predict the fair price for the dynamic data in future using machine learning algorithms.

9 Summary and Conclusions

Analyzing the data that is taken from uber related data from a csv file and performing various actions on it in order to visualize and access the data accordingly to the need is done , also the algorithms that are used for fare price detection[12] and optimal cab positioning are decided after knowing the best accuracy of each performed algorithm .From which we found out that Random forest has the best accuracy when it comes for fare price detection and k-means for optimal cab positioning.

Data that is already there is used in the first case and then a developed application that is used to take the requirement details from the users (which actually mean online data collection) and then perform operation on it to find the optimal positioning of a cab that is given for a particular location. We used the best accuracy algorithms in order to reduce the redundancy and increase the efficiency of the project in doing the tasks perfectly.

Acknowledgement: Not Applicable.
Funding Statement: The author(s) received no specific funding for this study.
Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] S. Silverstein, "These animated charts tell you everything about uber prices in 21 cities," *Business Insider*, vol. 16, 2014.

[2] "Discover what uberx is and how it works," accessed: 2018-07-18. [Online]. Available: https://www.uber.com/pt-BR/blog/o-que-e-uberx/

[3] P. Cohen, R. Hahn, J. Hall, S. Levitt et al., "Using big data to estimate consumer surplus: The case of uber," *National Bureau of Economic Research*, Tech. Rep., 2016.

[4] J. Chao, "Modeling and analysis of uber's rider pricing," 01 2019.

[5] R. Srinivas, B. Ankayarkanni, and R. S. B. Krishna, "Uber related data analysis using machine learning," *in 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS*), 2021, pp. 1148–1153.

[6] A. Brodeur and K. Nield, "An empirical analysis of taxi, lyft and uber rides: Evidence from weather shocks in nyc," *Journal of Economic Behavior & Organization*, vol. 152, pp.1–16,2018

[7] A. Kumar, "Python – replace missing values with mean, median & mode." [Online]. Available: https://vitalflux.com/pandas-imputemissing-values-mean-median-mode/

[8] B. Annamalai and S. Sundarraj, "An approach to predict taxi-passenger demand using quantitative histogram on uber data," *IOP Conference Series: Materials Science and Engineering*, vol.04, pp. 1–4, 2019.

[9] R. Pradhan, P. Mannepallli, and V. Rajpoot, "Analysing uber trips using pyspark," *IOP Conference Series: Materials Science and Engineering*, vol. 1119, p. 012013, 03 2021.

[10] L. Moreira-Matias, J. Gama, M. Ferreira et al., "Predicting taxi- passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1393–1402, 09 2013.

[11] A. Zheng and A. Casari, "Feature engineering for machine learning: principles and techniques for data scientists, " *O'Reilly Media*, Inc., 2018.

[12] M. K. Chen and M. Sheldon, "Dynamic pricing in a labor market: Surge pricing and flexible work on the uber platform," *Ec*, vol. 16, p. 455, 2016.

[13] H. B. Barlow, "Unsupervised learning,Neural computation", Vol.1, No.3,pp.295-311, 1989.

[14] L. Rokach,.and O. Maimon,"Clustering methods," *Data mining and knowledge discovery handbook, Springer*, pp.321-352, 2005.

[15] P. Baldi, "Auto encoders, unsupervised learning, and deep architectures,Proceedings of ICML workshop on unsupervised and transfer learning," pp.37-49, 2012.

[16] M. B. Rozenwald, A. A. Galitsyna, G. V. Sapunov, *et al.,"* A machine learning framework for the prediction of chromatin folding in Drosophila using epigenetic features". *PeerJ Comput Sci*. vol.6, no.307, 2020.

[17] X. An and W. Wang, "Knowledge management technologies and applications: A literature review". IEEE, pp.138-141, 2010.

[18] A. Berson, S. J. Smith, and K. Thearling, "Building Data Mining Applications for CRM". *New York: McGraw-Hill*, 1999.

[19] Ribeiro and A. R. D. Silva, "Survey on Cross-Platforms and Languages for Mobile Apps," *Eighth International Conference on the Quality of Information and Communications Technology*, 2012.

[20] S. S. Jagtap and D. B. Hanchate, "Development of Android Based Mobile App for PrestaShop eCommerce Shopping Cart (ALC) ," *International Research Journal of Engineering and Technology (IRJET)* , vol. 4, no. 7, pp. 2248–2254 , Jul. 2017.

[21] N. Litayem, B. Dhupia, and S. Rubab, "Review of Cross-Platforms for Mobile Learning Application Development," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 1, pp. 31–3