## JAYAPRAKASH NARAYAN COLLEGE OF ENGINEERING DHARMAPUR, MAHABUBNAGAR – 509001

AFFILIATED TO JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KUKATPALLY, HYDERABAD – 500072, Telangana, India

## “UBER RELATED DATA ANALYSIS USING MACHINE LEARNING”

A Dissertation on Major Project submitted to the Jawaharlal Nehru Technological University, Hyderabad in partial fulfilment of the requirement for the award of degree of

## BACHELOR OF TECHNOLOGY

## IN

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted By**

**MOHAMMAD SOHAIL ALI (19361A0537)**

**P.PAVAN TEJA (19361A0556)**

**P.HARSHA VARDHAN REDDY (19361A0521)**

**MOHAMMED AMER HUSSAIN (19361A0553)**

### Under the Guidance

**Mrs. M. Bharathi**

**Asst. Professor**

**JUNE 2023**

**Department of Computer Science and Engineering**

**JAYAPRAKASH NARAYAN COLLEGE OF ENGINEERING**

## DHARMAPUR, MAHABUBNAGAR – 509001

### Web: [www.jpnce.ac.in,](http://www.jpnce.ac.in/) Phone: 8886680021

2022-2023

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING JAYAPRAKASH NARAYAN COLLEGE OF ENGINEERING DHARMAPUR, MAHABUBNAGAR – 509001

**(Affiliated to J.N.T.U, Approved by A.I.C.T.E)**

**C ERTIFICATE**

This is to certify that Major Project report on “**UBER RELATED DATA ANALYSIS USING MACHINE LEARNING**” is a bonafide work done by **MOHAMMAD SOHAIL ALI (19361A0537), P. PAVAN TEJA(19361A0556), P. HARSHA VARDHAN REDDY(19361A0521), MOHAMMED AMER HUSSAIN (19361A0553).** In partial fulfillment of the requirement of the award for the degree of Bachelor of Technology in **“Computer Science and Engineering” J.N.T.U.,** Hyderabad during the year 2022-2023.

### Project Guide H.O.D

**Mrs. M. Bharathi Dr. K. Guru Raghavendra Reddy**

**Asst. Professor, Asst. Professor & Head,**

**Dept of C.S.E. Dept of C.S.E**

**External Examiner**

Department of Computer Science Engineering **Jayaprakash Narayan College of Engineering** Mahbubnagar - 509001, Telangana, India.

**Mrs. M. BHARATHI**

Assistant Professor

Department of Computer Science Engineering

Jayaprakash Narayan College of Engineering, Mahbubnagar

16.06. 2023

Supervisor’s Certificate

This is to certify that the work in the thesis entitled **“UBER RELATED DATA ANALYSIS USING MACHINE LEARNING”** by Mohammad Sohail Ali(19361A0537), P. Pavan Teja (19361A0556), P. Harsha Vardhan Reddy (19361A0521), Mohammad Amer Hussain (19361A0553) is a record of a work carried out by them under my supervision and guidance in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science Engineering**.

(Mrs. M. BHARATHI)

Supervisor

# ACKNOWLEDGEMENT

We owe a debt of gratitude to **Mrs. M. Bharathi, Asst. Professor**, CSE, JPNCE for her admirable guidance an inspirational both theoretically and practically and most importantly for the drive to complete projects successfully. Working under such an eminent guide was our privilege.

We express our sincere thanks to **Dr. K. Guru Raghavendra Reddy**,**Asst**. **Professor** and HOD, CSE, JPNCE of all kinds of consideration, support and encouragement in carrying out this project Successfully.

We would also like to thank **Dr. SUJEEVAN, Principal**, JPNCE for his cooperation and encouragement.

We are grateful to the department of Computer Science and Engineering for providing us with excellent lab and library facilities.

We thank our parents for the love, care and moral support without which we would have not been able to complete this project. It has been a constant source of inspiration for all my academic endeavor.

### PROJECT ASSOCIATES

Mohammad Sohail Ali (19361A0537)

P. Pavan Teja (19361A0556)

P. Harsha Vardhan Reddy (19361A0521)

Mohammed Amer Hussain (19361A0553)

# ABSTRACT

The paper explains the working of an Uber dataset, which contains data produced by Uber for New York City. Uber is defined as a P2P platform. The platform links you to drivers who can take you to your destination. The dataset includes primary data on Uber pickups with details including the date, time of the ride as well as longitude-latitude information , Using the information, the paper explains the use of the k-means clustering algorithm on the set of data and classify the various parts of New York City. Since the industry is booming and expected to grow shortly. Effective taxi dispatching will facilitate each driver and passenger to reduce the wait time to seek out one another. The model is employed to predict the demand on points of the city. To predict demand, time series analysis and forecasting models are utilized to identify patterns in ride requests based on historical data. These models take into account factors such as time of day, day of the week, holidays, and other external events to predict future demand and enable proactive resource allocation. Driver allocation and dispatch optimization are addressed through the application of clustering and routing algorithms. Clustering techniques are used to identify areas with high demand and group them into optimal zones for driver assignment. Routing algorithms are employed to determine the most efficient routes for drivers, considering factors like traffic conditions, distance, and time. The findings of this machine learning analysis provide valuable insights for Uber to optimize operations, improve efficiency, and enhance customer experience. The integration of machine learning techniques enables proactive decision-making, resource allocation, and service improvements, contributing to increased customer satisfaction and business growth. Overall, this study demonstrates the potential of machine learning in analyzing Uber-related data and showcases its application in generating predictive insights and optimizing operations in the ride-sharing industry. The results pave the way for further research and development of intelligent transportation systems, with the potential for broader applications in urban mobility and transportation planning.

# **CONTENTS**

**[ACKNOWLEDGEMENT I](#_Toc27059)V**

**[ABSTRACT V](#_Toc32571)**

**SNO. TOPIC PAGENO**

[1. INTRODUCTION 1](#_Toc18856)

[2. LITERATURE REVIEW 4](#_Toc418)

[3. METHODOLOGY 7](#_Toc28757)

[3.1 MODULES 7](#_Toc645)

[3.2 INTERFACE REQUIREMENTS 13](#_Toc27227)

[4. DESIGN 15](#_Toc2098)

[4.1 REQUIREMENTS 15](#_Toc18453)

[4.2 FUNCTIONAL REQUIREMENTS 15](#_Toc2211)

[4.3 NON-FUNCTIONAL REQUIREMENTS 16](#_Toc8452)

[4.4 SYSTEM CONFIGURATION 17](#_Toc31152)

[4.5 UML DIAGRAMS 23](#_Toc18257)

[5. IMPLEMENTATION 26](#_Toc25854)

[5.1 IMPORT REQUIRED LIBRARIES. 26](#_Toc27133)

[5.2 IMPORT RAW DATASET 27](#_Toc28)

[5.3 DATA PREPROCESSING 28](#_Toc20693)

[5.4 VISUALIZATION OF IMPORTED DATA 29](#_Toc24087)

[5.5 PREDICTED SCHEDULING OF CAB USING ALGORITHM 32](#_Toc19964)

[5.6 INTERFACE IMPLEMENTATION 39](#_Toc13694)

[6. RESULTS AND DISCUSSION 55](#_Toc7889)

[7. FUTURE SCOPE 61](#_Toc29441)

[8. CONCLUSION 62](#_Toc14059)

[9. BIBLIOGRAPHY 63](#_Toc5872)

**LIST OF FIGURES**

**FIGURE NO FIGURE DESCRIPTION PAGENO**

3.1 Visualize a histogram of trips travelled 8

3.2 Visualize a histogram of trips travelled every hour 9

3.3 Heatmap of frequency of cabs travelled during the hours 9

3.4 No of cabs travel per hour 10

4.1 Use Case diagram 23

4.2 Class diagram 24

4.3 Activity diagram 25

6.1 Input the latitude and the longitude 55

6.2 Inserting the latitude and the longitude points 55

6.3 Prediction of the cluster 56

6.4 Prediction of cluster along with the location 57

6.5 Prediction of the new points 58

6.6 Plotting the points through which cab has travelled. 59

6.7 Plotting the clusters on x-y graph of latitude and longitude. 59

6.8 Plotting the clusters on the map of New York City imported. 60

6.9 Predicting the pickup of the cab from the particular cluster. 60

# 1.INTRODUCTION

The Uber platform connects you with drivers who can take you to your destination or location. This dataset includes primary data on Uber collections with details that include the date, time of travel, as well as information on longitude and latitude in San Francisco and has operations in over 900 metropolitan areas worldwide. The prediction of the frequency of trips of data is by implementing a part of k-means clustering algorithm The standard algorithm describes the maximum variance within the group as the number of square distances Euclidean distances between the points and the corresponding centroid. The use of the digital computer has since moved to technology where the program involves the use of neural networks ,Examples of RNN (Recurrent Neural Network) and TDNN (Time delay Neural Network)for importing data from uber dataset which takes the data for forecasting on a time horizon. The ultimate aim of the project is to predict the pickup of the cab on the basis of clusters defined by the kmeans clustering algorithm. This algorithm is used to divide the dataset into k-groups. where k is defined as the number of groups provided by the user. The standard algorithm describes the maximum variance within the group as the number of square distances Euclidean distances between the points and the corresponding centroid.

With a huge database of drivers, as soon as a user requests for car, their algorithms match a user with the most suitable driver within a 15 second window to the nearest driver. Uber stores and analyses data on every single trip the users take which is leveraged to predict the demand for cars, set the fares and allocate sufficient resources. Data science team at Uber also performs in-depth analysis of the public transport networks across different cities so that they can focus on cities that have poor transportation and make the best use of the data to enhance customer service experience.

Uber is greedy about what data it collects and with many cheap relative storage options like Hadoop and Spark-it has got data about every single GPS point for every trip taken on Uber. Uber stores historic information about its system and capabilities to ease doing data science for its [data scientists](https://www.projectpro.io/article/how-to-learn-data-science-from-scratch-on-your-own-in-2021/420" \o " data scientists " \t "https://www.projectpro.io/article/how-uber-uses-data-science-to-reinvent-transportation/_blank)down the road. Keeping the change logs, versioning of database schemas helps data scientist answer every question on-hand. With the data Uber has, [data scientists can answer questions](https://www.projectpro.io/article/how-to-prepare-for-a-data-scientist-interview/233" \o "data scientists can answer questions" \t "https://www.projectpro.io/article/how-uber-uses-data-science-to-reinvent-transportation/_blank) like what did the Uber system look like at a particular point of time from a customer perspective, supply behaviour perspective, from inter-server communication perspective or even to the state of a database.

In fact, uber drivers continue to generate data for Uber even when they are not carrying any passengers because they transmit data back to the central platform at Uber which is used to draw inferences on traffic patterns. The data is stored into the database for supply and demand algorithm analysis. Driver data is used for autonomous car research, surge pricing, tracking the location of drivers, monitoring driver’s speed, motion and acceleration and identifying if a driver is working for a competing cab sharing company.

[Big data analysis](https://www.projectpro.io/article/5-big-data-use-cases-how-companies-use-big-data/155" \o "Big data analysis" \t "https://www.projectpro.io/article/how-uber-uses-data-science-to-reinvent-transportation/_blank) spans across diverse functions at Uber – machine learning, data science, marketing, fraud detection and more. Uber data consists of information about trips, billing, health of the infrastructure and other services behind its app. City operations teams use uber big data to calculate driver incentive payments and predict many other real time events. The complete process of data streaming is done through a Hadoop Hive based analytics [platform](https://www.projectpro.io/article/what-is-a-data-science-platform-and-why-does-your-business-need-one/333" \o "platform" \t "https://www.projectpro.io/article/how-uber-uses-data-science-to-reinvent-transportation/_blank) which gives right people and services with required data at right time.

### Problem Statement

With more than 8 million users, 1 billion Uber trips and 160,000+ people driving for Uber across 449 cities in 66 countries – Uber is the fastest growing startup standing at the top of its game. Tackling problems like poor transportation infrastructure in some cities, unsatisfactory customer experience, late cars, poor fulfilment, drivers denying to accept credit cards and more –Uber has “eaten the world” in less than 5 years and is a remarkable name to reckon when it comes to solving problems for people in transportation.

If you have ever booked an Uber, you might know how simple the process is –just press a button, set the pickup location, request a car, go for a ride and pay with a click of a button. The process is simple but there is a lot going on behind the scenes. The secret key driving growth of the $51 billion start-up, is the [big data](https://www.projectpro.io/article/top-20-big-data-project-ideas-for-beginners-in-2021/426" \o "big data" \t "https://www.projectpro.io/article/how-uber-uses-data-science-to-reinvent-transportation/_blank) it collects and leverages for insightful and intelligent decision making. While Uber moves people around the world without owning any cars, data moves Uber. With the foundation to build the most intelligent company on the planet by completely solving problems for riders –Big Data and Data Science are at the heart of everything Uber does - surge pricing, better cars, detecting fake rides, fake cards, fake ratings, estimating fares and driver ratings. Read on to understand how Uber makes clever use of big data and data science to reinvent transportation and logistics globally.

### Proposed System

Based on the problems of forecasting errors and risk of overfitting due to large datasets. The data analyzed and sent to the company is resulted as inefficient and ineffective. Thus to overcome the problem we are going to predict the pickup of cab from a coordinated cluster of points predicted by using applied k-means clustering algorithm. The k-means clustering algorithm adopted will effectively dispatch taxis to the cluster.This facilitates each driver and passenger to attenuate the wait-time to search out one another. Drivers don’t have enough info concerning wherever passengers and different taxis area unit and shall move.Therefore, a cab center will organize the taxicab fleet and with efficiency give out consistent request to the whole town. The system uses the latitude and longitude of the cab scheduled and also the day of the travel and the month.An unsupervised learning model is trained with this dataset and the model is employed to predict the pickup of the cab on the cluster

# LITERATURE REVIEW

Past few years have seen tremendous growth in uber related data analysis using machine learning. People are coming up with various methods to analyze uber related data such as A state in which the results, k-means clustering is used to estimate the most likely collection points at a given time and to predict the best hotspots of nightlife learning trends from previous Uber pickups. The center of the taxi service decides on the space of area to be targeted for the pickup of passengers. This can be justified by explaining that machine learning is the core of Uber and how it has impacted on tremendous growth x Bridging the supply demand gap x Reduction in ETA x Route Optimization Poulsen, L.K In this document applied an experiment of spatial analysis of Green cab and Uber to hotspots of New York to determine the competitive position of the NYCTLC. The resulted research showed that as demand of green cabs on the hotspots grew,the demand of Uber taxis on the hotspots also growed. This research recommends that NYCTLC creates a dashboard that analyzes and displays data in real time, as we believe this will increase its competitiveness compared to Uber. Uber is a recent taxi operator in New York and is constantly devouring the market share of the yellow and green taxis of the New York Taxi and Limousine Commission (NYCTLC).The NYCTLC is an agency of the New York City Government which licenses and regulates taxis and vehicle for hire industries and also app based companies. The commission was founded on March 2 ,1971 and their headquarters are based in New York.[1]. Faghih, S.S recommends a recent modeling approach in Manhattan, New York City, to capture the demand for electronic mail services, particularly the Uber application. Uber collection data is added to the Manhattan TAD level and at 15-minute time intervals. This aggregation allows the implementation of a modern approach to spatio-temporal modeling to obtain a spatial and temporal understanding of the demand. During a typical day, two spacetime models were developed using Uber collection data, the STAR and STAR and MSPE turns determine the output of the models. The results of the MSPE have shown that it is recommended to use the Lasso-Star system instead of the star design. A comparison between the demand for yellow and uber taxis in 2014 and 2015 in New York shows that the demand for uber has increased[2]. Ghuhaexplained the grouping of the sequences calculated and observed by using a small amount of memory and time was necessary for applications that needed to develop a data flow model to involve large data sets and consider categorizing the data in the form of clusters[3]. Ahmed, M., has shown that by using detailed data on taxis at the travel level and on the rental vehicle and data on complaints about the level of new complaints at the level of incidents, we study how Uber and Lyft enter damaged the quality of taxi services in New York City.

The overall effect of the organizations based on the scenario and in particular of the riding administrations was enormous and widespread. One of these effects is the expansion of the rivalry between Uber and Lyft over the quality of taxi administration. They use a new set of complaint data to measure (the lack of) quality of service that we have never been analyzed before. Focus on the quality dimensions generated by most of the complaints we demonstrate. The increased competition for these shared travel services has had an intuitive impact on the behavior of taxi drivers[4]. Wallsten, S, stated that the results of New York and Chicago are consistent with the possibility that taxis react to the new challenge by improving quality. In New York, the rise of Uber is linked to the reduction of objections to travel to the city. They discuss the competitive effect of sharing taxis in the taxi industry using the complete data set of the New York City Taxi and Limousine Commission for more than one billion taxi trips in complaints and details of New York, New York and Chicago Google Trends on the success of Uber's largest shared travel service.[5]. Sotiropoulos, D.N, represented that this document addresses the problem of grouping, by using a new approach to genetic algorithms that is highly scalable in large volumes of textual details, developing a coding scheme based on centroids. We apply k means clustering algorithm in this document. Clustering is the unsupervised machine learning algorithm used to solve grouping problems based on similarities. This technique has aroused interest in a wide range of scientific fields, which address clustering methods, to solve complex classification problems.[6]. Faghih, S.S said that the demand for electronic mail services is growing rapidly, particularly in large cities. Uber is the first and most famous email company in the United States and New York City. A comparison between the demand for yellow and Uber taxis in New York in 2014 and 2015 shows that the demand for Uber has increased. To study the forecast performance of the models, you choose to choose data for a typical day. Our goal in this document is to describe how these models can be used for forecasting Uber demand. The Uber data contains information about the position and time of the pick-ups and returns of each trip during a day. According to the available data, the Uber historical data of April 2014[7].Kumar, states that, k-means clustering is used to estimate the most likely collection points at a given time and to predict the best hotspots of nightlife learning trends from previous Uber pickups [8,9]. L.Liu, C.Andris, and C.Ratti planned for a strategy to disclose cabdrivers working patterns by inspecting their unbroken anatomy track[10].R-H Hwand focuses on GPS and the locality to pick up passengers , A venue to venue plot model referred to as an OFF-ON model [11].

# METHODOLOGY

## 3.1 MODULES

### Data Gathering

Data collection is the process of gathering and measuring information on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. Therefore, to solve this problem Data Preparation is done. It can also use some free data sets which are present on the internet. Kaggle and UCI Machine learning Repository are the repositories that are used the most for making Machine learning models. Kaggle is one of the most visited websites that is used for practicing machine learning algorithms.

### Data Pre-processing

It is a critical step for making the model work. In this study, we have classified data into its smallest section so that analysis can be performed more precisely. Data initialization consists of the following phases:

• **Association**: Data combination from various means is done.

• **Altering**: k-means clustering is used to alter data and classify into various clusters having similarities.

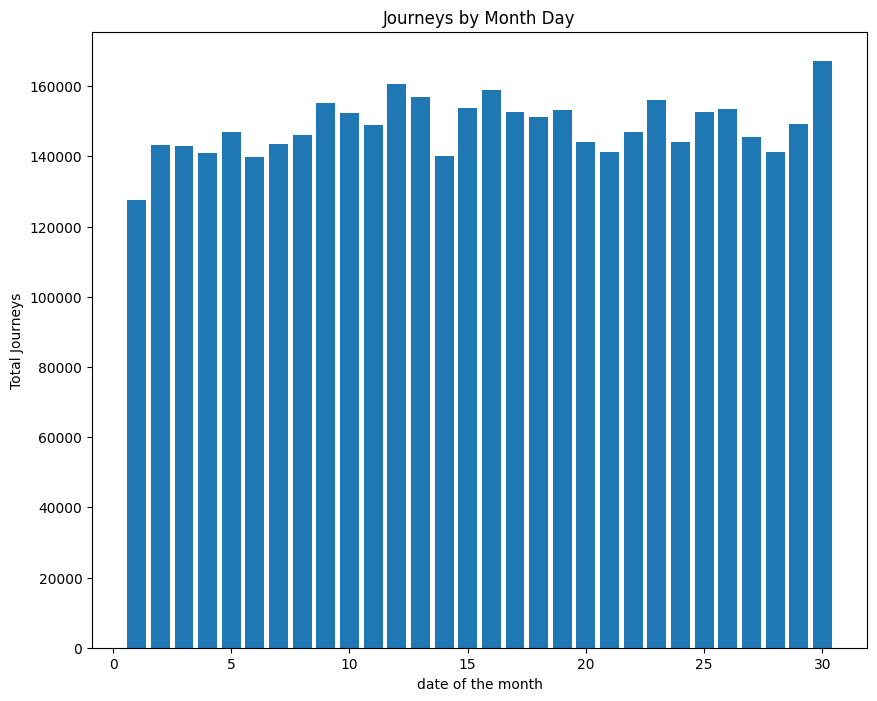
• **Clipping**: Data is selected based on requirements, and rest is used for analysis purpose.

• **Categorizing**: After successful application of the algorithm, clusters are categorized into various parts to process it further values) or duplication of data.

Date is extracted from the timestamp using built-in functions of pandas and datetime libraries. We take the hour during which an event occurred and map it to different categories to study whether a vehicle was observed during peak hours, early hours, regular hours, etc. After data initialization step, processed data is passed for training and analysis purpose.

### Data Visualization

Data visualization is defined as to evaluate the performance of a model by using graphs and metrics that calculate performance.Data visualization can be mainly used to categorize the data into new levels such that the algorithm used can be generalized to an observation of each output variable derived by an observed input variable.

Fig3.1- Visualize a histogram of trips travelled each day of the month

### frequency

Fig3.2- Visualize a histogram for frequency of trips travelled every hour

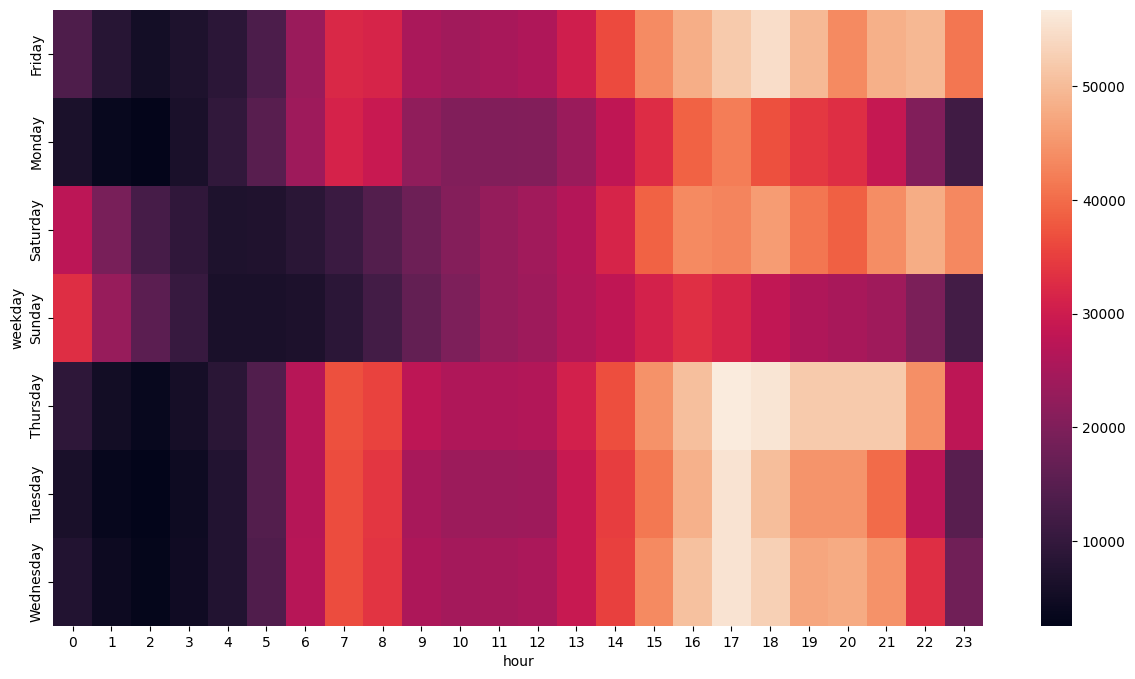
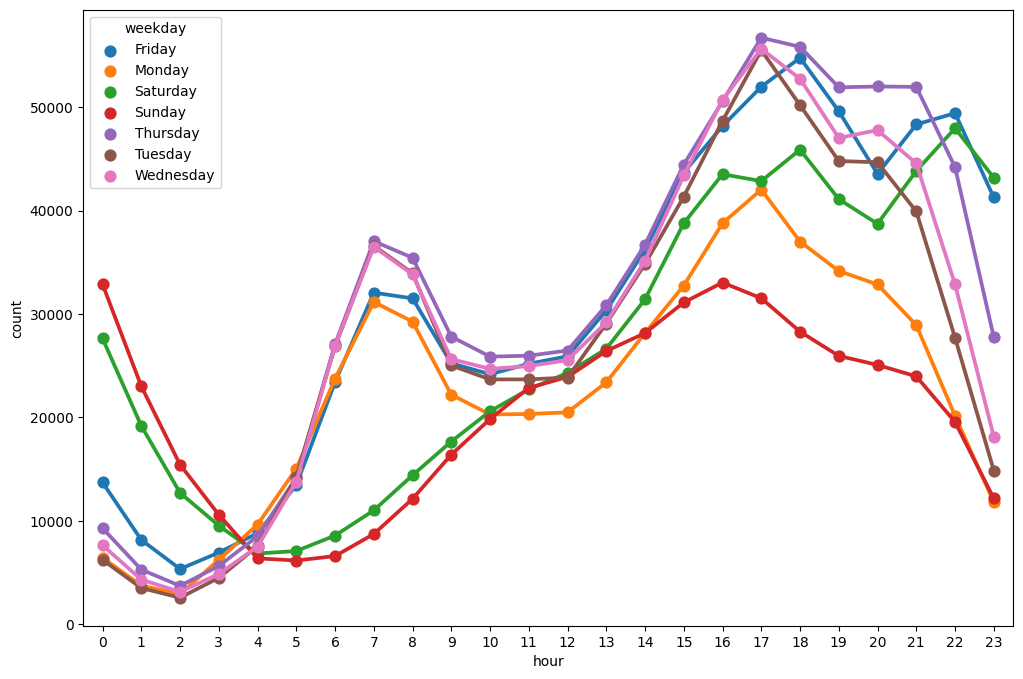


Fig3.3- Heatmap of frequency of cabs travelled during the hours

Fig3.4- No of cabs travel per hour

## **Algorithms**:

#### K-means Clustering

The algorithm used is involved on the concept of k-means clustering algorithm. which belongs to unsupervised learning. There is no labeled data for this clustering, unlike in supervised learning. K-Means performs division of objects into clusters that share similarities and are dissimilar to the objects belonging to another cluster

It works in the following manner:

• k points are initialized, also called as means.

• Each point is clustered to its closest possible point and then the mean point is revised.

• Same process is conducted for n number of iterations resulting in clusters we expect

Clustering is the method of grouping objects into groups based on similarities. This algorithm is used to divide a given data set into k groupsHere, k represents the number of groups and must be provided by the user. The idea behind the grouping of k-means is to identify the clusters in such a way to reduce total variation within the cluster. The standard algorithm describes the maximum variance within the group as the number of square distances Euclidean distances between the points and the corresponding centroid. The grouping can be classified into two groups.

#### K Medoids Clustering

The k-medoids problem is a [clustering](https://en.wikipedia.org/wiki/Data_clustering" \o "Data clustering) problem similar to [k-means](https://en.wikipedia.org/wiki/K-means" \o "K-means). Both the k-means and k-medoids algorithms are partitional (breaking the dataset up into groups) and attempt to minimize the distance between points labeled to be in a cluster and a point designated as the center of that cluster. In contrast to the k-means algorithm, k-medoids chooses actual data points as centers ([medoids](https://en.wikipedia.org/wiki/Medoids" \o "Medoids) or exemplars), and thereby allows for greater interpretability of the cluster centers than in k-means, where the center of a cluster is not necessarily one of the input data points (it is the average between the points in the cluster). Furthermore, k-medoids can be used with arbitrary dissimilarity measures, whereas k-means generally requires [Euclidean distance](https://en.wikipedia.org/wiki/Euclidean_distance" \o "Euclidean distance) for efficient solutions. Because k-medoids minimizes a sum of pairwise dissimilarities instead of a sum of [squared Euclidean distances](https://en.wikipedia.org/wiki/Squared_Euclidean_distance" \o "Squared Euclidean distance), it is more robust to noise and outliers than [k-means](https://en.wikipedia.org/wiki/K-means" \o "K-means).

k-medoids is a classical partitioning technique of clustering that splits the data set of n objects into k clusters, where the number k of clusters assumed known a priori (which implies that the programmer must specify k before the execution of a k-medoids algorithm). The "goodness" of the given value of k can be assessed with methods such as the [silhouette method](https://en.wikipedia.org/wiki/Silhouette_(clustering)" \o "Silhouette (clustering)).

The [medoid](https://en.wikipedia.org/wiki/Medoid" \o "Medoid) of a cluster is defined as the object in the cluster whose average dissimilarity to all the objects in the cluster is minimal, that is, it is a most centrally located point in the cluster.

#### Gaussian mixture models

Gaussian mixture models are a probabilistic model for representing [normally distributed](https://brilliant.org/wiki/multivariate-normal-distribution/" \o "normally distributed" \t "https://brilliant.org/wiki/gaussian-mixture-model/_blank) subpopulations within an overall population. [Mixture models](https://brilliant.org/wiki/mixture-model/" \o "Mixture models" \t "https://brilliant.org/wiki/gaussian-mixture-model/_blank) in general don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically. Since subpopulation assignment is not known, this constitutes a form of [unsupervised learning](https://brilliant.org/wiki/unsupervised-learning/" \o "unsupervised learning" \t "https://brilliant.org/wiki/gaussian-mixture-model/_blank).

For example, in modeling human height data, height is typically modeled as a normal distribution for each gender with a mean of approximately 5'10" for males and 5'5" for females. Given only the height data and not the gender assignments for each data point, the distribution of all heights would follow the sum of two scaled (different variance) and shifted (different mean) normal distributions. A model making this assumption is an example of a Gaussian mixture model (GMM), though in general a GMM may have more than two components. Estimating the parameters of the individual normal distribution components is a canonical problem in modeling data with GMMs.

GMMs have been used for feature extraction from speech data, and have also been used extensively in object tracking of multiple objects, where the number of mixture components and their means predict object locations at each frame in a video sequence

#### Agglomerative Hierarchical clustering

The agglomerative clustering is the most common type of hierarchical clustering used to group objects in clusters based on their similarity. It’s also known as AGNES (Agglomerative Nesting). The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. The result is a tree-based representation of the objects, named dendrogram.

The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the bottom-up approach. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram

## 3.2 INTERFACE REQUIREMENTS

The dataset we are utilizing comes from kaggle website which is open to everyone for free. Data set contains some null values and categorical values should be encoded to understand for ML Algorithm.

### Training and Testing the model on data

Raw data is taken into consideration and pre-processed with the help of k-means clustering for acquiring a prediction model.

* First, we consider the time and date of the ride and place of the customer so that we satisfy our factors for the processing of the dataset.
* Then, we consider meteorological data [12] for atmospheric conditions during that fraction of time. This can help us with factors like fog, rains (i.e., water on road), etc.
* The dataset includes primary data on Uber pickups with details including the date, time of the ride as well as longitude-latitude information are taken from the official website, and all these are considered as the pre-processed data for our model.
* Approximation was conducted before using this dataset. Various factors like temperature and visibility had almost similar values, which were approximated to a mode value for convenience.
* Then, the pre-processed data is passed further for feature engineering where the processing of the dataset can be made faster and efficiency can be increased to a great extent.
* After feature engineering, data being unlabelled, unsupervised learning is the best fit for this model. kmeans being most convincing and has high accuracy rate was chosen for analysis.
* After feature engineering, the dataset was then passed through k-means clustering to form various clusters and differentiate areas into low, medium, and high demand areas with their coordinates so that locating on maps becomes easy.
* Areas with high demand on point of the city were mapped using red pins so that they can be easily spotted customers/ users.

# 4. DESIGN

## 4.1 REQUIREMENTS

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide. In order to fully understand one’s project, it is very important that they come up with a SRS listing out their requirements, how are they going to meet it and how will they complete the project. It helps the team to save upon their time as they are able to comprehend how are going to go about the project. Doing this also enables the team to find out about the limitations and risks early on. Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable attributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources.

## 4.2 FUNCTIONAL REQUIREMENTS

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. Following are the functional requirements on the system:

1. All the data must be in the same format as structured data.
2. The data collected will be vectorized and sent across to the classifier.
3. Algorithms used in this project are K-Means Clustering, unsupervised learning

### Benefits of Functional requirements

* A functional requirement document helps you to define the functionality of a system or one of its subsystems.
* Functional requirements along with requirement analysis help identify missing requirements. They help clearly define the expected system service and behavior.
* Errors caught in the Functional requirement gathering stage are the cheapest to fix.
* Support user goals, tasks, or activities

## 4.3 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties such as reliability, response time and store occupancy. Non-functional requirements arise through the user needs, because of budget constraints, organizational policies and the need for interoperability with other software and hardware systems.

### Benefits of Non-functional requirements

* + - * The nonfunctional requirements ensure the software system follows legal and compliance rules.
      * They ensure the reliability, availability, and performance of the software system.
      * They ensure good user experience and ease of operating the software.
      * They help in formulating security policy of the software.
      * They ensure good user experience, ease of operating the software, and minimize the cost factor.
      * They serve as constraints or restrictions on the design of the system across the backlogs.
      * They ensure usability and effectiveness of the entire system

### Reliability Requirement

The system should accurately perform when a farmer gives inputs to the algorithm without causing any error.

### Usability Requirement

The system is designed for a user-friendly environment so that the users do not face any difficulty.

### Implementation Requirements:

The system is implemented using python by making use of Machine Learning algorithms such as K- Means Clustering, unsupervised learning.

## 4.4 SYSTEM CONFIGURATION

### Hardware System Configuration:

* Processor: i5 5th gen or above
* RAM: 8 GB for 64-bit.
* Hard disk space: =512GB.

### Software Configuration:

* Operating System: Windows 11
* Coding Language: Python.

Python is a simple, general purpose, high level, and object-oriented programming language. It is an interpreted scripting language also. Guido Van Rossum is known as the founder of Python programming. It is a general purpose, dynamic, high-level, and interpreted programming language. It supports an Object-Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures. It is easy to learn yet a powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.

Python supports multiple programming patterns, including object-oriented, imperative, and functional or procedural programming styles. Python is not intended to work in a particular area, such as web programming. That is why it is known as a multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc. We don't need to use data types to declare variables because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. Python makes the development and debugging fast because there is no compilation step included in Python development, and the edit-test-debug cycle is very fast. LIBRARIES: NumPy, pandas, scikit-learn NumPy, which stands for Numerical Python, is a library consisting of multidimensional

### Tools:

**Pandas:**Pandas are defined as an open-source library that provides high-performance data manipulation in Python. The name of Pandas is derived from the word Panel Data, which means an Econometrics from Multidimensional data. Data analysis requires lots of processing, such as restructuring, cleaning or merging, etc. There are different tools are available for fast data processing, such as NumPy, Skippy, Python, and Panda. But we prefer Pandas because working with Pandas is fast, simple and more expressive than other tools. Pandas is built on top of the NumPy package, means NumPy is required for operating the Pandas. Before Pandas, Python was capable for data preparation, but it only provided limited support for data analysis. So, Pandas came into the picture and enhanced the capabilities of data analysis. It can perform five significant steps required for processing and analysis of data irrespective of the origin of the data, i.e., load, manipulate, prepare, model, and analyze. Scikit-learn are an open-source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities.

**Numpy:** Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. In numpy, arrays allow a wide range of operations which can be performed on a particular array or a combination of Arrays. These operations include some basic Mathematical operations as well as Unary and Binary operations.

**Seaborn** is a library for making statistical graphics in Python. It builds on top of [matplotlib](https://matplotlib.org/) and integrates closely with [pandas](https://pandas.pydata.org/) data structures.Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

**Matplotlib** is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

**[Plotly Express](https://plotly.github.io/plotly_express/" \t "https://medium.com/plotly/_blank)** is a new high-level Python visualization library: it’s a wrapper for [Plotly.py](https://plot.ly/python/" \t "https://medium.com/plotly/_blank) that exposes a simple syntax for complex charts. Inspired by Seaborn and ggplot2, it was specifically designed to have a terse, consistent and easy-to-learn API: with just a single import, you can make richly interactive plots in just a single function call, including faceting, maps, animations, and trendlines. It comes with on-board datasets, color scales and themes, and just like Plotly.py, Plotly Express is totally free: with its permissive open-source MIT license, you can use it however you like (yes, even in commercial products!). Best of all, Plotly Express is fully compatible with the rest of Plotly ecosystem: use it in your [Dash](https://dash.plot.ly/" \t "https://medium.com/plotly/_blank) apps, export your figures to almost any file format [using Orca](https://medium.com/@plotlygraphs/plotly-py-end-of-summer-updates-5422c98b9058), or edit them in a GUI with the [JupyterLab Chart Editor](https://www.youtube.com/watch?v=zR7G2tNVo1Q" \t "https://medium.com/plotly/_blank)!

**The plotly.graph\_objects**module (typically imported as go) contains an [automatically-generated hierarchy of Python classes](https://plotly.com/python-api-reference/plotly.graph_objects.html" \l "graph-objects) which represent non-leaf nodes in this figure schema. The term "graph objects" refers to instances of these classes.The primary classes defined in the plotly.graph\_objects module are [Figure](https://plotly.com/python-api-reference/generated/plotly.graph_objects.Figure.html) and an [ipywidgets-compatible variant called FigureWidget](https://plotly.com/python/figurewidget/), which both represent entire figures. Instances of these classes have many convenience methods for Pythonically [manipulating their attributes](https://plotly.com/python/creating-and-updating-figures/) (e.g. .update\_layout() or .add\_trace(), which all accept ["magic underscore" notation](https://plotly.com/python/creating-and-updating-figures/" \l "magic-underscore-notation)) as well as [rendering them](https://plotly.com/python/renderers/) (e.g. .show()) and [exporting them to various formats](https://plotly.com/python/static-image-export/) (e.g. .to\_json() or .write\_image() or .write\_html()).

**folium**builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the Leaflet.js library. Manipulate your data in Python, then visualize it in a Leaflet map via folium.

**Sklearn** is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python

## ● Software: Anaconda

With over 20 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries. Search our cloud-based repository to find and install over 7,500 data science and machine learning packages. With the conda-install command, you can start using thousands of open-sources Conda, R, Python and many other packages. Individual Edition is an open source, flexible solution that provides the utilities to build, distribute, install, update, and manage software in a cross-platform manner. Conda makes it easy to manage multiple data environments that can be maintained and run separately without interference from each other. Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open- source packages and libraries. Search our cloud-based repository to find and install over 7,500 data science and machine learning packages. With the conda-install command, you can start using thousands of open-sourceConda, R, Python and many other packages. Individual Edition is an open source, flexible solution that provides the utilities to build, distribute, install, update, and manage software in a cross-platform manner. Conda makes it easy to manage multiple data environments that can be maintained and run separately without interference from each other.

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

Package versions in Anaconda are managed by the package management system [conda](https://en.wikipedia.org/wiki/Conda_(package_manager)). This package manager was spun out as a separate [open-source](https://en.wikipedia.org/wiki/Open_source) package as it ended up being useful on its own and for things other than Python. There is also a small, [bootstrap](https://en.wikipedia.org/wiki/Bootstrapping) version of Anaconda called Miniconda, which includes only conda, Python, the packages they depend on, and a small number of other packages

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPIas well as conda package and virtual environment manager. It also includes a GUI, Anaconda navigator as a graphical alternative to the command line interface. The big difference between conda and the pip package manager is in how package dependencies are manages, which is a significant challenge for python data science and the reason conda existsAnaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using [command-line commands.](https://en.wikipedia.org/wiki/Command-line_interface) Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

In contrast, conda analyses the current environment including everything currently installed and together with any version limitations specified. Open source packages can be installed from the Anaconda repository, Anaconda cloud, or the users own private repository or mirror using the conda install command. The default installation of Anaconda 2 includes python 2.7 and anaconda 3 includes python 3.7 however, it is possible to create new environments that include any version of python packaged with conda

## 4.5 UML DIAGRAMS

### Use case Diagram

The use case diagram describes what actions are to be performed by the user and it also used to analyse the system’s high-level instructions. A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses

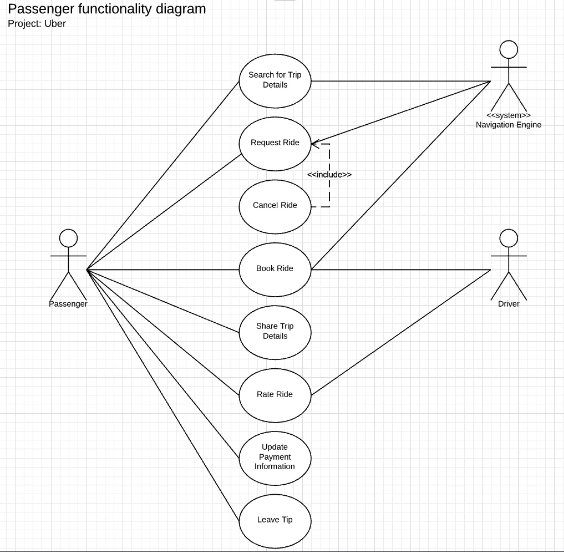


Fig 4.1- Use Case Diagram

### Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

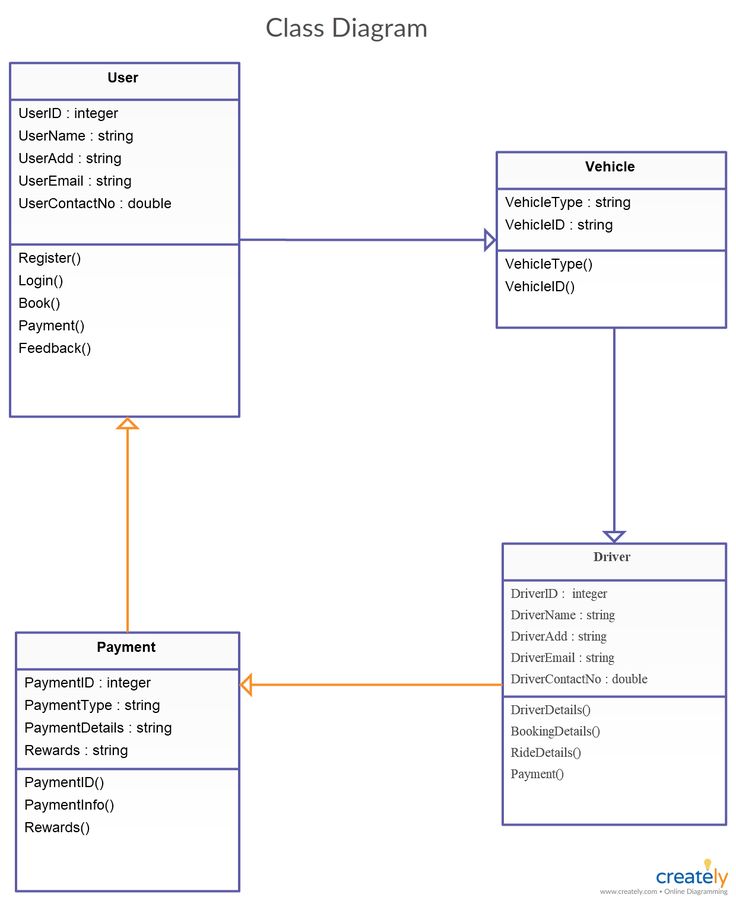


Fig 4.2- Class Diagram

### Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. It is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.



Fig 4.3- Activity Diagram

# 5.IMPLEMENTATION

## IMPORT REQUIRED LIBRARIES.

import pandas as pd

import seaborn as sns

import numpy as np

import matplotlib.pyplot as plt

import os

import plotly.express as px

import plotly.graph\_objects as go

**NumPy** is a python package used for performing the various numerical computations and processing of single dimensional and multi-dimensional array elements.

**Seaborn** is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data.

**Pandas** provide high performance and data manipulation.

**Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

**Python OS** module provides the facility to establish the interaction between the user and the operating system.

## IMPORT RAW DATASET

The dataset is downloaded from Kaggle Website.

path = r'C:\Users\moham\Documents\major project\New folder\archive'

files = ['uber-raw-data-aug14.csv',

         'uber-raw-data-apr14.csv',

         'uber-raw-data-jul14.csv',

         'uber-raw-data-jun14.csv',

         'uber-raw-data-may14.csv',

         'uber-raw-data-sep14.csv']

final = pd.DataFrame()

for file in files:

df = pd.read\_csv(path+"/"+file,encoding='utf-8')

 final = pd.concat([final,df])

CSV files are used to store a large number of variables or data. They are incredibly simplified spreadsheets - think Excel- only the content is stored in plaintext. And the CSV module is a built-in function that allows Python to parse these types of files

## 5.3 DATA PREPROCESSING

A huge amount of trip data will be collected fromUbe rfor training and testingdata. From the collected dataset the latitude and latitude will be clustered and classified based on the frequency of trips travelled by the cab during the day. When these criteria are considered, and data preprocess will be done on these datasets.

df['Date/Time'] = pd.to\_datetime(df['Date/Time'], format='%m/%d/%Y %H:%M:%S')

df.dtypes

df['weekday']=df['Date/Time'].dt.day\_name()

df['day']=df['Date/Time'].dt.day

df['minute']=df['Date/Time'].dt.minute

df['month']=df['Date/Time'].dt.month

df['hour']=df['Date/Time'].dt.hour

df.head()

## 5.4 VISUALIZATION OF IMPORTED DATA

Data visualization can be mainly used to categorize the data into new levels such that the algorithm used can be generalized to an observation of each output variable derived by an observed input variable

#Visualizeing a histogram for frequency of trips travelled each day of the month

plt.figure(figsize=(10,8))

plt.hist(df['day'],bins=30,rwidth=0.8,range=(0.5,30.5))

plt.xlabel('date of the month')

plt.ylabel('Total Journeys')

plt.title('Journeys by Month Day')

#Visualizeing a histogram of trips travelled each day of the week

colors = ['lightslategray',] \* 5

colors[0] = 'blue'

fig = go.Figure(data=[go.Bar(

    x=df['weekday'].value\_counts().index,

    y=df['weekday'].value\_counts().values,

marker\_color=colors

)])

fig.update\_layout(title\_text='Rush Day of Uber Trip')

#Visualizing data on graph on frequency of trips travelled during the day.

plt.hist(df['hour'])

plt.ylabel('frequency')

plt.xlabel('work hour')

plt.figure(figsize=(40,20))

for i,month in enumerate(df['month'].unique()):

plt.subplot(3,2,i+1)

df[df['month']==month]['hour'].hist()

#Data Visualization of Heatmap of frequency of cabs travelled during the hour.

def count\_rows(rows):

  return len(rows)

by\_cross = df.groupby(['weekday','hour']).apply(count\_rows)

by\_cross

pivot=by\_cross.unstack()

pivot

plt.figure(figsize=(15,8))

sns.heatmap(pivot)

#Plotting the collection of points through which cab has travelled during the course of the month.

plt.figure(figsize=(12,6))

plt.plot(df['Lon'],df['Lat'],'r+',ms=0.5)

plt.xlim(-74.2,-73.7)

plt.ylim(40.6,41)

## 5.5 PREDICTED SCHEDULING OF CAB USING ALGORITHM

The scheduling of the cab can be predicted on the basis of the location given by the user and the proposed method finds the nearest hotspot which is defined as a cluster of points analyzed by k-means clustering and gives info to the cab on the hotspot nearest to the location of the user and is booked to pickup the user.

### **5.5.1 KMeans algorithm**

from sklearn.cluster import KMeans

Km = KMeans(n\_clusters=5,random\_state=0)

clus=df[['Lat','Lon']]

cluster1=clus[:10000]

y\_predict=Km.fit\_predict(clus)

df['cluster']=y\_predict

centroids=Km.cluster\_centers\_

clocation=pd.DataFrame(centroids,columns=['Latitude','Longitude'])

#Plotting the centroids on x-y graph of latitude and longitude.

plt.grid(zorder=0)

plt.scatter(clocation['Latitude'],clocation['Longitude'],marker="X",c='Red',s=50)

plt.show()

#Plotting the centroids calculated by k-means on the map of New York City imported by Folium.

import folium

map=folium.Map(location=[clocation.Latitude.mean(),clocation.Longitude.mean()],zoom\_start =10, control\_scale=True)

for index, location\_info in clocation.iterrows():

folium.Marker([location\_info["Latitude"],location\_info["Longitude"]], popup=location\_info).add\_to(map)

Map

#Predicting the pickup of the cab from the particular cluster and showing the cluster coordinates.

new\_location= [(40.7898, -73.9795)]

Km.predict(new\_location)

### 5.5.2 Kmedoid algorithm

from sklearn\_extra.cluster import KMedoids

kmedoids = KMedoids(n\_clusters=5, random\_state=0)

clus=df[['Lat','Lon']]

clus

cluster1=clus[:10000]

cluster1

kmedoids.fit(cluster1)

medoid = kmedoids.cluster\_centers\_

medoid

dlocation=pd.DataFrame(medoid,columns=['Latitude','Longitude'])

Dlocation

#Plotting the centroids on x-y graph of latitude and longitude.

plt.grid(zorder=0)

plt.scatter(dlocation['Latitude'], dlocation['Longitude'],marker="X",c='Red',s=50)

plt.title('K-medoids Clustering')

plt.show()

#Plotting the clusters calculated by k-medoid on the map of New York City imported by Folium.

import folium

Map2 = folium.Map(location=[dlocation.Latitude.mean(), dlocation.Longitude.mean()], zoom\_start=10, control\_scale=True)

for index, location\_info in dlocation.iterrows():

folium.Marker([location\_info["Latitude"],location\_info["Longitude"]],popup=location\_info).a dd\_to(map2)

Map2

#Predicting the pickup of the cab from the particular cluster and showing the cluster coordinates.

new\_points = [[ 40.7481, -73.9892]] # New location points to be predicted

cluster\_labels = kmedoids.predict(new\_points)

cluster\_labels

### 5.5.3 Gaussian Mixture Models

from sklearn.mixture import GaussianMixture

gmm = GaussianMixture(n\_components=5)

gmm.fit(cluster1)

cluster\_centers = gmm.means\_

cluster\_centers

elocation=pd.DataFrame(cluster\_centers ,columns=['Latitude','Longitude'])

elocation

#Plotting the centroids on x-y graph of latitude and longitude.

plt.grid(zorder=0)

plt.scatter(elocation['Latitude'],elocation['Longitude'],marker="X",c='Red',s=50)

plt.title('Gaussian Mixture Models')

plt.show()

#Plotting the clusters calculated by Gaussian Mixture Models on the map of New York City imported by Folium.

import folium

Map3 = folium.Map(location=[elocation.Latitude.mean(), elocation.Longitude.mean()], zoom\_start=10, control\_scale=True)

for index, location\_info in elocation.iterrows():

folium.Marker([location\_info["Latitude"],location\_info["Longitude"]], popup=location\_info).add\_to(map3)

Map3

#Predicting the pickup of the cab from the particular cluster and showing the cluster coordinates.

new\_point = [[ 40.221472, -74.035064]] # New data point to be predicted

cluster\_label = gmm.predict(new\_point)

cluster\_label

### 5.5.4 Agglomerative Clustering

from sklearn.cluster import AgglomerativeClustering

n\_clusters = 5 # Number of clusters

agglomerative = AgglomerativeClustering(n\_clusters=n\_clusters)

agglomerative.fit(cluster1)

labels = agglomerative.labels\_

cluster\_centers\_ = [ ]

for label in range(n\_clusters):

cluster\_points = cluster1[labels == label]

cluster\_center = cluster\_points.mean(axis=0)

cluster\_centers\_.append(cluster\_center)

cluster\_centers\_ = np.array(cluster\_centers\_)

cluster\_centers\_

glocation=pd.DataFrame(cluster\_centers\_,columns=['Latitude','Longitude'])

glocation

#Plotting the centroids on x-y graph of latitude and longitude.

plt.grid(zorder=0)

plt.scatter(glocation['Latitude'],glocation['Longitude'],marker="X",c='Red',s=50)

plt.title('Agglomerative Clustering')

plt.show()

#Plotting the cluster calculated by Agglomerative on the map of New York City imported by Folium.

import folium

Map4 = folium.Map(location=[glocation.Latitude.mean(), glocation.Longitude.mean()], zoom\_start=10, control\_scale=True)

for index, location\_info in glocation.iterrows():

folium.Marker([location\_info["Latitude"],location\_info["Longitude"]], popup=location\_info).add\_to(map4)

Map4

## 5.6 Interface Implementation

#### 5.6.1 Flask

Flask is a web framework, it’s a Python module that lets you develop web applications easily. It’s has a small and easy-to-extend core: it’s a microframework that doesn’t include an ORM (Object Relational Manager) or such features.It does have many cool features like url routing, template engine. It is a WSGI web app framework.

from flask import Flask, render\_template, request

import folium

from folium.plugins import MarkerCluster

from sklearn.cluster import KMeans

import pandas as pd

app = Flask(\_name\_)

@app.route('/', methods=['GET', 'POST'])

def index():

if request.method == 'POST':

lat = request.form['lat']

lon = request.form['lon']

# perform clustering using KMeans

kmeans = KMeans(n\_clusters=7, random\_state=42).fit(X)

cluster = kmeans.predict([[lat, lon]])[0]

# generate map with cluster centroids

centroid = kmeans.cluster\_centers\_

map = folium.Map(location=[lat, lon], zoom\_start=15)

marker\_cluster = MarkerCluster().add\_to(map)

pt=-1

for point in range(0, len(centroid)):

color = 'blue' if point == cluster else 'green' # set the color of the marker based on the cluster

number = str(pt+1) # set the number of the marker

pt+=1

folium.Marker(

location=centroid[point],

popup=folium.Popup(str(centroid[point])),

icon=folium.DivIcon(

html=f'<div style="font-size: 12pt; font-weight: bold; color: white; background-color: {color}; border-radius: 50%; width: 25px; height: 25px; display: flex; justify-content: center; align-items: center;">{number}</div>',

icon\_size=(30, 30)

)

).add\_to(marker\_cluster)

folium.Marker( # add a marker for the predicted point

location=[lat, lon],

popup='Predicted point',

icon=folium.Icon(icon='star', color='red')

).add\_to(map)

map = map.repr\_html()

return render\_template('index.html', lat=lat, lon=lon, cluster=cluster, map=map)

else:

return render\_template('index.html')

if \_name=="\_\_main\_":

data = pd.read\_csv('/config/workspace/Dataset/uber-raw-data-sep14.csv')

X = data[['Lat', 'Lon']]

app.run(host="0.0.0.0")

### 5.6.2 HTML code

### Input Interface

The code is an HTML Uber Data Analysis. It allows users to input latitude and longitude coordinates and submit them for clustering prediction. The predicted cluster and its corresponding location information are displayed on the web page along with a Google Map

<!DOCTYPE html>

<html>

<head>

<title>Uber Data Analysis using K-Means</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f2f2f2;

}

h1 {

text-align: center;

margin-top: 50px;

color: #333333;

text-shadow: 2px 2px #cccccc;

}

form {

text-align: center;

margin-top: 50px;

border: 1px solid #cccccc;

padding: 20px;

border-radius: 5px;

background-color: #ffffff;

width: 50%;

margin: 0 auto;

}

input[type="text"] {

padding: 10px;

border-radius: 5px;

border: 1px solid #cccccc;

width: 50%;

margin-bottom: 20px;

font-size: 16px;

}

input[type="submit"] {

background-color: #4CAF50;

color: #ffffff;

padding: 10px;

border-radius: 5px;

border: none;

cursor: pointer;

font-size: 16px;

}

input[type="submit"]:hover {

background-color: #3e8e41;

}

p {

text-align: center;

margin-top: 50px;

font-size: 20px;

color: #333333;

}

#map {

height: 700px; /\* Adjust height to increase map size \*/

width: 100%;

margin-top: 50px;

border-radius: 5px;

overflow: hidden;

box-shadow: 2px 2px 5px #cccccc;

}

.error-message {

color: red;

margin-top: 20px;

text-align: center;

}

</style>

</head>

<body>

<h1>KMeans Clustering Demo</h1>

<form method="POST">

<label for="lat">Latitude:</label>

<br>

<input type="text" name="lat" required><br>

<label for="lon">Longitude:</label>

<br>

<input type="text" name="lon" required><br>

{% if error %}

<p class="error-message">{{ error }}</p>

{% endif %}

<input type="submit" value="Submit">

</form>

{% if lat and lon %}

<p>Input latitude: {{ lat }}</p>

<p>Input longitude: {{ lon }}</p>

{% endif %}

{% if cluster is not none %}

{% if cluster == 0 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near New York.</p>

{% elif cluster == 1 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near Rikers Island.</p>

{% elif cluster == 2 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near Manhattan.</p>

{% elif cluster == 3 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near NY 25A.</p>

{% elif cluster == 4 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near Park Avenue.</p>

{% elif cluster == 5 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near John F Kennedy International Airport.</p>

{% elif cluster == 6 %}

<p>Predicted cluster: {{ cluster }}</p>

<p>Near Evergreen Cemetery.</p>

{% endif %}

{% endif %}

{% if map %}

<div id="map">{{ map|safe }}</div>

{% endif %}

</body>

</html>

### Output Interface code

<!DOCTYPE html>

<html>

<head>

<title>Uber Data Clustering</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<style>

#map {

height: 500px;

width: 100%;

}

</style>

</head>

<body>

<h1>Uber Data Clustering</h1>

<form method="POST" action="{{ url\_for('predict') }}">

<label for="latitude">Latitude:</label>

<input type="text" id="latitude" name="latitude" required>

<br>

<label for="longitude">Longitude:</label>

<input type="text" id="longitude" name="longitude" required>

<br><br>

<button type="submit">Predict Cluster</button>

</form>

<br><br>

<div id="map"></div>

<script>

function initMap() {

var map = new google.maps.Map(document.getElementById('map'), {

zoom: 12,

center: {lat: 40.7589, lng: -73.9851} // default location - New York City

});

var labels = ['Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5'];

var colors = ['red', 'blue', 'green', 'cyan', 'magenta'];

{% if cluster\_centers %}

{% for i in range(cluster\_centers.shape[0]) %}

var center{{ i }} = new google.maps.Marker({

position: {lat: {{ cluster\_centers[i][0] }}, lng: {{ cluster\_centers[i][1] }}},

map: map,

label: labels[{{ i }}],

icon: {

path: google.maps.SymbolPath.CIRCLE,

scale: 10,

fillColor: colors[{{ i }}],

fillOpacity: 1,

strokeWeight: 0

}

});

{% endfor %}

{% endif %}

{% if prediction %}

var marker = new google.maps.Marker({

position: {lat: {{ prediction[0] }}, lng: {{ prediction[1] }}},

map: map

});

{% endif %}

}

</script>

<scriptasyncdefersrc="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY&callb ack=initMap"></script>

</body>

</html>

# RESULTS AND DISCUSSION

The results discussed are based based on the following figures of user interface

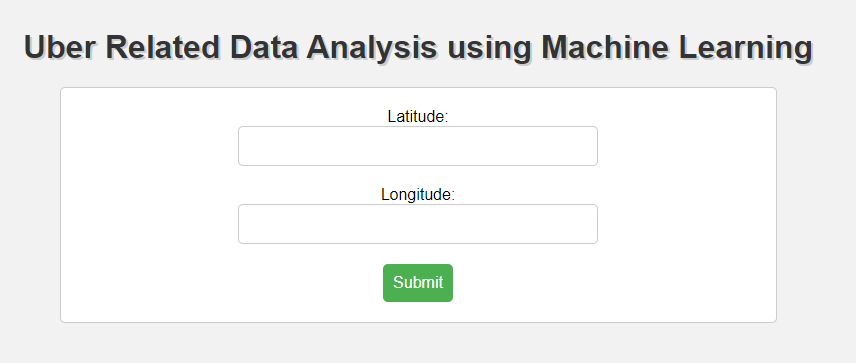


Fig 6.1- Input the latitude and the longitude for clustering prediction

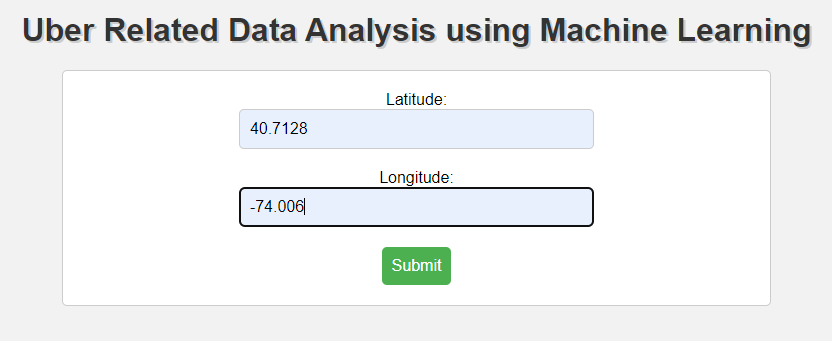


Fig 6.2-Inserting the latitude and the longitude points

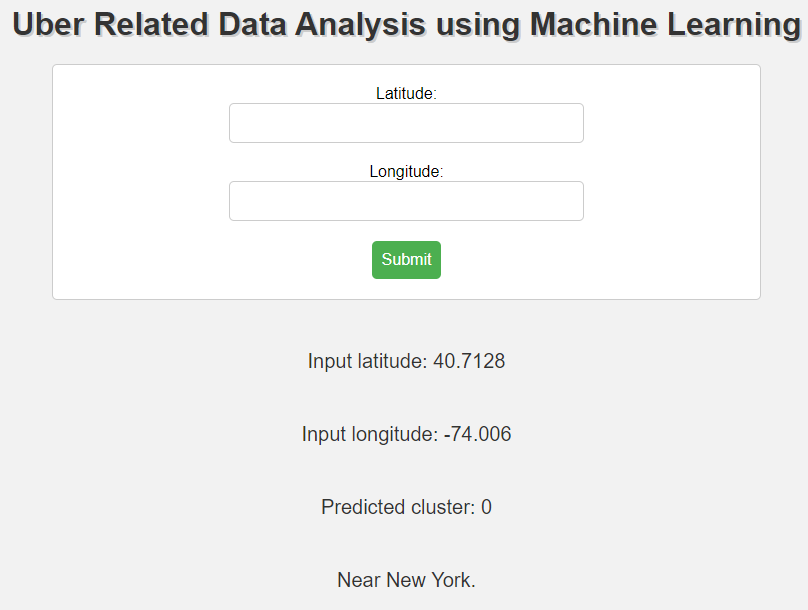


Fig 6.3- Prediction of the cluster

The new location points will predict the nearest cluster and show the point in the map given below. For every point it show the nearest cluster

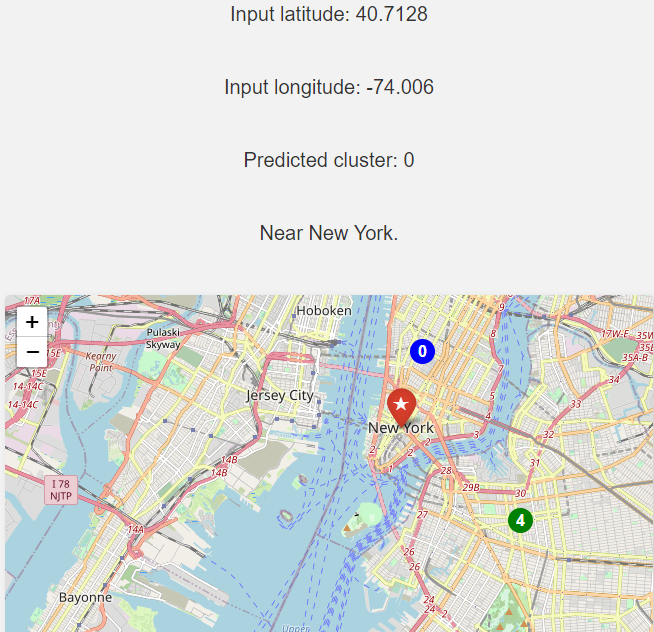
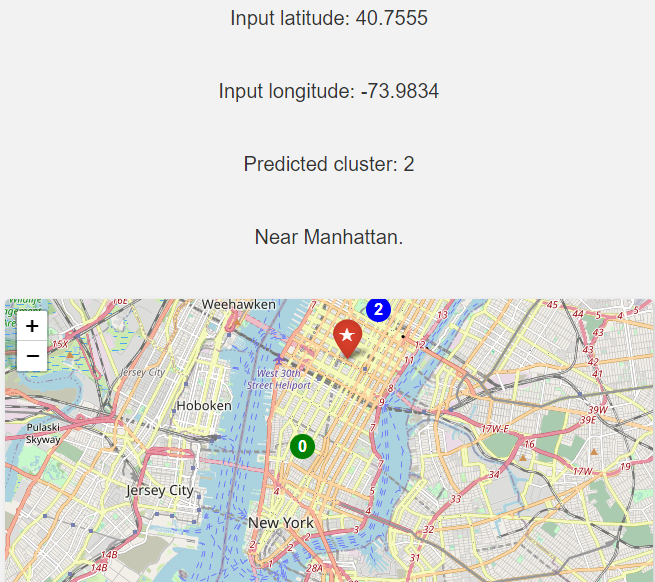


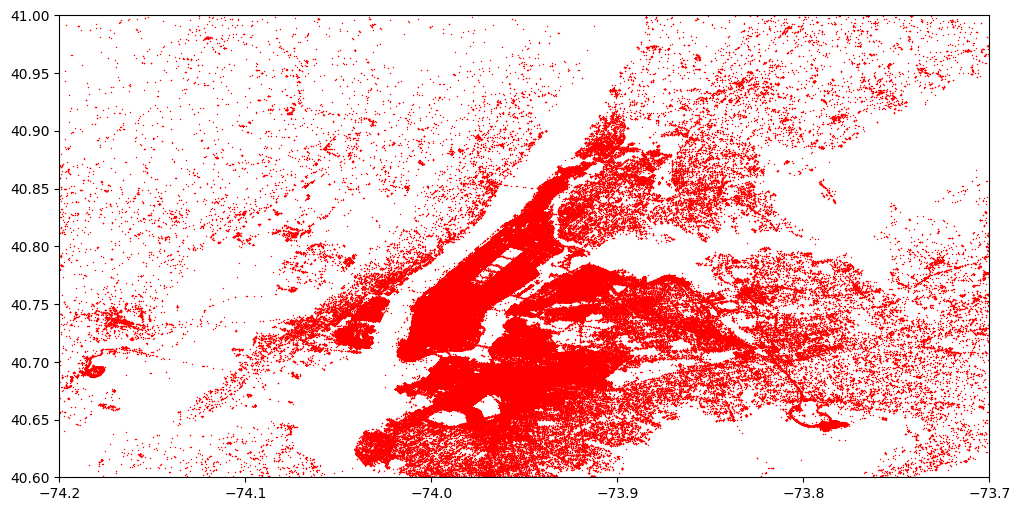
Fig 6.4 -prediction of cluster along with the location

Fig 6.5- prediction of the new points

For every point kmean predict the cluster which is nearest to the point and show the predicted cluster and illustrate the pin for the point given points

Result and discussion of the prediction in the background of this machine learning project

The program predicts the pickup location of the cab based on the clusters plotted using applied by KMeans, KMediod, Agglomerative clustering and Gaussian Mixture Models for appropriate cab scheduled for pickup.The results discussed are based on the following figures below

 Fig 6.6- Plotting the collection of points through which cab has travelled.

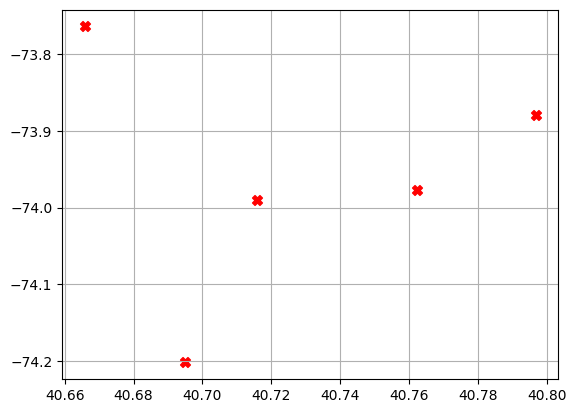


Fig 6.7-Plotting the clusters on x-y graph of latitude and longitude.

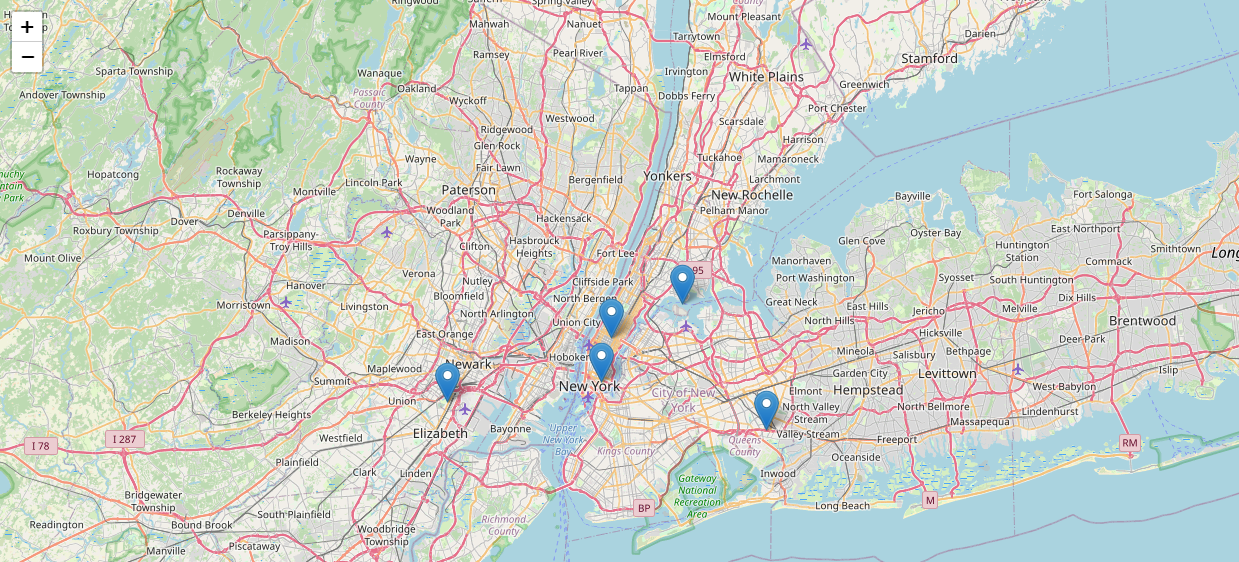


Fig 6.8-Plotting the clusters calculated by k-means on the map of New York City imported by Folium.

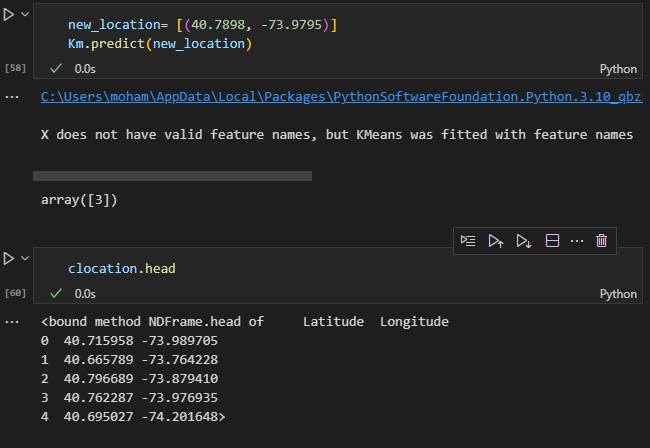


Fig 6.9-Predicting the pickup of the cab from the particular cluster and showing the cluster coordinates

# FUTURE SCOPE

The future work suggests that the system will provide the location to the user. The algorithm then records the time, latitude, longitude of the trip and assigns it to a cluster nearest to the passenger location where a cab is scheduled for pickup. We can also predict the passenger count on each district to deploy more cabs to the clustered coordinates using convolutional neural networks (CNN)

We can use this data for training a model using ML and building a smart AI based predictive system. Model can automatically send the insights to the authorities or drivers related to areas having most trips and passenger count in certain areas. This big data can be used to study passenger's behavior.

# CONCLUSION

The conclusion of the project is to project a basic outline of trips travelled with respect to latitude and longitude of locations and pinpoint the locations travelled with respect to the frequency of trips travelled by a uber cab during the day and also based on the cross analyzing of the dataset based on the latitude and longitude of the point travelled by the cab which is then analyzed by deploying kmeans clustering which classifies the locations on the basis of centroids and then orders the frequency of trips based on labels or clusters. By the location given by the user, the algorithm predicts the cluster nearest to the location so that cab can be assigned to the user for pickup. The merit of the project is that it explains the functioning of how cabs are assigned to passengers based on an unsupervised algorithm and also explains the key concepts of machine learning. The limitations of the project are that the algorithm deployed may be inefficient for huge data for over 10 years.

# BIBLIOGRAPHY

1. Poulsen, L.K., Dekkers, D., Wagenaar, N., Snijders, W., Lewinsky, B., Mukkamala, R.R.andVatrapu, R., 2016, June. Green Cabs vs. Uber in New York City. In 2016 IEEEInternational Congress on Big Data (BigData Congress) (pp. 222-229). IEEE.

2. Faghih, S.S., Safikhani, A., Moghimi, B. and Kamga, C., 2017. Predicting Short-Term UberDemand Using Spatio-Temporal Modeling: A New York City Case Study. arXiv preprintarXiv:1712.02001.

3.Guha, S. and Mishra, N., 2016. Clustering data streams. In Data stream management (pp.169-187). Springer, Berlin, Heidelberg.

4. Ahmed, M., Johnson, E.B. and Kim, B.C., 2018. The Impact of Uber and Lyft on TaxiService Quality Evidence from New York City. Available at SSRN 3267082.

5. Wallsten, S., 2015. The competitive effects of the sharing economy: how is Uber changingtaxis. Technology Policy Institute, 22, pp.1-21.

6. Sotiropoulos, D.N., Pournarakis, D.E. and Giaglis, G.M., 2016, July. A genetic algorithmapproach for topic clustering: A centroidbased encoding scheme. In 2016 7th InternationalConference on Information, Intelligence, Systems & Applications (IISA) (pp. 1-8). IEEE

7. Faghih, S.S., Safikhani, A., Moghimi, B. and Kamga, C., 2019. Predicting Short-TermUber Demand in New York City Using Spatiotemporal Modeling. Journal of Computing inCivil Engineering, 33(3), p.05019002.

8. Shah, D., Kumaran, A., Sen, R. and Kumaraguru, P., 2019, May. Travel Time EstimationAccuracy in Developing Regions: An Empirical Case Study with Uber Data in Delhi-NCR✱.In Companion Proceedings of The 2019 World Wide Web Conference (pp. 130-136). ACM.

9. Kumar, A., Surana, J., Kapoor, M. and Nahar, P.A., CSE 255 Assignment II PerfectingPassenger Pickups: An Uber Case Study.

10. L.Liu, C.Andris, and C.Ratti , “Uncovering cabdrivers behaviour patterns from their digital traces”,Compu. Environ.UrbanSyst.,vol.34,no.6,pp.541-548,2010

11. R.H.Hwang,Y.L.Hsueh , and Y.T.Chen,”An effective taxi recommender system model on a spatio-temporal factor analysis model,”Inf.Sci.,vol.314,pp.28-40,2015.

12. Vigneshwari, S., and M. Aramudhan. "Web information extraction on multiple ontologies based on concept relationships upon training the user profiles." In Artificial Intelligence and Evolutionary Algorithms in Engineering Systems, pp. 1-8. Springer, New Delhi, 2015.

13. L. Rayle, D. Dai, N. Chan, R. Cervero, and S. Shaheen, “Just a better taxi? a survey-based comparison of taxis, transit, and ridesourcing services in san francisco,” Transport Policy, vol. 45, 01 2016.

14. O. Flores and L. Rayle, “How cities use regulation for innovation: the case of uber, lyft and sidecar in san francisco,” Transportation research procedia, vol. 25, pp. 3756–3768, 2017. [15] H. A. Chaudhari, J. W. Byers, and E. Terzi, “Putting data in the driver’s seat: Optimizing earnings for on-demand ride-hailing,” in Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM, 2018, pp. 90–98.