Design of Bus Routes

# Rapaka Joseph Ravi Teja

# MM15B024

#### Section A: Data Collection

1. Taxi service provides the most flexible transportation service in a city. The data collection activity can provide **frequency** of taxi rides from one location to another location indicate high **demand** of travelling on the route over a **period of time**. **Peak timings** can be foundso they can charge higher than usual price. From the **clustering** **analysis** of data, we may get many clusters of taxi rides which is covered by the existing public buses. These **clusters** represent the group of passengers who is willing to pay a bit more for easier, faster, and more convenient way to reach home.

So, the 3 distinct benefits of the data collection activity that you would pitch to the taxi drivers are

1. Demand of Taxi’s in particular route.
2. Timings of high frequency of Taxi rides.
3. Clusters of demand of Taxi’s in Suburban areas
4. A) A value in decimal degrees to 5 decimal places is precise to 1.1132 **meter** at the equator (latitude). Assign a name for every 300 m2 area. Create a new **categorical** column such as **Pickup location** and **Drop location** with the assigned names in them when coordinate points fell in that area.

B) Another categorical attribute that can be useful is **Traffic –** Heavy, Medium, Less

C) **Hours -** Frequency of Taxi Rides in each and every hour.

#### Section B: Data Validation

1. To validate the given data let us consider Latitude and Longitudes attributes. Calculate the

mean (pickup\_latitude, pickup\_longitude)

mean (dropoff\_longitude, dropoff\_latitude) and find the variance of every row and plot the graph between means. The elements with high variance are the **Outliners** that should be removed.

Outliners = points where not in region of **(mean (dropoff\_longitude , pickup\_longitude) ± Standard Deviation).**

1. Normally spurious rows exist because some times when You combining multiple data sets into one big data set. There may **be mishandling of data**, Some times if two variables have no correlation but used to collect the data. Then you will have a chance of Spurious rows.

#### ****Section C: Post-analysis****

1. From clustering the data by DBSCAN clustering process we can find Clusters. Clustering means area where maximum cab rides taking place. This happens due to improper bus routes and bus timings. With the analysis of data, we can update the bus routes by joining the clusters from one end to the other end.
2. First remove all the Outliners from the data
3. Group the data according to certain period of time
4. Plot Clustering diagram by DBSCAN clustering process.

Various Stakeholders can be benefitted from the Analysis for example Auto association, Formation of clusters also means there is huge crowd residing there. So, there will be Companies, Offices, Restaurants etc can be developed.

1. **A**) Normally People Travelling through Aeroplanes and trains. They use cabs to go to destinated places. WE can find some relation data of Taxi rides from the data of people coming from Airports, Trains etc.

**B)** Next case people working in Companies, Offices, Colleges they also prefer going in cabs. We can find the data from No of companies in particular area with employer details and number of car owners etc.

#### Bus Route Design

1. we will find the routes by forming the chain of clusters over the map area.. First remove all the Outliners from the data
2. Group the data according to certain period of time
3. Plot Clustering diagram by DBSCAN clustering process.

Import pandas as pd, numpy as np, matplotlib.pyplot as plt

from sklearn.cluster import DBSCAN

import matplotlib.pyplot as plt

cars = pd.read\_csv(r'C:\Users\Dell\Desktop\train.csv')

cars.head()

cars.describe

#removed the Outliners fromt he data set

cars = cars.replace(0, np.nan)

cars = cars.dropna(how='all', axis=0)

cars = cars.replace(np.nan, 0)

# DSBAN clustering

kms\_per\_radian = 6371.0088

epsilon = 1.5 / kms\_per\_radian

coords = cars.as\_matrix(columns=['pickup\_longitude', 'pickup\_latitude'])

db = DBSCAN(eps=epsilon, min\_samples=1, algorithm='ball\_tree', metric='haversine').fit(np.radians(coords))

cluster\_labels = db.labels\_

num\_clusters = len(set(cluster\_labels))

clusters = pd.Series([coords[cluster\_labels == n] for n in range(num\_clusters)])

print('Number of clusters: {}'.format(num\_clusters))