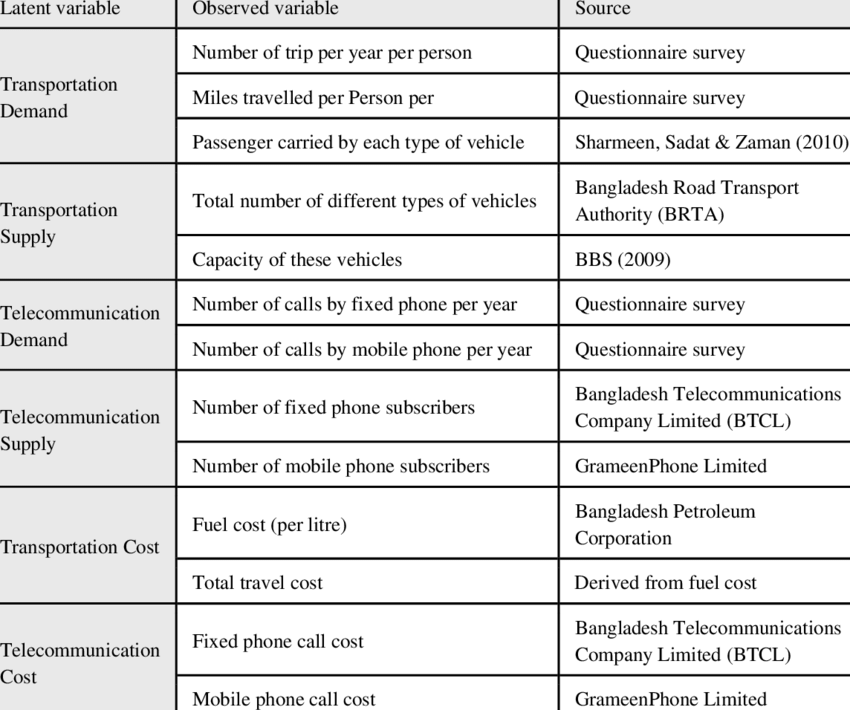
**PUBLIC TRANSPORTATION ANALYSIS:**

**CONTENT:**

Experts, in transportation utilize the capabilities of IBM Watson to streamline and improve urban transportation systems. By harnessing the power of Watsons intelligence and data analytics they can effectively. Optimize routes, schedules and passenger experiences. This ultimately leads to user friendly public transportation solutions in cities. The valuable insights provided by IBM Watson enable these experts to make decisions based on data benefiting both passengers and transit authorities. As a result the overall quality and sustainability of public transport services are enhanced.

**DATA SOURCE:**



**DATA COLLECTION:**

Collecting data for the analysis of transportation involves gathering an array of information, with the aim of evaluating and enhancing the effectiveness, safety and overall user satisfaction of public transit systems.

**DATA PROCESSING:**

The analysis of public transport data entails converting data into insights that can enhance the effectiveness, safety and overall quality of public transportation services.

**1.Data collection:**

As mentioned earlier information is gathered from a range of sources such, as sensors, surveys, GPS systems, weather reports and other relevant data providers.

**2 Data visualization:**

Presenting information using aids such, as charts, graphs and maps proves to be a way of conveying findings to stakeholders and decision makers.

**Creating a visualization:**

When it comes to analyzing public transportation data using IBMs tools and technologies one would usually rely on a data visualization platform such, as IBM Cognos, IBM Watson Analytics or IBM Data studio.

**Description:**

This visualization provides a depiction of how public transportation ridership fluctuates throughout the year enabling stakeholders to gain insights, into the trends and make choices based on data.

**Program:**

**%matplotlib inline**

**import numpy as np # linear algebra**

**import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)**

**import matplotlib.pyplot as plt**

**import datetime**

**import os**

**from math import sqrt**

**import warnings**

**## For Multiple Output in single cell**

**from IPython.core.interactiveshell import InteractiveShell**

**InteractiveShell.ast\_node\_interactivity = "all"**

**warnings.filterwarnings('ignore')**

**data = pd.read\_csv('/content/20140711.CSV')**

**data.shape**

**data.head(10)**

**out\_geo = pd.read\_csv('/content/output\_geo.csv')**

**out\_geo.shape**

**out\_geo.head()**

**from math import sin, cos, sqrt, atan2, radians**

**def calc\_dist(lat1,lon1):**

**## approximate radius of earth in km**

**R = 6373.0**

**dlon = radians(138.604801) - radians(lon1)**

**dlat = radians(-34.921247) - radians(lat1)**

**a = sin(dlat / 2)\*2 + cos(radians(lat1)) \* cos(radians(-34.921247)) \* sin(dlon / 2)\*2**

**c = 2 \* atan2(sqrt(a), sqrt(1 - a))**

**return R \* c**

**out\_geo['dist\_from\_centre'] = out\_geo[['latitude','longitude']].apply(lambda x: calc\_dist(\*x), axis=1)**

**out\_geo.head()**

**out\_geo['type'].fillna('street\_address',inplace=True)**

**out\_geo['type'] = out\_geo['type'].apply(lambda x: str(x).split(',')[-1])**

**out\_geo['type'].unique()**

**data['WeekBeginning'] = pd.to\_datetime(data['WeekBeginning']).dt.date**

**data['WeekBeginning'][1]**

**data= pd.merge(data,out\_geo,how='left',left\_on = 'StopName',right\_on = 'input\_string')**

**data.head(5)**

**data.shape**

**col = ['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning','NumberOfBoardings',**

**'latitude', 'longitude','postcode','type','dist\_from\_centre']**

**data = data[col]**

**grouped = data.groupby(['StopName','WeekBeginning','type'])**

**grouped = data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardings': ['sum', 'count','max']})**

**grouped.columns = ["\_".join(x) for x in grouped.columns.ravel()]**

**grouped.head(10)**

**grouped.columns**

**st\_week\_grp = pd.DataFrame(grouped).reset\_index()**

**st\_week\_grp.shape**

**st\_week\_grp.head()**

**st\_week\_grp1 = pd.DataFrame(st\_week\_grp.groupby('StopName')["WeekBeginning"].count()).reset\_index()**

**st\_week\_grp1.head()**

**fig,axrr=plt.subplots(2,2,figsize=(15,15))**

**ax=axrr[0][0]**

**ax.set\_title("No of Boardings")**

**data['NumberOfBoardings'].value\_counts().sort\_index().head(20).plot.bar(ax=axrr[0][0])**

**ax=axrr[0][1]**

**ax.set\_title("WeekBeginning")**

**data['WeekBeginning'].value\_counts().plot.area(ax=axrr[0][1])**

**ax=axrr[1][0]**

**ax.set\_title("most Busiest Route")**

**data['RouteID'].value\_counts().head(10).plot.bar(ax=axrr[1][0])**

**ax=axrr[1][1]**

**ax.set\_title("least Busiest Route")**

**data['RouteID'].value\_counts().tail(10).plot.bar(ax=axrr[1][1])**

**data['WeekBeginning'].value\_counts().mean()**

**x = data["dist\_from\_centre"]**

**distance\_10 = []**

**distance\_10\_50 = []**

**distance\_50\_100 = []**

**#distance\_100\_ = []**

**distance\_100\_more = []**

**total = 0**

**outlier = []**

**outlier\_ = 0**

**for i in x:**

**if(i<=10):**

**distance\_10.append(i)**

**total += 1**

**elif(i<=50):**

**distance\_10\_50.append(i)**

**total += 1**

**elif(i<=100):**

**distance\_50\_100.append(i)**

**total += 1**

**print(outlier\_)**

**y = len(distance\_10)+len(distance\_10\_50)+len(distance\_50\_100)**

**print(total)**

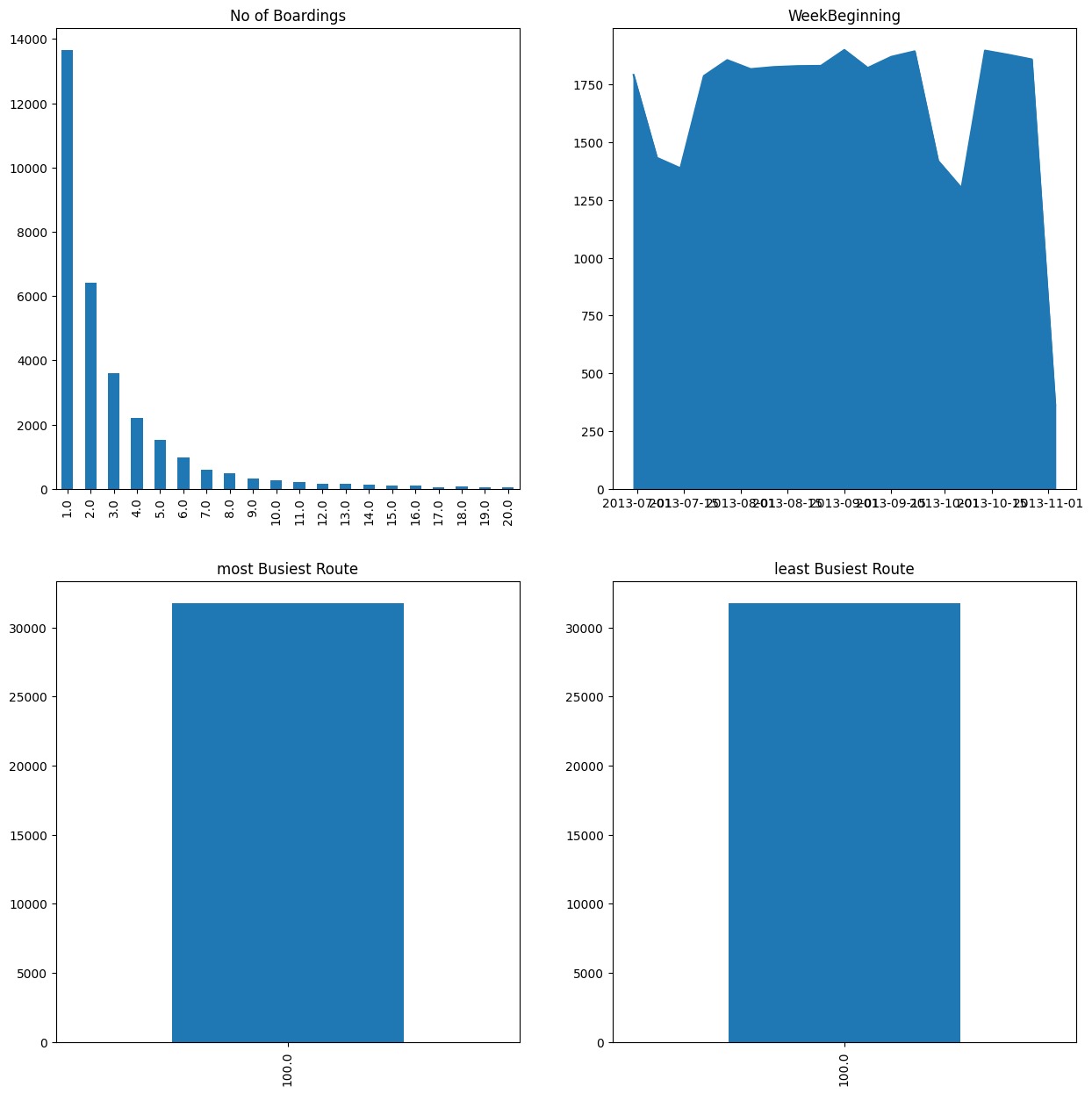
**print("passangers, boarding the buses in the radious of 10Km from the city center = ", (len(distance\_10)/total)\*100)**

**print("passanger, boarding the buses from the distance of 10Km to 50Km from the city center = ", (len(distance\_10\_50)/total)\*100)**

**print("passanger, boarding the buses from the distance of 50Km to 100 from the city center = ", (len(distance\_50\_100)/total)\*100)**

**grouped\_route = data.groupby(['RouteID']).agg({'NumberOfBoardings': ['sum', 'max']})**

**grouped\_route.columns = ["\_".join(x) for x in grouped\_route.columns.ravel()]**

**OUTPUT:**