SUBMITTED BY:Jayapriyadharshini.V

NM ID:au211521106068

# CODE FOR ANALYSIS OF DATA:

%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

data = pd.read\_csv('/content/cleaned\_data.csv')

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

data= pd.merge(data,out\_geo,how='left',left\_on = 'StopName',right\_on = 'input\_string')  
data.head(5)  
data.shape

(20431, 16)

#Columns to keep for further analysis  
col = ['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning','NumberOfBoardings',  
 'latitude', 'longitude','postcode','type']  
data = data[col]

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

# st\_week\_grp1 = pd.DataFrame(data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardings': ['sum', 'count']})).reset\_index()  
grouped = data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardings': ['sum', 'count','max']})  
grouped.columns = ["\_".join(x) for x in grouped.columns.ravel()]

<ipython-input-9-ac1d5add8db2>:3: FutureWarning: Index.ravel returning ndarray is deprecated; in a future version this will return a view on self.  
 grouped.columns = ["\_".join(x) for x in grouped.columns.ravel()]

grouped.head(10)  
grouped.columns

Index(['NumberOfBoardings\_sum', 'NumberOfBoardings\_count',  
 'NumberOfBoardings\_max'],  
 dtype='object')

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

StopName WeekBeginning type NumberOfBoardings\_sum \  
0 10 Holbrooks Rd 07-07-2013 street\_address 73.0   
1 10 Holbrooks Rd 08-04-2013 street\_address 245.0   
2 10 Holbrooks Rd 08-11-2013 street\_address 223.0   
3 10 Holbrooks Rd 09-01-2013 street\_address 194.0   
4 10 Holbrooks Rd 09-08-2013 street\_address 198.0   
  
 NumberOfBoardings\_count NumberOfBoardings\_max   
0 23 9.0   
1 26 100.0   
2 26 89.0   
3 25 73.0   
4 26 79.0

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

StopName WeekBeginning  
0 10 Holbrooks Rd 12  
1 10 Marion Rd 12  
2 10A Marion Rd 12  
3 11 Marion Rd 12  
4 11 Portrush Rd 12

#Gathering only the Stop Name which having all 54 weeks of Dat  
aa = list(st\_week\_grp1[st\_week\_grp1['WeekBeginning'] == 54]['StopName'])  
aa[1:10]

[]

bb = st\_week\_grp[st\_week\_grp['StopName'].isin(aa)]  
bb.head()  
bb.shape

(0, 6)

new\_data = data[data['StopName'].isin(aa)]  
new\_data.shape  
print("data without stopage removing: ", data.shape)  
print("data, after removing stoppage not having the data of whole 54 weeks: ", new\_data.shape)

data without stopage removing: (20431, 10)  
data, after removing stoppage not having the data of whole 54 weeks: (0, 10)

new\_data.head(2)  
filtered\_data = new\_data[new\_data['latitude'] <= 100]  
filtered\_data.shape

(0, 10)

data = filtered\_data.copy()  
data.shape

(0, 10)

#No of boarding for each stopage in all weeks  
#bb["StopName"].groupby(NumberOfBoardings\_sum)  
stopageName\_with\_boarding = bb.groupby(['StopName']).agg({'NumberOfBoardings\_sum': ['sum']})  
  
#stopageName\_with\_boarding.columns = ["\_".join(x) for x in stopageName\_with\_boarding.columns.ravel()]  
#stopageName\_with\_boarding.head()  
stopageName\_with\_boarding = pd.DataFrame(stopageName\_with\_boarding.reset\_index())

stopageName\_with\_boarding.columns = ["StopName", "Total\_boarding\_on\_the\_stopage"]  
#stopageName\_with\_boarding.shape  
stopageName\_with\_boarding.head()

Empty DataFrame  
Columns: [StopName, Total\_boarding\_on\_the\_stopage]  
Index: []

## save the aggregate data  
#bb.to\_csv('st\_week\_grp.csv', index=False)

data.nunique()  
#data.isnull().sum()  
#data['WeekBeginning'].unique()

TripID 0  
RouteID 0  
StopID 0  
StopName 0  
WeekBeginning 0  
NumberOfBoardings 0  
latitude 0  
longitude 0  
postcode 0  
type 0  
dtype: int64

DATA VISUALISATION USING IBM COGNOS:









