**PROJECT SUBMISSION FOR DATA SCIENCE WITH PYTHON**

**PROJECT REPORT**

**CUSTOMER SERVICE REQUESTS ANALYSIS**

Hello sir,

This is **ABIRAMI P.**

**Analysis Tasks to be performed (CODE FOR PROJECT SOLUTION)**

(Perform a service request data analysis of New York City 311 calls)

#Import necessary packages

import pandas as pd

import numpy as np

import scipy.stats as stats

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

import warning

#using set\_option function to display the needed no. of rows and columns

pd.set\_option ('display.max\_columns',30)

pd.set\_option ('display.max\_row',800)

1. Import a 311 NYC service request.

#Load the data set using pandas

df-pd.read-excel\_ (‘311\_Service\_Requests\_from\_2010\_to\_Present.excel’,low\_money=False)

#Head method to view first 3 rows of the dataset

df.head(3)

#understand the dataset

df.shape

df.info( )

#changing the column names for easy access

df.column – df.columns.str.replace(“ “, “ \_“).str. lower ()

#create new dataset with necessary columns

nyc\_dtaset – df [ [ ‘unique-key’, ‘created-date’, ‘closed-date’, ‘agency’, ‘agency – name’, ‘ complaint-type’, ‘descriptor’, ‘location-type’, ‘incident \_ zip’, ‘incident \_ address’ , ‘street\_name’ , ‘ cross \_street\_name\_1’ , ‘ cross \_street\_name\_2’ , ‘ address\_tpe’ , ‘citt’, ‘resolution\_desciption’, ,resolution\_action\_updated\_date; , ‘community\_board’ , ‘borough’ , ‘x\_coordinate\_(state\_plane)’ , ‘ y\_coordinate\_(state\_plate)’ , ‘park\_borough’ , ‘latitude’ , ‘longitute’ , ‘location’] ]

#check for duplicates and NA values

nyc\_dataset.duplicated( ).sum( )

nyc\_dataset.isna().sum()

#drop na values in necessary colimns

nyc\_dataset.dropna(subset-[‘city’, ‘latitude’], inplace-true

#write the cleaned dataset back

nyc\_dataset.to\_csv(‘nyc\_cleaned.csv’, index-False)

1. Provide major insights/patterns that you can offer in a visual format (graphs or tables); at least 4 major conclusions that you can come up with after generic data mining.

#Load the cleaned dataset

nyc – pd.read\_csv(‘nyc\_cleaned.csv’, parse\_dates=[‘created\_date’, ‘closed\_date’, ‘resolution\_action\_updated\_date’])

nyc.head(3)

#calculating the response time in hrs and in minutes

nyc[‘request\_closing\_timehrs’] - nyc [ ‘close\_date’].values-nyc[created\_date’].values

nyc[‘request\_closing\_time\_mins’] - nyc[ ‘request\_closing\_time\_hrs’]/np.timedelta64(1,’m’)

nyc.head()

1. Order the complaint types based on the average ‘Request Closing Time’, grouping them for different locations.

#visualizing complaint types based on the no of requests

Nyc[‘city’]= nyc [‘city’].str.lower( )str.lower( ).str.replace(“ “, “ \_ “)

Txt={‘weight’:’bold)

Plt.figure(figsize=(12,7))

Sns.countplot (x=’complaint\_type’, data=nyc,der=nyc[‘complaint\_type’].value\_count().index)

Plt.xticks(rotation=90)

Plt.title (“complaint types and no. of requests per complaint \_type”’ fontdict=txt)

Plt.xlable(‘complaint type’, fontdict=txt,labelpad=40)

Plt,ylabel(‘request’, fontdict=txt,labelpad=30)

Plt.show()

# visualizing no. of requests from top 15 cities

City\_top1=nyc [‘city’].value\_count().head(15)

Top15=city\_top.index

Txt={‘weight’:’bold’}

Plt.fingure(figsize=(28,7))

Sns.countplot(x=’city’,data=nyc[nyc.city.isin(top15)], oder=top15)

Plt.title(‘top 15 cities with no. of complaint request “, fontdict =txt)

Plt.xlabel(‘city’ , fontdict=txt,labelpad=20)

Plt.xticks(rotation=90)

Plt.ylabel(‘requests’, fontdict=txt, labelpad=30)

Plt.show()

#visualizing complaint types with average response time

Viz2=nyc [[‘complaint \_typ’, ‘request \_closing\_time\_mins’]]

C2= viz2.groupby(‘complaint\_type)(‘request\_closing\_time\_mins’].mean().toto\_frame()

C2=c2.sort\_value(‘request \_closing\_time\_mins’)

C2[‘complaint\_type’]=c2.index

Txt={‘weight’:’bond’}

Plt.figure(fingsize=(16,8))

Sns.barplot(x=’complaint\_type’, y=’request \_closing\_time\_mins’, data=c2)

Plt.title(“complaint types with average request resolved time (mins)”, fontdict=txt)

Plt.xlabel(“complaint type”, fontdict=txt,labelpad=30)

Plt.ylabel(“average request resolved time (mins)”, fontdict=txt,labelpad=30)

Plt.xticks(rotation=90)

Plt.show()

1. Perform a statistical test for the following:

#groping complaint by cities and finding mean response time for each complaint type

#sorting the mean response time of different complaint type for each city

City\_complainttype\_group=nyc.groupby{[‘city’, ‘complaint\_type’]} [‘request\_closing\_time\_mins’].mean().unstack(level=1)

City\_complaintype\_group=city\_complaintype\_group.T

Col=city\_complaintype\_group.columns

For I in col:

Exec(“{}-city\_complaintype\_group[‘{}’.sort\_values()”.format(I,i))

#visualizing the top 6cities with the mean response time sorted for different complait type

Plt.figure(figsize=(20,10))

Plt.subplots\_adjust(hspace=1.6,wspace=0.5)

Plt.suptitle(“top 6cities with more no. of complaints and their response time “, fontweight+”bold”, fontsize=”25”,y+1.1)

Txt={‘weight’:’bold’}

Plt.subplot(2,3,1)

Plt.title(‘brooklyn average complaint response time ‘, fontdict=txt, y=1.1)

Brooklyn.dopna().plot.bar()

Plt.xlabel(‘complaint type’,fontdict=txt,labelpad=20)

Plt.ylabel(average response time (mins) fontdict=txt,labelpad=30)

Plt.ylim(0,800)

Plt.subplot(2,3,2)

Plt.title(‘new york average complaint response time’, fontdict=txt,y=1.1)

New\_york.dropna().plot.bar()

Plt.xlabel(‘complaint type’, fontdict=txt, labelpad=20)

Plt.ylabel(‘average responsetime(mins)’, fontdict-txt,labelpad=30)

Plt.ylim(0,800)

Plt.subplot(2,3,3)

Plt.title(‘bronx average complaint response time’, fontdict=txt,y=1.1)

bronx.dropna().plot.bar()

Plt.xlabel(‘complaint type’, fontdict=txt, labelpad=20)

Plt.ylabel(‘average responsetime(mins)’, fontdict-txt,labelpad=30)

Plt.ylim(0,800)

Plt.subplot(2,3,4)

Plt.title(‘staten island average complaint response time’, fontdict,y=1.1)

Staten island.dropna().plot.bar()

Plt.xlabel(‘complaint type’, fontdict=txt, labelpad=20)

Plt.ylabel(‘average responsetime(mins)’, fontdict-txt,labelpad=30)

Plt.ylim(0,800)

Plt.subplot(2,3,5)

Plt.title(‘jamaica average complaint response time’, fontdict,y=1.1)

jamaica.dropna().plot.bar()

Plt.xlabel(‘complaint type’, fontdict=txt, labelpad=20)

Plt.ylabel(‘average responsetime(mins)’, fontdict-txt,labelpad=30)

Plt.ylim(0,800)

Plt.subplot(2,3,6)

Plt.title(‘astoria average complaint response time’, fontdict,y=1.1)

astoria.dropna().plot.bar()

Plt.xlabel(‘complaint type’, fontdict=txt, labelpad=20)

Plt.ylabel(‘average responsetime(mins)’, fontdict-txt,labelpad=30)

Plt.ylim(0,800)

Plt.show()

Please note: For the below statements you need to state the Null and Alternate and then provide a statistical test to accept or reject the Null Hypothesis along with the corresponding ‘p-value’.