**Comcast Telecom Consumer Complaints .**

**DESCRIPTION**

Comcast is an American global telecommunication company. The firm has been providing terrible customer service. They continue to fall short despite repeated promises to improve. Only last month (October 2016) the authority fined them a $2.3 million, after receiving over 1000 consumer complaints.  
The existing database will serve as a repository of public customer complaints filed against Comcast.  
It will help to pin down what is wrong with Comcast's customer service.

**Data Dictionary**

* Ticket #: Ticket number assigned to each complaint
* Customer Complaint: Description of complaint
* Date: Date of complaint
* Time: Time of complaint
* Received Via: Mode of communication of the complaint
* City: Customer city
* State: Customer state
* Zip code: Customer zip
* Status: Status of complaint
* Filing on behalf of someone

**Analysis Task**

To perform these tasks, you can use any of the different Python libraries such as NumPy, SciPy, Pandas, scikit-learn, matplotlib, and BeautifulSoup.

- Import data into Python environment.  
- Provide the trend chart for the number of complaints at monthly and daily granularity levels.  
- Provide a table with the frequency of complaint types.

* Which complaint types are maximum i.e., around internet, network issues, or across any other domains.

- Create a new categorical variable with value as **Open**and **Closed**. Open & Pending is to be categorized as Open and Closed & Solved is to be categorized as Closed.  
- Provide state wise status of complaints in a stacked bar chart. Use the categorized variable from Q3. Provide insights on:

* Which state has the maximum complaints
* Which state has the highest percentage of unresolved complaints

- Provide the percentage of complaints resolved till date, which were received through the Internet and customer care calls.

The analysis results to be provided with insights wherever applicable.

**Step By Step**

**Step 1.** Import all important library (NumPy, SciPy, Pandas, scikit-learn, matplotlib, and BeautifulSoup and more)

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.offline as py

import plotly.graph\_objs as go

**Step 2. Read Data**

Read Data in csv format

df=pd.read\_csv("Comcast\_telecom\_complaints\_data.csv")

**Step 3.** df.info()

The info() function is used to print a concise summary of a DataFrame. This method prints information about a DataFrame including the index dtype and column dtypes, non-null values and memory usage.

**Step 4.** df.head()

df. head() Returns the first 5 rows of the dataframe.

**Step 5**. df['Date']=pd.to\_datetime(df['Date'])

Pandas to\_datetime() method helps to convert string Date time into Python Date time object

**Step 6**. df['Month']=df['Date'].dt.month\_name()

dt. month\_name() function to return the month names of the underlying datetime data in the given series object. Return the names of the month in English language.

**Step 7**. df['Date'].value\_counts()

Use Series.value\_counts() function to find the unique value counts of each element in the given Series object.

**Step 8**. dates=df.groupby(['Date'])['Ticket #'].count()

**groupby**() :- Pandas groupby is used for grouping the data according to the categories and apply a function to the categories. It also helps to aggregate data efficiently.

**count**() :- The count() function is used to count elements on a list as well as a string.

grouping the Date and ticket in dates variable and count the elements on the list .

**Step 9.** daily=pd.DataFrame(dates).reset\_index()

Pandas reset\_index() is a method to reset index of a Data Frame. reset\_index() method sets a list of integer ranging from 0 to length of data as index.

Create a new Data Frame there has two col. "Data & Ticket #" then reset there index.

**Step 10**. daily.head()

Show Daliy new data frame

**Step 11**. daily.plot(x='Date',y='Ticket #' ,kind='line')

ploting daily Data Frame there x is 'Date' and y is 'Tickets' and there graphs are line format.kind function are show the graphs formats deside like that bar,line,box and more type of graphs.

**Step 12.** month=df.groupby('Month')['Ticket #'].count()

Grouping the two columns "Month" and "Tickets" . and count there elements and grouping month wise and store month variable

**Step 13.** month=pd.DataFrame(month).reset\_index()

Create a data frame there named are month and reset there index.

**Step 14.** month

show month data frame

**Step 15.** month.plot(x='Month',y='Ticket #',kind='bar')

ploting month Data Frame there x is 'Month' and y is 'Tickets #' and there graphs are bar format.

kind function are show the graphs formats deside like that bar,line,box and more type of graphs.

**Step 16**. df['Customer Complaint'].value\_counts()[:9]

Pandas Series.value\_counts() function return a Series containing counts of unique values.

The resulting object will be in descending order so that the first element is the most frequently

Top 9 Values Out of Customer Complaint.

**Step 17.** df['Customer Complaint'].value\_counts()[:9].plot.bar()

ploting top 9 Customer Complaint values in bar graph

**Step 18.** df['NewStatus']=['Open' if st=='Open' or st=='Pending' else 'Close' for st in df['Status']]

Create a new column "NewStatus".put a loop in it there loop are if st=='Open' or "pending" they return 'Open' else 'Close' for st in df['Status']

['Open' if st=='Open' or st=='Pending' else 'Close' for st in df['Status']]

**Step 19.** df['Status'].unique()

Show Unique element in Status column

**Step 20.** state\_complain=df.groupby(['State','NewStatus']).size().unstack()

Unstack :- unstack is similar to stack method, It also works with multi-index objects in dataframe, producing a reshaped DataFrame with a new inner-most level of column labels.

size() function count the number of elements along a given axis.

grouping two column 'State' and 'newStatus'and count number of elements and unstack(it are reshaped data frame ) and store new variable (state\_complain).

**Step 21.** state\_complain

show state\_complain on new dataframe open ya close status

**Step 22.** state\_complain.plot.bar(stacked=True,figsize=(10,7))

plt.show()

ploting state\_complain in bar graph. Close and open are bar stacked

**Step 23.** df.groupby('State').size().sort\_values(ascending=False)[:5]

will be grouped by state in the dataframe then sort the value in descending order and top 5 value get with by index

**Step 24.** unresolved\_data=df.groupby(['State','NewStatus']).size().unstack().fillna(0).sort\_values(by='Open',ascending=False)

Groupby State and NewStatus, then size came out, then unstacked after that, then filled the null value with 0 and then sorted according to the value of open

**Step 25.** unresolved\_data['unresolve\_cmp\_prct']=unresolved\_data['Open']/unresolved\_data['Open'].sum()\*100

A dataframe named unresolve\_cmp\_prct is created, then the perccentage of the open value is obtained.

**Step 26.** unresolved\_data

show data frame with unsolve Complaint in percantage format

**Step 27.** resolved\_data=df.groupby(['Received Via', 'NewStatus']).size().unstack()

grouping two column 'Received Via' and 'NewStatus'and count number of elements and unstack(it are reshaped data frame ) and store new variable (resolved\_data).

**Step 28.** resolved\_data

Show resolved\_data

**Step 29.** resolved\_data['resolved']=resolved\_data['Close']/resolved\_data['Close'].sum()\*100

A dataframe named resolved\_data is created, then the perccentage of the close value is obtained.

**Step 30.** resolved\_data.plot(kind='bar',figsize=(10,7))

Ploting resolved\_data in bar graph there value are Close, Open, Resolved

**Step 31.** internet\_issue=df[df['Customer Complaint'].str.contains('internet')].count()

Create new dataframe and count internet complaint from Customer Complaint

**Step 32.** internet\_issue

Show internet\_issue list

**Results**

* Internet has 843 close and 262 Open Complaint And Customer Care Call has 864 close and 255 open Complaint
* Georgia State has maximum complsints.
* Georgia has the highest percentage of unresolved complaints