```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sb

import datetime

from matplotlib import dates as mpl_dates


#adding csv file

df = pd.read_csv('Comcast_telecom_complaints_data.csv')



df['Date_month_year'] = pd.to_datetime(df['Date_month_year'])

df['Date_Dup'] = df['Date_month_year'].dt.strftime('%m-%d-%Y')

df['Date_Dup']


df['month'] = pd.DatetimeIndex(df['Date_month_year']).month

ext = df['month']

temp = df['month'].value_counts()

sor_tmp = temp.sort_index()


day = df['Date_month_year'].dt.strftime('%m-%d-%Y')

cnt =day.value_counts();

sor_cnt = cnt.sort_index()


#providng the trend chart for the number of complaints at monthly complaints
```

```
month = ["Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"]
color = 'tab:red'
fig, ax1 = plt.subplots()
plt.title('Monthly Compliants')
ax1.set_xlabel('Months')
ax1.set_ylabel('Monthly Compliant data')
ax1.plot(month, sor_tmp,color=color)
#providng the trend chart for the number of complaint daily granularity levels
color = "tab:blue"
fig, ax2 = plt.subplots()
plt.title('Date wise complaints ')
ax2.set_xlabel('Dates')
ax2.set_ylabel('Daily Compliant data')
ax2.plot(sor_cnt.index.values, sor_cnt,color=color)

#extracting the count of words from the customer complaints
int_cnt = df['Customer Complaint'].str.extract('(Internet)',expand=False).str.strip().count()
int_cnt

nw_cnt =df['Customer Complaint'].str.extract('(Network)',expand=False).str.strip().count()
nw_cnt

bl_cnt =df['Customer Complaint'].str.extract('(Bill)',expand=False).str.strip().count()
bl_cnt
```

```
ser_cnt = df['Customer Complaint'].str.extract('(Service)',expand=False).str.strip().count()

ser_cnt


oth_cnt = len(df.index) - (int_cnt + nw_cnt + bl_cnt + ser_cnt)

oth_cnt


#plotting the bar graph to see the max complaints received

df3 = pd.DataFrame({"Data":["Internet","Network","Billing","Servicing","Others"],

                    "Count of tickets from diff. modes":[int_cnt, nw_cnt, bl_cnt, ser_cnt, oth_cnt]})

df3

df3_max = df3.max()

df3_max


df3.plot.bar(x='Data', y='Count of tickets from diff. modes')


#replacing pending -> Open and Resolved -> Closed

stat_opn = df['Status'].replace("Pending","Open")

stat_cls = df['Status'].replace("Solved","Closed")

stat_opn

stat_cls

df['Open'] = stat_opn

df['Close'] = stat_cls



opn_cnt = df.loc[df['Open']=='Open']
```

```
opn_cnt


cls_cnt = df.loc[df['Close']=='Closed']

cls_cnt


#creating dataframes from open and close status

df1 = df[df['Open'] == 'Open']

print (df1)


df2 = df[df['Close'] == 'Closed']

print (df2)


#taking counts for observing the max number of complaints received

stat_count = df['State'].value_counts()

opn_stat_count = df1['State'].value_counts()

opn_max_count = opn_stat_count.index.values

opn_max_count


max_stat_count = stat_count[0]

max_stat_name = stat_count.index.values

max_stat_name[0]

print("State ",max_stat_name[0]," has maximum complaints with the count ",max_stat_count)


unres = df1['State'].value_counts()

total_unres = unres.sum()
```

```
total_unres



#plotting pie chart for highest number of unresolved tickets observation in the state

fig = plt.figure(figsize=(16,12))

plt.pie(tot_opn,labels = opn_max_count,autopct='%1.1f%%')

plt.title('Unresolved Tickets')

plt.axis('equal')

plt.show()


#taking count of states with status still open and close

tot_opn = df1['State'].value_counts()

tot_cls = df2['State'].value_counts()

tot_opn

tot_cls

total_opn_pd = pd.DataFrame(tot_opn)

total_cls_pd = pd.DataFrame(tot_cls)

total_opn_pd.rename(columns = {'State':'Open Count'}, inplace = True)

total_cls_pd.rename(columns = {'State':'Close Count'}, inplace = True)



#removing duplicate state values

sta_dup = pd.DataFrame(df['State'].drop_duplicates())

sort_dup = sta_dup.sort_values('State')

sort_list =sort_dup.values.tolist()
```

```
#merging open and close ticket values and replacing NAN values with '0'

df_mer = pd.merge(total_cls_pd,total_opn_pd ,left_index=True,right_index=True, how="outer")

df_tot_mer = df_mer.fillna(0)

df_tot_mer


#plotting stacked bar chart for ticket status open and close for states

N=np.arange(43);

width =0.35


fig = plt.figure(figsize=(12,6))

p1 = plt.bar(N,df_tot_mer['Open Count'],width)

p2 = plt.bar(N,df_tot_mer['Close Count'],width, bottom=df_tot_mer['Open Count'])


plt.title('State wise open and close details in stacked graph')

plt.ylabel('Tickets Count')

plt.xlabel('States')

plt.xticks(range(len(df_tot_mer.index)), df_tot_mer.index, rotation=90)

plt.yticks(np.arange(0,500,50))

plt.show()


#taking the count of complaints received via Internet and Customer Care

internet_call = df2['Received Via'].str.extract('(Internet)',expand=False).str.strip().count()

customer_call = df2['Received Via'].str.extract('(Customer Care Call)',expand=False).str.strip().count()

internet_call
```

customer_call

#plotting a pie chart for calculating percentage of complaints received via Internet and Customer Care

plt.pie([internet_call,customer_call],labels = ['internet_call','customer_call'],autopct='%1.1f%%')

plt.title('complaints received via Internet and Customer Care')

plt.axis('equal')

plt.show()