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 Zipcode: 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.

```
import numpy as np
import pandas as pd
```

```
df = pd.read_csv("/content/drive/MyDrive/Data/simplilearn/Comcast_telecom_complaints_data.
```

```
df.head(2)
```

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
0	250635	Comcast Cable Internet Speeds	22-04-15	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2224 entries, 0 to 2223
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   Ticket #                     2224 non-null   object
 1   Customer Complaint           2224 non-null   object
 2   Date                         2224 non-null   object
 3   Date_month_year              2224 non-null   object
 4   Time                         2224 non-null   object
 5   Received Via                 2224 non-null   object
 6   City                         2224 non-null   object
 7   State                        2224 non-null   object
 8   Zip code                     2224 non-null   int64 
 9   Status                       2224 non-null   object
 10  Filing on Behalf of Someone  2224 non-null   object
dtypes: int64(1), object(10)
memory usage: 191.2+ KB
```

Data types of int and object. No null values.

```
df.shape
```

```
(2224, 11)
```

```
df.isna().sum()
```

```
Ticket #                     0
Customer Complaint           0
Date                         0
Date_month_year              0
Time                         0
Received Via                 0
City                         0
State                        0
Zip code                     0
Status                       0
Filing on Behalf of Someone  0
dtype: int64
```

Observation 1: No null values

```
df.describe(include='object')
```

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	St
count	2224	2224	2224	2224	2224	2224	2224	2
unique	2224	1841	91	91	2190	2	928	
top	323897	Comcast	24-06-15	24-Jun-15	9:55:33 PM	Customer Care Call	Atlanta	Geo

```
# check duplicates
df[df.duplicated()].shape
```

```
(0, 11)
```

No duplicates

convert date types

```
# convert dates to date time object
df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%y')
df.head(2)
```

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State
0	250635	Comcast Cable Internet Speeds	2015-04-22	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland

```
from datetime import datetime

#extracting year, month and day from date
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
df['Month_Name'] = df['Month'].apply(lambda x: datetime.strptime(str(x), "%m").strftime("%
```

```
df['Day'] = df['Date'].dt.day

df[['Date', 'Date_month_year', 'Month', 'Month_Name']].head(2)
```

	Date	Date_month_year	Month	Month_Name
0	2015-04-22	22-Apr-15	4	Apr
1	2015-08-04	04-Aug-15	8	Aug

```
# drop redundant date columns
#df.drop(['Date', 'Date_month_year'], axis=1, inplace=True)


df['Date'].min()
```

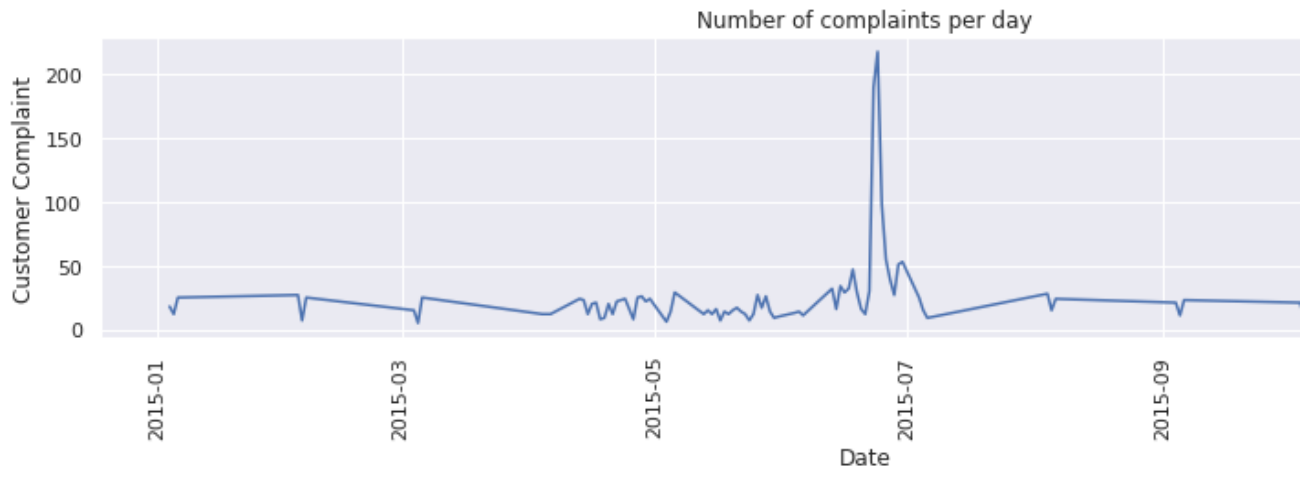
```
Timestamp('2015-01-04 00:00:00')
```

analysis

Provide the trend chart for the number of complaints at monthly and daily granularity levels.

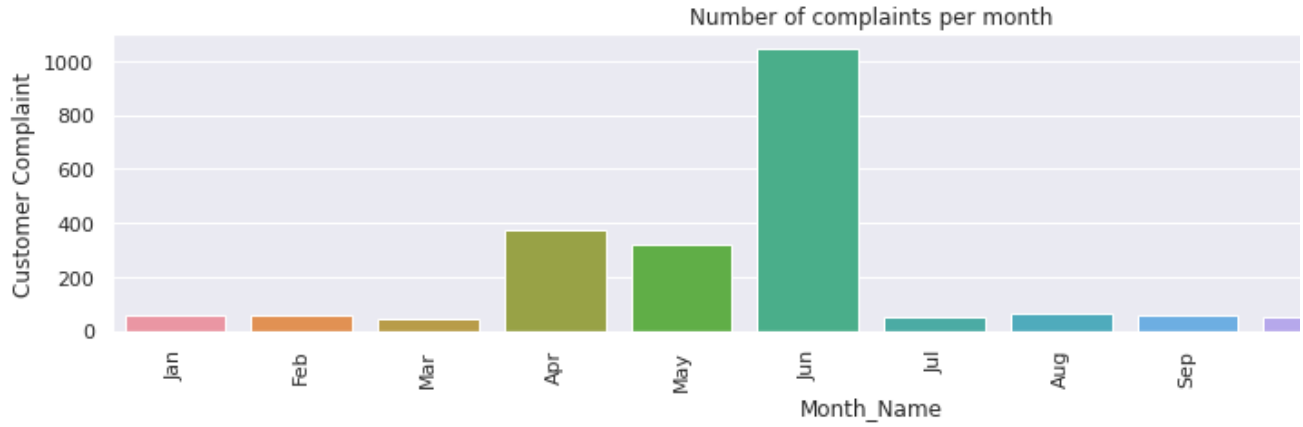
```
import seaborn as sns

sns.set(rc={'figure.figsize':(15, 3)})
ax = sns.lineplot(data=df.groupby(['Date'])['Customer Complaint'].count())
ax.set_title('Number of complaints per day')
ax.tick_params(axis='x', rotation=90)
```




```
sns.set(rc={'figure.figsize':(15, 3)})
data = df[['Customer Complaint', 'Year', 'Month_Name', 'Month']].groupby(['Year', 'Month',
```

```
ax = sns.barplot(data=data, x='Month_Name', y='Customer Complaint')
ax.set_title('Number of complaints per month')
ax.tick_params(axis='x', rotation=90)
```

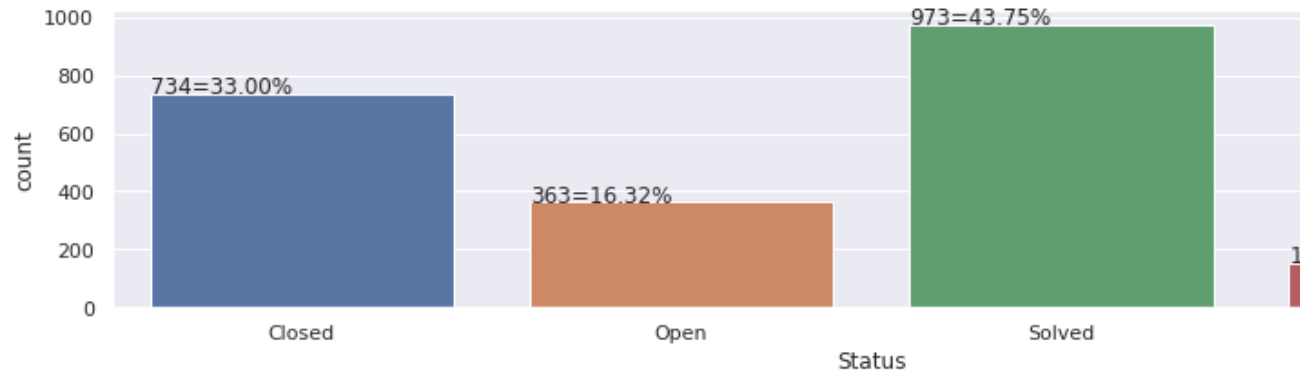


Observation - Majority of complaints are in June

```
df.groupby(['Status'])['Customer Complaint'].count()
```

```
Status
Closed     734
Open       363
Pending    154
Solved     973
Name: Customer Complaint, dtype: int64
```

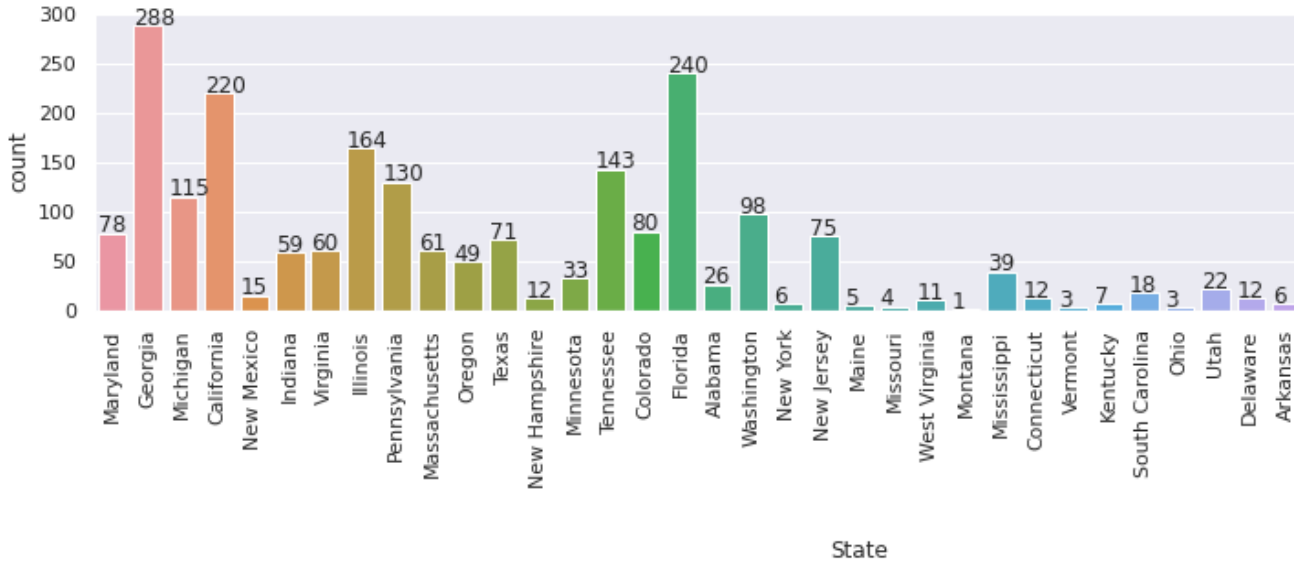
```
sns.set(rc={'figure.figsize':(15, 3)})
ax = sns.countplot(x='Status', data=df)
for p in ax.patches:
  ax.annotate('{}={:.2f}%'.format(p.get_height(), p.get_height()/df.shape[0]*100), (p.get_
```



Which state has the maximum complaints

```
sns.set(rc={'figure.figsize':(15, 3)})
```

```
ax = sns.countplot(data=df, x='State')
for p in ax.patches:
  ax.annotate(f'{p.get_height()}', (p.get_x(), p.get_height()+1))
ax.tick_params(axis='x', rotation=90)
```



Georgia has the maximum number of complaints of 288

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

```
df['Status'].unique()
```

```
array(['Closed', 'Open', 'Solved', 'Pending'], dtype=object)
```

```
# set the Closed and the Solved ones as Resolved and others as Open
data = df[['State', 'Status', 'Customer Complaint']]
data['Status'] = data['Status'].apply(lambda x: np.where(x=='Closed' or x=='Solved', 'Clos
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWarnir
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  This is separate from the ipykernel package so we can avoid doing imports until
```

```
data['Status'].unique()
```

```
array(['Closed', 'Open'], dtype=object)
```

```
# count the number of resolved and open complaints state wise
```

```
data_stat_count = data[['State', 'Status', 'Customer Complaint']].groupby(['State', 'Statu
data_stat_count.head(2)
```

		Customer Complaint
State	**Status**	
Alabama	**Closed**	17
	Open	9

```
# calculate the total complaints state wise
data_total_comp = data[['State', 'Customer Complaint']].groupby(['State']).count()
data_total_comp.sort_values(by=['Customer Complaint'], ascending=False).head(2)
```

	Customer Complaint
State	
Georgia	288
Florida	240

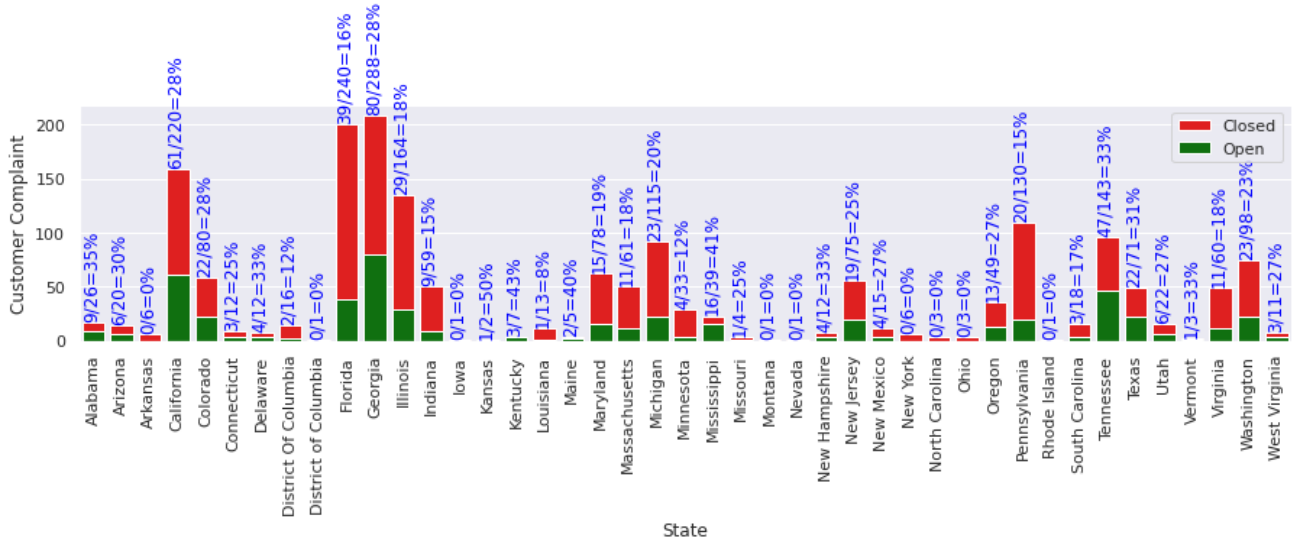
```
# calculate the percentage state wise
data_stat_count['Percent'] = data_stat_count['Customer Complaint']/data_total_comp['Custom
data_stat_count['Total_Count'] = data_stat_count['Customer Complaint']+data_total_comp['Cu
data_stat_count = data_stat_count.reset_index()
data_stat_count.head(4)
```

	State	Status	Customer Complaint	Percent	Total_Count
0	Alabama	Closed	17	0.653846	26
1	Alabama	Open	9	0.346154	26
2	Arizona	Closed	14	0.700000	20
3	Arizona	Open	6	0.300000	20

```
data_stat_closed = data_stat_count.reset_index()
data_stat_closed = data_stat_closed[data_stat_closed['Status']=='Closed']
data_stat_open = data_stat_closed['State'].copy().to_frame()
data_stat_open['Customer Complaint'] = 0
data_stat_open['Percent'] = 0
temp_data = data_stat_count.reset_index()
temp_data = temp_data[temp_data['Status']=='Open']
for state in data_stat_open['State']:
  if len(temp_data.loc[temp_data['State']==state, 'Customer Complaint'].values) != 0:
    data_stat_open.loc[data_stat_open['State']==state, 'Customer Complaint'] = temp_data.l
    data_stat_open.loc[data_stat_open['State']==state, 'Percent'] = round(temp_data.loc[te
data_stat_open['Status'] = 'Open'
data_stat_open['Total_Count'] = data_stat_closed['Total_Count'].copy()
```

```
sns.set(rc={'figure.figsize':(15, 3)})
ax = sns.barplot(data=data_stat_closed, x='State', y='Customer Complaint', color='Red')
ax.patches[0].set_label('Closed')
ax = sns.barplot(data=data_stat_open, x='State', y='Customer Complaint', color='Green')
for p, val, tot, per in zip(ax.patches, data_stat_open['Customer Complaint'].values,
                            data_stat_closed['Customer Complaint'].values, data_stat_open['Perc
  ax.annotate(f'{val}/{val+tot}={per}%', (p.get_x(), p.get_height()+5), color='Blue', rota
ax.patches[-1].set_label('Open')
ax.tick_params(axis='x', rotation=90)
ax.legend()
```

```
<matplotlib.legend.Legend at 0x7f27a11b39d0>
```



As can be seen from graph, the maximum number of total complaints received is by Georgia(total 0f 288 and open count as 80).

Kansas has the highest number of unresolved complaints percentage (50%).

Provide a table with the frequency of complaint types.

```
df['Customer Complaint'].unique()
```

```
array(['Comcast Cable Internet Speeds',
       'Payment disappear - service got disconnected',
       'Speed and Service', ..., 'complaint about comcast',
       'Extremely unsatisfied Comcast customer',
       'Comcast, Ypsilanti MI Internet Speed'], dtype=object)
```

Please tell how to approach this