Comcast_Solution_

March 26, 2021

1 *Comcast Telecom Consumer Complaints*

[1]:
```
# Import the required Libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df = pd.read_csv("Comcast_telecom_complaints_data.csv")  # Importing the data
↪into Python Environment.
```

[3]:
```
df.head()
```

[3]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State	Zip code	Status	Filing on Behalf of Someone
0	250635	Comcast Cable Internet Speeds	22-04-15	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland	21009	Closed	No
1	223441	Payment disappear - service got disconnected	04-08-15	04-Aug-15	10:22:56 AM	Internet	Acworth	Georgia	30102	Closed	No
2	242732	Speed and Service	18-04-15	18-Apr-15	9:55:47 AM	Internet	Acworth	Georgia	30101	Closed	Yes
3	277946	Comcast Imposed a New Usage Cap of 300GB that …	05-07-15	05-Jul-15	11:59:35 AM	Internet	Acworth	Georgia	30101	Open	Yes
4	307175	Comcast not working and no service to boot	26-05-15	26-May-15	1:25:26 PM	Internet	Acworth	Georgia	30101	Solved	No

[4]:
```
df.isna().sum() # To find null values
```

[4]:
```
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
```

[5]:
```
df.describe() # To view statistical details of the dataframe.
```

[5]:
```
           Zip code
count   2224.000000
mean   47994.393435
std    28885.279427
min     1075.000000
25%    30056.500000
50%    37211.000000
75%    77058.750000
max    99223.000000
```

2 *Extracting of Monthly and Daily Count Tickets*

[6]:
```
df_dmy = df.groupby("Date_month_year")
```

[7]:
```
df_dmy.size()
```

[7]:
```
Date_month_year
04-Apr-15    12
04-Aug-15    28
04-Dec-15    15
04-Feb-15    27
04-Jan-15    18
             ..
29-May-15    14
30-Apr-15    24
30-Jun-15    53
30-May-15     9
31-May-15    10
Length: 91, dtype: int64
```

[8]:
```
df.sort_values(["Date_month_year"], axis=0,ascending=True, inplace=True,␣
↪kind='quicksort', na_position='last')
```

[9]:
```
df.head()
```

[9]:
```
     Ticket #                                    Customer Complaint      Date  \
1416   218108  Comcast Business Phone/Internet Contract Disag…  04-04-15
1483   217985         bait and switch services for monetary gain  04-04-15
584    217999                        Misleading information given  04-04-15
561    218043                                   comcast services  04-04-15
1892   218168  Multiple Unauthorized and Unwarranted Credit C…  04-04-15

     Date_month_year        Time        Received Via        City       State  \
1416       04-Apr-15  6:39:55 PM            Internet      Newnan     Georgia
1483       04-Apr-15  4:07:36 PM            Internet      Orcutt  California
584        04-Apr-15  4:21:46 PM            Internet  Des Moines  Washington
561        04-Apr-15  5:32:05 PM            Internet      Denver    Colorado
1892       04-Apr-15  8:10:35 PM  Customer Care Call   Shoreview   Minnesota

      Zip code  Status Filing on Behalf of Someone
1416     30265  Closed                          No
1483     93455  Closed                          No
584      98148  Closed                         Yes
561      80227  Closed                          No
1892     55126  Closed                          No
```

[10]:
```
df['month'] = pd.DatetimeIndex(df['Date_month_year']).month
```

[11]:
```
df.head()
```

[11]:
```
     Ticket #                                    Customer Complaint      Date  \
1416   218108  Comcast Business Phone/Internet Contract Disag…  04-04-15
1483   217985         bait and switch services for monetary gain  04-04-15
584    217999                        Misleading information given  04-04-15
561    218043                                   comcast services  04-04-15
1892   218168  Multiple Unauthorized and Unwarranted Credit C…  04-04-15

     Date_month_year        Time        Received Via        City       State  \
1416       04-Apr-15  6:39:55 PM            Internet      Newnan     Georgia
1483       04-Apr-15  4:07:36 PM            Internet      Orcutt  California
584        04-Apr-15  4:21:46 PM            Internet  Des Moines  Washington
561        04-Apr-15  5:32:05 PM            Internet      Denver    Colorado
1892       04-Apr-15  8:10:35 PM  Customer Care Call   Shoreview   Minnesota

      Zip code  Status Filing on Behalf of Someone  month
1416     30265  Closed                          No      4
1483     93455  Closed                          No      4
```

```
584      98148  Closed                          Yes      4
561      80227  Closed                           No      4
1892     55126  Closed                           No      4
```

[12]:
```
df_month = df.groupby('month')
df_month_size = df_month.size()
df_month_size
```

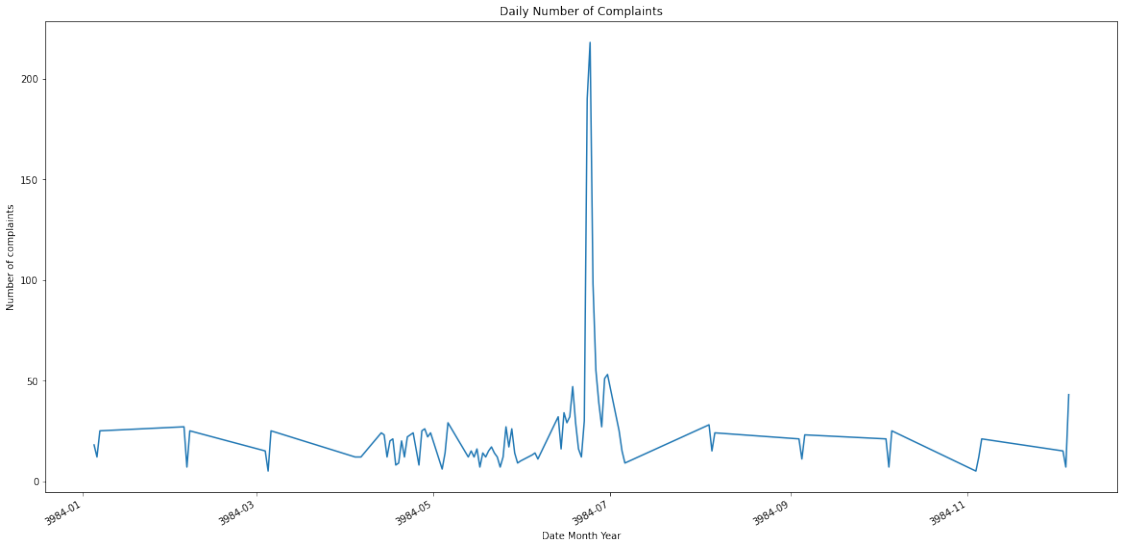
[12]:
```
month
1       55
2       59
3       45
4      375
5      317
6     1046
7       49
8       67
9       55
10      53
11      38
12      65
dtype: int64
```

[13]:
```
df["Date_month_year"] = pd.to_datetime(df["Date_month_year"])
```

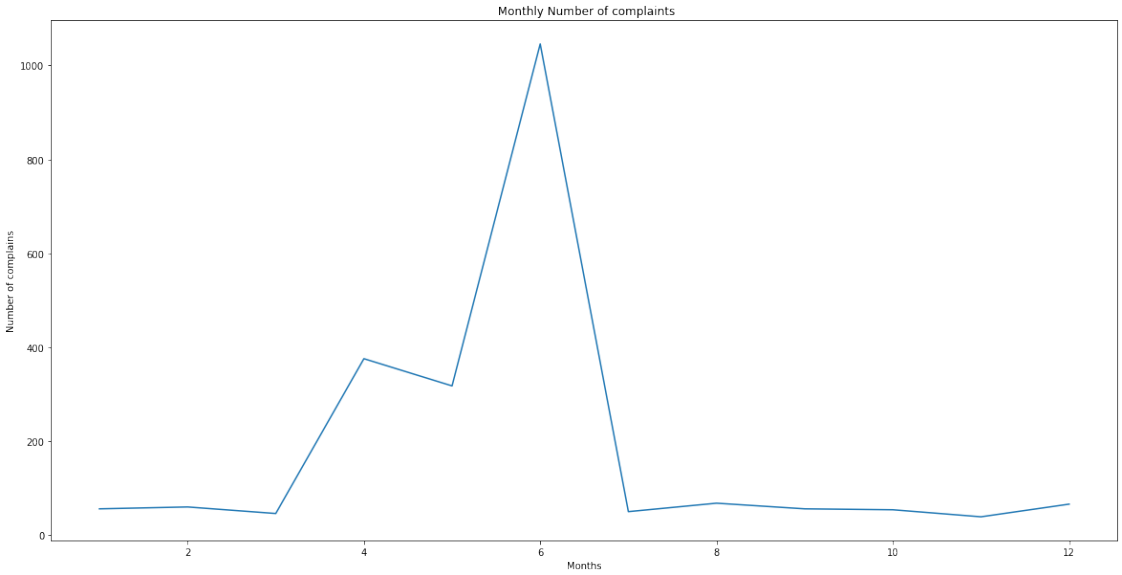
3 *Trend chart for the number of complaints at monthly and daily granularity levels.*

[14]:
```
plt.figure(figsize=(20,10))
df['Date_month_year'].value_counts().plot()
plt.title('Daily Number of Complaints')
plt.xlabel('Date Month Year')
plt.ylabel('Number of complaints')
plt.show()
```

```
/usr/local/lib/python3.7/site-
packages/pandas/plotting/_matplotlib/converter.py:256:
MatplotlibDeprecationWarning:
The epoch2num function was deprecated in Matplotlib 3.3 and will be removed two
minor releases later.
  base = dates.epoch2num(dt.asi8 / 1.0e9)
```



[15]:
```
plt.figure(figsize=(20,10))
df_month_size.plot()
plt.title(' Monthly Number of complaints')
plt.xlabel('Months')
plt.ylabel('Number of complains')
plt.show()
```

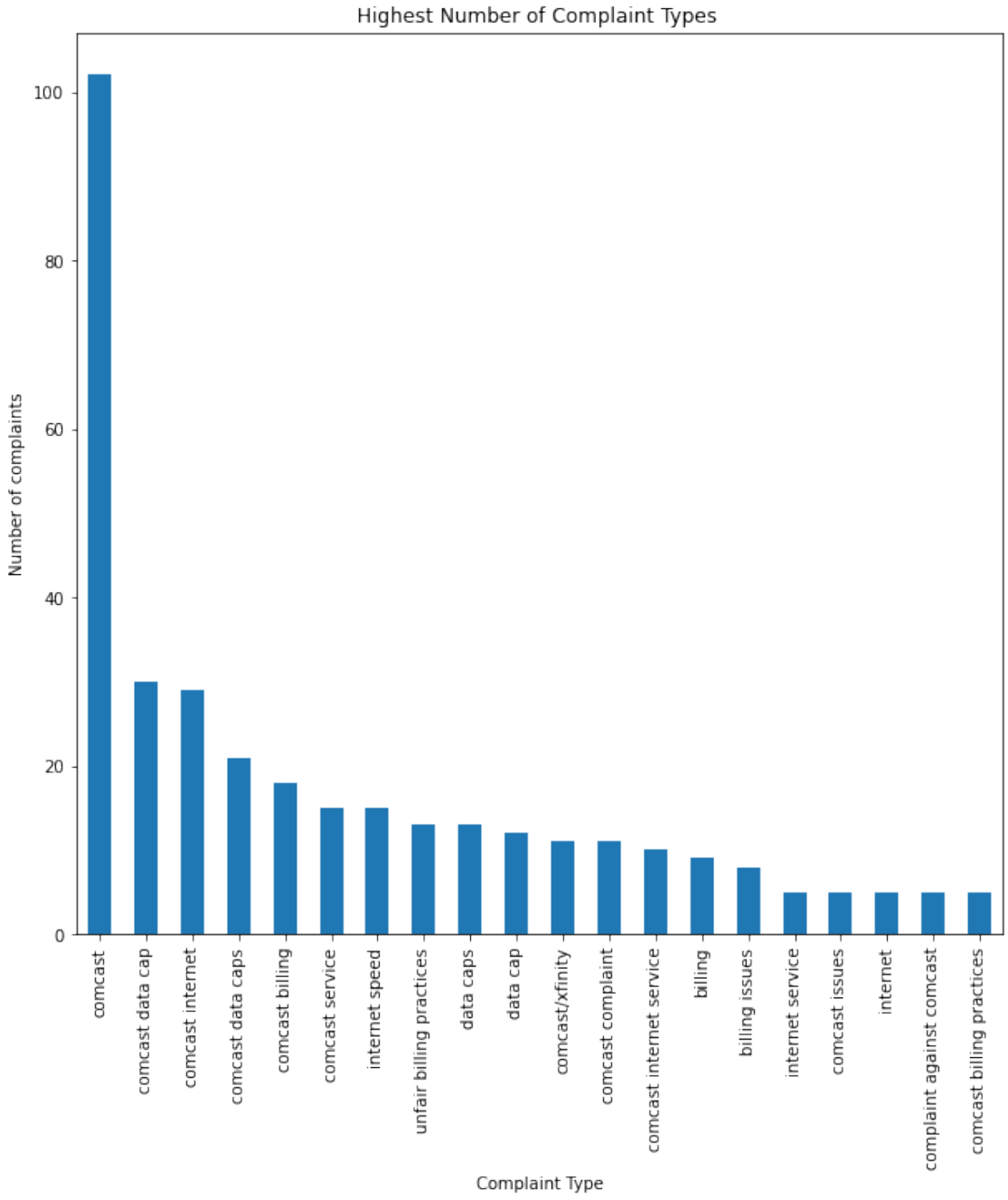


4 *Table with the frequency of complaint types.*

[16]:
```
complaint_type = df['Customer Complaint'].str.lower().value_counts()
complaint_type.head(n=20)
```

[16]:
```
comcast                      102
comcast data cap              30
comcast internet              29
comcast data caps             21
comcast billing               18
comcast service               15
internet speed                15
unfair billing practices      13
data caps                     13
data cap                      12
comcast/xfinity               11
comcast complaint             11
comcast internet service      10
billing                        9
billing issues                 8
internet service               5
comcast issues                 5
internet                       5
complaint against comcast      5
comcast billing practices      5
Name: Customer Complaint, dtype: int64
```

[17]:
```
complaint_type_head = complaint_type.head(n=20)
plt.figure(figsize=(10,10))
complaint_type_head.plot.bar()
plt.title('Highest Number of Complaint Types ')
plt.xlabel('Complaint Type')
plt.ylabel('Number of complaints')
plt.show()
```



5 *A new categorical variable with value as Open and Closed.*

[18]:
```
df.groupby('Status').size()
```

[18]:
```
Status
Closed     734
Open       363
Pending    154
Solved     973
dtype: int64
```

[19]:
```
df['New Status'] = df['Status']
```

[20]:
```
# Open & Pending is to be categorized as Open and Closed & Solved is to be␣
 ↪categorized as Closed.
df['New Status'].replace(('Pending', 'Solved'), ('Open', 'Closed'),␣
 ↪inplace=True)
df.groupby('New Status').size()
```

[20]:
```
New Status
Closed    1707
Open       517
dtype: int64
```

[21]:
```
df.groupby('State').size()
```

[21]:
```
State
Alabama                 26
Arizona                 20
Arkansas                 6
California             220
Colorado                80
Connecticut             12
Delaware                12
District Of Columbia    16
District of Columbia     1
Florida                240
Georgia                288
Illinois               164
Indiana                 59
Iowa                     1
Kansas                   2
Kentucky                 7
Louisiana               13
Maine                    5
Maryland                78
Massachusetts           61
Michigan               115
Minnesota               33
Mississippi             39
Missouri                 4
```

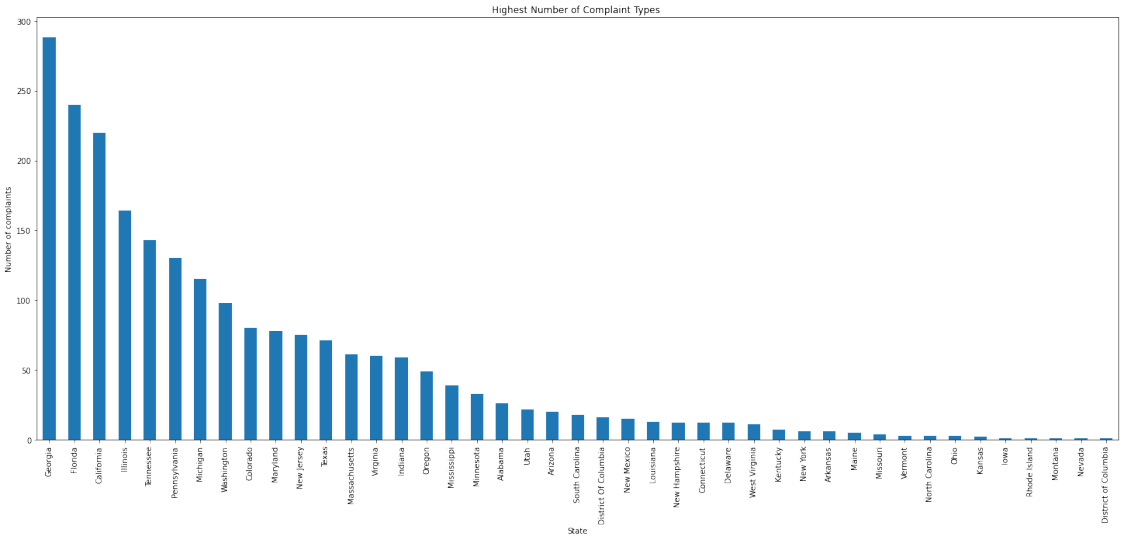
```
Montana                   1
Nevada                    1
New Hampshire            12
New Jersey               75
New Mexico               15
New York                  6
North Carolina            3
Ohio                      3
Oregon                   49
Pennsylvania            130
Rhode Island              1
South Carolina           18
Tennessee               143
Texas                    71
Utah                     22
Vermont                   3
Virginia                 60
Washington               98
West Virginia            11
dtype: int64
```

[22]:
```
df['State'].value_counts()
```

[22]:
```
Georgia                 288
Florida                 240
California              220
Illinois                164
Tennessee               143
Pennsylvania            130
Michigan                115
Washington               98
Colorado                 80
Maryland                 78
New Jersey               75
Texas                    71
Massachusetts            61
Virginia                 60
Indiana                  59
Oregon                   49
Mississippi              39
Minnesota                33
Alabama                  26
Utah                     22
Arizona                  20
South Carolina           18
District Of Columbia     16
New Mexico               15
```

```
Louisiana                13
New Hampshire            12
Connecticut              12
Delaware                 12
West Virginia            11
Kentucky                  7
New York                  6
Arkansas                  6
Maine                     5
Missouri                  4
Vermont                   3
North Carolina            3
Ohio                      3
Kansas                    2
Iowa                      1
Rhode Island              1
Montana                   1
Nevada                    1
District of Columbia      1
Name: State, dtype: int64
```

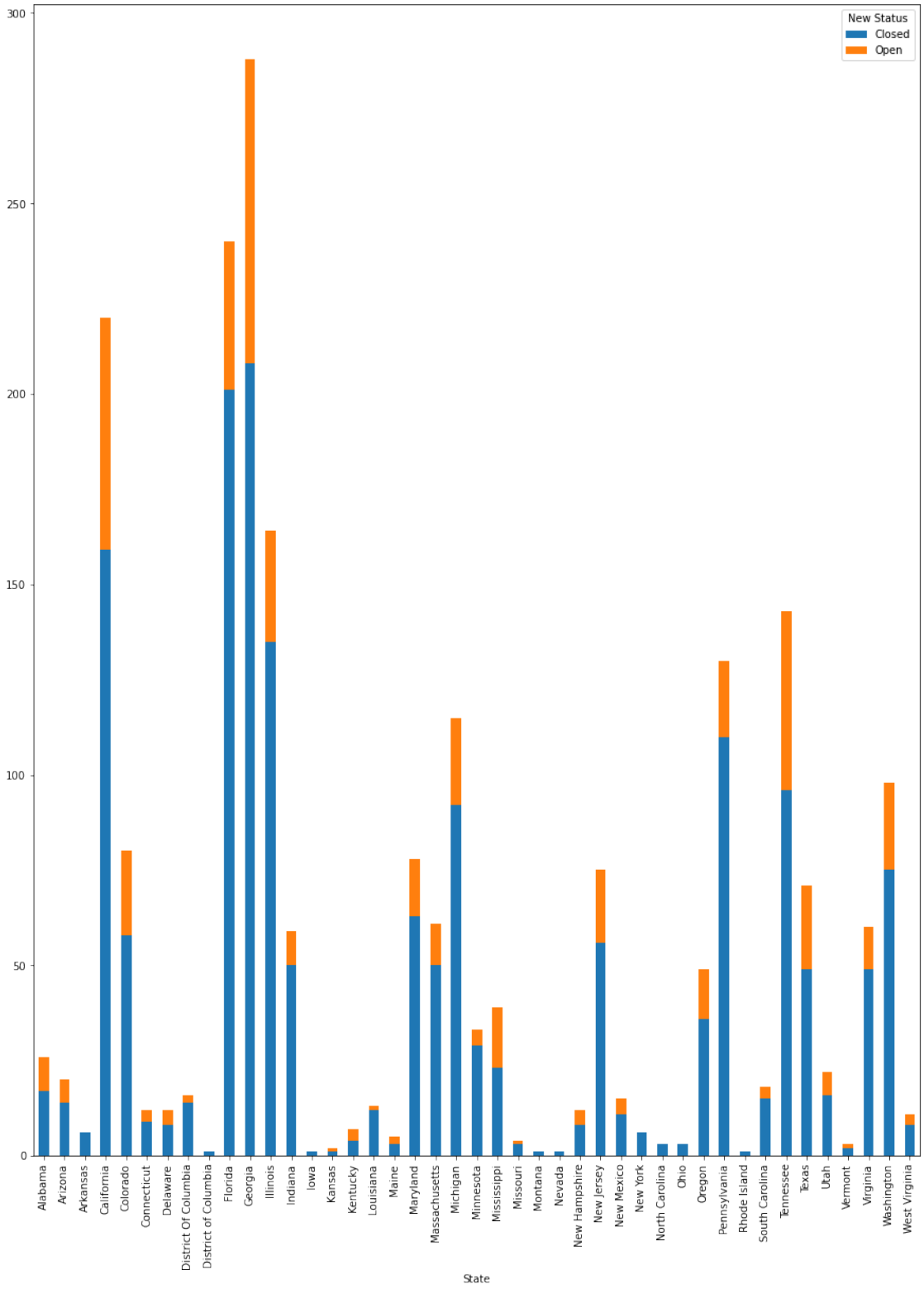
[23]:
```
plt.figure(figsize=(25,10))
df['State'].value_counts().plot.bar()
plt.title('Highest Number of Complaint Types ')
plt.xlabel('State')
plt.ylabel('Number of complaints')
plt.show()
```



6 *State wise status of complaints in a stacked bar chart.*

[24]:
```
print(df.groupby(['State', 'New Status']).size().unstack().plot(kind='bar',␣
↪figsize = (15,20), stacked=True))
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```


New Status
Closed
Open
300
250
200
150
100
50
0
Alabama
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
District Of Columbia
District of Columbia
Florida
Georgia
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
Ohio
Oregon
Pennsylvania
Rhode Island
South Carolina
Tennessee
Texas
Utah
Vermont
Virginia
Washington
West Virginia
State

[25]:
```
complaint_by_state = df.groupby(['State','New Status']).size().unstack().
↪fillna(0)
complaint_by_state
```

[25]:

New Status	Closed	Open
State		
Alabama	17.0	9.0
Arizona	14.0	6.0
Arkansas	6.0	0.0
California	159.0	61.0
Colorado	58.0	22.0
Connecticut	9.0	3.0
Delaware	8.0	4.0
District Of Columbia	14.0	2.0
District of Columbia	1.0	0.0
Florida	201.0	39.0
Georgia	208.0	80.0
Illinois	135.0	29.0
Indiana	50.0	9.0
Iowa	1.0	0.0
Kansas	1.0	1.0
Kentucky	4.0	3.0
Louisiana	12.0	1.0
Maine	3.0	2.0
Maryland	63.0	15.0
Massachusetts	50.0	11.0
Michigan	92.0	23.0
Minnesota	29.0	4.0
Mississippi	23.0	16.0
Missouri	3.0	1.0
Montana	1.0	0.0
Nevada	1.0	0.0
New Hampshire	8.0	4.0
New Jersey	56.0	19.0
New Mexico	11.0	4.0
New York	6.0	0.0
North Carolina	3.0	0.0
Ohio	3.0	0.0
Oregon	36.0	13.0
Pennsylvania	110.0	20.0
Rhode Island	1.0	0.0
South Carolina	15.0	3.0
Tennessee	96.0	47.0
Texas	49.0	22.0
Utah	16.0	6.0
Vermont	2.0	1.0
Virginia	49.0	11.0

```
Washington                75.0  23.0
West Virginia              8.0   3.0
```

7 Georgia state has the maximum complaints.

8 *To find the percentage of resolved and un-resolved Complaints.*

[26]:
```
complaint_by_state.iloc[0,1]

closed_cont = []
opened_cont = []

for i in range(0, len(complaint_by_state)):
     closed_cont.append(complaint_by_state.iloc[i,0])

for i in range(0, len(complaint_by_state)):
     opened_cont.append(complaint_by_state.iloc[i,1])
```

[27]:
```
closed_complaints = np.array(closed_cont)
opened_complaints = np.array(opened_cont)
```

[28]:
```
percentage_opened = (opened_complaints/(opened_complaints+closed_complaints)) *␣
 ↪100
percentage_closed = (closed_complaints/(opened_complaints+closed_complaints)) *␣
 ↪100
percentage_opened
```

[28]:
```
array([34.61538462, 30.        ,  0.        , 27.72727273, 27.5       ,
       25.        , 33.33333333, 12.5       ,  0.        , 16.25      ,
       27.77777778, 17.68292683, 15.25423729,  0.        , 50.        ,
       42.85714286,  7.69230769, 40.        , 19.23076923, 18.03278689,
       20.        , 12.12121212, 41.02564103, 25.        ,  0.        ,
        0.        , 33.33333333, 25.33333333, 26.66666667,  0.        ,
        0.        ,  0.        , 26.53061224, 15.38461538,  0.        ,
       16.66666667, 32.86713287, 30.98591549, 27.27272727, 33.33333333,
       18.33333333, 23.46938776, 27.27272727])
```

[29]:
```
dictionary = complaint_by_state.to_dict() # Converting to dictionary
```

[30]:
```
c = {}  # Closed
o = {}  # Opened

for k, v in dictionary.items(): # k represents keys and v represents values in␣
 ↪dictionary
```

```
    if k == 'Closed':
        c.update(v)
    else:
        o.update(v)
```

[31]:
```
# Finding a list with only states names
states = []
for keys in c:
    states.append(keys)
```

[32]:
```
print("Length of states is {} and length of percentage open list is {}".␣
↪format(len(states), len(percentage_opened)))
```

```
Length of states is 43 and length of percentage open list is 43
```

[33]:
```
# Function to find the index of the maximum percentage value
def find_max_index(percentage_list):

    result = np.where(percentage_list == np.amax(percentage_list))
    return result[0]
```

[34]:
```
# Converting items in a list to a single integer
def convert(list_values):
    #convert to string first
    s = [str(i) for i in list_values]
    # Join list items using join()
    res = int("".join(s))
    return res
```

[35]:
```
opened_max_index = convert(find_max_index(percentage_opened))
# Closed_max_index = find_max_index(percentage_closed) # This is not needed␣
↪because these are already resolved cases.
                                                       # Finding the max number␣
↪will give 8 total states.
                                                       # They have ONLY closed␣
↪cases so the percentage is 100%.
```

[36]:
```
print('Highest Unresolved Percentage is',percentage_opened.max(),'% from',␣
↪states[opened_max_index])
```

```
Highest Unresolved Percentage is 50.0 % from Kansas
```

9 Kansas has the highest unresolved percentage.

10 *The percentage of complaints resolved till date, which were received through the Internet and customer care calls.*

[37]:
```
comcast_received = df.groupby(['Received Via', 'New Status']).size()
print(comcast_received)
```

```
Received Via        New Status
Customer Care Call  Closed        864
                    Open          255
Internet            Closed        843
                    Open          262
dtype: int64
```

[38]:
```
closed = []
opened = []
for counter, value in enumerate(comcast_received):
    if counter % 2 == 0:
        closed.append(value)
    else:
        opened.append(value)
```

[39]:
```
closed = np.array(closed)
opened = np.array(opened)
```

[40]:
```
percentage_open = (opened/(opened+closed)) * 100
percentage_close = (closed/(opened+closed)) * 100
```

[41]:
```
print(percentage_open)
```

```
[22.78820375 23.71040724]
```

[42]:
```
 # Customer Care Call - 22.78% Open, Internet - 23.71% Open
```

[43]:
```
print(percentage_close)
```

```
[77.21179625 76.28959276]
```

[44]:
```
# Customer Care Call - 77.21% Close, Internet - 76.28% Close
```