

## SCREEN SHOTS -:

localhost 8888/notebooks/MyPythoncode/project.ipynb

jupyter project Last Checkpoint: Last Saturday at 5:50 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

### Task 1: Import data into Python environment

```
In [2]: comcast_tele_consumer=pd.read_csv("C:/Users/91706/OneDrive/Desktop/KhushalPractice/Comcast_telecom_complaints_data.csv")

In [3]: # check whether data imported or not
comcast_tele_consumer.head()
```

```
Out[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

```
In [4]: # Step 1: Check the duplicate columns or Variables with duplicate names and delete such columns
comcast_tele_consumer.shape

Out[4]: (2224, 11)
```

localhost 8888/notebooks/MyPythoncode/project.ipynb

jupyter project Last Checkpoint: Last Saturday at 5:50 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
In [5]: comcast_tele_consumer.columns

Out[5]: Index(['Ticket #', 'Customer Complaint', 'Date', 'Date_month_year', 'Time',
            'Received Via', 'City', 'State', 'Zip code', 'Status',
            'Filing on Behalf of Someone'],
            dtype='object')
```

```
In [6]: # Hence no duplicate names found we will go to step 2
```

```
In [7]: # Step 2: Check for 0 columns or single value
comcast_tele_consumer.describe()
```

```
Out[7]:
```

	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

```
In [8]: # No Zero Columns or single Value Found, so we proceed to step 3
```

```
project - Jupyter Notebook x Assessment x +
localhost:8888/notebooks/MyPythoncode/project.ipynb
File Edit View Insert Cell Kernel Widgets Help
Run Code
In [9]: # Step 3: Missing value Treatment
comcast_tele_consumer.isnull().sum().sort_values(ascending=False)

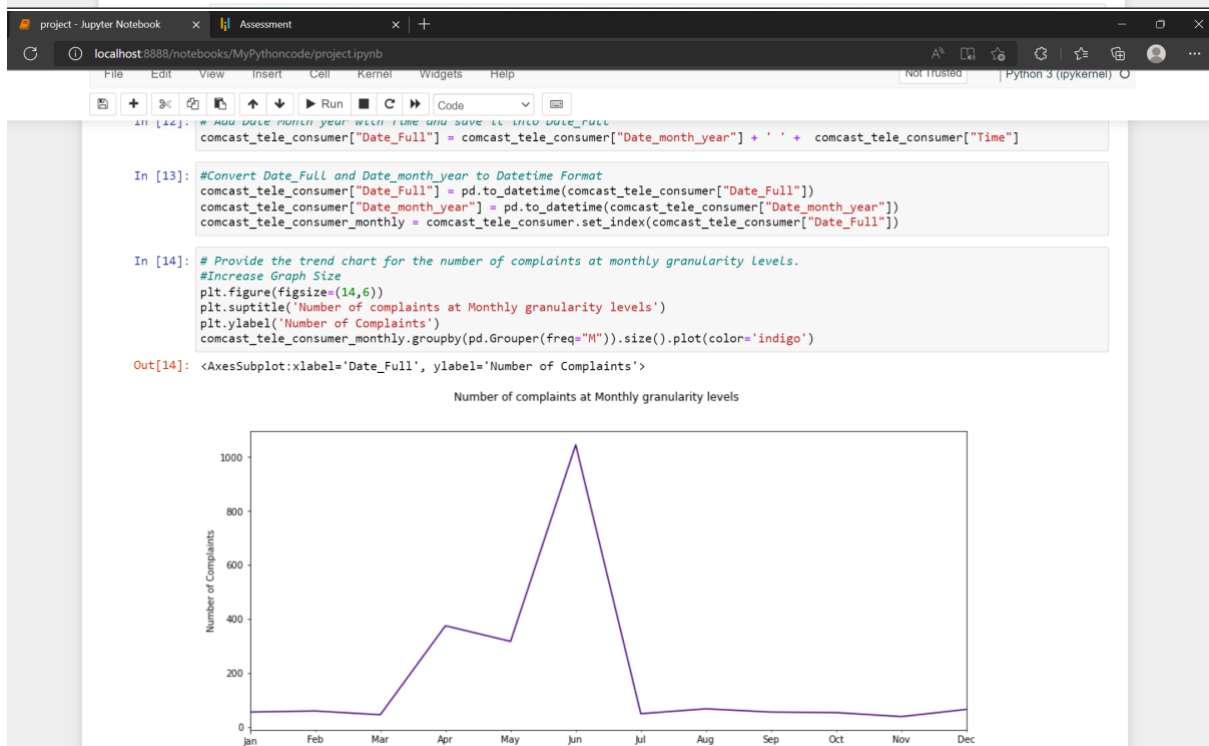
Out[9]: Ticket # 0
Customer Complaint 0
Date 0
Date_month_year 0
Time 0
Received Via 0
City 0
State 0
Zip code 0
Status 0
Filing on Behalf of Someone 0
dtype: int64

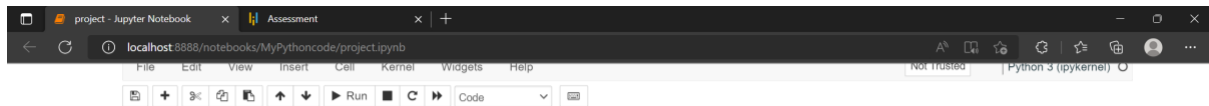
In [10]: # No Missing value found so we proceed to step

Task 2: Provide the trend chart for the number of complaints at monthly and daily granularity levels.

In [11]: comcast_tele_consumer.dtypes

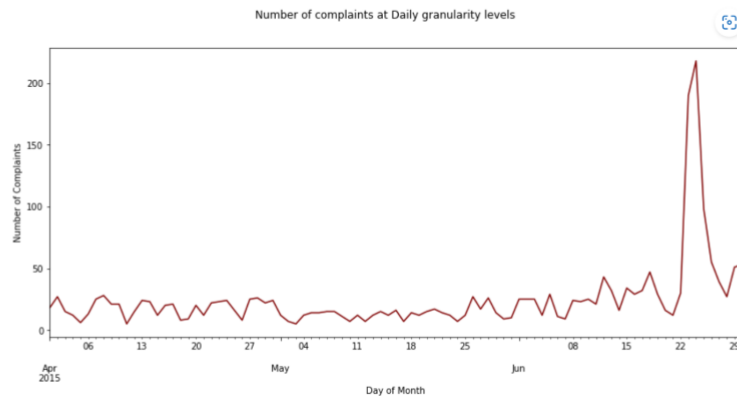
Out[11]: Ticket # object
Customer Complaint object
Date object
Date_month_year object
Time object
Received Via object
City object
State object
Zip code int64
Status object
Filing on Behalf of Someone object
dtype: object
```





```
In [15]: # Provide the trend chart for the number of complaints at daily granularity levels.
comcast_tele_consumer['Day of Month'] = pd.to_datetime(comcast_tele_consumer['Date'])
comcast_tele_consumer_daily = comcast_tele_consumer.set_index(comcast_tele_consumer['Day of Month'])
#Increase Graph Size
plt.figure(figsize=(14,6))
plt.suptitle('Number of complaints at Daily granularity levels')
plt.ylabel('Number of Complaints')
comcast_tele_consumer_daily.groupby(pd.Grouper(freq="D")).size().plot(color='maroon')

Out[15]: <AxesSubplot: xlabel='Day of Month', ylabel='Number of Complaints'>
```



### Task 3: Provide a table with the frequency of complaint types.

```
In [16]: # To get the frequency of complaint types first we have to see all complaint types and check for duplicate, case sensitive
# Incomplete data so that we can make analytics better
comcast_tele_consumer_complaint_type = comcast_tele_consumer["Customer Complaint"].value_counts()
```

```
In [17]: comcast_tele_consumer_complaint_type.head(10)
```

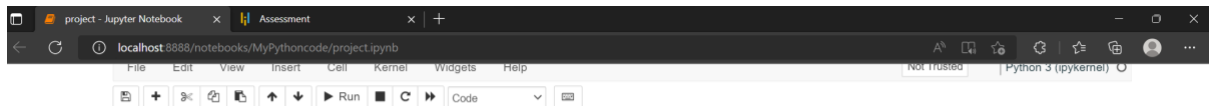
```
Out[17]: Comcast      83
Comcast Internet    18
Comcast Data Cap    17
comcast             13
Comcast Billing      11
Data Caps           11
Comcast Data Caps    11
Unfair Billing Practices  9
Comcast data cap      8
Comcast internet      8
Name: Customer Complaint, dtype: int64
```

```
In [18]: # Better to convert all data into upper case or sentence case so duplicate value will be shorted
comcast_tele_consumer_complaint_type = comcast_tele_consumer["Customer Complaint"].str.upper().value_counts()
```

```
In [19]: # Data is huge so we have showed only top 25 Complaint Types. We can show clearly in this table that COMCAST, COMCAST DATA CAP
# COMCAST INTERNET are the Highest top 3 complaint types
comcast_tele_consumer_complaint_type.head(25)
```

```
Out[19]: 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
```



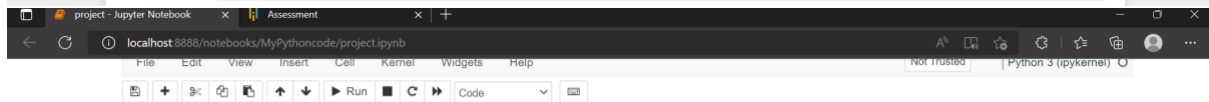


**Task 5: Create a new categorical variable with value as Open and Closed. Open & Pending is to be categorized as Open and Closed & Solved is to be categorized as Closed.**

```
In [20]: # Check how many unique values are under Status Column
comcast_tele_consumer['Status'].unique()

Out[20]: array(['Closed', 'Open', 'Solved', 'Pending'], dtype=object)

In [21]: # Convert as per Instruction (Task 5) into New Column without changing the main data so that we can use the main data
# in Future
comcast_tele_consumer['New_Status'] = ["Open" if Status=="Open" or Status=="Pending" else
                                       "Closed" for Status in comcast_tele_consumer["Status"]]
```



```
Out[24]:
```

New_Status	Closed	Open
Alabama	17	9
Arizona	14	6
Arkansas	6	0
California	159	61
Colorado	58	22
Connecticut	9	3
Delaware	8	4
District Of Columbia	14	2
District of Columbia	1	0
Florida	201	39
Georgia	208	80
Illinois	135	29
Indiana	50	9
Iowa	1	0
Kansas	1	1
Kentucky	4	3
Louisiana	12	1

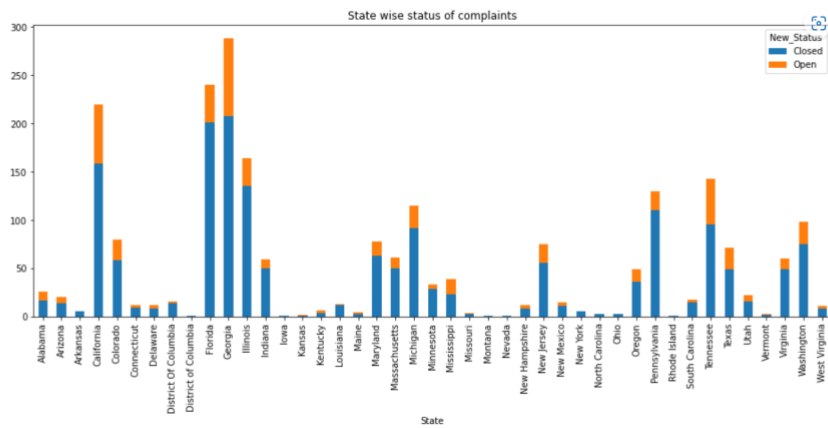
Maine	3	2
Maryland	63	15
Massachusetts	50	11
Michigan	92	23
Minnesota	29	4
Mississippi	23	16
Missouri	3	1
Montana	1	0
Nevada	1	0
New Hampshire	8	4
New Jersey	56	19
New Mexico	11	4
New York	6	0
North Carolina	3	0
Ohio	3	0
Oregon	36	13
Pennsylvania	110	20
Rhode Island	1	0
South Carolina	15	3
Tennessee	96	47
Texas	49	22
Utah	16	6
Vermont	2	1
Virginia	49	11
Washington	75	23
West Virginia	8	3

Maine	3	2
Maryland	63	15
Massachusetts	50	11
Michigan	92	23
Minnesota	29	4
Mississippi	23	16
Missouri	3	1
Montana	1	0
Nevada	1	0
New Hampshire	8	4
New Jersey	56	19
New Mexico	11	4
New York	6	0
North Carolina	3	0
Ohio	3	0
Oregon	36	13
Pennsylvania	110	20
Rhode Island	1	0
South Carolina	15	3
Tennessee	96	47
Texas	49	22
Utah	16	6
Vermont	2	1
Virginia	49	11
Washington	75	23
West Virginia	8	3

### Task 6- Provide state wise status of complaints in a stacked bar chart. Use the categorized variable from

```
In [25]: pd.crosstab(comcast_tele_consumer.State,comcast_tele_consumer.New_Status).plot(kind='bar',figsize=(16,6),
stacked=True,
title='State wise status of complaints')
```

```
Out[25]: <AxesSubplot:title=('center':'State wise status of complaints'), xlabel='State'>
```



project - Jupyter Notebook x Assessment x +

localhost:8888/notebooks/MyPythoncode/project.ipynb

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel)

State

**Task7- Which state has the maximum complaints - Georgia has maximum number of complaints**

**Task 8- Which state has the highest percentage of unresolved complaints**

```
In [26]: comcast_tele_consumer_unresolved_complaints = comcast_tele_consumer[comcast_tele_consumer['New_Status']=='Open']
```

```
In [27]: comcast_tele_consumer_unresolved_complaint_count = comcast_tele_consumer_unresolved_complaints.State.value_counts()
comcast_tele_consumer_unresolved_complaint_count
```

```
Out[27]: Georgia      80
California  61
Tennessee  47
Florida    39
Illinois   29
Michigan   23
Washington 23
Texas      22
Colorado   22
Pennsylvania 20
New Jersey 19
Mississippi 16
Maryland   15
Oregon     13
Massachusetts 11
Virginia   11
Indiana    9
Alabama    9
Arizona    6
Utah        6
New Hampshire 4
Delaware    4
```

project - Jupyter Notebook x Assessment x +

localhost:8888/notebooks/MyPythoncode/project.ipynb

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3 (ipykernel)

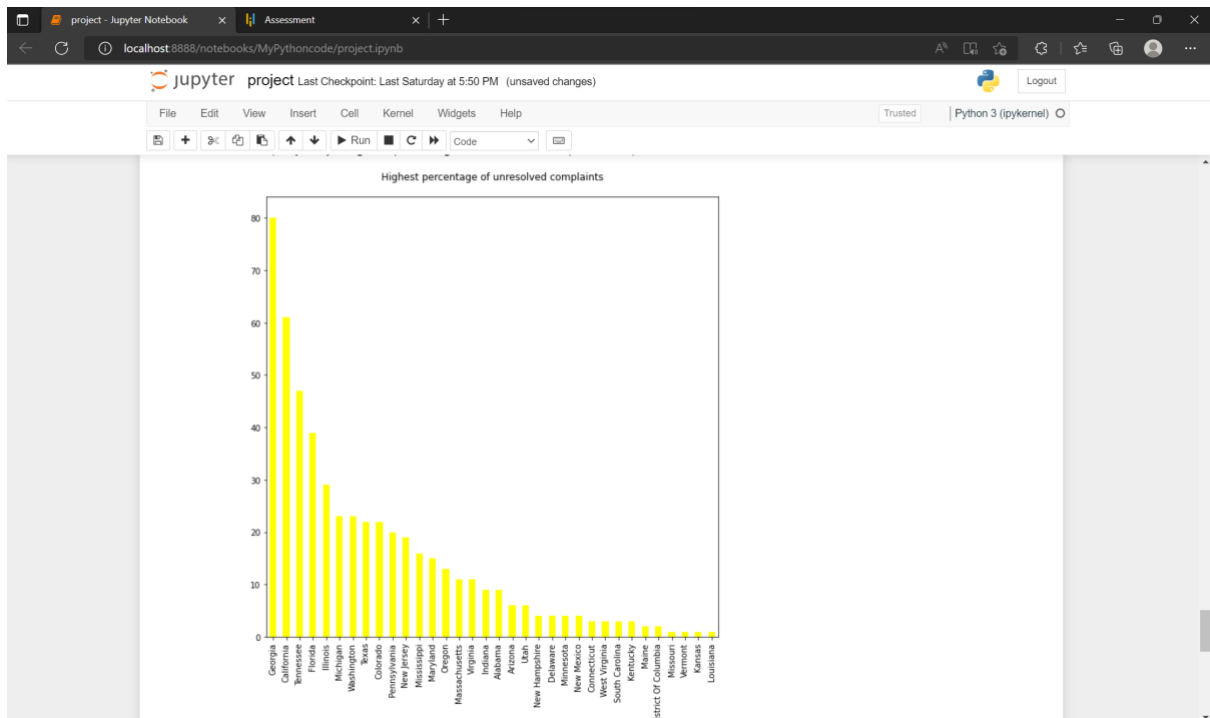
Jupyter project Last Checkpoint: Last Saturday at 5:50 PM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3 (ipykernel)

```
New Hampshire 4
Delaware 4
Minnesota 4
New Mexico 4
Connecticut 3
West Virginia 3
South Carolina 3
Kentucky 3
Maine 2
District Of Columbia 2
Missouri 1
Vermont 1
Kansas 1
Louisiana 1
Name: State, dtype: int64
```

```
In [28]: # Georgia has the Highest Number of unresolved complaints
# Show this by Bar Graph
comcast_tele_consumer_unresolved_complaint_count.plot(kind='bar',figsize=(10,10),color="yellow")
plt.title('Highest percentage of unresolved complaints\n')
```



project - Jupyter Notebook x Assessment x +

localhost:8888/notebooks/MyPythoncode/projectIpyb

jupyter project Last Checkpoint: Last Saturday at 5:50 PM (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

**Task 8.1: Provide the percentage of complaints resolved till date, which were received through the Internet and customer care calls.**

```
In [29]: # Check unique values in Received Via Column
comcast_tele_consumer["Received Via"].unique()

Out[29]: array(['Customer Care Call', 'Internet'], dtype=object)

In [30]: # So there are only two values in that columns so no need to sort we can directly proceed to task 6

In [31]: comcast_tele_consumer.New_Status.value_counts()

Out[31]: Closed    1707
Open         517
Name: New_Status, dtype: int64

In [32]: # Used autopct='%1.1f%%' to show percentage under the pie chart
myexplode = [0.2, 0]
plt.title('Complaints Status through the Internet & Customer Care Calls\n')
comcast_tele_consumer.New_Status.value_counts().plot(kind='pie', explode = myexplode, autopct='%1.1f%%',
figsize = (14,6))
```

