In [1]:
```
import pandas as pd
```

In [2]:
```
import os
os.path.abspath('/home/smaher/Downloads/DataScienceInterview')
```

Out[2]:
```
'/home/smaher/Downloads/DataScienceInterview'
```

In [3]:
```
data = pd.read_csv('customer_complaints.csv')
```

In [4]:
```
data
```

Out[4]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	Stat
0	250635	Comcast Cable Internet Speeds	22-04-15	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Marylar
1	223441	Payment disappear - service got disconnected	4/8/2015	4-Aug-15	10:22:56 AM	Internet	Acworth	Georg
2	242732	Speed and Service	18-04-15	18-Apr-15	9:55:47 AM	Internet	Acworth	Georg
3	277946	Comcast Imposed a New Usage Cap of 300GB that ...	5/7/2015	5-Jul-15	11:59:35 AM	Internet	Acworth	Georg
4	307175	Comcast not working and no service to boot	26-05-15	26-May-15	1:25:26 PM	Internet	Acworth	Georg
...	...	...	...	...	...	...	...	
2219	213550	Service Availability	4/2/2015	4-Feb-15	9:13:18 AM	Customer Care Call	Youngstown	Floric
2220	318775	Comcast Monthly Billing for Returned Modem	6/2/2015	6-Feb-15	1:24:39 PM	Customer Care Call	Ypsilanti	Michiga
2221	331188	complaint about comcast	6/9/2015	6-Sep-15	5:28:41 PM	Internet	Ypsilanti	Michiga
2222	360489	Extremely unsatisfied Comcast customer	23-06-15	23-Jun-15	11:13:30 PM	Customer Care Call	Ypsilanti	Michiga
2223	363614	Comcast, Ypsilanti MI Internet Speed	24-06-15	24-Jun-15	10:28:33 PM	Customer Care Call	Ypsilanti	Michiga

2224 rows × 11 columns

In [5]:
```
data.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 [7]:
```
data.head()
```

Out[7]:

	Ticket #	Customer Complaint	Date	Date_month_year	Time	Received Via	City	State	c
0	250635	Comcast Cable Internet Speeds	22-04-15	22-Apr-15	3:53:50 PM	Customer Care Call	Abingdon	Maryland	21
1	223441	Payment disappear - service got disconnected	4/8/2015	4-Aug-15	10:22:56 AM	Internet	Acworth	Georgia	30
2	242732	Speed and Service	18-04-15	18-Apr-15	9:55:47 AM	Internet	Acworth	Georgia	30
3	277946	Comcast Imposed a New Usage Cap of 300GB that ...	5/7/2015	5-Jul-15	11:59:35 AM	Internet	Acworth	Georgia	30
4	307175	Comcast not working and no service to boot	26-05-15	26-May-15	1:25:26 PM	Internet	Acworth	Georgia	30

In [9]:
```
data.shape[0]
```

Out[9]:
```
2224
```

In [10]:
```
data.isnull().sum()
```

Out[10]:
```
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 [11]:
```
### Convert the columns names so that they don't have space and are more re
data.columns = [i.lower().replace(" ","_").replace("-","_") for i in data.c
data.columns
```

Out[11]:
```
Index(['ticket_#', 'customer_complaint', 'date', 'date_month_year', 'time',
       'received_via', 'city', 'state', 'zip_code', 'status',
       'filing_on_behalf_of_someone'],
      dtype='object')
```

Cleaning Text Data in Python

Text data contains a lot of noise either in the form of symbols or in the form of punctuations and stopwords. Therefore, it becomes necessary to clean the text, not just for making it more understandable but also for getting better insights

Here, we have 11 columns in our dataset out of which two columns (‘customer_complaint’, 'received_via', ‘city’,'state','status','filing_on_behalf_of_someone) contain textual data. So, let’s start with the ‘customer_complaint’ column first and take a look at the text present in this column:

In [13]:
```
data['customer_complaint'].unique()
```

Out[13]:
```
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)
```

Take a close look at the title of products. Some product customer_complaint contain repeating names separated by three consecutive commas (...,,,-). So, let’s clean the customer_complaint of the products:

In [19]:
```
data['customer_complaint']=data['customer_complaint'].apply(lambda x: x.spl
```

In [22]:
```
data['cleaned']=data['customer_complaint'].apply(lambda x: x.lower())
```

Lowercase the customer_complaint

In [24]:
```
data['cleaned']
```

Out[24]:
```
0                            comcast cable internet speeds
1              payment disappear - service got disconnected
2                                           speed and service
3       comcast imposed a new usage cap of 300gb that ...
4                comcast not working and no service to boot
                                ...
2219                                    service availability
2220             comcast monthly billing for returned modem
2221                                 complaint about comcast
2222                 extremely unsatisfied comcast customer
2223                   comcast, ypsilanti mi internet speed
Name: cleaned, Length: 2224, dtype: object
```

Remove digits and words containing digits

we need to remove numbers and words containing digits from the customer_complaint. I am doing this because digits and words containing digits do not give much importance to the main words. To do this, I am using regular expressions with lambda functions.

In [26]:
```
import re
data['cleaned']=data['cleaned'].apply(lambda x: re.sub('\w*\d\w*','', x))
```

Remove Punctuations

Punctuations are the marks in English like commas, hyphens, full stops, etc. These are important for English grammar but not for text analysis. Therefore, they need to be removed:

In [28]:
```
import string
data['cleaned']=data['cleaned'].apply(lambda x: re.sub('[%s]' % re.escape(s
```

Here, *string.punctuations* function contains all the punctuations and we use regular expressions to search them in the text and remove them. Finally, we still have some extra spaces present in the data. Let's remove them:

In [31]:
```
# Removing extra spaces
data['cleaned']=data['cleaned'].apply(lambda x: re.sub(' +',' ',x))
```

Let's how our text looks after cleaning:

In [32]:
```
for index,text in enumerate(data['cleaned'][35:40]):
  print('customer_complaint %d:\n'%(index+1),text)
```

```
customer_complaint 1:
 comcast
customer_complaint 2:
 no service
customer_complaint 3:
 comcast
customer_complaint 4:
 internet billing and servie issues
customer_complaint 5:
 comcast blocking directv signals
```

Preparing Text Data for Exploratory Data Analysis (EDA)

We have already cleaned our data and have our corpus ready

we use "Document Term Matrix" provides the frequency of a word in a corpus (collection of documents), which in this case are customer_complaint. It helps in analyzing the occurrence of words in different documents in a dataset.

we use following step before moving to the next EDA step

1. Stopwords Removal
2. Lemmatization
3. Create Document Term Matrix

Stopwords are the most common words of a language like 'I', 'this', 'is', 'in' which do not add much value to the meaning of a document.

Lemmatization is a systematic process of reducing a token to its lemma. It uses vocabulary, word structure, part of speech tags, and grammar relations to convert a word to its base form.

so we use NLP SpaCy for the removal of stopwords and lemmatization.

In [34]:
```
# Importing spacy
```

```
import spacy

# Loading model
nlp = spacy.load('en_core_web_sm',disable=['parser', 'ner'])

# Lemmatization with stopwords removal
data['lemmatized']=data['cleaned'].apply(lambda x: ' '.join([token.lemma_ f
```

We have removed stopwords and lemmatized our reviews successfully. Let's group them according to the products:

In [37]:
```
data_grouped=data[['customer_complaint','lemmatized']].groupby(by='customer_
data_grouped.head()
```

Out[37]:

customer_complaint	lemmatized
(Comcast is not my complaint!) Cyber Tele-marketing is my complaint!	comcast complaint cyber telemarketing complaint
10 Days No Service - 12 Appointments Comcast Hasn't Shown Up - Comcast Has Not Fixed Home Infinity X1 - In 10 Days- Cannot Make 911 Calls	day service appointment comcast not show com...
2 months and Comcast has not fixed problem	month comcast fix problem
2+ Day Degraded Services	day degraded service
300 GB monthly allowance	gb monthly allowance

It's time to create a Document Term Matrix.

In [45]:
```
# Creating Document Term Matrix
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(analyzer='word')
data=cv.fit_transform(data_grouped['lemmatized'])
df_dtm = pd.DataFrame(data.toarray(), columns=cv.get_feature_names())
df_dtm.index=data_grouped.index
df_dtm.head(10)
```

```
/root/miniconda3/envs/venv/lib/python3.10/site-packages/sklearn/utils/depre
cation.py:87: FutureWarning: Function get_feature_names is deprecated; get_
feature_names is deprecated in 1.0 and will be removed in 1.2. Please use g
et_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[45]:

customer_complaint	able	abuse	abysmal	acceptable	access	accidentally	accord	account
(Comcast is not my complaint!) Cyber Tele-marketing is my complaint!	0	0	0	0	0	0	0	0
10 Days No Service - 12 Appointments Comcast Hasn't Shown Up - Comcast Has Not Fixed Home Infinity X1 - In 10 Days- Cannot Make 911 Calls	0	0	0	0	0	0	0	0
2 months and Comcast has not fixed problem	0	0	0	0	0	0	0	0
2+ Day Degraded Services	0	0	0	0	0	0	0	0
300 GB monthly allowance	0	0	0	0	0	0	0	0
300 gb cap	0	0	0	0	0	0	0	0
300GB/month Data Cap	0	0	0	0	0	0	0	0
300GB/month data cap 'trial' for several years now	0	0	0	0	0	0	0	0
60 day delay in cancelling service agreement	0	0	0	0	0	0	0	0
60 days to close my account	0	0	0	0	0	0	0	1

10 rows × 1145 columns

Finally, we have completed all the procedures required before starting our analysis, and we have our dataset present in the exact format needed for the exploration stage.

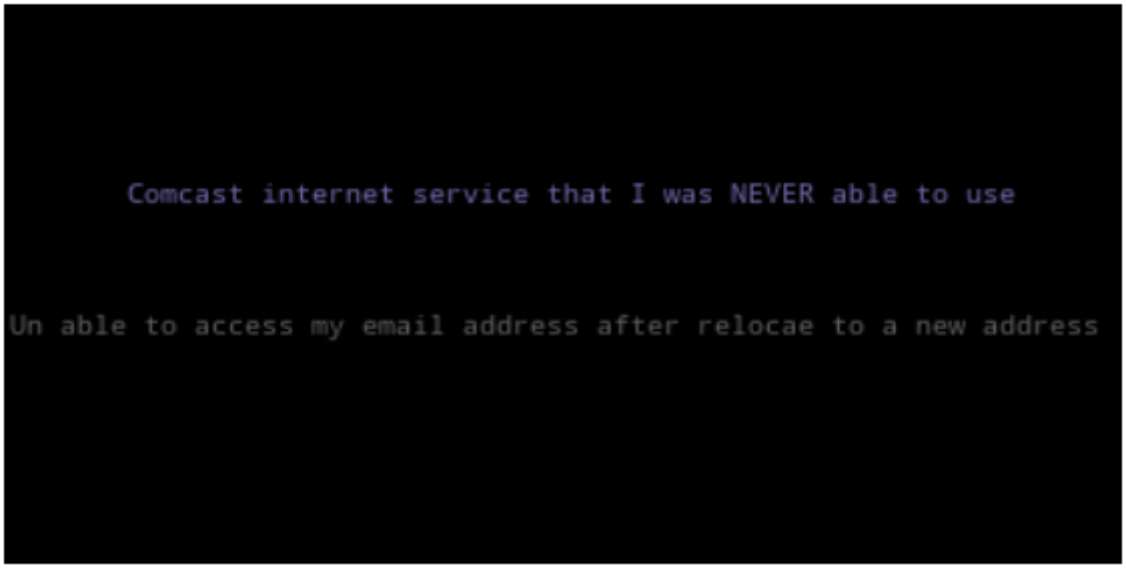
Exploratory Data Analysis on customer_complaints

Word clouds can be generated using the wordcloud library. So, let's plot word clouds for each product:

In [62]:
```
# For visualizations
import matplotlib.pyplot as plt
# Importing wordcloud for plotting word clouds and textwrap for wrapping lo
from wordcloud import WordCloud
from textwrap import wrap

# Function for generating word clouds
def generate_wordcloud(data,title):
  wc = WordCloud(width=400, height=200, max_words=30,colormap="Dark2").gene
  plt.figure(figsize=(10,8))
  plt.imshow(wc, interpolation='bilinear')
```

```
  plt.axis("off")
  plt.title('\n'.join(wrap(title,60)),fontsize=13)
  plt.show()

# Transposing document term matrix
df_dtm=df_dtm.transpose()

# Plotting word cloud for each product
for index,product in enumerate(df_dtm.columns):
  generate_wordcloud(df_dtm[product].sort_values(ascending=False),product)
```

able



abuse