In [1]:

```
!pip install --upgrade google-api-python-client google-auth-httplib2 google-auth-oauthlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: google-api-python-client in /usr/local/lib/python3.7/dist-packages (2.51.0)
Requirement already satisfied: google-auth-httplib2 in /usr/local/lib/python3.7/dist-packages (0.1.0)
Requirement already satisfied: google-auth-oauthlib in /usr/local/lib/python3.7/dist-packages (0.5.2)
Requirement already satisfied: uritemplate<5,>=3.0.1 in /usr/local/lib/python3.7/dist-packages (from google-api-python-client) (3.0.1)
Requirement already satisfied: google-auth<3.0.0dev,>=1.16.0 in /usr/local/lib/python3.7/dist-packages (from google-api-python-client) (1.35.0)
Requirement already satisfied: google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5 in /usr/local/lib/python3.7/dist-packages (from google-api-python-clien
t) (1.31.6)
Requirement already satisfied: httplib2<1dev,>=0.15.0 in /usr/local/lib/python3.7/dist-packages (from google-api-python-client) (0.17.4)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from google-auth-httplib2) (1.15.0)
Requirement already satisfied: protobuf<4.0.0dev,>=3.12.0 in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.3
1.5->google-api-python-client) (3.17.3)
Requirement already satisfied: setuptools>=40.3.0 in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5->goog
le-api-python-client) (57.4.0)
Requirement already satisfied: googleapis-common-protos<2.0dev,>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.
0.0dev,>=1.31.5->google-api-python-client) (1.56.2)
Requirement already satisfied: requests<3.0.0dev,>=2.18.0 in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.3
1.5->google-api-python-client) (2.23.0)
Requirement already satisfied: packaging>=14.3 in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5->google-
api-python-client) (21.3)
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5->google-api-python-
client) (2022.1)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from google-auth<3.0.0dev,>=1.16.0->google-api-python-client) (4.2.4)
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-auth<3.0.0dev,>=1.16.0->google-api-python-client) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3.0.0dev,>=1.16.0->google-api-python-client) (4.8)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging>=14.3->google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<
3.0.0dev,>=1.31.5->google-api-python-client) (3.0.9)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3.0.0dev,>=1.16.0->google-api-pyth
on-client) (0.4.8)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0dev,>=2.18.0->google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.
3.0,<3.0.0dev,>=1.31.5->google-api-python-client) (2022.6.15)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0dev,>=2.18.0->google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<
3.0.0dev,>=1.31.5->google-api-python-client) (2.10)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0dev,>=2.18.0->google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.
3.0,<3.0.0dev,>=1.31.5->google-api-python-client) (3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests<3.0.0dev,>=2.18.0->google-api-core!=2.0.
*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5->google-api-python-client) (1.24.3)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist-packages (from google-auth-oauthlib) (1.3.1)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib) (3.2.0)
```

In [2]:

```
# Scrape Or Download Comments Using Python Through The Youtube Data API
# Watch the youtube video for explaination
# https://youtu.be/B9uCX2s7y7A

api_key = "AIzaSyDLXD0m2l0R8GWiRdUthv2DimwihAQPfOA" # Replace this dummy api key with your own.

from apiclient.discovery import build
youtube = build('youtube', 'v3', developerKey=api_key)

import pandas as pd

ID = "vLHpfulLcUE" # Replace this YouTube video ID with your own.

box = [['Name', 'Comment', 'Time', 'Likes', 'Reply Count']]
```

```
def scrape_comments_with_replies():
    data = youtube.commentThreads().list(part='snippet', videoId=ID, maxResults='100', textFormat="plainText").execute()

    for i in data["items"]:

        name = i["snippet"]['topLevelComment']["snippet"]["authorDisplayName"]
        comment = i["snippet"]['topLevelComment']["snippet"]["textDisplay"]
        published_at = i["snippet"]['topLevelComment']["snippet"]['publishedAt']
        likes = i["snippet"]['topLevelComment']["snippet"]['likeCount']
        replies = i["snippet"]['totalReplyCount']

        box.append([name, comment, published_at, likes, replies])

        totalReplyCount = i["snippet"]['totalReplyCount']

        if totalReplyCount > 0:

            parent = i["snippet"]['topLevelComment']["id"]

            data2 = youtube.comments().list(part='snippet', maxResults='100', parentId=parent,
                                            textFormat="plainText").execute()

            for i in data2["items"]:
                name = i["snippet"]["authorDisplayName"]
                comment = i["snippet"]["textDisplay"]
                published_at = i["snippet"]['publishedAt']
                likes = i["snippet"]['likeCount']
                replies = ""

                box.append([name, comment, published_at, likes, replies])

    while ("nextPageToken" in data):

        data = youtube.commentThreads().list(part='snippet', videoId=ID, pageToken=data["nextPageToken"],
                                             maxResults='100', textFormat="plainText").execute()

        for i in data["items"]:
            name = i["snippet"]['topLevelComment']["snippet"]["authorDisplayName"]
            comment = i["snippet"]['topLevelComment']["snippet"]["textDisplay"]
            published_at = i["snippet"]['topLevelComment']["snippet"]['publishedAt']
            likes = i["snippet"]['topLevelComment']["snippet"]['likeCount']
            replies = i["snippet"]['totalReplyCount']

            box.append([name, comment, published_at, likes, replies])

            totalReplyCount = i["snippet"]['totalReplyCount']

            if totalReplyCount > 0:

                parent = i["snippet"]['topLevelComment']["id"]

                data2 = youtube.comments().list(part='snippet', maxResults='100', parentId=parent,
                                                textFormat="plainText").execute()

                for i in data2["items"]:
                    name = i["snippet"]["authorDisplayName"]
                    comment = i["snippet"]["textDisplay"]
                    published_at = i["snippet"]['publishedAt']
```

```
                    likes = i["snippet"]['likeCount']
                    replies = ''

                    box.append([name, comment, published_at, likes, replies])

    df = pd.DataFrame({'Name': [i[0] for i in box], 'Comment': [i[1] for i in box], 'Time': [i[2] for i in box],
                       'Likes': [i[3] for i in box], 'Reply Count': [i[4] for i in box]})

    df.to_csv('youtube-comments.csv', index=False, header=False)

    return "Successful! Check the CSV file that you have just created."
```

In [3]:
```
scrape_comments_with_replies()
```

Out[3]:
```
'Successful! Check the CSV file that you have just created.'
```

In [4]:
```
import pandas as pd
pd.options.display.max_rows = 10

df = pd.read_csv('youtube-comments.csv', index_col=0)
df
```

Out[4]:

Name	Comment	Time	Likes	Reply Count
Rahul J	How you guys are convince him to do that-?	2022-06-18T11:24:09Z	0	0.0
Preetam Shetty	2 cough syrups, on the rocks 🤣	2022-06-17T19:48:37Z	0	1.0
Последователь Джины	means	2022-06-18T22:22:42Z	0	NaN
Asmit Sharma	Better actor than Starkids lok	2022-06-16T06:12:35Z	0	0.0
nobody	Two cough syrup on the rocks....😀😀😀😀😀😂	2022-06-15T05:58:59Z	0	0.0
...	...	...	...	...
Saurabh Nimje	*"Gotte mu me aa gaye...!!"*\nAre you expectin...	2022-05-21T18:19:15Z	2	NaN
Akash Srivastava	Yeah what was it	2022-05-21T14:08:34Z	2	NaN
anilesh singh	What was it?	2022-05-21T13:35:09Z	2	NaN
Aarjav Doshi	💯✅	2022-05-21T09:36:13Z	0	0.0
Soham Palkar	1.💯	2022-05-21T09:35:28Z	0	0.0

1631 rows × 4 columns

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Comment', 'Time', 'Likes', 'Reply Count'], dtype='object')
```

In [6]:
```
df['Comment'].nunique()
```

Out[6]: 1614

In [7]:
```
# Load library
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

# download the set of stop words the first time
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[7]: True

In [8]:
```
# Load stop words
stop_words = stopwords.words('english')

# Show stop words
stop_words[:10]
```

Out[8]: ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're"]

In [9]:
```
#### Now creating some functions to do text processing
# Removing hashtags and mentions
def get_hashtags(text):
    hashtags = re.findall(r'\#\w+',text.lower())
    return hashtags
def get_mentions(text):
    mentions = re.findall(r'\@\w+',text.lower())
    return mentions

# Cleaning up the text of the tweets
def remove_content(text):
    text = re.sub(r"http\S+", "", text) #remove urls
    text=re.sub(r'\S+\.com\S+','',text) #remove urls
    text=re.sub(r'\@\w+','',text) #remove mentions
    text =re.sub(r'\#\w+','',text) #remove hashtags
    return text

def process_tweet(tweet):
    """
    tweets cleaning by
    1) lowering the case of the tweet,
    2) removing unwanted symbols and replacing them with a whitespace,
    3) split sentences into words according to whitespaces and then
    4) join back with a single whitespace as separator between various words
    """
    return " ".join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])", " ",tweet.lower()).split())

def process_text(text, stem=False): #clean text
    text=remove_content(text)
    lemmatizer=WordNetLemmatizer()
    text = re.sub('[^A-Za-z]', ' ', text.lower()) #remove non-alphabets
    text = re.sub(r'@[A-Za-z0-9]+', '', str(text)) # remove @mentions
    text = re.sub(r'#', '',  str(text)) # remove the '#' symbol
```

```
    text = re.sub(r'RT[\s]+', '',  str(text)) # remove RT
    text = re.sub(r'https?\/\/S+', '',  str(text)) # remove the hyperlink
    text = re.sub(r'http\S+', '',  str(text)) # remove the hyperlink
    text = re.sub(r'www\S+', '',  str(text)) # remove the www
    text = re.sub(r'pic+', '',  str(text)) # remove the pic
    text = re.sub(r'com', '',  str(text)) # remove the pic
    text = re.sub(r"\bamp\b", ' ', text.lower()) #remove "amp" which is coming from the translation of &
    text = re.sub(r"\bco\b", ' ', text.lower()) #remove "co" which was one of the top words found below
    tokenized_text = word_tokenize(text) #tokenize
    #tokenized_text = [lemmatizer.lemmatize(word) for word in tokenized_text]
    clean_text = [
         word for word in tokenized_text
         if (word not in stop_words and len(word)>1)
    ]
    if stem:
        clean_text=[stemmer.stem(word) for word in clean_text]
    clean_text = [lemmatizer.lemmatize(word) for word in clean_text]
    return ' '.join(clean_text)

#functions used to remove search terms from all the tweets
#function to remove duplicates from a string - in this case the string is the keywords used to scrape the tweets
def removeDupWithoutOrder(string):
    words = string.lower().split()
    return " ".join(sorted(set(words), key=words.index)).replace('OR', '').replace('  ', ' ')

#function to search for string i.e. remove specific words (search_terms in this case)
def remove_search(text, search_terms):
    query = text.lower()
    querywords = query.split()
    resultwords  = [word for word in querywords if word.lower() not in search_terms]
    return ' '.join(resultwords)

# define function to plot frequency of bi-grams, tri-grams, single words, phrases etc
from sklearn.feature_extraction.text import CountVectorizer
def plot_topn(sentences, ngram_range=(1,3), top=20,firstword=''):
    c=CountVectorizer(ngram_range=ngram_range)
    X=c.fit_transform(sentences)
    words=pd.DataFrame(X.sum(axis=0),columns=c.get_feature_names()).T.sort_values(0,ascending=False).reset_index()
    res=words[words['index'].apply(lambda x: firstword in x)].head(top)
    pl=px.bar(res, x='index',y=0)
    pl.update_layout(yaxis_title='count',xaxis_title='Phrases')
    pl.show('png')
```

In [10]:
```
import re
re.compile('<title>(.*)</title>')
```

Out[10]:
```
re.compile(r'<title>(.*)</title>', re.UNICODE)
```

In [11]:
```
# removing useless content (hashtags, mentions)
df['Comment']=df['Comment'].apply(lambda x: remove_content(x))
```

In [12]:
```
# Several functions applied here: processing the comments to remove punctuation, hashtags, mentions

df['cleaned_comments']=df['Comment'].apply(lambda x: process_tweet(x))
```

In [13]:
```
df
```

Out[13]:

Name	Comment	Time	Likes	Reply Count	cleaned_comments
Rahul J	How you guys are convince him to do that-?	2022-06-18T11:24:09Z	0	0.0	how you guys are convince him to do that
Preetam Shetty	2 cough syrups, on the rocks 🤣	2022-06-17T19:48:37Z	0	1.0	2 cough syrups on the rocks
Последователь Джины	means	2022-06-18T22:22:42Z	0	NaN	means
Asmit Sharma	Better actor than Starkids lok	2022-06-16T06:12:35Z	0	0.0	better actor than starkids lok
nobody	Two cough syrup on the rocks....😀😀😀😀😀😂	2022-06-15T05:58:59Z	0	0.0	two cough syrup on the rocks
...	...	...	...	...	...
Saurabh Nimje	*"Gotte mu me aa gaye...!!"*\nAre you expectin...	2022-05-21T18:19:15Z	2	NaN	gotte mu me aa gaye are you expecting this twe...
Akash Srivastava	Yeah what was it	2022-05-21T14:08:34Z	2	NaN	yeah what was it
anilesh singh	What was it?	2022-05-21T13:35:09Z	2	NaN	what was it
Aarjav Doshi	💯✅	2022-05-21T09:36:13Z	0	0.0	
Soham Palkar	1.💯	2022-05-21T09:35:28Z	0	0.0	1

1631 rows × 5 columns

In [14]:
```
from wordcloud import WordCloud,ImageColorGenerator
from PIL import Image
import urllib
import requests
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import matplotlib.pyplot as plt
```

In [15]:
```
comment_words = ''
stopwords = set(STOPWORDS)

# iterate through the csv file
for val in df.cleaned_comments:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()

    # Converts each token into lowercase
    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()

    comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 1000, height = 800,
                background_color='white', colormap='Set2',
                collocations=False,
                stopwords = stopwords,
```

```
                normalize_plurals=False,
                min_font_size = 12).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (10, 10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

In [16]:

```
import pandas as pd
import numpy as np
import string
import re
import nltk
from textblob import TextBlob
from wordcloud import WordCloud
import matplotlib.pyplot as plt
```

In [17]:
```
def getSubjectivity(text):
    return TextBlob( str(text)).sentiment.subjectivity

def getPolarity(text):
    return TextBlob( str(text)).sentiment.polarity
```

In [18]:
```
df.dropna(subset=['cleaned_comments'], inplace = True)
df.reset_index(drop=True, inplace=True)
```

In [19]:
```
df['Subjectivity'] = df['cleaned_comments'].apply(getSubjectivity)
df['Polarity'] = df['cleaned_comments'].apply(getPolarity)
df.head()
```

Out[19]:

	Comment	Time	Likes	Reply Count	cleaned_comments	Subjectivity	Polarity
0	How you guys are convince him to do that-?	2022-06-18T11:24:09Z	0	0.0	how you guys are convince him to do that	0.0	0.0
1	2 cough syrups, on the rocks 🤣	2022-06-17T19:48:37Z	0	1.0	2 cough syrups on the rocks	0.0	0.0
2	means	2022-06-18T22:22:42Z	0	NaN	means	0.0	0.0
3	Better actor than Starkids lok	2022-06-16T06:12:35Z	0	0.0	better actor than starkids lok	0.5	0.5
4	Two cough syrup on the rocks....😀😀😀😀😀😂	2022-06-15T05:58:59Z	0	0.0	two cough syrup on the rocks	0.0	0.0

In [20]:
```
# Create a function to compute negative (-1), neutral (0) and positive (+1) analysis
def get_Polarity_Analysis(score):
    if score < 0:
      return 'Negative'
    elif score == 0:
      return 'Neutral'
    else:
      return 'Positive'
def get_Subjectivity_Analysis(score):
    if score >  0:
      return 'Opinion'
    else:
      return 'Fact'

df['Analysis_Polarity'] = df['Polarity'].apply(get_Polarity_Analysis)

df['Analysis_Subjectivity'] = df['Subjectivity'].apply(get_Subjectivity_Analysis)
```

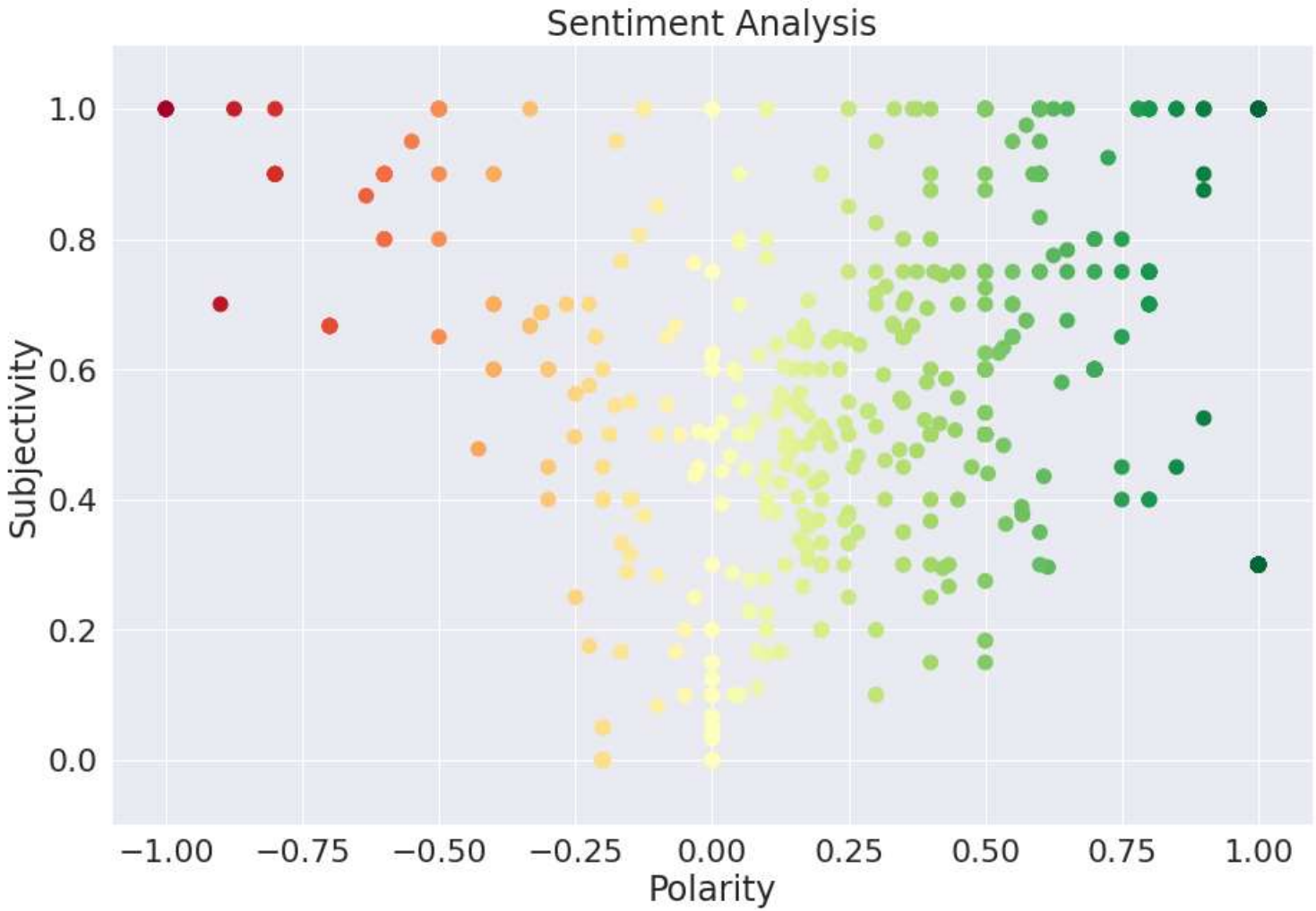
In [21]:
```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(font_scale=2)
```

In [22]:
```
plt.figure(figsize=(15,10))

# plt.style.use('seaborn-pastel')
```

```
plt.scatter(df['Polarity'], df['Subjectivity'], c=df['Polarity'], s=100, cmap='RdYlGn')

plt.xlim(-1.1, 1.1)
plt.ylim(-0.1, 1.1)
plt.title('Sentiment Analysis')
plt.xlabel('Polarity')
plt.ylabel('Subjectivity')
plt.show(),
```

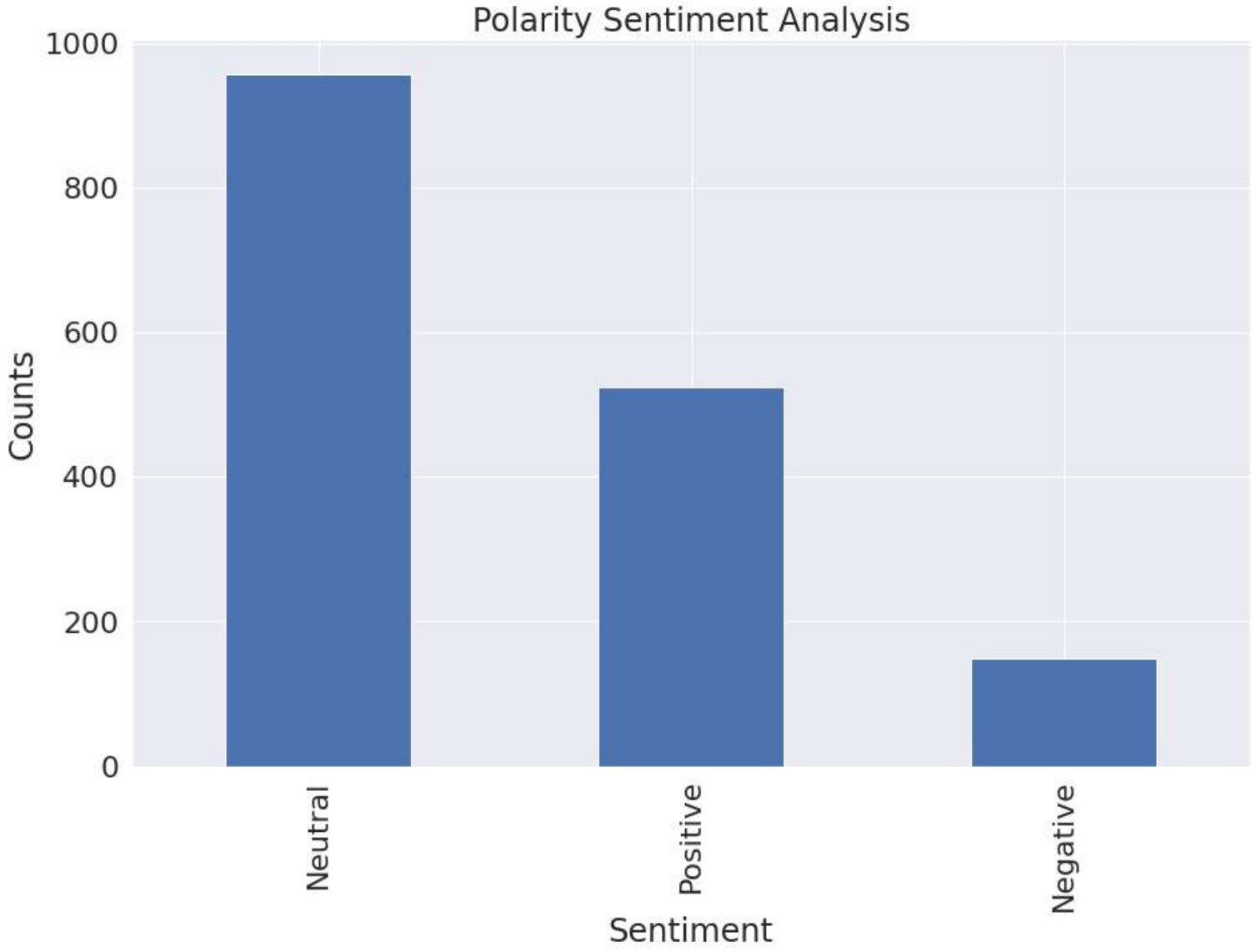


Out[22]: (None,)

In [23]:
```
# Plotting and visualizing the counts
plt.figure(figsize=(15,10))

plt.title('Polarity Sentiment Analysis')
plt.xlabel('Sentiment')
plt.ylabel('Counts')
```

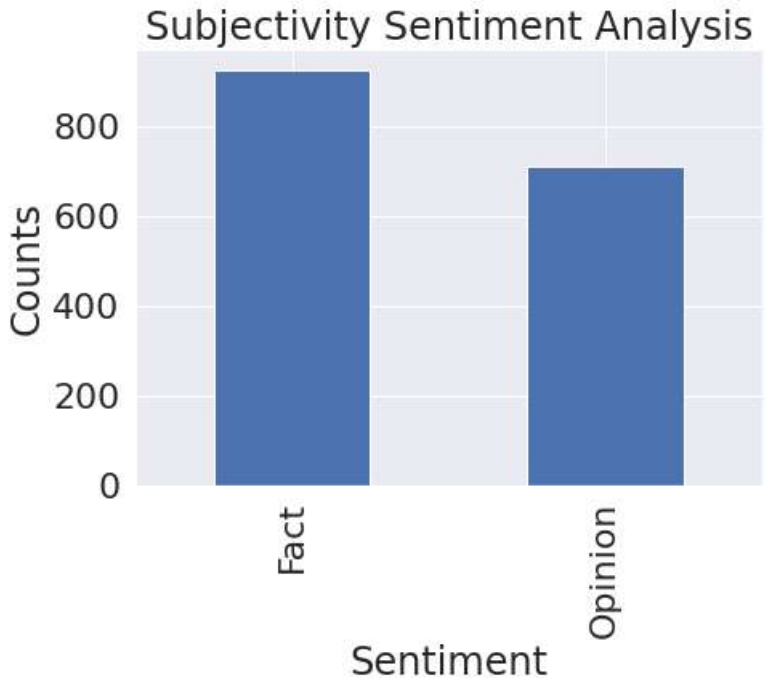
```
df['Analysis_Polarity'].value_counts().plot(kind = 'bar')
plt.show()
```



In [24]:

```
# Plotting and visualizing the counts
plt.figure(figsize=(7,5))

plt.title('Subjectivity Sentiment Analysis')
plt.xlabel('Sentiment')
plt.ylabel('Counts')
df['Analysis_Subjectivity'].value_counts().plot(kind = 'bar')
plt.show()
```


Subjectivity Sentiment Analysis
Counts
800
600
400
200
0
Fact
Opinion
Sentiment

In [24]: