In [1]:

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

In [4]:

```
import codecs
path = '/home/kalihacker/Downloads/covid-19_vaccine_tweets_with_sentiment.csv'

with codecs.open(path, 'r', 'utf-8', 'ignore') as f:
    df = pd.read_csv(f)
df[0:2]
```

Out[4]:

	tweet_id	label	tweet_text
0	1.360342e+18	1	4,000 a day dying from the so called Covid-19 ...
1	1.382896e+18	2	Pranam message for today manifested in Dhyan b...

In [5]:

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /home/kalihacker/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[5]:

```
True
```

In [7]:

```
from nltk.corpus import stopwords
from nltk.tokenize import RegexpTokenizer
from nltk.stem.porter import PorterStemmer
tokenizer=RegexpTokenizer(r'\w+')

ps=PorterStemmer()

en_stop=set(stopwords.words('english'))
```

In [8]:

```
def getCleanedText(text):
    text=text.lower()
    tokens=tokenizer.tokenize(text)
    new_tokens=[token for token in tokens if token not in en_stop]
    stemmed_tokens=[ps.stem(tokens) for tokens in new_tokens]
    clean_text=" ".join(stemmed_tokens)
    return clean_text
```

In [9]:

```
df['tweet_text']=df['tweet_text'].apply(getCleanedText)
df['tweet_text'].head()
```

Out[9]:

```
0    4 000 day die call covid 19 vaccin dailybeast ...
1    pranam messag today manifest dhyan meenapranam...
2    hyderabad base bharatbiotech sought fund gover...
3    confirm chines vaccin dont high protect rate a...
4    lab studi suggest pfizer moderna vaccin protec...
Name: tweet_text, dtype: object
```

In [12]:

```
#Create a new function to get the subjectivity
def getSubjectivity(tweet_text):
    return TextBlob(tweet_text).sentiment.subjectivity
```

In [14]:

```
#Create a function to get the polority
def getPolarity(tweet_text):
    return TextBlob(tweet_text).sentiment.polarity
```

In [15]:

```
#Create two new columns
df['Subjectivity'] = df['tweet_text'].apply(getSubjectivity)
df['Polarity'] = df['tweet_text'].apply(getPolarity)
```

In [16]:

```
#Show the dataframe with the new columns
df
```

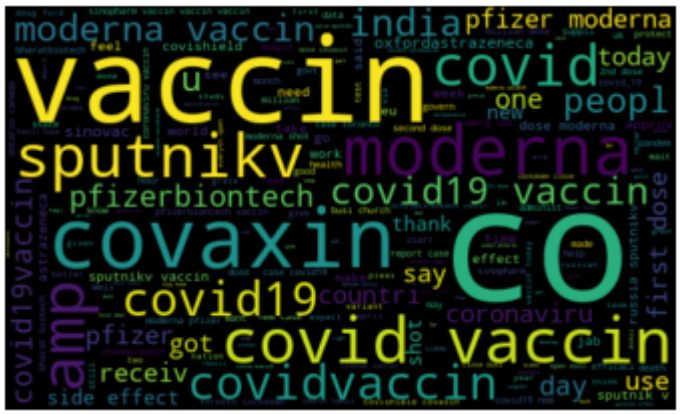
Out[16]:

	tweet_id	label	tweet_text	Subjectivity	Polarity
0	1.360342e+18	1	4 000 day die call covid 19 vaccin dailybeast ...	0.000000	0.000000
1	1.382896e+18	2	pranam messag today manifest dhyan meenapranam...	0.600000	0.333333
2	1.375673e+18	2	hyderabad base bharatbiotech sought fund gover...	1.000000	-0.800000
3	1.381311e+18	1	confirm chines vaccin dont high protect rate a...	0.313333	0.020000
4	1.362166e+18	3	lab studi suggest pfizer moderna vaccin protec...	0.000000	0.000000
...	...	...	...	...	...
5995	1.370975e+18	2	swamy39 dr swamy39 jee mani peopl like get cov...	0.000000	0.000000
5996	1.379827e+18	3	happi fulli vaccin covid 19 im readi serv vacc...	0.000000	0.000000
5997	1.384789e+18	2	serum institut india announc cost covishield v...	0.000000	0.000000
5998	1.382355e+18	1	___batshitcrazi barrowfordhead bectulli mine y...	0.000000	0.000000
5999	1.380051e+18	2	smart sympathi attend oblong noth educ feedbac...	0.642857	0.214286

6000 rows × 5 columns

In [20]:

```
#plot the word cloud
allWords = ''.join([twts for twts in df['tweet_text']])
wordCloud = WordCloud(width = 500,height =300,random_state = 21,max_font_size =
119).generate(allWords)
plt.imshow(wordCloud,interpolation = "bilinear")
plt.axis('off')
plt.show()
```



In [21]:

```
#Create a function to compute the negative,neutral and positive analysis
def getAnalysis(score):
    if score < 0:
        return 'Negative'
    elif score == 0:
        return 'Neutral'
    else:
        return 'Positive'
df['Analysis'] = df['Polarity'].apply(getAnalysis)
df
```

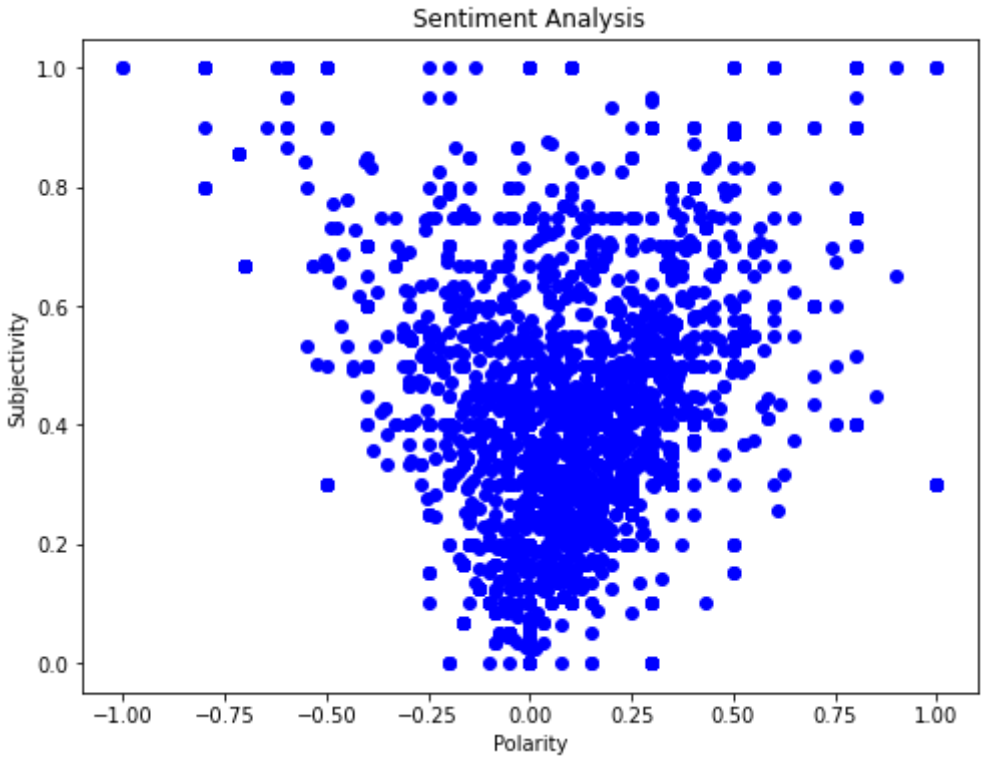
Out[21]:

	tweet_id	label	tweet_text	Subjectivity	Polarity	Analysis
0	1.360342e+18	1	4 000 day die call covid 19 vaccin dailybeast ...	0.000000	0.000000	Neutral
1	1.382896e+18	2	pranam messag today manifest dhyan meenapranam...	0.600000	0.333333	Positive
2	1.375673e+18	2	hyderabad base bharatbiotech sought fund gover...	1.000000	-0.800000	Negative
3	1.381311e+18	1	confirm chines vaccin dont high protect rate a...	0.313333	0.020000	Positive
4	1.362166e+18	3	lab studi suggest pfizer moderna vaccin protec...	0.000000	0.000000	Neutral
...	...	...	...	...	...	...
5995	1.370975e+18	2	swamy39 dr swamy39 jee mani peopl like get cov...	0.000000	0.000000	Neutral
5996	1.379827e+18	3	happi fulli vaccin covid 19 im readi serv vacc...	0.000000	0.000000	Neutral
5997	1.384789e+18	2	serum institut india announc cost covishield v...	0.000000	0.000000	Neutral
5998	1.382355e+18	1	___batshitcrazi barrowfordhead bectulli mine y...	0.000000	0.000000	Neutral
5999	1.380051e+18	2	smart sympathi attend oblong noth educ feedbac...	0.642857	0.214286	Positive

6000 rows × 6 columns

In [22]:

```
#Plot the polarity and subjectivity
plt.figure(figsize = (8,6))
for i in range(0,df.shape[0]):
    plt.scatter(df['Polarity'][i],df['Subjectivity'][i],color = 'Blue')
plt.title('Sentiment Analysis')
plt.xlabel('Polarity')
plt.ylabel('Subjectivity')
plt.show()
```



In [23]:

```
#Get the percentage of positive tweets
pos_tweets = df[df.Analysis == 'Positive']
pos_tweets = pos_tweets['tweet_text']
round((pos_tweets.shape[0]/df.shape[0])*100,2)
```

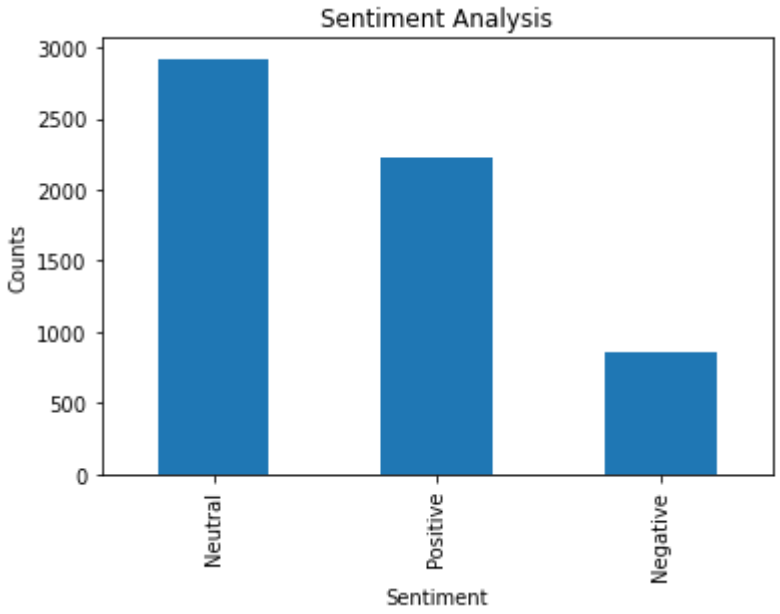
Out[23]:

37.03

In [24]:

```
#Show the value counts
df['Analysis'].value_counts()
#Plot and visualize the counts

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



In [28]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(ngram_range=(1,2))
X_cv=cv.fit_transform(df['tweet_text']).toarray()
X=X_cv
y=df['label']
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=12
3)
RF=RandomForestClassifier()
RF_model=RF.fit(X_train,y_train)
print("Train Accuracy",RF_model.score(X_train,y_train))
print("Test Accuracy",RF_model.score(X_test,y_test))
```

```
Train Accuracy 0.9997916666666666
Test Accuracy 0.715
```

In [ ]:

```

```