SENTIMENT ANALYSIS FOR MARKETING

TEAM MEMBER

NAME- HARISH S M

PHASE 5

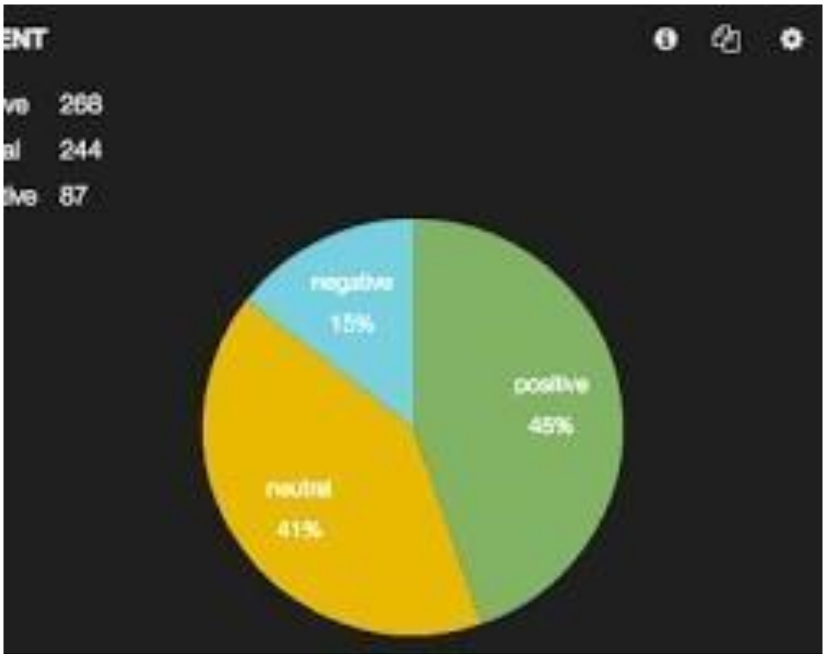
PROJECT DOCUMENTATION AND SUBMISSION

TITLE-**SENTIMENT ANALYSIS FOR MARKETING**

Two important aspects in sentiment analysis for marketing are:

Understanding Customer Emotions:

Sentiment analysis helps marketers gauge customer emotions towards products, services, or marketing campaigns. It's crucial to not only identify whether a sentiment is positive, negative, or neutral but also to comprehend the underlying emotions. Understanding the emotional tone, such as happiness, frustration, excitement, or disappointment, provides nuanced insights.



Contextual Analysis:

Context is paramount in sentiment analysis for marketing. The same phrase or word can carry different sentiments based on the context it's used in. Analyzing the context helps in accurate sentiment interpretation. For instance, the phrase "small size" might be positive when referring to portable gadgets but negative when describing a product meant to be large. Contextual analysis involves understanding the industry-specific jargon, sarcasm, idiomatic expressions, and cultural nuances.

- DATASET:

Dataset Link:https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

tweet_id	airline_sentiment	airline_sentiment...	negativereason	negativereaso
567588279b 570310600b	negative 63% neutral 21% Other (2363) 16%	0.34 – 1	[null] 37% Customer Service ... 20% Other (6268) 43%	0
570306133677760513	neutral	1.0		
570301130888122368	positive	0.3486		0.0
570301083672813571	neutral	0.6837		
570301031407624196	negative	1.0	Bad Flight	0.7033
570300817074462722	negative	1.0	Can't Tell	1.0
570300767074181121	negative	1.0	Can't Tell	0.6842
570300616901320704	positive	0.6745		0.0
570300248553349120	neutral	0.634		
570299953286942721	positive	0.6559		
570295459631263746	positive	1.0		
570294189143031808	neutral	0.6769		0.0
570289724453216256	positive	1.0		
570289584061480960	positive	1.0		
570287408438120448	positive	0.6451		
570285904809598977	positive	1.0		
570282469121007616	negative	0.6842	Late Flight	0.3684
570277724385734656	positive	1.0		
570276917301137409	negative	1.0	Bad Flight	1.0
570270684619923457	positive	1.0		
570267956648792064	positive	1.0		
570265883513384960	negative	0.6705	Can't Tell	0.3614
570264145116819457	positive	1.0		
570259420287868928	positive	1.0		
570258822297579520	neutral	1.0		
570256553502068736	negative	1.0	Customer Service Issue	0.3557
570249102404923392	negative	1.0	Customer Service Issue	1.0
570239632807370753	negative	1.0	Can't Tell	0.6614
570217831557677057	neutral	0.6854		
570207886493782019	negative	1.0	Bad Flight	1.0
570124596180955136	neutral	0.615		0.0
570114021854212096	negative	1.0	Flight Booking Problems	1.0
570094701371469825	neutral	1.0		
570088404156698625	negative	1.0	Customer Service Issue	1.0
570084582780899328	negative	1.0	Customer Service Issue	1.0
570076792993611776	positive	1.0		
570051991277342720	neutral	0.6207		
570051381534396416	positive	1.0		

Sentiment analysis in marketing using NLP techniques can provide valuable insights into customer opinions and reactions. Here's how you can approach it:

1. ****Data Collection:**** Gather customer feedback, reviews, social media comments, and any other textual data related to your products or services.

2. ****Text Preprocessing:**** Clean and preprocess the text data. This step involves removing special characters, stopwords, and performing tasks like tokenization and lemmatization to prepare the text for analysis.

3. ****Sentiment Analysis:**** Utilize NLP techniques and sentiment analysis algorithms to determine the sentiment of the text. There are various

methods, including rule-based approaches and machine learning-based models, such as Support Vector Machines (SVM) or Recurrent Neural Networks (RNNs).

4. **Aspect-Based Sentiment Analysis:** For more detailed insights, perform aspect-based sentiment analysis. This technique breaks down the text into aspects (features or attributes) and analyzes the sentiment associated with each aspect. This can be incredibly useful for product reviews where customers might comment on different features.

5. **Entity Recognition:** Identify entities mentioned in the text, such as product names, brands, or people. Understanding which entities are associated with positive or negative sentiments can provide targeted insights.

6. **Visualization:** Visualize the sentiment data using charts or graphs. Visualization can make complex data more understandable and help in identifying patterns and trends.

7. **Feedback Analysis:** Categorize the sentiment into different categories (positive, negative, neutral) and analyze the volume of feedback in each category. Additionally, look for common themes or keywords in negative feedback, which can help in identifying areas for improvement.

8. **Feedback Loop:** Use the insights gained from sentiment analysis to improve marketing strategies, customer service, or product development. Address negative sentiments and

leverage positive sentiments in marketing campaigns.

Importing the libraries and loading the data

Import numpy as np # linear algebra
Import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
Import matplotlib.pyplot as plt

Input data files are available in the "../input/" directory.
For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory

Import os
Print(os.listdir("../input"))
Import re
Import nltk
From nltk.corpus import stopwords

```
From sklearn.model_selection import
train_test_split
From mlxtend.plotting import
plot_confusion_matrix
From sklearn.tree import DecisionTreeClassifier
From sklearn.ensemble import
RandomForestClassifier
From sklearn.metrics import
accuracy_score,confusion_matrix,classification_rep
ort

Df= pd.read_csv("../input/Tweets.csv")
Df.head()
Tweet_id    airline_sentiment
    airline_sentiment_confidence    negativereason
    negativereason_confidence airline
    airline_sentiment_gold  name
    negativereason_gold     retweet_count   text
    tweet_coord     tweet_created   tweet_location
    user_timezone
```

0 570306133677760513 neutral 1.0000 NaN
NaN Virgin America NaN cairdin NaN
0 @VirginAmerica What @dhepburn said.
NaN 2015-02-24 11:35:52 -0800 NaN
Eastern Time (US & Canada)
1 570301130888122368 positive 0.3486 NaN
0.0000 Virgin America NaN jnardino
NaN 0 @VirginAmerica plus you've added
commercials t… NaN 2015-02-24 11:15:59 -0800
NaN Pacific Time (US & Canada)
2 570301083672813571 neutral 0.6837 NaN
NaN Virgin America NaN yvonnalynn
NaN 0 @VirginAmerica I didn't today…
Must mean I n… NaN 2015-02-24 11:15:48 -0800
Lets Play Central Time (US & Canada)
3 570301031407624196 negative 1.0000 Bad
Flight 0.7033 Virgin America NaN jnardino
NaN 0 @VirginAmerica it's really
aggressive to blast… NaN 2015-02-24 11:15:36 -
0800 NaN Pacific Time (US & Canada)

4 570300817074462722 negative 1.0000
Can't Tell 1.0000

Data Preprocessing

The first step should be to check the shape of the dataframe and then check the number of null values in each column.

In this way we can get an idea of the redundant columns in the data frame depending on which columns have the highest number of null values.

Print("Shape of the dataframe is",df.shape)
Print("The number of nulls in each column are \n", df.isna().sum())
Shape of the dataframe is (14640, 15)
The number of nulls in each column are
Tweet_id 0
Airline_sentiment 0
Airline_sentiment_confidence 0
Negativereason 5462

Negativereason_confidence 4118
Airline 0
Airline_sentiment_gold 14600
Name 0
Negativereason_gold 14608
Retweet_count 0
Text 0
Tweet_coord 13621
Tweet_created 0
Tweet_location 4733
User_timezone 4820
Dtype: int64
To get a better idea, lets calculate the percentage of nulls or NA values in each column

```
Print("Percentage null or na values in df")
((df.isnull() | df.isna()).sum() * 100 /
df.index.size).round(2)
```

Percentage null or na values in df
Tweet_id 0.00
Airline_sentiment 0.00

Airline_sentiment_confidence 0.00
Negativereason 37.31
Negativereason_confidence 28.13
Airline 0.00
Airline_sentiment_gold 99.73
Name 0.00
Negativereason_gold 99.78
Retweet_count 0.00
Text 0.00
Tweet_coord 93.04
Tweet_created 0.00
Tweet_location 32.33
User_timezone 32.92
Dtype: float64

To get a better idea, lets calculate the percentage of nulls or NA values in each column

```
Print("Percentage null or na values in df")
((df.isnull() | df.isna()).sum() * 100 /
df.index.size).round(2)
```

Percentage null or na values in df

Tweet_id	0.00
Airline_sentiment	0.00
Airline_sentiment_confidence	0.00
Negativereason	37.31
Negativereason_confidence	28.13
Airline	0.00
Airline_sentiment_gold	99.73
Name	0.00
Negativereason_gold	99.78
Retweet_count	0.00
Text	0.00
Tweet_coord	93.04
Tweet_created	0.00
Tweet_location	32.33
User_timezone	32.92

Dtype: float64

Tweet_coord , airline_sentiment_gold, negativereason_gold have more than 90% missing data. It will be better to delete these columns as they will not provide any constructive information.

Del df['tweet_coord']
Del df['airline_sentiment_gold']
Del df['negativereason_gold']
Df.head()

Airline sentiments for each airline

Firstly lets calculate the total number of tweets for each airline
Then, we are going to get the barplots for each airline with respect to sentiments of tweets (positive,negative or neutral).
This will give us a clearer idea about the airline sentiments , airlines relationship.
Print("Total number of tweets for each airline \n ",df.groupby('airline')['airline_sentiment'].count().sort_values(ascending=False))
Airlines= ['US Airways','United','American','Southwest','Delta','Virgin America']
Plt.figure(1,figsize=(12, 12))

```
For I in airlines:
    Indices= airlines.index(i)
    Plt.subplot(2,3,indices+1)
    New_df=df[df['airline']==i]

Count=new_df['airline_sentiment'].value_counts()
    Index = [1,2,3]
    Plt.bar(Index,count, color=['red', 'green', 'blue'])
    Plt.xticks(Index,['negative','neutral','positive'])
    Plt.ylabel('Mood Count')
    Plt.xlabel('Mood')
    Plt.title('Count of Moods of '+i)
```

Total number of tweets for each airline

```
  Airline
United            3822
US Airways        2913
American          2759
Southwest         2420
Delta             2222
Virgin America     504
Name: airline_sentiment, dtype: int64
```

Airline sentiments for each airline¶
Firstly lets calculate the total number of tweets for each airline
Then, we are going to get the barplots for each airline with respect to sentiments of tweets (positive,negative or neutral).
This will give us a clearer idea about the airline sentiments , airlines relationship.

```
Print("Total number of tweets for each airline \n
",df.groupby('airline')['airline_sentiment'].count().so
rt_values(ascending=False))
Airlines= ['US
Airways','United','American','Southwest','Delta','Virgi
n America']
Plt.figure(1,figsize=(12, 12))
For I in airlines:
    Indices= airlines.index(i)
    Plt.subplot(2,3,indices+1)
    New_df=df[df['airline']==i]

Count=new_df['airline_sentiment'].value_counts()
```

Index = [1,2,3]
Plt.bar(Index,count, color=['red', 'green', 'blue'])
Plt.xticks(Index,['negative','neutral','positive'])
Plt.ylabel('Mood Count')
Plt.xlabel('Mood')
Plt.title('Count of Moods of '+i)

Total number of tweets for each airline

Airline	
United	3822
US Airways	2913
American	2759
Southwest	2420
Delta	2222
Virgin America	504

Name: airline_sentiment, dtype: int64

Most used words in Positive and Negative tweets¶

From wordcloud import WordCloud,STOPWORDS

The goal is to firstly get an idea of the most frequent words in negative tweets.

Get idea about most frequent words in positive tweets.
Wordcloud for Negative sentiments of tweets
Wordcloud is a great tool for visualizing nlp data.
The larger the words in the wordcloud image , the more is the frequency of that word in our text data.

```
New_df=df[df['airline_sentiment']=='negative']
Words = ' '.join(new_df['text'])
Cleaned_word = " ".join([word for word in
words.split()
                        If 'http' not in word
                           And not word.startswith('@')
                           And word != 'RT'
                        ])
Wordcloud = WordCloud(stopwords=STOPWORDS,
                      Background_color='black',
                      Width=3000,
                      Height=2500
                     ).generate(cleaned_word)
Plt.figure(1,figsize=(12, 12))
```

```
Plt.imshow(wordcloud)
Plt.axis('off')
Plt.show()
```

Wordcloud for positive reasons

The code for getting positive sentiments is completely same with the one for negative sentiments. Just replace negative with positive in the first line. Easy, right!

```
New_df=df[df['airline_sentiment']=='positive']
Words = ' '.join(new_df['text'])
Cleaned_word = " ".join([word for word in
words.split()
                    If 'http' not in word
                       And not word.startswith('@')
                       And word != 'RT'
                    ])
Wordcloud = WordCloud(stopwords=STOPWORDS,
                Background_color='black',
                Width=3000,
```

```
                 Height=2500
                ).generate(cleaned_word)
Plt.figure(1,figsize=(12, 12))
Plt.imshow(wordcloud)
Plt.axis('off')
Plt.show()
```

Calculate highest frequency words in positive tweets

```
Def freq(str):

    # break the string into list of words
    Str = str.split()
    Str2 = []

    # loop till string values present in list str
    For I in str:

        # checking for the duplicacy
        If I not in str2:
```

```
        # insert value in str2
        Str2.append(i)

    For I in range(0, len(str2)):
      If(str.count(str2[i])>50):
        Print('Frequency of', str2[i], 'is :',
str.count(str2[i]))

Print(freq(cleaned_word))
```

Frequency of to is : 923
Frequency of the is : 924
Frequency of time is : 59
Frequency of I is : 574
Frequency of fly is : 54
Frequency of this is : 143
Frequency of ☺ is : 96
Frequency of it is : 166
Frequency of was is : 226
Frequency of and is : 416
Frequency of an is : 74
Frequency of good is : 75

Frequency of have is : 124
Frequency of Thank is : 231
Frequency of at is : 178
Frequency of thanks is : 218
Frequency of get is : 111
Frequency of me is : 196
Frequency of service is : 100
Frequency of you! Is : 129
Frequency of Thanks is : 177
Frequency of as is : 57
Frequency of thank is : 204
Frequency of will is : 85
Frequency of our is : 64
Frequency of up is : 66
Frequency of guys is : 76
Frequency of got is : 85
Frequency of made is : 55
None

Words like Thanks, best, customer , love, flying , good are understandably present in the most frequent words of positive tweets.

However, other than these, most of the words are stop words and need to be filtered. We will do so later.

Lets try and visualize the reasons for negative tweets first !!

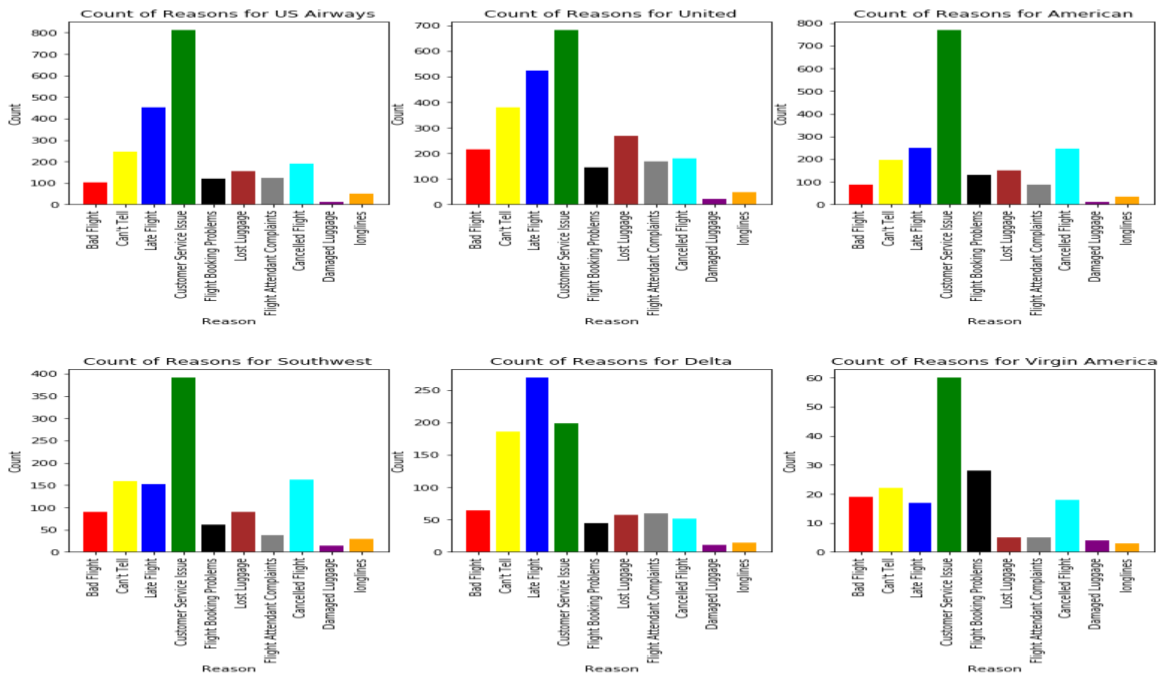
What are the reasons for negative sentimental tweets for each airline ?

We will explore the negative reason column of our dataframe to extract conclusions about negative sentiments in the tweets by the customers

```
#get the number of negative reasons
Df['negativereason'].nunique()

NR_Count=dict(df['negativereason'].value_counts(s
ort=False))
Def NR_Count(Airline):
   If Airline=='All':
```

```
        A=df
    Else:
        A=df[df['airline']==Airline]
    Count=dict(a['negativereason'].value_counts())

Unique_reason=list(df['negativereason'].unique())
    Unique_reason=[x for x in Unique_reason if str(x)
!= 'nan']

Reason_frame=pd.DataFrame({'Reasons':Unique_re
ason})

Reason_frame['count']=Reason_frame['Reasons'].a
pply(lambda x: count[x])
    Return Reason_frame
Def plot_reason(Airline):

    A=NR_Count(Airline)
    Count=a['count']
    Index = range(1,(len(a)+1))
```

```
    Plt.bar(Index,count,
color=['red','yellow','blue','green','black','brown','gray
','cyan','purple','orange'])
    Plt.xticks(Index,a['Reasons'],rotation=90)
    Plt.ylabel('Count')
    Plt.xlabel('Reason')
    Plt.title('Count of Reasons for '+Airline)

Plot_reason('All')
Plt.figure(2,figsize=(13, 13))
For I in airlines:
    Indices= airlines.index(i)
    Plt.subplot(2,3,indices+1)
    Plt.subplots_adjust(hspace=0.9)
    Plot_reason(i)
```



Is there a relationship between negative sentiments and date

```
Date = df.reset_index()
#convert the Date column to pandas
datetime
Date.tweet_created =
pd.to_datetime(date.tweet_created)
```

#Reduce the dates in the date column to only the date and no time stamp using the ‘dt.date’ method
Date.tweet_created =
date.tweet_created.dt.date
Date.tweet_created.head()
Df = date
Day_df =
df.groupby([‘tweet_created’,’airline’,’airline_sentiment’]).size()
# day_df = day_df.reset_index()
Day_df

Our next step will be to plot this and get better visualization for negative tweets.

```
Day_df = day_df.loc(axis=0)[:,:,'negative']

#groupby and plot data
Ax2 =
day_df.groupby(['tweet_created','airline'
]).sum().unstack().plot(kind = 'bar',
color=['red', 'green',
'blue','yellow','purple','orange'], figsize =
(15,6), rot = 70)
Labels =
['American','Delta','Southwest','US
Airways','United','Virgin America']
Ax2.legend(labels = labels)
Ax2.set_xlabel('Date')
Ax2.set_ylabel('Negative Tweets')
Plt.show()
```

Preprocessing the tweet text data

Now, we will clean the tweet text data and apply classification algorithms on it

```
Def tweet_to_words(tweet):
    Letters_only = re.sub("[^a-zA-Z]", " ",tweet)
    Words = letters_only.lower().split()
    Stops = set(stopwords.words("english"))
    Meaningful_words = [w for w in words if not w in stops]
    Return( " ".join( meaningful_words ))
```

Df['clean_tweet']=df['text'].apply(lambda x: tweet_to_words(x))

The data is split in the standard 80,20 ratio.

```
Train,test =
train_test_split(df,test_size=0.2,random_state=42)
Train_clean_tweet=[]
For tweet in train['clean_tweet']:
   Train_clean_tweet.append(tweet)
Test_clean_tweet=[] for
For tweet in test['clean_tweet']:
   Test_clean_tweet.append(tweet)
From sklearn.feature_extraction.text
import CountVectorizer
V = CountVectorizer(analyzer = "word")
```

Train_features=
v.fit_transform(train_clean_tweet)
Test_features=v.transform(test_clean_tw
eet)

Prediciting sentiments from tweet text data

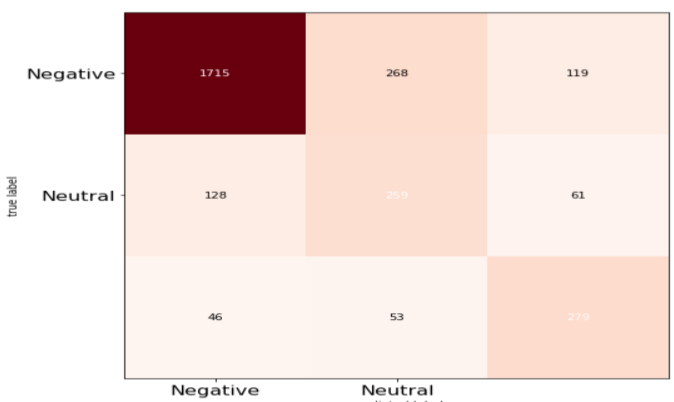
Decision Tree Classifier
Random Forest Classifier
Classifiers = [
DecisionTreeClassifier(),

RandomForestClassifier(n_estimators=2
00)]
Dense_features=train_features.toarray()

```
Dense_test= test_features.toarray()
Accuracy=[]
Model=[]
For classifier in Classifiers:
    Try:
        Fit =
classifier.fit(train_features,train['airline_s
entiment'])
        Pred = fit.predict(test_features)
    Except Exception:
        Fit =
classifier.fit(dense_features,train['airline
_sentiment'])
        Pred = fit.predict(dense_test)
    Accuracy =
accuracy_score(pred,test['airline_sentim
ent'])
```

```
    Accuracy.append(accuracy)

Model.append(classifier.__class__.__na
me__)
    Print('Accuracy of
'+classifier.__class__.__name__+'is
'+str(accuracy))

Print(classification_report(pred,test['airli
ne_sentiment']))
    Cm=confusion_matrix(pred ,
test['airline_sentiment'])
    Plt.figure()

Plot_confusion_matrix(cm,figsize=(12,8),
hide_ticks=True,cmap=plt.cm.Reds)
```

```
    Plt.xticks(range(2), [‘Negative’,
‘Neutral’, ‘Positive’],
fontsize=16,color=’black’)
    Plt.yticks(range(2), [‘Negative’,
‘Neutral’, ‘Positive’], fontsize=16)
    Plt.show()
```



As we you can see above we have plotted the confusion matrix for predicted sentiments and actual sentiments (negative,neutral and positive)

Random Forest Classifier gives us the best accuracy score, precision scores according to the classification report. The confusion matrix shows the TP,TN,FP,FN for all the 3 sentiments(negative,neutral and positive),Here also Random Forest Classifier gives better results than the Decision Tree Classifier.

…