**AIM: SENTIMENT ANALYSIS & TEXT CLASSIFICATION**

**THEORY:**

**SENTIMENT ANALYSIS:**

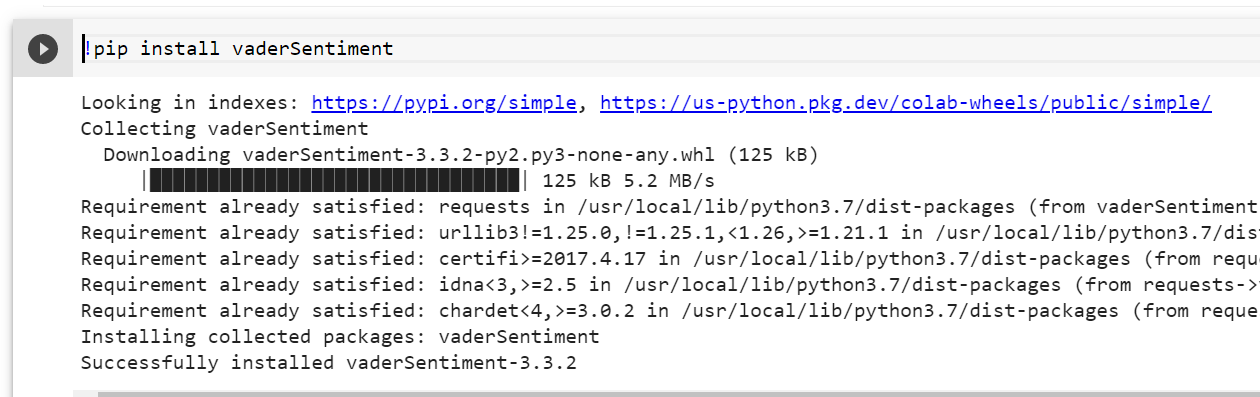
Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.

**TEXT CLASSFICATION:**

Text classification also known as text tagging or text categorization is the process of categorizing text into organized groups. By using Natural Language Processing (NLP), text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content.

1. **SENTIMENT ANALYSIS USING VADERSENTIMENT LIB: [ENGLISH]**

**SOURCE CODE:**



from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

sid\_obj = SentimentIntensityAnalyzer()

# polarity\_scores method of SentimentIntensityAnalyzer

# object gives a sentiment dictionary.

# which contains pos, neg, neu, and compound scores.

def senntiment\_sentences(sentences):

  sentiment\_dict = sid\_obj.polarity\_scores(sentence)

  print("Overall sentiment dictionary is : ", sentiment\_dict)

  print("sentence was rated as ", sentiment\_dict['neg']\*100, "% Negative")

  print("sentence was rated as ", sentiment\_dict['neu']\*100, "% Neutral")

  print("sentence was rated as ", sentiment\_dict['pos']\*100, "% Positive")

  print("Sentence Overall Rated As", end = " ")

    # decide sentiment as positive, negative and neutral

  if sentiment\_dict['compound'] >= 0.05 :

    print("Positive")

  elif sentiment\_dict['compound'] <= - 0.05 :

    print("Negative")

  else:

    print("Neutral")

print("\n1st statement :")

sentence = "Geeks For Geeks is the best portal for the computer science engineering students."

senntiment\_sentences(sentence)

print("\n2nd Statement :")

sentence = "study is going on as usual"

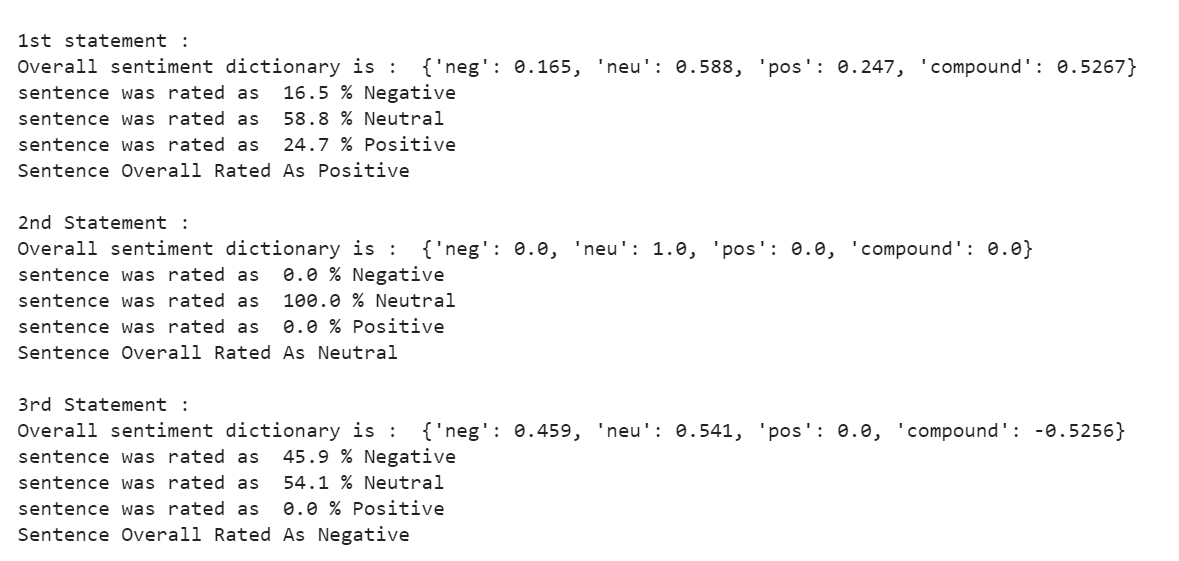
senntiment\_sentences(sentence)

print("\n3rd Statement :")

sentence = "I am very sad today."

senntiment\_sentences(sentence)

**OUTPUT:**



1. **SENTIMENT ANALYSIS ON US AIRELINE REVIEWS:**

**SOURCE CODE:**

import pandas as pd

import matplotlib.pyplot as plt

from tensorflow.keras.preprocessing.text import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad\_sequences

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM,Dense, Dropout, SpatialDropout1D

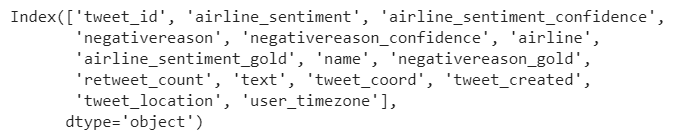
from tensorflow.keras.layers import Embedding

df = pd.read\_csv("./Tweets.csv")

df.head()



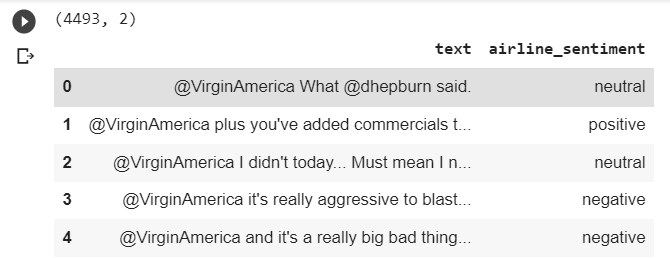
df.columns



tweet\_df = df[['text','airline\_sentiment']]

print(tweet\_df.shape)

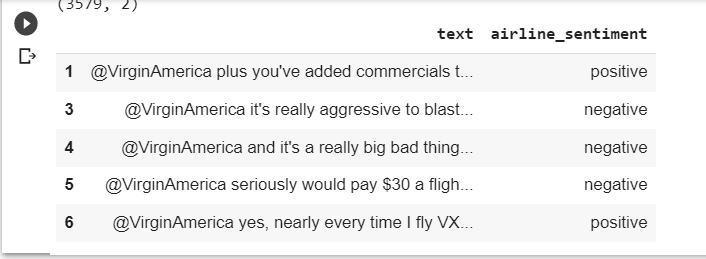
tweet\_df.head(5)



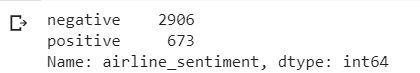
tweet\_df = tweet\_df[tweet\_df['airline\_sentiment'] != 'neutral']

print(tweet\_df.shape)

tweet\_df.head(5)



tweet\_df["airline\_sentiment"].value\_counts()



sentiment\_label = tweet\_df.airline\_sentiment.factorize()

sentiment\_label



tweet = tweet\_df.text.values

tokenizer = Tokenizer(num\_words=5000)

tokenizer.fit\_on\_texts(tweet)

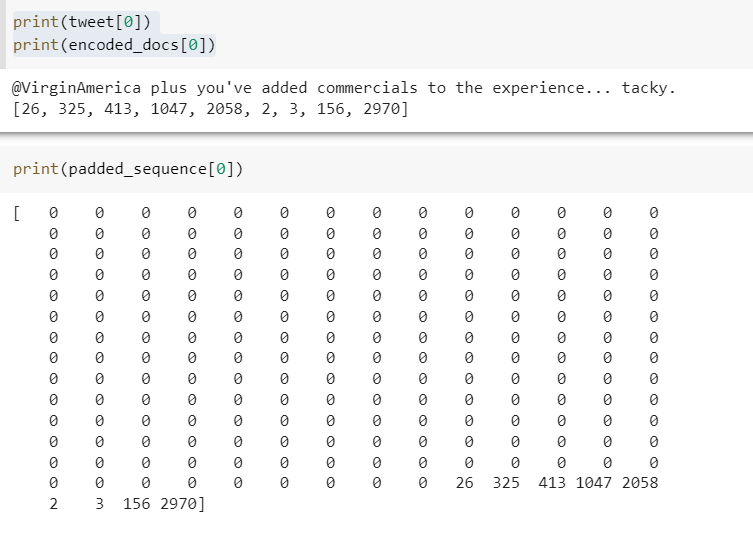
vocab\_size = len(tokenizer.word\_index) + 1

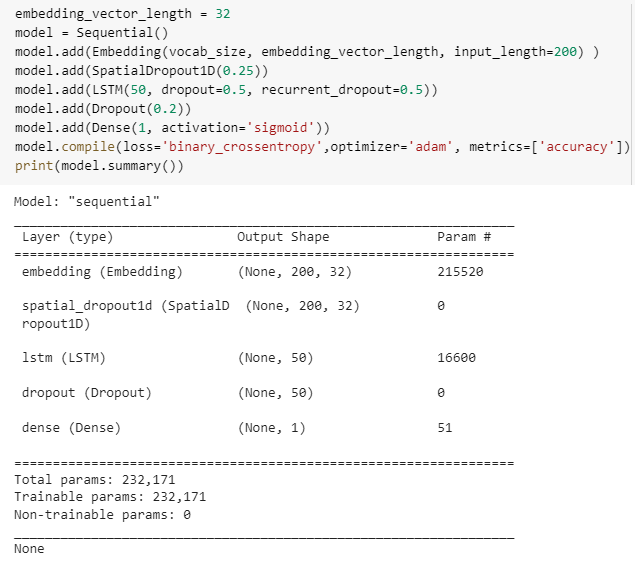
encoded\_docs = tokenizer.texts\_to\_sequences(tweet)

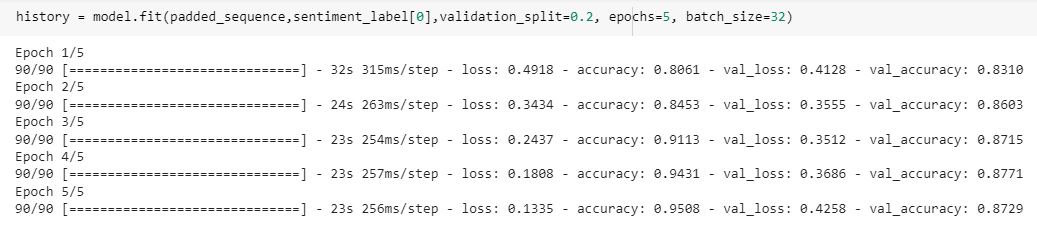
padded\_sequence = pad\_sequences(encoded\_docs, maxlen=200)

print(tokenizer.word\_index)

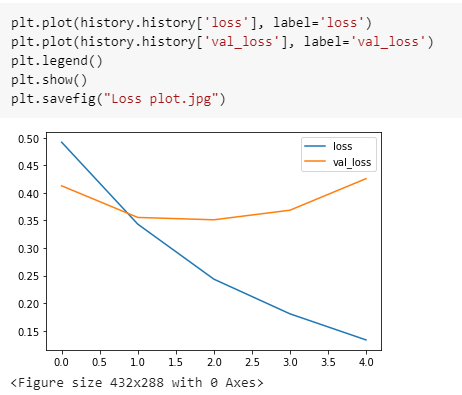


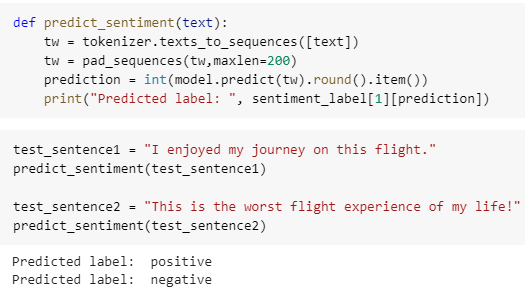






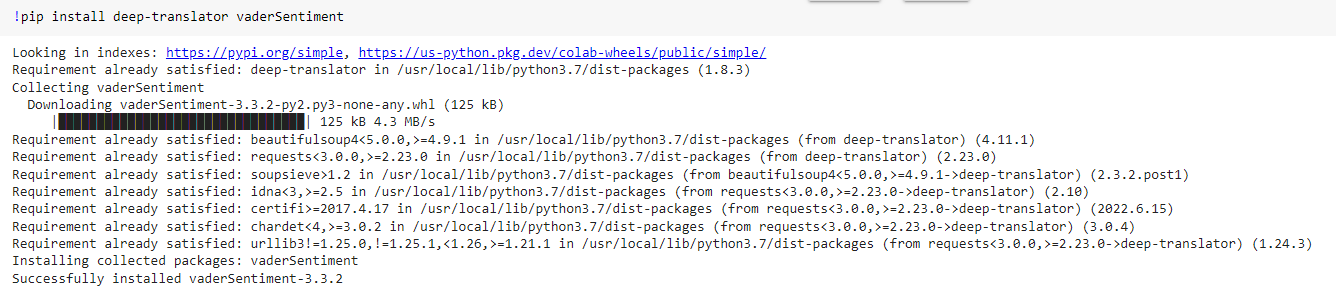






1. **SENTIMENT ANALYIS OF HINDI LANGUAGE:**

**SOURCE CODE:**



# codecs provides access to the internal Python codec registry

import codecs

# This is to translate the text from Hindi to English

from deep\_translator import GoogleTranslator

# This is to analyse the sentiment of text

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Read the hindi text into 'sentences'

sentences = ['गोवा की यात्रा बहुत अच्छी रही।',

             'समुद्र तट बहुत गर्म थे।',

             'मुझे समुद्र तट पर खेलने में बहुत मजा आया।',

             'मेरी बेटी बहुत गुस्से में थी।']

# with codecs.open('SampleHindiText.txt', encoding='utf-8') as f:

#   sentences = f.readlines()

for sentence in sentences:

  translated\_text = GoogleTranslator(source='auto', target='en').translate(sentence)

  #print(translated\_text)

  analyzer = SentimentIntensityAnalyzer()

  sentiment\_dict = analyzer.polarity\_scores(translated\_text)

  print("\nTranslated Sentence=",translated\_text, "\nDictionary=",sentiment\_dict)

  if sentiment\_dict['compound'] >= 0.05 :

      print("It is a Positive Sentence")

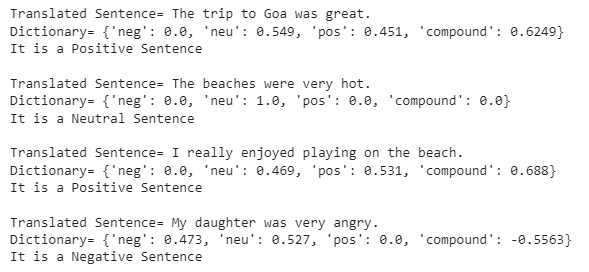
  elif sentiment\_dict['compound'] <= - 0.05 :

      print("It is a Negative Sentence")

  else :

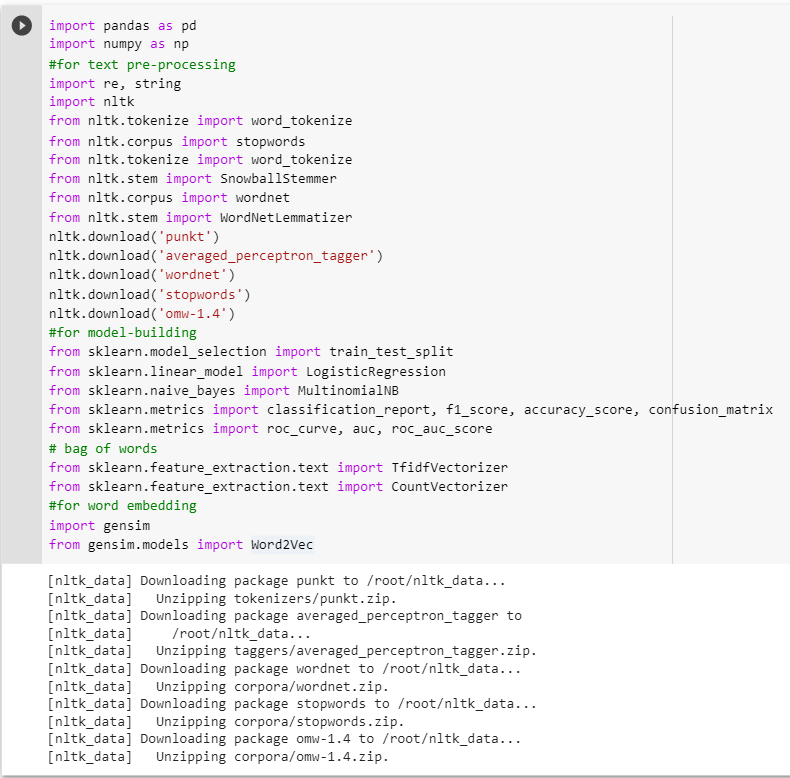
    print("It is a Neutral Sentence")

**OUTPUT:**

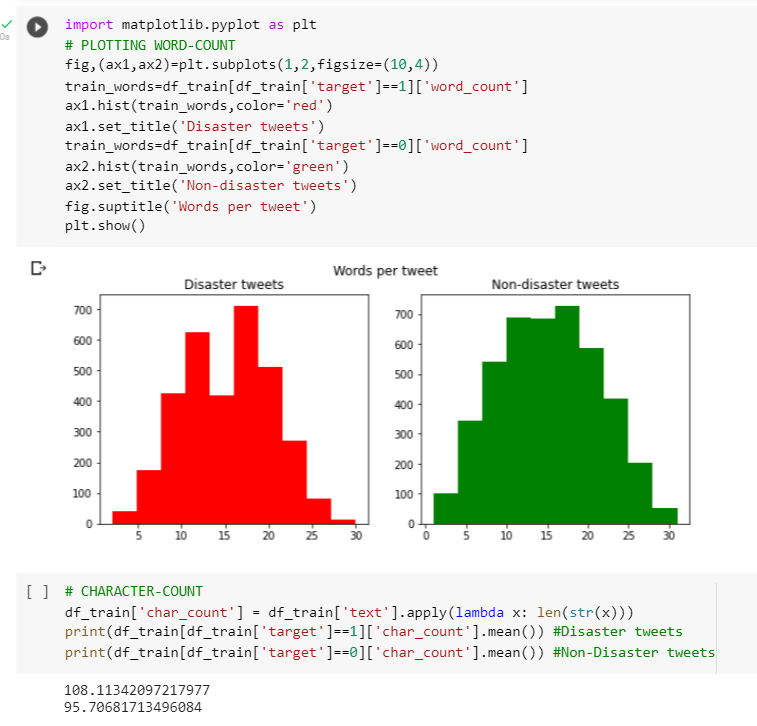


1. **TEXT CLASSFICATION IN NLP:**

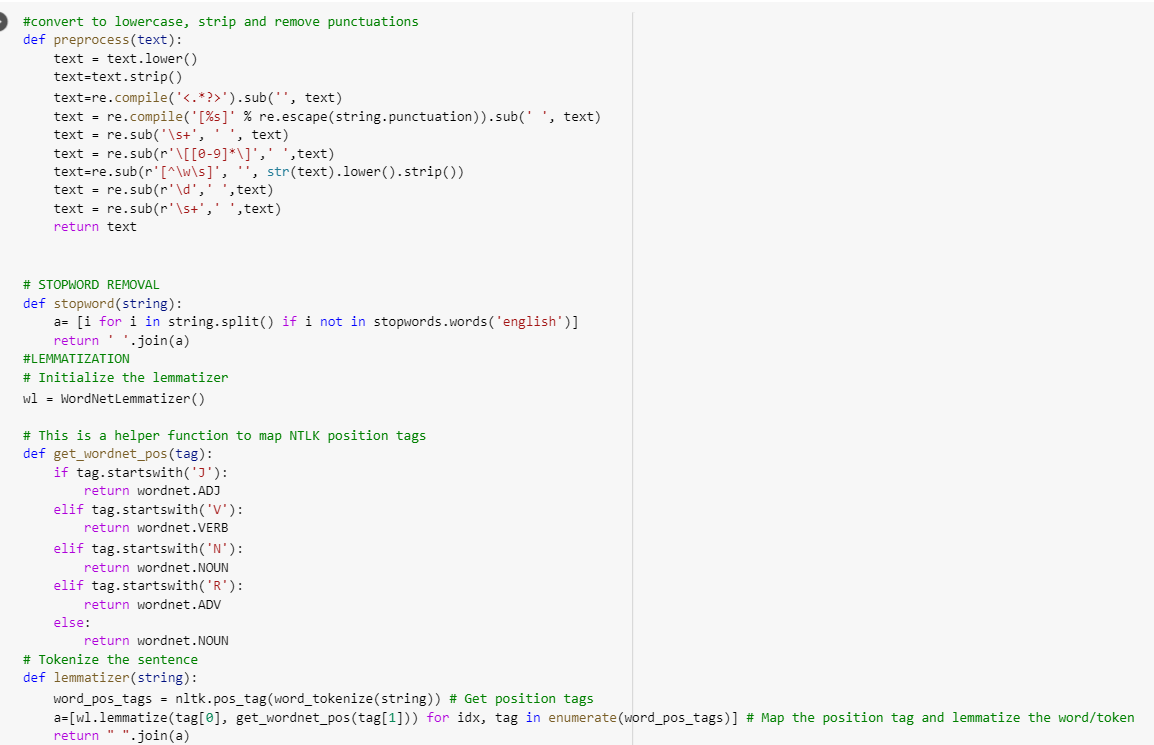
**SOURCE CODE:**

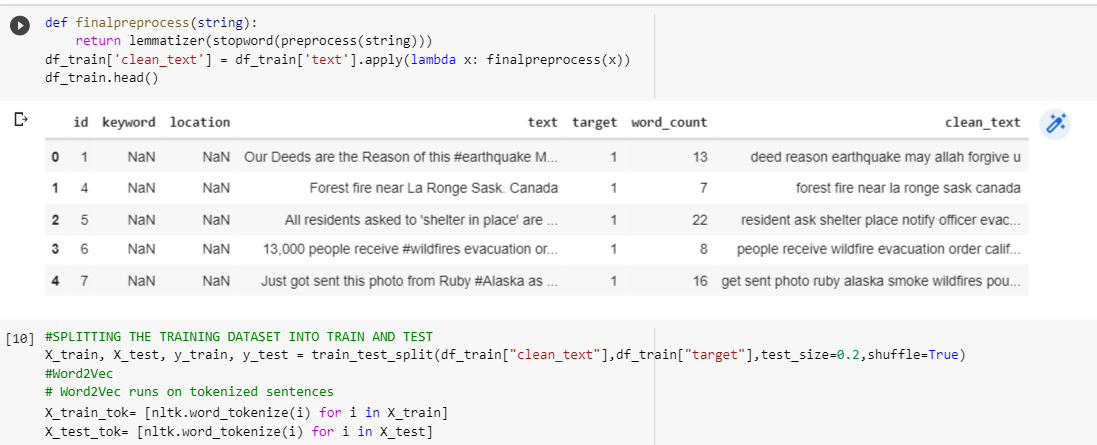


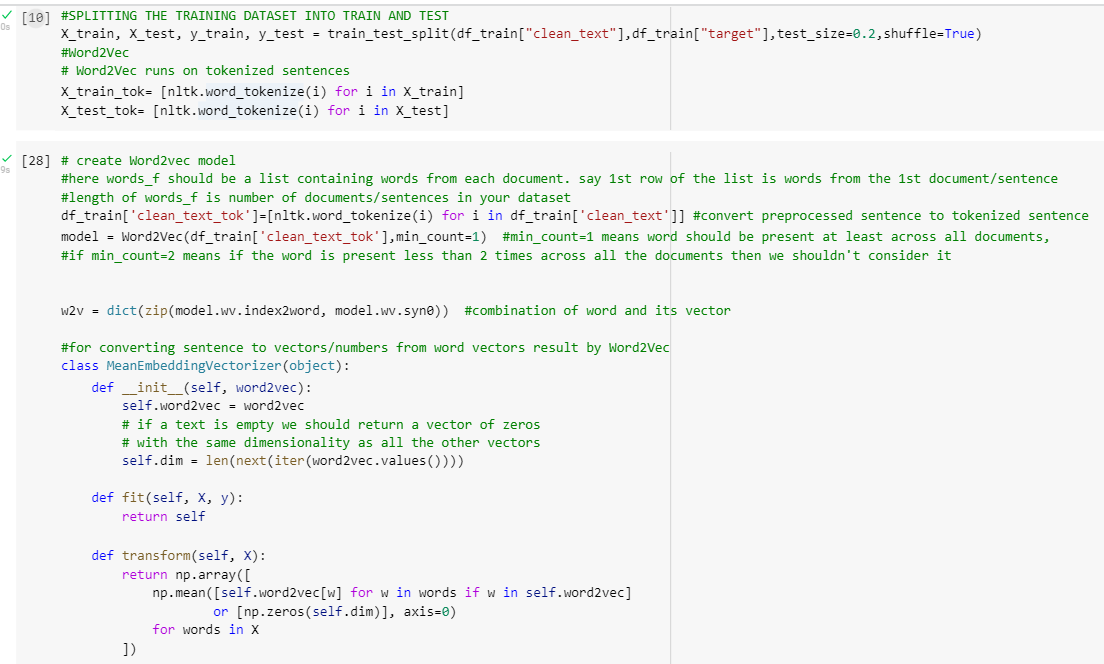


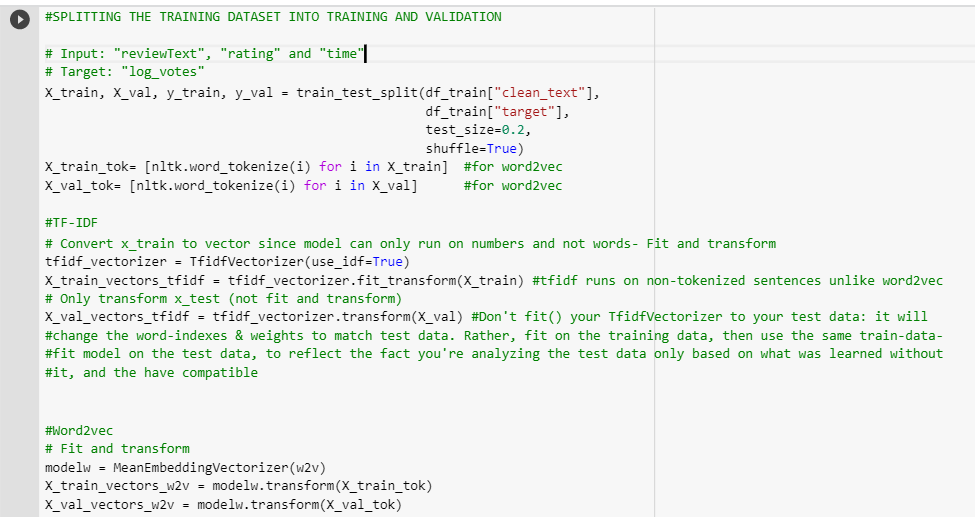


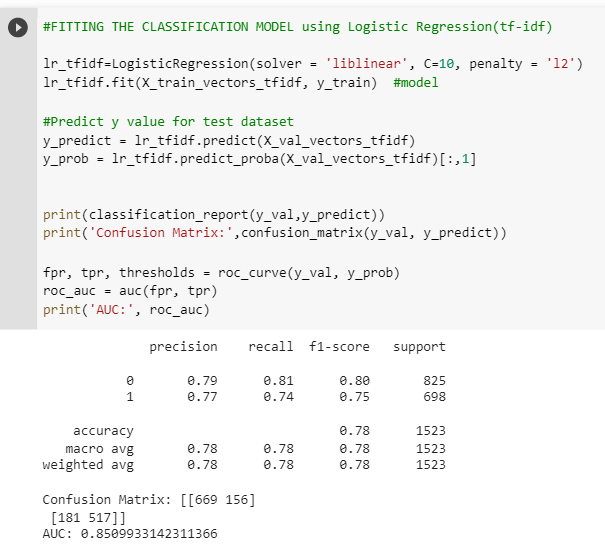


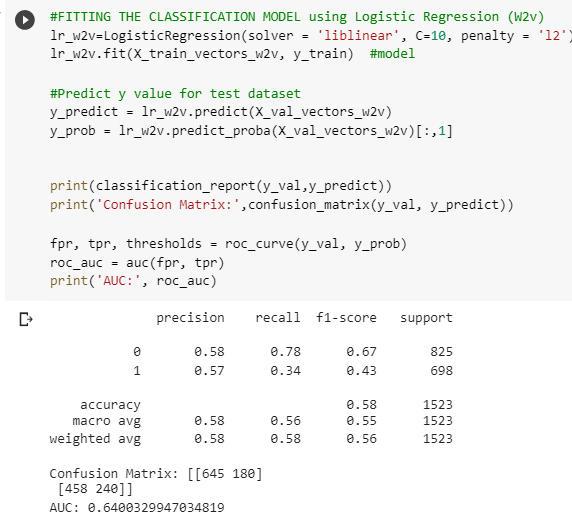


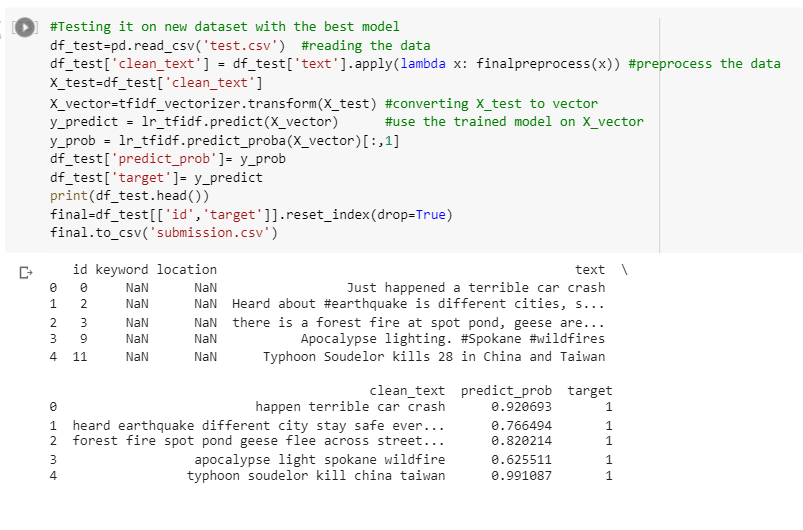






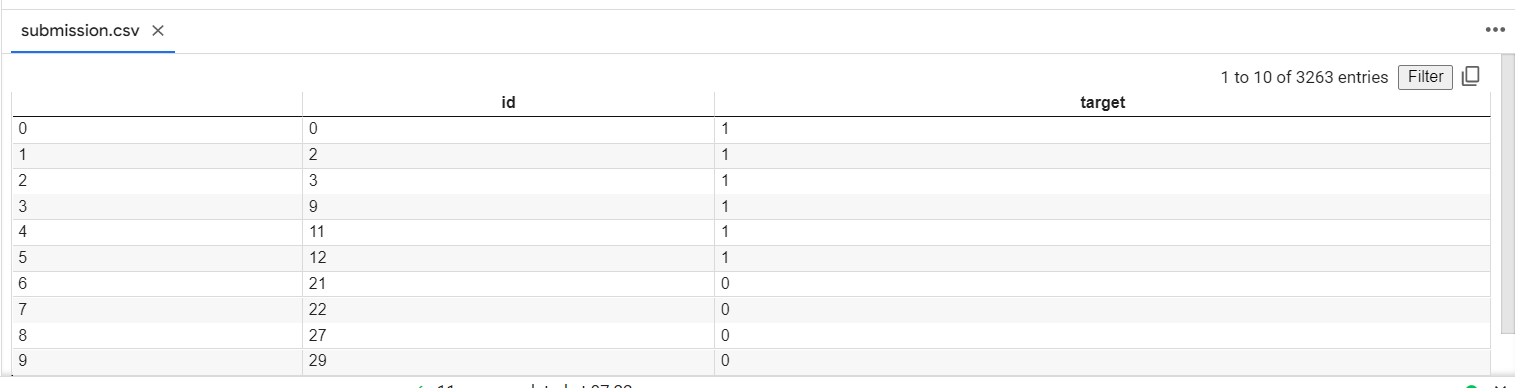






**OUTPUT:**

1. **REAL DISATER, 0- NO REAL DISATER [TWEETS]**



**CONCLUSION:**

From this tutorial, I have learned & implemented the sentiment analysis of text and text classification in python.