Identify the Sentiments

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations. Brands can use this data to measure the success of their products in an objective manner. In this challenge, you are provided with tweet data to predict sentiment on electronic products of netizens.

```
#Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import re
import pickle
%matplotlib inline
import subprocess
```

```
!pip install bert-serving-client
!pip install -U bert-serving-server[http]
```

```
    Collecting bert-serving-client
      Downloading https://files.pythonhosted.org/packages/1f/09/aae1405378a848b2e87769ad8
    Requirement already satisfied: pyzmq>=17.1.0 in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from
    Installing collected packages: bert-serving-client
    Successfully installed bert-serving-client-1.10.0
    Collecting bert-serving-server[http]
      Downloading https://files.pythonhosted.org/packages/b0/bd/cab677bbd0c5fb08b72e46837
         |████████████████████████████████| 71kB 3.9MB/s
    Requirement already satisfied, skipping upgrade: numpy in /usr/local/lib/python3.6/di
    Collecting GPUtil>=1.3.0
      Downloading https://files.pythonhosted.org/packages/ed/0e/5c61eedde9f6c87713e89d794
    Requirement already satisfied, skipping upgrade: six in /usr/local/lib/python3.6/dist
    Requirement already satisfied, skipping upgrade: pyzmq>=17.1.0 in /usr/local/lib/pyth
    Requirement already satisfied, skipping upgrade: termcolor>=1.1 in /usr/local/lib/pyt
    Collecting flask-json; extra == "http"
      Downloading https://files.pythonhosted.org/packages/6f/2d/4c21d98b11f3a206fabbdd965
    Collecting flask-cors; extra == "http"
      Downloading https://files.pythonhosted.org/packages/69/7f/d0aeaaafb5c3c76c8d2141dbe
    Requirement already satisfied, skipping upgrade: flask; extra == "http" in /usr/local
    Requirement already satisfied, skipping upgrade: bert-serving-client; extra == "http"
    Collecting flask-compress; extra == "http"
      Downloading https://files.pythonhosted.org/packages/de/eb/6bb0f8cb872167752eab8b06b
    Requirement already satisfied, skipping upgrade: Jinja2>=2.10.1 in /usr/local/lib/pyt
    Requirement already satisfied, skipping upgrade: click>=5.1 in /usr/local/lib/python3
    Requirement already satisfied, skipping upgrade: itsdangerous>=0.24 in /usr/local/lib
    Requirement already satisfied, skipping upgrade: Werkzeug>=0.15 in /usr/local/lib/pyt
    Collecting brotli
      Downloading https://files.pythonhosted.org/packages/b4/d3/7c98f05b7b9103e2f3a112ba4
         |████████████████████████████████| 358kB 16.6MB/s
    Requirement already satisfied, skipping upgrade: MarkupSafe>=0.23 in /usr/local/lib/p
    Building wheels for collected packages: GPUtil, flask-compress
      Building wheel for GPUtil (setup.py) ... done
      Created wheel for GPUtil: filename=GPUtil-1.4.0-cp36-none-any.whl size=7411 sha256=
      Stored in directory: /root/.cache/pip/wheels/3d/77/07/80562de4bb0786e5ea186911a2c8
      Building wheel for flask-compress (setup.py) ... done
      Created wheel for flask-compress: filename=Flask_Compress-1.7.0-cp36-none-any.whl s
      Stored in directory: /root/.cache/pip/wheels/1d/b7/18/2b88ed33c5ef53868d1bfb0d3f2f
    Successfully built GPUtil flask-compress
    Installing collected packages: GPUtil, flask-json, flask-cors, brotli, flask-compress
    Successfully installed GPUtil-1.4.0 bert-serving-server-1.10.0 brotli-1.0.9 flask-co
```

```
# Download and unzip the pre-trained model
!wget http://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip
!unzip uncased_L-12_H-768_A-12.zip
```

```
--2020-10-26 20:40:02--  http://storage.googleapis.com/bert_models/2018_10_18/uncase
Resolving storage.googleapis.com (storage.googleapis.com)... 74.125.195.128, 74.125.
Connecting to storage.googleapis.com (storage.googleapis.com)|74.125.195.128|:80... 
HTTP request sent, awaiting response... 200 OK
Length: 407727028 (389M) [application/zip]
Saving to: ‘uncased_L-12_H-768_A-12.zip’

uncased_L-12_H-768_ 100%[===================>] 388.84M   120MB/s    in 3.2s

2020-10-26 20:40:06 (120 MB/s) - ‘uncased_L-12_H-768_A-12.zip’ saved [407727028/4077

Archive:  uncased_L-12_H-768_A-12.zip
   creating: uncased_L-12_H-768_A-12/
  inflating: uncased_L-12_H-768_A-12/bert_model.ckpt.meta
  inflating: uncased_L-12_H-768_A-12/bert_model.ckpt.data-00000-of-00001
  inflating: uncased_L-12_H-768_A-12/vocab.txt
  inflating: uncased_L-12_H-768_A-12/bert_model.ckpt.index
  inflating: uncased_L-12_H-768_A-12/bert_config.json
```

```
!pwd
!ls
```

```
/content
sample_data  uncased_L-12_H-768_A-12  uncased_L-12_H-768_A-12.zip
```

```
# Start the BERT server
bert_command='bert-serving-start -model_dir /content/uncased_L-12_H-768_A-12'
process=subprocess.Popen(bert_command.split(),stdout=subprocess.PIPE)
```

```
!nohup bert-serving-start -model_dir=./uncased_L-12_H-768_A-12 > out.file 2>&1 &
```

```
%tensorflow_version 1.x
```

```
TensorFlow 1.x selected.
```

```
import tensorflow
print(tensorflow.__version__)
```

```
1.15.2
```

```
from bert_serving.client import BertClient
bc = BertClient()


# Load the training dataset
df = pd.read_csv('./train.csv')
print(df.head())

        id  label                                              tweet
     0   1      0  #fingerprint #Pregnancy Test https://goo.gl/h1...
     1   2      0  Finally a transparant silicon case ^^ Thanks t...
     2   3      0  We love this! Would you go? #talk #makememorie...
     3   4      0  I'm wired I know I'm George I was made that wa...
     4   5      1  What amazing service! Apple won't even talk to...


# Create a list of punctuation marks
#Ref google
puncts = [',', '.', '"', ':', ')', '(', '-', '!', '?', '|', ';', "'", '$', '&', '/', '[',
 '•', '_', '{', '}', '©', '^', '®', '`',  '<', '→', '°', '€', '™', '›',  '♥', '←', '×', '§
 '“', '★', '”', '–', '●', 'â', '►', '−', '¢', '²', '¬', '░', '¶', '↑', '±', '¿', '▾', '═',
 '▒', '：', '¼', '⊕', '▼', '▪', '†', '■', '’', '▀', '¨', '▄', '♫', '☆', 'é', '¯', '♦', '¤
 '·', ')', '↓', '、', '│', '（', '»', '，', '♪', '╩', '╚', '³', '・', '╦', '╣', '╔', '╗',


# Code to replace punctuations with whitespaces
def clean_text(x):
    x = str(x)
    for punct in puncts:
        if punct in x:
            x = x.replace(punct, ' ')
    return x



df.tweet=df.tweet.apply(lambda x:clean_text(x))


df.tweet=df.tweet.apply(lambda x:re.sub(r'http\S+','',x))


df.tweet=df.tweet.apply(lambda x:re.sub(r'@[\w]*','',x))


df.tweet=df.tweet.apply(lambda x:x.lower())


df.tweet=df.tweet.apply(lambda x:' '.join(x.split()))


df.head()
```

	id	label	tweet
0	1	0	fingerprint pregnancy test goo gl h1mfqv andro...
1	2	0	finally a transparant silicon case thanks to m...

```
# Compute embeddings for training tweets using Bert Client encode function
# The model returns 768-dimensional embeddings
tweets=df.tweet
tweet_list=[word for word in tweets]
embeddings=bc.encode(tweet_list)
```

```
    /usr/local/lib/python3.6/dist-packages/bert_serving/client/__init__.py:299: UserWarni
    here is what you can do:
    - disable the length-check by create a new "BertClient(check_length=False)" when you
    - or, start a new server with a larger "max_seq_len"
      '- or, start a new server with a larger "max_seq_len"' % self.length_limit)
```

```
print(embeddings.shape)
```

```
    (7920, 768)
```

```
# save bert_train_new for reuse as it would take a really long time for conversion
pickle_out=open('bert_train.pickle','wb')
pickle.dump(embeddings,pickle_out)
pickle_out.close()
```

```
#loading test dataset
df1=pd.read_csv('test.csv')
df1.head()
```

	id	tweet
0	7921	I hate the new #iphone upgrade. Won't let me d...
1	7922	currently shitting my fucking pants. #apple #i...
2	7923	I'd like to puts some CD-ROMS on my iPad, is t...
3	7924	My ipod is officially dead. I lost all my pict...
4	7925	Been fighting iTunes all night! I only want th...

```
df1.tweet=df1.tweet.apply(lambda x:clean_text(x))
df1.tweet=df1.tweet.apply(lambda x:re.sub(r'http\S+','',x))
df1.tweet=df1.tweet.apply(lambda x:re.sub(r'@[\w]*','',x))
df1.tweet=df1.tweet.apply(lambda x:x.lower())
df1.tweet=df1.tweet.apply(lambda x:' '.join(x.split()))
```

```
# Compute embeddings for training tweets using Bert Client encode function
# The model returns 768-dimensional embeddings
test_tweets=df1.tweet
test_tweet_list=[word for word in test_tweets]
```

```
test_embeddings=bc.encode(test_tweet_list)
```

```
print(test_embeddings.shape)
```

```
    (1953, 768)
```

```
# save bert_train_new for reuse as it would take a really long time for conversion
pickle_out1=open('bert_test.pickle','wb')
pickle.dump(test_embeddings,pickle_out1)
pickle_out1.close()
```

Spacy Word2Vec

```
pd.set_option('display.max_colwidth',100)
```

```
!python -m spacy download en_vectors_web_lg
```

```
    Collecting en_vectors_web_lg==2.1.0
      Downloading https://github.com/explosion/spacy-models/releases/download/en_vectors_
         |████████████████████████████████| 661.8MB 1.1MB/s
    Requirement already satisfied: spacy<3.0.0,>=2.1.0 in /usr/local/lib/python3.6/dist-p
    Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.6/dist-p
    Requirement already satisfied: thinc==7.4.0 in /usr/local/lib/python3.6/dist-packages
    Requirement already satisfied: catalogue<1.1.0,>=0.0.7 in /usr/local/lib/python3.6/di
    Requirement already satisfied: wasabi<1.1.0,>=0.4.0 in /usr/local/lib/python3.6/dist-
    Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.6/di
    Requirement already satisfied: srsly<1.1.0,>=1.0.2 in /usr/local/lib/python3.6/dist-p
    Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.6/dist
    Requirement already satisfied: plac<1.2.0,>=0.9.6 in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.6/
    Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.6/dist-p
    Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (
    Requirement already satisfied: blis<0.5.0,>=0.4.0 in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: importlib-metadata>=0.20; python_version < "3.8" in /u
    Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-pac
    Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/
    Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-packages (
    Building wheels for collected packages: en-vectors-web-lg
      Building wheel for en-vectors-web-lg (setup.py) ... done
      Created wheel for en-vectors-web-lg: filename=en_vectors_web_lg-2.1.0-cp36-none-any
      Stored in directory: /tmp/pip-ephem-wheel-cache-c0k7l_x1/wheels/ce/3e/83/59647d0b4
    Successfully built en-vectors-web-lg
    Installing collected packages: en-vectors-web-lg
    Successfully installed en-vectors-web-lg-2.1.0
    ✓ Download and installation successful
    You can now load the model via spacy.load('en_vectors_web_lg')
```

```
!python -m spacy download en_vectors_web_lg
```

```
Requirement already satisfied: en_vectors_web_lg==2.1.0 from https://github.com/explo
Requirement already satisfied: spacy<3.0.0,>=2.1.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: plac<1.2.0,>=0.9.6 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: thinc==7.4.0 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: wasabi<1.1.0,>=0.4.0 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: srsly<1.1.0,>=1.0.2 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.6/dist
Requirement already satisfied: catalogue<1.1.0,>=0.0.7 in /usr/local/lib/python3.6/di
Requirement already satisfied: blis<0.5.0,>=0.4.0 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.6/di
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.6/
Requirement already satisfied: importlib-metadata>=0.20; python_version < "3.8" in /u
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-packages (
✓ Download and installation successful
You can now load the model via spacy.load('en_vectors_web_lg')
```

```
import spacy
# Load the largest english language vector collection from Spacy
nlp = spacy.load('en_vectors_web_lg')
```

```
df['label'].value_counts(normalize = True)
```

```
0    0.744192
1    0.255808
Name: label, dtype: float64
```

```
# Function to lemmatize the tokens to their basic forms to normalize the tweet text
# and focus on key words for the classification tasks

def lemmatization(texts):
    output = []
    for i in texts:
        s = [token.lemma_ for token in nlp(i)]
        output.append(' '.join(s))
    return output
```

```
%%time
df.tweet=lemmatization(df.tweet)

df.tweet=df.tweet.str.replace('-PRON-','')
```

```
CPU times: user 1.35 s, sys: 4.36 ms, total: 1.35 s
Wall time: 1.36 s
```

```
%%time
df1.tweet=lemmatization(df1.tweet)
df1.tweet=df1.tweet.str.replace('-PRON-','')
```

```
CPU times: user 318 ms, sys: 84 µs, total: 318 ms
Wall time: 318 ms
```

```
nlp('having').vector
```

```
         1.3405e-01,  3.7844e-04,  5.4900e-02, -2.5173e-01, -3.5485e-01,
        -3.7260e-01, -1.7240e-03,  1.1956e+00, -4.1293e-01,  3.5877e-01,
         5.1265e-03, -2.9626e-01, -2.4748e-01,  1.6286e-01,  7.5768e-02,
         1.3535e-02, -7.9647e-02, -4.9073e-01, -1.0783e-01, -6.3812e-02,
        -1.3171e-01,  1.8626e-01,  2.4554e-01,  2.5685e-01,  3.0148e-01,
        -4.8167e-01,  4.0285e-01, -4.7838e-02, -8.0964e-02, -5.6645e-01,
         2.1666e-01,  1.1220e-01,  1.5485e-02,  3.1444e-01, -4.7426e-01,
         3.0210e-01, -3.6470e-01, -3.4347e-01,  9.9283e-02, -8.5861e-02,
        -8.2277e-02, -2.5866e-02, -4.7161e-02, -2.1301e-01,  2.6880e-01,
         1.8113e-01, -2.0620e-01, -2.4319e-02, -1.5963e-01,  8.6472e-02,
         1.8116e-01,  1.2205e-01, -4.6879e-01,  2.7622e-01,  3.7899e-02,
         4.4370e-03,  2.6413e-01,  2.2721e-01, -1.7805e-02,  2.7563e-01,
         2.6386e-01,  1.7431e-01, -6.1444e-02, -2.1381e-02, -2.7438e-02,
         1.6243e-01,  2.8102e-01, -3.5839e-02,  1.7901e-01,  3.2328e-01,
        -1.1513e-01,  1.3440e-01, -1.8181e-01, -5.0755e-01,  1.9801e-02,
        -3.0611e-01,  2.8132e-01, -3.1478e-02,  1.8076e-01, -8.5850e-03,
        -1.4519e-02,  2.8539e-02, -2.0772e-01,  1.8872e-01, -3.3428e-02,
         2.5718e-01,  4.4756e-01, -2.1874e-01,  4.7900e-02,  1.4013e-01,
        -3.2908e-01,  3.1017e-02, -1.5771e-02, -2.7796e-01,  2.0601e-01,
        -1.3484e+00, -1.3698e-01, -8.7260e-02, -6.8283e-02,  1.7768e-01,
         1.2368e-01,  2.2966e-01,  3.7684e-03,  3.6778e-02, -1.9610e-01,
        -4.0696e-01, -1.2112e-01,  2.8510e-01, -2.4706e-01,  4.0122e-01,
         2.9606e-01,  1.7297e-01,  5.7350e-01,  6.2956e-02,  3.7901e-01,
        -1.9420e-02, -1.4721e-01, -3.1434e-01, -2.4116e-01, -2.2703e-01,
        -1.5893e-03,  1.8312e-01, -3.2423e-01,  1.5497e-01,  3.3933e-01,
        -2.3480e-01,  1.5851e-02,  2.7963e-01,  1.8745e-02, -1.5975e-01,
        -1.5019e+00,  5.0632e-02,  3.6933e-02,  1.0450e-01, -7.0496e-02,
        -2.0645e-01, -7.0083e-02, -8.1474e-02,  1.8476e-01, -9.9499e-02,
        -3.0478e-01,  7.6468e-02, -2.3014e-01, -7.0870e-02, -7.0931e-02,
         8.1447e-02,  8.0975e-02,  4.3891e-01,  1.9877e-01, -3.2176e-01,
         2.1967e-01, -5.7821e-01,  3.0394e-01, -1.2663e-01, -1.0427e-01,
        -2.4780e-01,  2.6204e-01,  6.2570e-02,  9.1614e-02,  1.8825e-02,
        -2.6012e-01, -4.1146e-01,  2.7580e-01, -4.9186e-03, -8.3340e-02,
        -1.1895e-01, -3.8721e-01,  4.7886e-02,  9.9593e-02, -2.6970e-01,
        -7.3007e-03, -3.7161e-01, -6.0079e-01,  4.3112e-02, -1.7589e-01,
        -3.2411e-01, -2.6899e-01,  7.3743e-01, -1.7653e-01, -1.7557e-01,
         1.6940e-01,  1.9966e-02, -1.3267e-01,  5.9843e-01,  2.3689e-01,
         1.1431e-02, -8.2624e-02,  2.5213e-01, -5.1019e-01,  1.7412e-01,
         4.0625e-01, -1.0041e-03,  2.7558e-01,  7.2856e-03,  3.6192e-01,
         9.3313e-02, -4.0080e-01, -2.0661e-01, -5.1045e-03,  1.5150e-01,
        -2.6760e-01,  2.6065e-01, -3.8441e-01, -4.5888e-02, -3.4107e-01,
         2.3661e-01, -2.5816e-01, -1.6351e-01,  1.4184e-01,  1.7698e-01,
        -1.1873e-01, -7.8805e-02, -2.2065e-01,  2.1354e-01,  8.3310e-02,
        -1.3151e-02,  1.6681e-01,  6.7123e-02, -1.4861e-01, -7.7549e-02,
         1.3314e-01, -2.5016e-01,  3.0317e-02, -4.2529e-02,  2.6820e-01,
        -2.5129e-01,  1.7177e-01,  8.6223e-02, -1.0212e-01,  1.1251e-01,
        -6.6374e-02,  3.7500e-02,  2.6159e-01,  6.3398e-01, -7.4445e-02,

        -2.2132e-03, -3.4139e-02,  1.3005e-01, -3.4528e-01,  2.7955e-02,
         1.4248e-01, -2.3346e-01,  3.2881e-01,  1.5303e-01,  1.7503e-01,
         1.3949e-01, -5.0988e-02, -9.5092e-02, -5.1364e-02,  2.5831e-01,
         3.1437e-01,  4.3509e-01, -3.9043e-01,  5.4367e-01,  2.8549e-01,
         7.8270e-01, -2.2442e-02,  1.1466e-01,  5.1672e-01, -2.9182e-01,
        -4.3049e-02, -7.7364e-02, -3.9407e-01, -2.5879e-01, -3.4362e-01,
         2.9721e-01, -2.6811e-01,  8.9689e-02,  1.2101e-01,  5.0895e-01,
         2.8325e-01,  4.3377e-01,  9.8544e-02,  5.9706e-02, -1.3283e-02,
        -1.0903e-01,  2.1455e-01, -2.9188e-01,  1.6256e-01,  2.1777e-01,
        -1.4039e-01, -8.1819e-03, -3.7918e-01, -2.1583e-01, -1.8292e-01,
        -5.0702e-02, -7.3112e-02, -1.6639e-03, -1.7232e-02,  3.5350e-02],
       dtype=float32)
```

```
# Convert cleaned tweets into Spacy word vectors
# The model returns 300-dimensional embeddings

tweets=df.tweet
tweet_list=[nlp(word).vector for word in tweets]
X_tr=np.array(tweet_list)


test_tweets = df1.tweet
test_word_vec = [nlp(word).vector for word in test_tweets]
X_te = np.array(test_word_vec)
```

```
print(X_tr.shape, X_te.shape)
```

```
     (7920, 300) (1953, 300)
```

```
# Save Spacy_train_new
pickle_out = open("Spacy_train.pickle","wb")
pickle.dump(X_tr, pickle_out)
pickle_out.close()

# Save Spacy_test_new
pickle_out = open("Spacy_test.pickle","wb")
pickle.dump(X_te, pickle_out)
pickle_out.close()
```

```
# Additional 'Optional' step for text normalization
# Import spaCy's language model
nlp1 = spacy.load('en', disable=['parser', 'ner'])
```

```
%tensorflow_version 1.x
```

```
     TensorFlow 1.x selected.
```

```
!pip install tensorflow-hub
```

```
     Requirement already satisfied: tensorflow-hub in /usr/local/lib/python3.6/dist-packag
     Requirement already satisfied: protobuf>=3.8.0 in /usr/local/lib/python3.6/dist-packa
     Requirement already satisfied: numpy>=1.12.0 in /usr/local/lib/python3.6/dist-package
     Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.6/dist-packages
     Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (
!pip install tensorflow_gpu==1.5.0
```

```
Collecting tensorflow_gpu==1.5.0
  Downloading https://files.pythonhosted.org/packages/d5/8b/094add4d2d667ddfef867285
    |████████████████████████████████| 201.9MB 85kB/s
Requirement already satisfied: protobuf>=3.4.0 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: absl-py>=0.1.6 in /usr/local/lib/python3.6/dist-packag
Collecting tensorflow-tensorboard<1.6.0,>=1.5.0
  Downloading https://files.pythonhosted.org/packages/cc/fa/91c06952517b4f1bc075545b
    |████████████████████████████████| 3.0MB 50.0MB/s
Requirement already satisfied: numpy>=1.12.1 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages (
Requirement already satisfied: werkzeug>=0.11.10 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: bleach==1.5.0 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: html5lib==0.9999999 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr/loc
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-packages (
Installing collected packages: tensorflow-tensorboard, tensorflow-gpu
  Found existing installation: tensorflow-gpu 1.2.0
    Uninstalling tensorflow-gpu-1.2.0:
      Successfully uninstalled tensorflow-gpu-1.2.0
Successfully installed tensorflow-gpu-1.5.0 tensorflow-tensorboard-1.5.1
```

```
import tensorflow_hub as hub
```

```
import tensorflow as tf

elmo = hub.Module("https://tfhub.dev/google/elmo/2", trainable=True)
```

```
tf.test.gpu_device_name()
```

```
'/device:GPU:0'
```

```
def elmo_convert(x):
  embeddings=elmo(x.tolist(),signature='default',as_dict=True)['elmo']
  with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    sess.run(tf.tables_initializer())
    # return average of ELMo features
    return sess.run(tf.reduce_mean(embeddings,1))
```

```
# Creating batches of 100 tweets to feed into elmo model at a time as it consumes high com
list_train = [df[i:i+100] for i in range(0,df.shape[0],100)]
list_test = [df1[i:i+100] for i in range(0,df1.shape[0],100)]
```

```
# Extract ELMo embeddings
elmo_train=[elmo_convert(x['tweet']) for x in list_train]
```

```
elmo_test=[elmo_convert(x['tweet']) for x in list_test]
```

```
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
INFO:tensorflow:Saver not created because there are no variables in the graph to r
```

```
# Concatenating converted batches into single array of train and test dataset embeddings
elmo_train_new=np.concatenate(elmo_train,axis=0)
elmo_test_new=np.concatenate(elmo_test,axis=0)
```

```
# save elmo_train
pickle_out = open("elmo_train.pickle","wb")
pickle.dump(elmo_train_new, pickle_out)
pickle_out.close()

# save elmo_test
pickle_out = open("elmo_test.pickle","wb")
pickle.dump(elmo_test_new, pickle_out)
pickle_out.close()
```

```
df_train=pd.read_csv('train.csv')
```

```
#adding tweet length
df_train['tweet_len_total']=df_train.tweet.str.len()
```

```
df_train.head()
```

	id	label	tweet	tweet_len_total
0	1	0	#fingerprint #Pregnancy Test https://goo.gl/h1...	128
1	2	0	Finally a transparant silicon case ^^ Thanks t...	131
2	3	0	We love this! Would you go? #talk #makememorie...	123
3	4	0	I'm wired I know I'm George I was made that wa...	112
4	5	1	What amazing service! Apple won't even talk to...	124

```
#punctuation length as  a feature
# Function to calculate the total length of punctuation marks in a tweet
def puncts_len(x):
    punct_list = []
    x = str(x)
    for punct in puncts:
        for char in x:
            if punct==char:
                punct_list.append(punct)
    return len(punct_list)
```

```
df_train['punc_len']=df_train.tweet.apply(lambda x:puncts_len(x))
```

```
df_train.head()
```

	id	label	tweet	tweet_len_total	punc_len
0	1	0	#fingerprint #Pregnancy Test https://goo.gl/h1...	128	16
1	2	0	Finally a transparant silicon case ^^ Thanks t...	131	17
2	3	0	We love this! Would you go? #talk #makememorie...	123	18

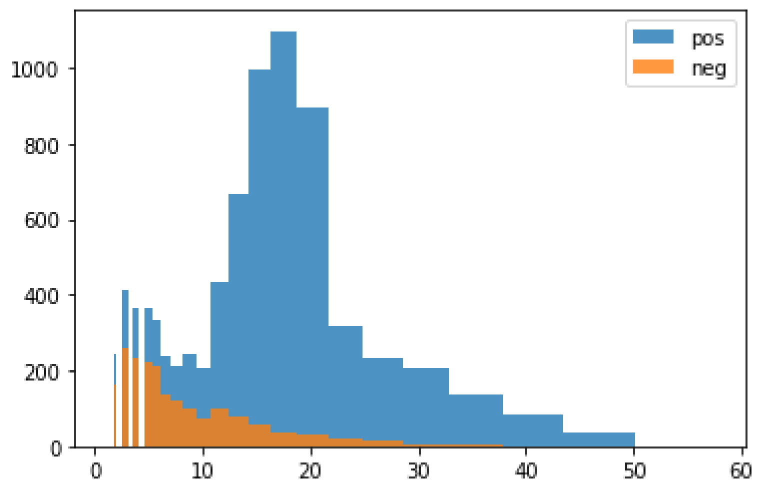
```
df_train.punc_len.describe()
```

```
count    7920.000000
mean       13.956692
std         8.406173
min         0.000000
25%         7.000000
50%        15.000000
75%        18.000000
max        59.000000
Name: punc_len, dtype: float64
```

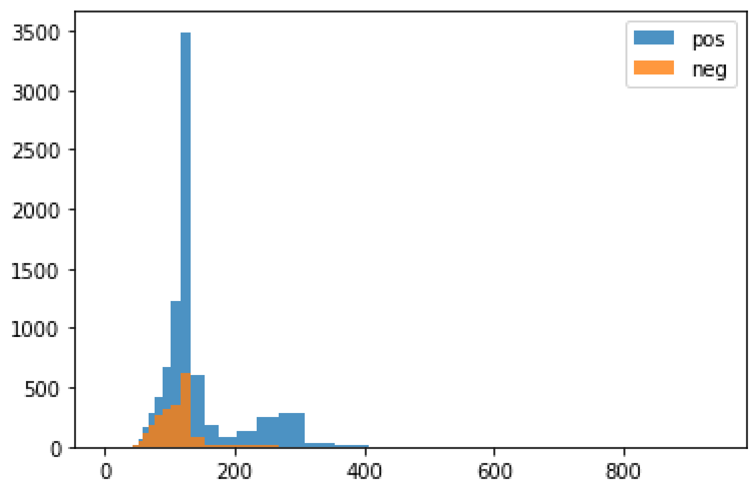
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7920 entries, 0 to 7919
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   id               7920 non-null   int64 
 1   label            7920 non-null   int64 
 2   tweet            7920 non-null   object
 3   tweet_len_total  7920 non-null   int64 
 4   punc_len         7920 non-null   int64 
dtypes: int64(4), object(1)
memory usage: 309.5+ KB
```

```
bins=1.15**(np.arange(0,30))
plt.hist(df_train.punc_len,bins=bins,alpha=0.8)
plt.hist(df_train[df_train.label==1]['punc_len'],bins=bins,alpha=0.8)
plt.legend(('pos','neg'))
plt.show()
```



```
bins = 1.15**(np.arange(0,50))
plt.hist(df_train['tweet_len_total'],bins=bins,alpha=0.8)
plt.hist(df_train[df_train['label']==1]['tweet_len_total'],bins=bins,alpha=0.8)
plt.legend(('pos','neg'))
plt.show()
```



```
df_train.to_csv('more_features_train.csv')


# Similarly for test dataset loading new dataframe
df_test= pd.read_csv('test.csv')



#adding tweet length
df_test['tweet_len_total']=df_test.tweet.str.len()


df_test['punc_len']=df_test.tweet.apply(lambda x:puncts_len(x))


df_test.to_csv('more_features_test.csv')


# Load Spacy_train Vectors
pickle_in = open("Spacy_train.pickle","rb")
spacy_train = pickle.load(pickle_in)
# Load Spacy_test Vectors
pickle_in = open("Spacy_test.pickle","rb")
spacy_test = pickle.load(pickle_in)



# Load BERT_train Vectors
pickle_in = open("bert_train.pickle","rb")
bert_train = pickle.load(pickle_in)

# Load BERT_test Vectors
pickle_in = open("bert_test.pickle","rb")
bert_test = pickle.load(pickle_in)
```

```
# Load ELMo_train Vectors
pickle_in = open("elmo_train.pickle","rb")
elmo_train = pickle.load(pickle_in)

# Load ELMo_test Vectors
pickle_in = open("elmo_test.pickle","rb")
elmo_test = pickle.load(pickle_in)



# Create Spacy + BERT Vectors
sb_train=np.hstack((spacy_train,bert_train))


sb_test=np.hstack((spacy_test,bert_test))


# save spacy_bert_train
pickle_out = open("Spacy_bert_train.pickle","wb")
pickle.dump(sb_train, pickle_out)
pickle_out.close()
# save Spacy_bert_test
pickle_out = open("Spacy_bert_test.pickle","wb")
pickle.dump(sb_test, pickle_out)
pickle_out.close()



# Create BERT + ELMo Vectors
bert_elmo_train = np.hstack((bert_train, elmo_train))
bert_elmo_test = np.hstack((bert_test, elmo_test))

print(bert_elmo_train.shape, bert_elmo_test.shape)

# save bert_elmo_train
pickle_out = open("bert_elmo_train.pickle","wb")
pickle.dump(bert_elmo_train, pickle_out)
pickle_out.close()

# save bert_elmo_test
pickle_out = open("bert_elmo_test.pickle","wb")
pickle.dump(bert_elmo_test, pickle_out)
pickle_out.close()
```

```
     (7920, 1792) (1953, 1792)
```

```
# Create Spacy + ELMo Vectors
spacy_elmo_train = np.hstack((spacy_train, elmo_train))
spacy_elmo_test = np.hstack((spacy_test, elmo_test))

print(spacy_elmo_train.shape, spacy_elmo_test.shape)

# save Spacy_elmo_train
pickle_out = open("Spacy_elmo_train.pickle","wb")
pickle.dump(spacy_elmo_train, pickle_out)
```

```
pickle_out.close()

# save Spacy_elmo_test
pickle_out = open("Spacy_elmo_test.pickle","wb")
pickle.dump(spacy_elmo_test, pickle_out)
pickle_out.close()

     (7920, 1324) (1953, 1324)
```

```
# Create Spacy + BERT + ELMo Vectors
spacy_bert_elmo_train = np.hstack((spacy_train, bert_train, elmo_train))
spacy_bert_elmo_test = np.hstack((spacy_test, bert_test, elmo_test))

print(spacy_bert_elmo_train.shape, spacy_bert_elmo_test.shape)

# save Spacy_bert_elmo_train
pickle_out = open("Spacy_bert_elmo_train.pickle","wb")
pickle.dump(spacy_bert_elmo_train, pickle_out)
pickle_out.close()

# save Spacy_bert_elmo_test
pickle_out = open("Spacy_bert_elmo_test.pickle","wb")
pickle.dump(spacy_bert_elmo_test, pickle_out)
pickle_out.close()

     (7920, 2092) (1953, 2092)
```

```
# Load any variation of word embeddings from Spacy, BERT and ELMo and assign it to X varia
pickle_in = open("Spacy_bert_elmo_train.pickle","rb")
```

```
X=pickle.load(pickle_in)
```

```
X.shape

     (7920, 2092)
```

```
# Load the training dataset into a dataframe
df = pd.read_csv('more_features_train.csv')
print(df.head())

        Unnamed: 0  id  ...  tweet_len_total punc_len
     0           0   1  ...              128       16
     1           1   2  ...              131       17
     2           2   3  ...              123       18
     3           3   4  ...              112       17
     4           4   5  ...              124        5

     [5 rows x 6 columns]
```

```
df.label.value_counts()
```

```
0    5894
1    2026
Name: label, dtype: int64
```

```
y=df.label
```

```
from sklearn.preprocessing import MinMaxScaler
mms=MinMaxScaler(feature_range=(-1,1))
```

```
tweet_len=mms.fit_transform(np.array(df.tweet_len_total).reshape(-1,1))
```

```
punc_len=mms.fit_transform(np.array(df.punc_len).reshape(-1,1))
```

```
final_X=np.hstack((X,tweet_len,punc_len))
```

```
print(final_X.shape)
```

```
(7920, 2094)
```

```
# Split the training dataset into train and test subsets
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(final_X, y, test_size=0.1, random_stat
```

```
#we can apply any classification model
from sklearn import svm
svc=svm.LinearSVC()
```

```
from sklearn.pipeline import Pipeline
```

```
text_clf=Pipeline([('clf',svc)])
```

```
text_clf.fit(X_train,y_train)
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/svm/_base.py:947: ConvergenceWarning:
  "the number of iterations.", ConvergenceWarning)
Pipeline(memory=None,
         steps=[('clf',
                 LinearSVC(C=1.0, class_weight=None, dual=True,
                           fit_intercept=True, intercept_scaling=1,
                           loss='squared_hinge', max_iter=1000,
                           multi_class='ovr', penalty='l2', random_state=None,
                           tol=0.0001, verbose=0))],
         verbose=False)
```

```
# Make predictions
# make predictions
predictions = text_clf.predict(X_test)
print(predictions)
```

```
[0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 0 1 1 0 1 1 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1
 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0
 0 1 0 0 0 1 0 1 1 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1
 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1
 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1
 0 0 1 0 0 0 0 1 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 0 0 1 1
 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 1 1 0
 0 0 0 0 0 1 0 1 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 1
 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 1 1 0 0 1
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0
 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0
 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0
 1 0 1 0 0 0 0 1 1 0 0 0 1 1 1 0 0 1 1 0 1 0 0 0 1 1 1 0 1 1 0 0 0 0 0 0 1
 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 1 1 0 1 0 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1
 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 1
 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0]
```

```
from sklearn import metrics
```

```
metrics.confusion_matrix(y_test,predictions)
```

```
array([[545,  38],
       [ 61, 148]])
```

```
metrics.classification_report(y_test,predictions)
```

```
'              precision    recall  f1-score   support\n\n           0       0.90
0.93      0.92       583\n           1       0.80      0.71      0.75       209\n\n
accuracy                           0.88       792\n   macro avg       0.85      0.82
```

```
metrics.accuracy_score(y_test,predictions)
```

```
0.875
```

```
# Loading test dataset
df1 = pd.read_csv('more_features_test.csv')
print(df1.head())
```

```
   Unnamed: 0    id  ... tweet_len_total  punc_len
0           0  7921  ...              77         6
1           1  7922  ...             115        13
2           2  7923  ...             104         9
3           3  7924  ...             129         4
4           4  7925  ...              70         6
```

```
     [5 rows x 5 columns]
```

```
# Dropping tweet column as it is no longer required for final submission leaving only the
df1 = df1.drop(['tweet'],axis=1)
print(df1.head())
```

```
        Unnamed: 0    id  tweet_len_total  punc_len
     0           0  7921               77         6
     1           1  7922              115        13
     2           2  7923              104         9
     3           3  7924              129         4
     4           4  7925               70         6
```

```
# Loading corresponding test tweets embeddings as loaded for the training dataset
pickle_in = open("Spacy_bert_elmo_test.pickle","rb")
test_X = pickle.load(pickle_in)
print(test_X.shape)
```

```
     (1953, 2092)
```

```
# Preparing test dataset for predictions by adding text features to the tweet embeddings d
tweet_len_arr_test = np.array(df1['tweet_len_total'])
tweet_punct_arr_test = np.array(df1['punc_len'])
print(tweet_len_arr_test.shape, tweet_punct_arr_test.shape)

tweet_len_norm_test = mms.fit_transform(tweet_len_arr_test.reshape(-1, 1))
tweet_punct_norm_test = mms.fit_transform(tweet_punct_arr_test.reshape(-1, 1))
print(tweet_len_norm_test.shape, tweet_punct_norm_test.shape)

test_X = np.hstack((test_X, tweet_len_norm_test, tweet_punct_norm_test))
print(test_X.shape)
```

```
     (1953,) (1953,)
     (1953, 1) (1953, 1)
     (1953, 2094)
```

```
# Making predictions using trained model for the test dataset for final submission
test_predictions = text_clf.predict(test_X)
print(test_predictions)
```

```
     [1 1 0 ... 0 1 0]
```

```
df1.columns
```

```
     Index(['Unnamed: 0', 'id', 'tweet_len_total', 'punc_len', 'label'], dtype='object')
```

```
# Adding predicted labels to the test dataframe
df1['label'] = test_predictions
df1.drop(['Unnamed: 0','tweet_len_total', 'punc_len'],axis=1,inplace=True)
print(df1.head())
```

```
        id  label
     0  7921      1
     1  7922      1
     2  7923      0
     3  7924      1
     4  7925      1
```

```
# Saving the final predicted submission file to csv
df1.to_csv('ALL_SVM.csv', index=False)
```