In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:
```
DATASET_ENCODING = "ISO-8859-1"
df = pd.read_csv("C:\\Users\\arunk\\OneDrive\\Desktop\\twitter_new.csv", encoding=D
df.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

In [ ]:
```
df.shape
```

Out[ ]:
```
(1000000, 6)
```

In [ ]:
```
df_copy = df.copy()
```

In [ ]:
```
df_copy['word counts'] = df_copy['Tweet'].apply(lambda x : len(x.split()))
df_copy.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet	v co
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	

In [ ]:
```
def char_counts(x):
    x = x.split()
    x = ''.join(x)
    return len(x)
```

In [ ]:
```
df_copy['char counts'] = df_copy['Tweet'].apply(lambda x : char_counts(x))
df_copy.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet	v co
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	

In [ ]:
```
df_copy['average_word_len'] = df_copy['char counts']/df_copy['word counts']
df_copy.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet	v co
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	

In [ ]:
```
from nltk.corpus import stopwords
```

In [ ]:
```
stop_words = stopwords.words('english')
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : ' '.join([t for t in x.split() if t not
```

In [ ]:
```
df.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	upset can't update Facebook texting it... migh...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times ball. Managed sav...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	whole body feels itchy like fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, behaving all. i'm mad. he...

In [ ]:
```
def remove(x):
    x = x.split()
    for i in range(len(x)):
        if(x[i].startswith('@')):
            x[i] = x[i].replace('@','')
        elif(x[i].startswith('#')):
            x[i].replace('#','')
    return ' '.join(x)
```

In [ ]:
```
remove('@robbiebronniman Sounds like great night')
```

Out[ ]:
```
'robbiebronniman Sounds like great night'
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : remove(x))
```

In [ ]:
```
df.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	switchfoot http://twitpic.com/2y1zl - Awww, th...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	upset can't update Facebook texting it... migh...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	Kenichan I dived many times ball. Managed save...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	whole body feels itchy like fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	nationwideclass no, behaving all. i'm mad. her...

In [ ]:
```
df_copy.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet	v co
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...	
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...	
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...	
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire	
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....	

In [ ]:
```
df_copy['numeric_counts'] = df_copy['Tweet'].apply(lambda x : len([t for t in x.spl
```

In [ ]:
```
df_copy[df_copy['numeric_counts']==5]
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
8733	0	1548341270	Fri Apr 17 20:41:16 PDT 2009	NO_QUERY	HayleyyFitch	@sydneohhh aww no its not hot at all! today i...
22698	0	1557548816	Sun Apr 19 04:46:29 PDT 2009	NO_QUERY	4mariam	I Went 2 school 4 2 hours only , the nurse cal...
29842	0	1563599632	Sun Apr 19 23:28:05 PDT 2009	NO_QUERY	alyssawang	2 exams, 1 paper, 1 quiz, 1 project, +hw all i...
70248	0	1693599250	Sun May 03 23:05:15 PDT 2009	NO_QUERY	nkotb427	@donniewahlberg wanted 2 b there but had 2 com...
108294	0	1824234530	Sun May 17 00:32:40 PDT 2009	NO_QUERY	sparkay	its nearly 330 in the am, i've barely slept &a...
...	...	...	...	...	...	...
895891	4	1692853483	Sun May 03 20:56:34 PDT 2009	NO_QUERY	MichaelCBeck	I did it!!! I finished Wildflower triathalon i...
916100	4	1753238118	Sun May 10 00:42:20 PDT 2009	NO_QUERY	KimOFDakiraAve	*sigh* Off 2 bed 2 try 2 get these crummy 2 hr...
947169	4	1822942647	Sat May 16 20:44:46 PDT 2009	NO_QUERY	MTVnHollyWEST23	@Lyriqal Im a solid 21 and a HALF I luv my age...
953792	4	1824713036	Sun May 17 02:35:59 PDT 2009	NO_QUERY	TomJensen76	@MianneBagger ok. I seriously wld luv 2 do it ...

	Sentiment	Id	Date	Flag	User	Tweet
979918	4	1833974703	Mon May 18 01:08:59 PDT 2009	NO_QUERY	akwikkel	13 weeks of "fun in the sun" with th...

69 rows × 10 columns

In [ ]:
```
df_copy.iloc[12969]
```

Out[ ]:
```
Sentiment                                                       0
Id                                                     1552060384
Date                                  Sat Apr 18 10:41:00 PDT 2009
Flag                                                     NO_QUERY
User                                                     springit89
Tweet           house feels so empty with 2 out of 3 roommates...
word counts                                                    18
char counts                                                    80
average_word_len                                         4.444444
numeric_counts                                                  2
Name: 12969, dtype: object
```

In [ ]:
```
def remove_digit(x):
    x = x.split()
    for t in range(0,len(x)):
        if(x[t].isdigit()):
            x[t] = ''
    return ' '.join(x)
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : remove_digit(x))
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : str(x).lower())
```

In [ ]:
```
df.iloc[12969]
```

Out[ ]:
```
Sentiment                                                       0
Id                                                     1552060384
Date                                  Sat Apr 18 10:41:00 PDT 2009
Flag                                                     NO_QUERY
User                                                     springit89
Tweet           house feels empty   roommates gone! im sooo lo...
Name: 12969, dtype: object
```

In [ ]:
```
contractions = {
 "ain't": "am not",
 "aren't": "are not",
 "can't": "cannot",
 "can't've": "cannot have",
 "'cause": "because",
 "could've": "could have",
```

```
"couldn't": "could not",
"couldn't've": "could not have",
"didn't": "did not",
"doesn't": "does not",
"don't": "do not",
"hadn't": "had not",
"hadn't've": "had not have",
"hasn't": "has not",
"haven't": "have not",
"he'd": "he would",
"he'd've": "he would have",
"he'll": "he will",
"he'll've": "he will have",
"he's": "he is",
"how'd": "how did",
"how'd'y": "how do you",
"how'll": "how will",
"how's": "how does",
"i'd": "i would",
"i'd've": "i would have",
"i'll": "i will",
"i'll've": "i will have",
"i'm": "i am",
"i've": "i have",
"isn't": "is not",
"it'd": "it would",
"it'd've": "it would have",
"it'll": "it will",
"it'll've": "it will have",
"it's": "it is",
"let's": "let us",
"ma'am": "madam",
"mayn't": "may not",
"might've": "might have",
"mightn't": "might not",
"mightn't've": "might not have",
"must've": "must have",
"mustn't": "must not",
"mustn't've": "must not have",
"needn't": "need not",
"needn't've": "need not have",
"o'clock": "of the clock",
"oughtn't": "ought not",
"oughtn't've": "ought not have",
"shan't": "shall not",
"sha'n't": "shall not",
"shan't've": "shall not have",
"she'd": "she would",
"she'd've": "she would have",
"she'll": "she will",
"she'll've": "she will have",
"she's": "she is",
"should've": "should have",
"shouldn't": "should not",
"shouldn't've": "should not have",
"so've": "so have",
```

```
 "so's": "so is",
 "that'd": "that would",
 "that'd've": "that would have",
 "that's": "that is",
 "there'd": "there would",
 "there'd've": "there would have",
 "there's": "there is",
 "they'd": "they would",
 "they'd've": "they would have",
 "they'll": "they will",
 "they'll've": "they will have",
 "they're": "they are",
 "they've": "they have",
 "to've": "to have",
 "wasn't": "was not",
 " u ": " you ",
 " ur ": " your ",
 " n ": " and ",
 "won't": "would not",
 'dis': 'this',
 'bak': 'back',
 'brng': 'bring',
 "won't":"would not",
  'dis':'this',
 "bak":"back",
 "brng":'bring',
 "i'v" : 'i have'
 }
```

In [ ]:
```
def expand(x):
    if type(x) is str:
        for key in contractions:
            value = contractions[key]
            x = x.replace(key,value)
        return x
    else:
        return x
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : expand(x))
```

In [ ]:
```
import re
```

Email removals and count

In [ ]:
```
 # re.findall(r'([a-z0-9+._-]+@[a-z0-9+._-]+\.[a-z0-9+_-]+)',x)
df_copy['emails'] = df_copy['Tweet'].apply(lambda x : re.findall(r'([a-z0-9+._-]+@[
df_copy['email_count'] = df_copy['emails'].apply(lambda x : len(x))
df_copy[df_copy['email_count']==1]
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
4054	0	1468735195	Tue Apr 07 03:26:47 PDT 2009	NO_QUERY	gabbyeight	I want a new laptop. HP TX2000 is the bomb. :...
7917	0	1470004086	Tue Apr 07 08:03:42 PDT 2009	NO_QUERY	ElleDell	who stole elledell@gmail.com?
8496	0	1470213042	Tue Apr 07 08:41:06 PDT 2009	NO_QUERY	missatari	@alexistehpom really? did you send out all th...
10290	0	1550847940	Sat Apr 18 07:26:12 PDT 2009	NO_QUERY	caaaitysarah	@LaureyStack awh...that's kinda sad lol add m...
16413	0	1555904035	Sat Apr 18 21:02:02 PDT 2009	NO_QUERY	kluo	@jilliancyork got 2 bottom of it, human error...
...	...	...	...	...	...	...
994661	4	1835590994	Mon May 18 06:20:03 PDT 2009	NO_QUERY	maggiewhitley	sewing up some orders for my Etsy shop: www.ma...
995131	4	1835658485	Mon May 18 06:28:50 PDT 2009	NO_QUERY	tiphaniebrooke	@krabumple t.tart@antigirl.com - yes, definite...
998805	4	1836339846	Mon May 18 07:49:17 PDT 2009	NO_QUERY	dogstrust	@ideasuk We release places in Oct/Nov, after b...
999377	4	1836456791	Mon May 18 08:01:54 PDT 2009	NO_QUERY	jnozzi	@drewmccormack If you'd like to discuss in mor...

	Sentiment	Id	Date	Flag	User	Tweet
999425	4	1836459020	Mon May 18 08:02:08 PDT 2009	NO_QUERY	ExclusiveSoRaw	@YoungCartoon Yea I bet....I told u I want my ...

215 rows × 12 columns

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : re.sub(r'([a-z0-9+._-]+@[a-z0-9+._-]+\.[
```

Removal of special characters

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : re.sub(r'[^\w]+'," ",x))
```

Remove URLs

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : re.sub(r'(http|https|ftp|ssh)://([\w_-]+
```

In [ ]:
```
df.head()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	switchfoot http twitpic com 2y1zl awww that is...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	upset cannot update facebook texting it might ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	kenichan i dived many times ball managed save ...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	whole body feels itchy like fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	nationwideclass no behaving all i am mad here ...

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : ' '.join(x.split()))
```

Removal of HTML tags

In [ ]:
```
from bs4 import BeautifulSoup
```

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : BeautifulSoup(x,'lxml').get_text().strip
```

Removal of Accented Characters

In [ ]:
```
import unicodedata
```

In [ ]:
```
def remove_accented(x):
 x = unicodedata.normalize('NFKD',x).encode('ascii','ignore').decode('utf-8','ignor
 return x
```

In [ ]:
```
df.tail()
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
999995	4	1879942807	Thu May 21 23:36:19 PDT 2009	NO_QUERY	divabat	healingsinger thank you needed
999996	4	1879942922	Thu May 21 23:36:20 PDT 2009	NO_QUERY	nick1975	vactress http bit ly cadea maybe
999997	4	1879942975	Thu May 21 23:36:21 PDT 2009	NO_QUERY	znmeb	brat13 hell windows price range unless free
999998	4	1879943113	Thu May 21 23:36:22 PDT 2009	NO_QUERY	virmani	jigardoshi neah wish reminiscing read post las...
999999	4	1879943219	Thu May 21 23:36:24 PDT 2009	NO_QUERY	redcomet81	msteagan and way i rewatched sun goddess last ...

In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : remove_accented(x))
```

Common Occurring words Removal

In [ ]:
```
text = ' '.join(df['Tweet'])
len(text)
```

Out[ ]:
```
53821807
```

In [ ]:
```
text = text.split()
```

In [ ]:
```
freq_comm = pd.Series(text).value_counts()
fre = freq_comm[:20]
df['Tweet'] = df['Tweet'].apply(lambda x : ' '.join([w for w in x.split() if w not
df.sample(10)
```

Out[ ]:

	Sentiment	Id	Date	Flag	User	Tweet
739369	0	2265711140	Sun Jun 21 07:09:27 PDT 2009	NO_QUERY	ksapopstar	oh dear
470457	0	2176427026	Mon Jun 15 03:50:43 PDT 2009	NO_QUERY	kdotcom	best massage life last night went f neck upper...
801493	4	1468086768	Mon Apr 06 23:38:25 PDT 2009	NO_QUERY	dEE486	jmac777 real mature bout nd ya average abc rhymes
900751	4	1694107624	Mon May 04 01:07:06 PDT 2009	NO_QUERY	fate_22	_mmmichelle france cheri
201633	0	1971937902	Sat May 30 08:13:03 PDT 2009	NO_QUERY	Emz_musicjunkie	has exam monday
687563	0	2251309335	Sat Jun 20 03:05:23 PDT 2009	NO_QUERY	mariemontemayor	jjayarzadon okay mexiland
910083	4	1751465685	Sat May 09 19:30:54 PDT 2009	NO_QUERY	bigjsl	nice pc authority review tivo vod service here...
675123	0	2247995306	Fri Jun 19 19:56:35 PDT 2009	NO_QUERY	erickmetz	soaked bone sitting car
964548	4	1827534335	Sun May 17 10:54:57 PDT 2009	NO_QUERY	annemul	mel future husband dj ing fashion show
277424	0	1991272555	Mon Jun 01	NO_QUERY	misskittycharms	believe pie left house

Sentiment	Id	Date	Flag	User	Tweet
		06:44:00 PDT 2009			

Rare occuring words removal

In [ ]:
```
rare20 = freq_comm.tail(100)
```

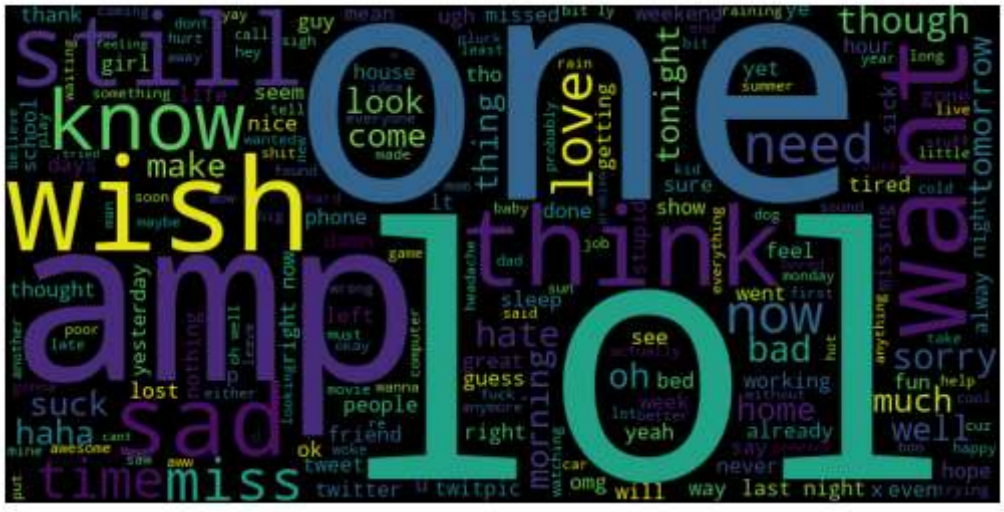
In [ ]:
```
df['Tweet'] = df['Tweet'].apply(lambda x : ' '.join([w for w in x.split() if w not
```

In [ ]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
%matplotlib inline
```

In [ ]:
```
neg_Tweet = df[df["Sentiment"]==0]
```

In [ ]:
```
neg_text = ' '.join(neg_Tweet['Tweet'])
```

In [ ]:
```
wc = WordCloud(width = 800,height = 400).generate(neg_text)
plt.imshow(wc)
plt.axis('off')
plt.show()
```



In [ ]:
```
pos_Tweet = df[df['Sentiment']==4]
```

In [ ]:
```
pos_text = " ".join(pos_Tweet['Tweet'])
```

In [ ]:
```
if not pos_text.strip():  # Check if pos_text is empty or just whitespace
    print("No words found in pos_text.")
else:
    pos_wc = WordCloud(width = 800, height = 400).generate(pos_text)
    plt.imshow(pos_wc, interpolation = 'bilinear')
    plt.axis('off')
```

In [ ]:
```
df['Sentiment'].value_counts()
```

Out[ ]:
```
Sentiment
0    800000
4    200000
Name: count, dtype: int64
```

Training algorithm for sentiment analysis

In [ ]:
```
from sklearn.svm import LinearSVC
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

In [ ]:
```
X = df['Tweet']
y = df['Sentiment']
```

In [ ]:
```
tfidf = TfidfVectorizer(norm = 'l1')
```

In [ ]:
```
X = tfidf.fit_transform(X)
```

In [ ]:
```
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2, random_state=
```

In [ ]:
```
X_train
```

Out[ ]:
```
<800000x460290 sparse matrix of type '<class 'numpy.float64'>'
        with 6034061 stored elements in Compressed Sparse Row format>
```

In [ ]:
```
model_svm = LinearSVC()
model_svm.fit(X_train,y_train)
```

Out[ ]:
```
▾ LinearSVC
LinearSVC()
```

In [ ]:
```
y_pred = model_svm.predict(X_test)
```

In [ ]:
```
acc_score = accuracy_score(y_test,y_pred)
print(acc_score)
```

```
0.851915
```

In [ ]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.86      0.97      0.91    160000
           4       0.76      0.38      0.51     40000

    accuracy                           0.85    200000
   macro avg       0.81      0.67      0.71    200000
weighted avg       0.84      0.85      0.83    200000
```

In [ ]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[155248   4752]
 [ 24865  15135]]
```

In [ ]:
```
x = ['i am really happy. that you came with me']
```

In [ ]:
```
model_svm.predict(tfidf.transform(x))[0]
```

Out[ ]:
```
4
```

Hyperparameter tuning

In [ ]:
```
params = {
    'penalty' : ['l1','l2'],
    'loss' : ['higne','squared_hinge'],
    'dual' : ['auto',True,False],
    'tol' : [0.1,0.01,0.001],
    'C' : [1.0,2.0]
 }
```

In [ ]:
```
gs = GridSearchCV(estimator = LinearSVC(), param_grid=params)
```

In [ ]:
```
gs.fit(X,y)
```

Out[ ]:
```
GridSearchCV
estimator: LinearSVC
LinearSVC
```

In [ ]:
```
gs.best_params_
```

Out[ ]:
```
{'C': 1.0, 'dual': True, 'loss': 'squared_hinge', 'penalty': 'l2', 'tol': 0.01}
```

In [ ]:
```
model_svm_h = LinearSVC(penalty='l1',C = 1.0, dual=False, loss = 'squared_hinge', t
```

In [ ]:
```
model_svm_h.fit(X_train,y_train)
```

Out[ ]:
```
LinearSVC(dual=False, penalty='l1', tol=0.1)
```

In [ ]:
```
y_pred_n = model_svm_h.predict(X_test)
```

In [ ]:
```
print(accuracy_score(y_test,y_pred_n))
```

```
0.85086
```

In [ ]:
```
print(classification_report(y_test,y_pred_n))
```

```
              precision    recall  f1-score   support

           0       0.86      0.97      0.91    160000
           4       0.75      0.38      0.51     40000

    accuracy                           0.85    200000
   macro avg       0.81      0.68      0.71    200000
weighted avg       0.84      0.85      0.83    200000
```

In [ ]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[155248   4752]
 [ 24865  15135]]
```

In [ ]:
```
x = ['i am really happy. that you came with me']
```

In [ ]:
```
model_svm.predict(tfidf.transform(x))[0]
```

Out[ ]:
```
4
```

In [ ]:
```
def run_svm(df):
    X = df['Tweet']
    y = df['Sentiment']
    tfidf = TfidfVectorizer(norm = 'l1',ngram_range = (1,5),analyzer = 'word', max_
    X = tfidf.fit_transform(X)
    X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2, random_st
    print('shape of X ',X.shape)
    clf = LinearSVC(penalty='l1',C = 1.0, dual=False, loss = 'squared_hinge', tol=0
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    print('Accuracy Score ', accuracy_score(y_test,y_pred))
    print('Classification report ',classification_report(y_test,y_pred))
    return tfidf,clf
tfidf,clf = run_svm(df)
```

```
shape of X  (1000000, 5000)
Accuracy Score  0.842695
Classification report                precision    recall  f1-score   support

           0       0.85      0.97      0.91    160000
           4       0.74      0.33      0.46     40000

    accuracy                           0.84    200000
   macro avg       0.80      0.65      0.68    200000
weighted avg       0.83      0.84      0.82    200000
```

In [ ]:
```
clf.predict(tfidf.transform(x))
```

Out[ ]:
```
array([0], dtype=int64)
```

In [ ]:
```
import pickle
```

In [ ]:
```
pickle.dump(clf,open('clf.pkl','wb'))
pickle.dump(tfidf,open('tfidf.pkl','wb'))
```

In [ ]:
```
clf = pickle.load(open('clf.pkl','rb'))
tfidf = pickle.load(open('tfidf.pkl','rb'))
```

In [ ]:
```
x = ['i am really happy. that you came with me']
x = tfidf.transform(x)
```

In [ ]:
```
clf.predict(x)
```

Out[ ]:
```
array([0], dtype=int64)
```

Logistic Regression

In [ ]:
```
from sklearn.linear_model import LogisticRegression
```

In [ ]:
```
lr = LogisticRegression()
```

In [ ]:
```
def run_lr(df):
    X = df['Tweet']
    y = df['Sentiment']
    tfidf = TfidfVectorizer(norm = 'l1',ngram_range = (1,5),analyzer = 'word', max_
    X = tfidf.fit_transform(X)
    X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2, random_st
    print('shape of X ',X.shape)
    clf = LogisticRegression()
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    print('Accuracy Score ', accuracy_score(y_test,y_pred))
    print('Classification report ',classification_report(y_test,y_pred))
    return tfidf,clf
tfidf_lr,clf_lr = run_lr(df)
```

```
shape of X  (1000000, 10000)
Accuracy Score  0.84476
Classification report                precision    recall  f1-score   support

           0       0.85      0.97      0.91    160000
           4       0.76      0.33      0.46     40000

    accuracy                           0.84    200000
   macro avg       0.80      0.65      0.68    200000
weighted avg       0.83      0.84      0.82    200000
```

Decision Tree

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
def run_decision_tree(df):
    X = df['Tweet']
    y = df['Sentiment']
    tfidf = TfidfVectorizer(norm = 'l1',ngram_range = (1,5),analyzer = 'word', max_
    X = tfidf.fit_transform(X)
    X_train,X_test,y_train,y_test = train_test_split(X, y, test_size=0.2, random_st
    print('shape of X ',X.shape)
    clf = DecisionTreeClassifier(criterion='entropy')
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    print('Accuracy Score ', accuracy_score(y_test,y_pred))
    print('Classification report ',classification_report(y_test,y_pred))
    return tfidf,clf
tfidf_dt,clf_dt = run_lr(df)
```

```
shape of X  (1000000, 10000)
Accuracy Score  0.84476
Classification report                precision    recall  f1-score   support

           0       0.85      0.97      0.91    160000
           4       0.76      0.33      0.46     40000

    accuracy                           0.84    200000
   macro avg       0.80      0.65      0.68    200000
weighted avg       0.83      0.84      0.82    200000
```