In [1]:
```python
import pandas as pd
dataset = pd.read_csv('hate_speech.csv')
dataset.head()
```

Out[1]:

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

In [2]:
```python
dataset.label.value_counts()
```

Out[2]:
```
0    3000
1    2242
Name: label, dtype: int64
```

In [3]:
```python
for index, tweet in enumerate(dataset["tweet"] [10:15]):
    print(index+1,"-", tweet)
```

```
1 -  â  #ireland consumer price index (mom) climbed from previous 0.2% to
0.5% in may   #blog #silver #gold #forex
2 - we are so selfish. #orlando #standwithorlando #pulseshooting #orlandos
hooting #biggerproblems #selfish #heabreaking   #values #love #
3 - i get to see my daddy today!!   #80days #gettingfed
4 - ouch...junior is angryð#got7 #junior #yugyoem   #omg
5 - i am thankful for having a paner. #thankful #positive
```

In [4]:
```python
import re
def clean_text(text):
    text = re.sub(r'[^a-zA-Z\']','', text)
    text = re.sub(r'[^\x00-\x7F]+','', text)
    text = text.lower()
    return text
```

In [5]:
```python
dataset['clean_text'] = dataset.tweet.apply(lambda x: clean_text(x))
```

In [29]:
```
dataset.head(10)
```

Out[29]:

	id	label	tweet	clean_text	word_count	any_
0	1	0	@user when a father is dysfunctional and is s...	userwhenafatherisdysfunctionalandissoselfishhe...	1	
1	2	0	@user @user thanks for #lyft credit i can't us...	useruserthanksforlyftcreditican'tusecausetheyd...	1	
2	3	0	bihday your majesty	bihdayyourmajesty	1	
3	4	0	#model i love u take with u all the time in ...	modeliloveutakewithuallthetimeinur	1	
4	5	0	factsguide: society now #motivation	factsguidesocietynowmotivation	1	
5	6	0	[2/2] huge fan fare and big talking before the...	hugefanfareandbigtalkingbeforetheyleavechaosan...	1	
6	7	0	@user camping tomorrow @user @user @user @use...	usercampingtomorrowuseruseruseruseruseruseruse...	1	
7	8	0	the next school year is the year for exams.ð□□...	thenextschoolyearistheyearforexamscan'tthinkab...	1	
8	9	0	we won!!! love the land!!! #allin #cavs #champ...	wewonlovethelandallincavschampionsclevelandcle...	1	
9	10	0	@user @user welcome here ! i'm it's so #gr...	useruserwelcomeherei'mit'ssogr	1	

In [6]:
```
from nltk.corpus import stopwords
len(stopwords.words('english'))
```

Out[6]:
```
179
```

In [7]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\nihar\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[7]:
```
True
```

In [8]:
```
def gen_freq(text):
    word_list = []
    for tw_words in text.split():
        word_list.extend(tw_words)
    word_freq = pd.Series(word_list).value_counts()
    stop = stopwords.words('english')
    word_freq = word_freq.drop(stop, errors='ignore')
    return word_freq
```

In [9]:
```
def any_neg(words):
    for word in words:
        if word in ['n', 'no', 'non', 'not'] or re.search(r"\wn't", wor
            return 1
        else:
            return 0
```

In [10]:
```
def any_rare(words,rare_100):
      for word in words:
        if word in rare_100:
            return 1
        else:
            return 0
```

In [11]:
```
def is_question(words):
      for word in words:
        if word in ["when","what","how","why","who"]:
            return 1
        else:
            return 0
```

In [12]:
```
word_freq=gen_freq(dataset.clean_text.str)
rare_100=word_freq[-100:]
dataset['word_count']=dataset.clean_text.str.split().apply(lambda x:ler
dataset['any_neg']=dataset.clean_text.str.split().apply(lambda x:any_ne
dataset['is_question']=dataset.clean_text.str.split().apply(lambda x:is
dataset['any_rare']=dataset.clean_text.str.split().apply(lambda x:any_r
dataset['char_count']=dataset.clean_text.apply(lambda x:len(x))
```

In [13]:
```
dataset.head(10)
```

Out[13]:

	id	label	tweet	clean_text	word_count	any_
0	1	0	@user when a father is dysfunctional and is s...	userwhenafatherisdysfunctionalandissoselfishhe...	1	
1	2	0	@user @user thanks for #lyft credit i can't us...	useruserthanksforlyftcreditican'tusecausetheyd...	1	
2	3	0	bihday your majesty	bihdayyourmajesty	1	
3	4	0	#model i love u take with u all the time in ...	modeliloveutakewithuallthetimeinur	1	
4	5	0	factsguide: society now #motivation	factsguidesocietynowmotivation	1	
5	6	0	[2/2] huge fan fare and big talking before the...	hugefanfareandbigtalkingbeforetheyleavechaosan...	1	
6	7	0	@user camping tomorrow @user @user @user @use...	usercampingtomorrowuseruseruseruseruseruseruse...	1	
7	8	0	the next school year is the year for exams.ð□□...	thenextschoolyearistheyearforexamscan'tthinkab...	1	
8	9	0	we won!!! love the land!!! #allin #cavs #champ...	wewonlovethelandallincavschampionsclevelandcle...	1	
9	10	0	@user @user welcome here ! i'm it's so #gr...	useruserwelcomeherei'mit'ssogr	1	

In [18]:
```
from sklearn.model_selection import train_test_split
X = dataset[['word_count', 'any_neg','any_rare','char_count','is_questi
y = dataset.label
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2,
```

In [21]:
```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model = model.fit(X_train, y_train)
pred = model.predict(X_test)
```

In [22]:
```
model.predict(X_test[5:10])
```

Out[22]:
```
array([1, 1, 1, 1, 1], dtype=int64)
```

In [23]:
```
from sklearn.metrics import accuracy_score
print("Accuracy:", accuracy_score(y_test, pred)*100, '%')
```

```
Accuracy: 44.518589132507145 %
```

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
clf_rf = RandomForestClassifier()
clf_rf.fit(X_train,y_train)
rf_pred=clf_rf.predict(X_test).astype(int)
```

In [25]:
```
from sklearn.metrics import classification_report, confusion_matrix, \
accuracy_score
print(confusion_matrix(y_test,rf_pred))
print(classification_report(y_test,rf_pred))
print("Accuracy:",accuracy_score(y_test, rf_pred))
```

```
[[410 189]
 [233 217]]
              precision    recall  f1-score   support

           0       0.64      0.68      0.66       599
           1       0.53      0.48      0.51       450

    accuracy                           0.60      1049
   macro avg       0.59      0.58      0.58      1049
weighted avg       0.59      0.60      0.59      1049

Accuracy: 0.5977121067683508
```

In [26]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(class_weight='balanced')
logreg.fit(X_train, y_train)
```

Out[26]:
```
LogisticRegression(class_weight='balanced')
```

In [27]:
```
y_pred = logreg.predict(X_test)
```

In [28]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.65      0.54      0.59       599
           1       0.50      0.62      0.55       450

    accuracy                           0.57      1049
   macro avg       0.58      0.58      0.57      1049
weighted avg       0.59      0.57      0.57      1049
```

In [ ]:
```

```