```
import pandas as pd
dataset=pd.read_csv("hate_speech.csv")
dataset
```

	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
...	...	...	...
5237	31935	1	lady banned from kentucky mall. @user #jcpenn...
5238	31947	1	@user omfg i'm offended! i'm a mailbox and i'...
5239	31948	1	@user @user you don't have the balls to hashta...
5240	31949	1	makes you ask yourself, who am i? then am i a...
5241	31961	1	@user #sikh #temple vandalised in in #calgary,...

5242 rows × 3 columns

Next steps: Generate code with dataset | View recommended plots | New interactive sheet

```
dataset.shape
```

```
(5242, 3)
```

```
dataset.label.value_counts()
```

label	count
0	3000
1	2242

```
import pandas as pd
dataset=pd.read_csv("hate_speech.csv")
dataset
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 #orlandoshooting #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
```

```
import re

# Clean text from noise
def clean_text(text):
    text = re.sub(r'[^a-zA-Z\s]', ' ', text)

    text = re.sub(r'[^\x00-\x7F]+', ' ', text)
    text = text.lower()
    return text
```

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

```
from nltk.corpus import stopwords
len(stopwords.words('english'))
```

```
179
```

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
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
```

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

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

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

```
word_freq=gen_freq(dataset.clean_text.str)
rare_100=word_freq[-100:]
dataset['word_count']=dataset.clean_text.str.split().apply(lambda x:len(x))
dataset['any_neg']=dataset.clean_text.str.split().apply(lambda x:any_neg(x))
dataset['is_question']=dataset.clean_text.str.split().apply(lambda x:is_question(x))
dataset['any_rare']=dataset.clean_text.str.split().apply(lambda x:any_rare(x,rare_100))
dataset['char_count']=dataset.clean_text.apply(lambda x:len(x))
```

```
dataset.head(10)
```

	id	label	tweet	clean_text	word_count	any_neg	is_question	any_rare	char_count
0	1	0	@user when a father is dysfunctional and is s...	user when a father is dysfunctional and is s...	18	0	0	0	102
1	2	0	@user @user thanks for #lyft credit i can't us...	user user thanks for lyft credit i can t us...	21	0	0	0	122
2	3	0	bihday your majesty	bihday your majesty	3	0	0	0	21
3	4	0	#model i love u take with u all the time in ...	model i love u take with u all the time in ...	12	0	0	0	86
4	5	0	factsguide: society now #motivation	factsguide society now motivation	4	0	0	0	39
5	6	0	[2/2] huge fan fare and big talking before the...	huge fan fare and big talking before the...	18	0	0	0	116
6	7	0	@user camping tomorrow @user @user @user @use...	user camping tomorrow user user user use...	11	0	0	0	74
7	8	0	the next school year is the year for exams.ð□□...	the next school year is the year for exams ...	21	0	0	0	143
8	9	0	we won!!! love the land!!! #allin #cavs #champ...	we won love the land allin cavs champ...	10	0	0	0	87
			@user @user welcome here ! i'm it's	user user welcome here i m it s					

Next steps: Generate code with dataset | View recommended plots | New interactive sheet

```
from sklearn.model_selection import train_test_split
X=dataset[['word_count','any_neg','any_rare','char_count','is_question']]
y=dataset.label
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model = model.fit(X_train, y_train)
pred = model.predict(X_test)
model.predict(X_test[5:10])
```

```
array([1, 1, 1, 1, 1])
```

```
from sklearn.metrics import accuracy_score
print("Accuracy:",accuracy_score(y_test,pred)*100,"%")
```

```
Accuracy: 42.89799809342231 %
```

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

```
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
print(classification_report(y_test,rf_pred))
print("Accuracy:",accuracy_score(y_test,rf_pred)*100,"%")
```

```
              precision    recall  f1-score   support

           0       0.64      0.69      0.67       599
           1       0.54      0.49      0.52       450

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

Accuracy: 60.43851286939943 %
```

```
from sklearn.linear_model import LogisticRegression
logreg=LogisticRegression(class_weight='balanced')
logreg.fit(X_train,y_train)
```

```
LogisticRegression
LogisticRegression(class_weight='balanced')
```

```
y_pred = logreg.predict(X_test)
print('accuracy %s' % accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred))
```

```
accuracy 0.5662535748331744
                precision    recall  f1-score   support

             0       0.63      0.58      0.60       599
             1       0.49      0.55      0.52       450

      accuracy                           0.57      1049
     macro avg       0.56      0.56      0.56      1049
  weighted avg       0.57      0.57      0.57      1049
```

```
Start coding or generate with AI.
```