```
import nltk
from nltk import word_tokenize
import sys

nltk.download('stopwords')
nltk.download('punkt')
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
True
```

```
from google.colab import drive
import pandas as pd

drive.mount('/content/gdrive/', force_remount=True)
```

```
Mounted at /content/gdrive/
```

Step 1

```
df = pd.read_csv('/content/gdrive/My Drive/Colab_Notebooks/federalist.csv')
df = df.astype({"author":'category'})
print(df[:10])
authors = {}
for author in df['author']:
  if author in authors.keys():
    authors[author] = authors.get(author, 0) + 1
  else:
    authors[author] = 1
for author in authors:
  print(author + " : " + str(authors.get(author)))
```

```
     author                                               text
0  HAMILTON  FEDERALIST. No. 1 General Introduction For the...
1       JAY  FEDERALIST No. 2 Concerning Dangers from Forei...
2       JAY  FEDERALIST No. 3 The Same Subject Continued (C...
3       JAY  FEDERALIST No. 4 The Same Subject Continued (C...
4       JAY  FEDERALIST No. 5 The Same Subject Continued (C...
5  HAMILTON  FEDERALIST No. 6 Concerning Dangers from Disse...
6  HAMILTON  FEDERALIST. No. 7 The Same Subject Continued (...
7  HAMILTON  FEDERALIST No. 8 The Consequences of Hostiliti...
8  HAMILTON  FEDERALIST No. 9 The Union as a Safeguard Agai...
9   MADISON  FEDERALIST No. 10 The Same Subject Continued (...
```

```
HAMILTON : 49
JAY : 5
MADISON : 15
HAMILTON AND MADISON : 3
HAMILTON OR MADISON : 11
```

Step 2

```
from sklearn.model_selection import train_test_split
X = df.text.values
y = df.author.values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8
print(X_train.shape)
print(X_test.shape)
```

```
(66,)
(17,)
```

Step 3

```
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.corpus import stopwords
stopwords = set(stopwords.words('english'))
vectorizer = TfidfVectorizer(stop_words=stopwords)
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
print(X_train.shape)
print(X_test.shape)
```

```
(66, 7876)
(17, 7876)
```

Step 4

```
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, cc

naive_bayes = BernoulliNB()
naive_bayes.fit(X_train, y_train)

pred = naive_bayes.predict(X_test)
print('accuracy score: ', accuracy_score(y_test, pred))
```

```
    accuracy score:  0.5882352941176471
```

Step 5

```
vectorizer_v2 = TfidfVectorizer(min_df=2, max_df=0.5, ngram_range=(1, 2), stop_words =
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8
X_train = vectorizer_v2.fit_transform(X_train)
X_test = vectorizer_v2.transform(X_test)

naive_bayes.fit(X_train, y_train)

pred = naive_bayes.predict(X_test)
print('accuracy score: ', accuracy_score(y_test, pred))
```

```
    accuracy score:  0.9411764705882353
```

Step 6

No Parameters

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(X_train, y_train)

# evaluate
pred = classifier.predict(X_test)
print('accuracy score: ', accuracy_score(y_test, pred))
```

```
    accuracy score:  0.5882352941176471
```

With Parameters

```
classifier = LogisticRegression(class_weight='balanced', C = 2, )
classifier.fit(X_train, y_train)

# evaluate
pred = classifier.predict(X_test)
print('accuracy score: ', accuracy_score(y_test, pred))
```

```
    accuracy score:  0.8235294117647058
```

Step 7

```
1 from sklearn.neural_network import MLPClassifier
2 classifier = MLPClassifier(random_state = 1234, max_iter=400, hidden_layer_sizes=(30, 2
3 classifier.fit(X_train, y_train)
4 pred = classifier.predict(X_test)
5 print('accuracy score: ', accuracy_score(y_test, pred))
```

```
accuracy score:  0.8823529411764706
```