In [24]:
```
import pandas as pd
df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category',}) # changes to categorical
df.head() #display 1st few rows
```

Out[24]:

	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...

In [25]:
```
df[df.isin(df.author.values)].stack().value_counts() # display counts of author
```

Out[25]:
```
HAMILTON               49
MADISON                15
HAMILTON OR MADISON    11
JAY                     5
HAMILTON AND MADISON    3
dtype: int64
```

In [26]:
```
train = df.sample(frac=0.8,random_state=1234) #divides up the data into train
test = df.drop(train.index) # test data
print(train.shape, test.shape) # shape of train and test
```

```
(66, 2) (17, 2)
```

In [3]:
```
from nltk.corpus import stopwords
from sklearn.naive_bayes import MultinomialNB
import pandas as pd
from nltk import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
import math
stopwords = stopwords.words('english')
vectorizer = TfidfVectorizer(stop_words=stopwords)

df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category'}) # changes to categorical
#train = df.sample(frac=0.8,random_state=1234)
#test = df.drop(train.index)
A = df.text
B = df.author
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.2, train_size=0.
A_train = vectorizer.fit_transform(A_train)  # fit and transform the train data, deals
A_test = vectorizer.transform(A_test)

print(A_train.shape, A_test.shape) # shape of test and train
```

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

In [8]:
```
from nltk.corpus import stopwords
```

```
import pandas as pd
from nltk import word_tokenize
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import math
stopwords = stopwords.words('english')
naive_bayes = MultinomialNB()

df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category'}) # changes to categorical
train = df.sample(frac=0.8,random_state=1234)
test = df.drop(train.index)
vectorizer = TfidfVectorizer(stop_words=stopwords)
A = df.text
B = df.author
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.2, train_size=0.
A_train = vectorizer.fit_transform(A_train)  # fit and transform the train data
A_test = vectorizer.transform(A_test)

naive_bayes.fit(A_train, B_train)
prior_p = sum(B_train == 1)/len(B_train)
pred = naive_bayes.predict(A_test) # a Bernoulli Naïve Bayes model
print('Accuracy score: ', accuracy_score(B_test, pred)) # prints accuracy score
```

```
Accuracy score:  0.5882352941176471
```

In [11]:

```
from nltk.corpus import stopwords
import pandas as pd
from nltk import word_tokenize
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import math
from nltk.util import ngrams
stopwords = stopwords.words('english')
naive_bayes = MultinomialNB()

df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category'}) # changes to categorical
vectorizer = TfidfVectorizer(stop_words=stopwords, ngram_range=(2, 2))
MAX_FEATURES = 1000 # max features variable
vectorizer.max_features = MAX_FEATURES
A = df.text
B = df.author
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.2, train_size=0.
A_train = vectorizer.fit_transform(A_train)  # vectorization
A_test = vectorizer.transform(A_test)


naive_bayes.fit(A_train, B_train)
prior_p = sum(B_train == 1)/len(B_train)
pred = naive_bayes.predict(A_test) # Bernoulli Naïve Bayes model
print('Accuracy score: ', accuracy_score(B_test, pred)) # prints accuracy score
```

```
Accuracy score:  0.5882352941176471
```

In [12]:
```
from sklearn.linear_model import LogisticRegression
from nltk.corpus import stopwords
import pandas as pd
from nltk import word_tokenize
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import math
from nltk.util import ngrams
stopwords = stopwords.words('english')
naive_bayes = MultinomialNB()

df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category'}) # changes to categorical
vectorizer = TfidfVectorizer(stop_words=stopwords, ngram_range=(2, 2), max_features=10
A = df.text
B = df.author
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.2, train_size=0.
A_train = vectorizer.fit_transform(A_train)  # vectorization
A_test = vectorizer.transform(A_test)

naive_bayes.fit(A_train, B_train)
prior_p = sum(B_train == 1)/len(B_train)

# Logistic regression
clf = LogisticRegression(C=2.5, n_jobs=4, solver='lbfgs', random_state=17, verbose=1)
clf.fit(A_train, B_train)
pred = clf.predict(A_test)

print('Accuracy score: ', accuracy_score(B_test, pred)) # prints accuracy score
```

```
[Parallel(n_jobs=4)]: Using backend LokyBackend with 4 concurrent workers.
Accuracy score:  0.6470588235294118
[Parallel(n_jobs=4)]: Done   1 out of   1 | elapsed:    7.5s finished
```

In [15]:
```
from sklearn.linear_model import LogisticRegression
from nltk.corpus import stopwords
import pandas as pd
from nltk import word_tokenize
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, l
from sklearn.pipeline import Pipeline
from sklearn.neural_network import MLPClassifier
import math
from nltk.util import ngrams
stopwords = stopwords.words('english')
naive_bayes = MultinomialNB()

df = pd.read_csv('federalist.csv') # reads in csv file
df = df.astype({"author":'category'}) # changes to categorical
vectorizer = TfidfVectorizer(stop_words=stopwords, ngram_range=(2, 2), max_features=10
A = vectorizer.fit_transform(df.text)
B = df.author
```

```
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.2, train_size=0.

classifier = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(15, 2), ran
classifier.fit(A_train, B_train)
pred = classifier.predict(A_test)
print('accuracy score: ', accuracy_score(B_test, pred))
```

```
accuracy score:  0.6470588235294118
```