```
import pandas as pd

df = pd.read_csv('Data/ham-spam.csv')
df.head()
```

| | IsSpam | Text |
|---|---|---|
| **0** | 0 | key issues going forwarda year end reviews rep... |
| **1** | 0 | congrats contratulations the execution the cen... |
| **2** | 0 | key issues going forwardall under control set... |
| **3** | 0 | epmi files protest entergy transcoattached our... |
| **4** | 0 | california power please contact kristin walsh ... |

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   IsSpam  1000 non-null   int64 
 1   Text    1000 non-null   object
dtypes: int64(1), object(1)
memory usage: 15.8+ KB
```

Check for duplicate rows in the dataset.

```
df.groupby('IsSpam').describe()
```

| | Text | | | |
|---|---|---|---|---|
| | count | unique | top | freq |
| **IsSpam** | | | | |
| **0** | 500 | 499 | paso firm capacity award memorandumlouise del... | 2 |
| **1** | 500 | 500 | you get your order immediately via gra levitr ... | 1 |

```
df = df.drop_duplicates()
df.groupby('IsSpam').describe()
```

| | Text | | | |
|---|---|---|---|---|
| | count | unique | top | freq |
| **IsSpam** | | | | |
| **0** | 499 | 499 | brazil commercial update version delete previ... | 1 |
| **1** | 500 | 500 | you get your order immediately via gra levitr ... | 1 |

```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(ngram_range=(1, 2), stop_words='english')
x = vectorizer.fit_transform(df['Text'])
y = df['IsSpam']
```

Show the vocabulary that `CountVectorizer` built from the training e-mails.

```
vectorizer.vocabulary_
```

```
{'key': 50036,
 'issues': 48310,
 'going': 40119,
 'forwarda': 37692,
 'year': 99702,
 'end': 31413,
 'reviews': 77698,
 'report': 76304,
```

```
        'needs': 60510,
        'generating': 39353,
        'like': 52433,
        'mid': 57723,
        'documenting': 27881,
        'business': 12428,
        'unit': 93760,
        'performance': 66771,
        'review': 77618,
        'completion': 18806,
        'david': 23666,
        'john': 49105,
        'work': 98732,
        'plan': 67886,
        'generation': 39362,
        'nim': 61313,
        'employees': 31202,
        'hpl': 44127,
        'transition': 92241,
        'ongoing': 63430,
        'officially': 63018,
        'transferred': 92194,
        'regardsdelainey': 75386,
        'key issues': 50047,
        'issues going': 48337,
        'going forwarda': 40142,
        'forwarda year': 37693,
        'year end': 99733,
        'end reviews': 31450,
        'reviews report': 77707,
        'report needs': 76342,
        'needs generating': 60519,
        'generating like': 39356,
        'like mid': 52526,
        'mid year': 57738,
        'year documenting': 99731,
        'documenting business': 27882,
        'business unit': 12542,
        'unit performance': 93775,
        'performance review': 66798,
        'review completion': 77630,
        'completion david': 18810,
        'david john': 23680,
        'john work': 49173,
        'work plan': 98799,
        'plan generation': 67922,
        'generation nim': 39379,
        'nim issues': 61314,
        'issues employees': 48325,
```

```
text = vectorizer.transform(['Why pay MORE for * expensive meds when you can ...123... order them online and save $$$?'])
text = vectorizer.inverse_transform(text)
print(text)
```

```
[array(['expensive', 'meds', 'online', 'order', 'order online', 'pay',
       'save'], dtype='<U401')]
```

Split the dataset so that 80% can be used for training and 20% for testing.

```
from sklearn.model_selection import train_test_split

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 MultinomialNB

model = MultinomialNB()
model.fit(x_train, y_train)
```

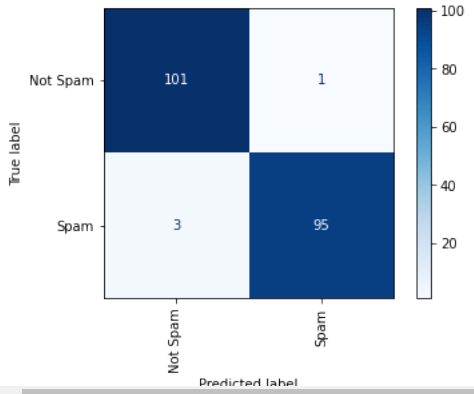
```
MultinomialNB()
```

Validate the trained model with the 20% of the dataset aside for testing and show a confusion matrix.

```
%matplotlib inline
from sklearn.metrics import plot_confusion_matrix

plot_confusion_matrix(model, x_test, y_test, display_labels=['Not Spam', 'Spam'], cmap='Blues', xticks_rotation='vertical')
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x227576de460>
```




```
model.score(x_test, y_test)
```

```
0.98
```

```
from sklearn.metrics import roc_auc_score

probabilities = model.predict_proba(x_test)
roc_auc_score(y_test, probabilities[:, 1])
```

```
0.9992997198879552
```

## Use the model to classify e-mails

```
message = vectorizer.transform(['Can you attend a code review on Tuesday? Need to make sure the logic is rock solid.'])
model.predict(message)[0]
```

```
0
```

```
model.predict_proba(message)[0][0]
```

```
0.9999170457201042
```

```
message = vectorizer.transform(['Why pay more for expensive meds when you can order them online and save $$$?'])
model.predict(message)[0]
```

```
1
```

```
model.predict_proba(message)[0][0]
```

```
0.00021423891260677753
```

```
model.predict_proba(message)[0][1]
```

```
0.9997857610873945
```