Spam email detection

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df = pd.read_csv("spam.csv")
df
```

Out[2]:

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ü b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

In [3]:
```
df.Category.value_counts()
```

Out[3]:
```
Category
ham     4825
spam     747
Name: count, dtype: int64
```

Impliment Train test split

In [6]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df.Message, df.Category, test_size=0.2, random_state=1)
```

In [7]:
```
X_train.shape
```

Out[7]:
```
(4457,)
```

In [8]:
```
X_test.shape
```

Out[8]:
```
(1115,)
```

In [12]:
```
X_train.values
```

Out[12]:
```
array(["Hi , where are you? We're at  and they're not keen to go out i kind of am but feel i shouldn't so ca
n we go out tomo, don't mind do you?",
       'If you r @ home then come down within 5 min',
       "When're you guys getting back? G said you were thinking about not staying for mcr",
       ...,
       'CERI U REBEL! SWEET DREAMZ ME LITTLE BUDDY!! C YA 2MORO! WHO NEEDS BLOKES',
       'Text & meet someone sexy today. U can find a date or even flirt its up to U. Join 4 just 10p. REPLY
with NAME & AGE eg Sam 25. 18 -msg recd@thirtyeight pence',
       'K k:) sms chat with me.'], dtype=object)
```

Create bag of words representation using CountVectorizer

In [9]:
```
from sklearn.feature_extraction.text import CountVectorizer

v = CountVectorizer()
```

In [14]:
```
X_train_cv = v.fit_transform(X_train.values)
X_train_cv.toarray()
```

Out[14]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [16]:
```
X_train_cv.shape
```

Out[16]:
```
(4457, 7711)
```

In [15]:
```
v.get_feature_names_out()
```

Out[15]:
```
array(['00', '000', '008704050406', ..., 'zyada', 'èn', '‧ud'],
      dtype=object)
```

Train the naive bayes model

In [17]:
```
from sklearn.naive_bayes import MultinomialNB

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

Out[17]:
```
MultinomialNB
MultinomialNB()
```

In [19]:
```
X_test_cv = v.transform(X_test)
X_test_cv.toarray()
```

Out[19]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

In [20]:
```
model.score(X_test_cv, y_test)
```

Out[20]:
```
0.989237668161435
```

In [21]:
```
from sklearn.metrics import classification_report

y_pred = model.predict(X_test_cv)

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

         ham       0.99      1.00      0.99       968
        spam       0.98      0.94      0.96       147

    accuracy                           0.99      1115
   macro avg       0.98      0.97      0.98      1115
weighted avg       0.99      0.99      0.99      1115
```

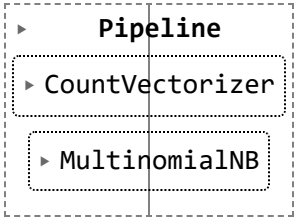
Train the model using sklearn pipeline

In [22]:
```
from sklearn.pipeline import Pipeline

clf = Pipeline([
    ('vectorizer', CountVectorizer()),
    ('nb', MultinomialNB())
])
```

In [23]:
```
clf.fit(X_train, y_train)
```

Out[23]:



In [24]:
```
y_pred = clf.predict(X_test)

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

         ham       0.99      1.00      0.99       968
        spam       0.98      0.94      0.96       147

    accuracy                           0.99      1115
   macro avg       0.98      0.97      0.98      1115
weighted avg       0.99      0.99      0.99      1115
```

In [ ]:
```

```