Chapter 4 - Exercise 3: Spam or ham

Cho dữ liệu spam.csv

Yêu cầu: đọc dữ liệu về, chuẩn hóa dữ liệu (nếu cần) và áp dụng thuật toán Naive Bayes để thực hiện việc dự đoán khả năng email là spam hay không dựa trên các thuộc tính v2

1. Tạo X_train, X_test, y_train, y_test từ dữ liệu đọc được với tỷ lệ dữ liệu test là 0.2
2. Áp dụng thuật toán Naive Bayer => kết quả
3. Đánh giá mô hình
4. Ghi mô hình
5. Đọc mô hình vừa ghi => dự đoán kết quả cho câu 6
6. Cho dữ liệu Test: x_new = np.array(['Dear Ms. Phuong. I will come on time.', 'URGENT! We are trying to contact you. Today is the last day of sale. Discount up to 50%']) => sẽ là ham hay spam?

In [1]:
```
import numpy as np
import pandas as pd
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
```

In [2]:
```
data = pd.read_csv("spam.csv", encoding='latin-1')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
v1            5572 non-null object
v2            5572 non-null object
Unnamed: 2    50 non-null object
Unnamed: 3    12 non-null object
Unnamed: 4    6 non-null object
dtypes: object(5)
memory usage: 217.7+ KB
```

In [3]:
```
data['v1'].head()
```

Out[3]:
```
0     ham
1     ham
2    spam
3     ham
4     ham
Name: v1, dtype: object
```

In [4]:
```
source = data['v2']
type(source)
```

Out[4]:
```
pandas.core.series.Series
```

In [5]:
```
source[:5]
```

Out[5]:
```
0    Go until jurong point, crazy.. Available only ...
1                        Ok lar... Joking wif u oni...
2    Free entry in 2 a wkly comp to win FA Cup fina...
3    U dun say so early hor... U c already then say...
4    Nah I don't think he goes to usf, he lives aro...
Name: v2, dtype: object
```

In [6]:
```
data.groupby('v1').v2.count()
```

Out[6]:
```
v1
ham     4825
spam     747
Name: v2, dtype: int64
```

In [7]:
```
target = data['v1']
type(target)
```

Out[7]:
```
pandas.core.series.Series
```

In [8]:
```
# ham = 0, spam = 1
```

In [9]:
```
target = target.replace("ham", 0)
```

In [10]:
```
target = target.replace("spam", 1)
```

In [11]:
```
target[:5]
```

Out[11]:
```
0    0
1    0
2    1
3    0
4    0
Name: v1, dtype: int64
```

In [12]:
```
temp = pd.DataFrame(target)
```

In [13]:
```
temp.head()
```

Out[13]:

	v1
0	0
1	0
2	1
3	0
4	0

In [14]:
```
text_data = np.array(source)
text_data
```

Out[14]:
```
array(['Go until jurong point, crazy.. Available only in bugis n great world la
e buffet... Cine there got amore wat...',
       'Ok lar... Joking wif u oni...',
       "Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Tex
t FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075ove
r18's",
       ..., 'Pity, * was in mood for that. So...any other suggestions?',
       "The guy did some bitching but I acted like i'd be interested in buying
something else next week and he gave it to us for free",
       'Rofl. Its true to its name'], dtype=object)
```

In [15]:
```
target_data = np.array(target)
target_data
```

Out[15]:
```
array([0, 0, 1, ..., 0, 0, 0], dtype=int64)
```

In [16]:
```
count = CountVectorizer()
count.fit(text_data)
bag_of_words = count.transform(text_data)
bag_of_words
```

Out[16]:
```
<5572x8672 sparse matrix of type '<class 'numpy.int64'>'
	with 73916 stored elements in Compressed Sparse Row format>
```

In [17]:
```
X = bag_of_words.toarray()
X
```

Out[17]:
```
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 [18]:
```
X.shape
```

Out[18]:
```
(5572, 8672)
```

```
In [19]: y = np.array(target)
```

```
In [20]: y.shape
```

```
Out[20]: (5572,)
```

```
In [21]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

```
In [22]: clf = MultinomialNB()
         model = clf.fit(X_train, y_train)
```

```
In [23]: y_pred = clf.predict(X_test)
```

```
In [24]: print('score Scikit learn - train: ', model.score(X_train,y_train))
```

```
score Scikit learn - train:  0.9923715503702042
```

```
In [25]: print('score Scikit learn: ', model.score(X_test,y_test))
```

```
score Scikit learn:  0.9820627802690582
```

```
In [26]: from sklearn.metrics import accuracy_score
         print("Accuracy is ", accuracy_score(y_test,y_pred)*100,"%")
```

```
Accuracy is  98.20627802690582 %
```

```
In [27]: # Nhận xét: Cả training và testing đều có Score cao
```

```
In [28]: from sklearn.metrics import confusion_matrix
```

```
In [29]: confusion_matrix(y_test, y_pred, labels=[0, 1])
```

```
Out[29]: array([[956,  15],
                [  5, 139]], dtype=int64)
```

```
In [30]: # Đánh giá model
         from sklearn. metrics import classification_report, roc_auc_score, roc_curve
```

```
In [31]: print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.99      0.98      0.99       971
           1       0.90      0.97      0.93       144

   micro avg       0.98      0.98      0.98      1115
   macro avg       0.95      0.97      0.96      1115
weighted avg       0.98      0.98      0.98      1115
```

In [32]:
```
# Nhận xét: Có precision cao, recall cao
```

In [33]:
```
y_prob = model.predict_proba(X_test)
y_prob
```

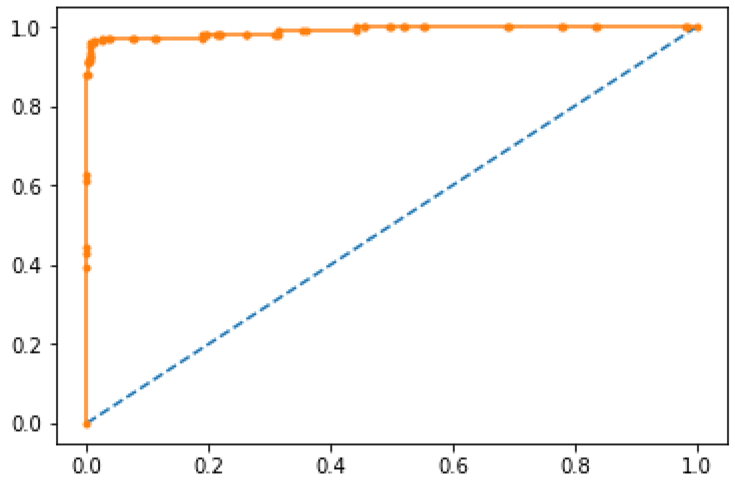
Out[33]:
```
array([[9.99998085e-01, 1.91524638e-06],
       [9.99972774e-01, 2.72257198e-05],
       [9.99995131e-01, 4.86919355e-06],
       ...,
       [9.99999996e-01, 3.55736738e-09],
       [9.94209881e-01, 5.79011898e-03],
       [9.99979973e-01, 2.00266666e-05]])
```

In [34]:
```
roc_auc_score(y_test, y_prob[:, 1])
```

Out[34]:
```
0.99075266048747
```

In [35]:
```
import matplotlib.pyplot as plt
```

In [36]:
```
# calculate roc curve
fpr, tpr, thresholds = roc_curve(y_test, y_prob[:, 1])
# plot no skill
plt.plot([0, 1], [0, 1], linestyle='--')
plt.plot(fpr, tpr, marker='.')
plt.show()
```



In [37]:
```
# ROC cao
# Dựa trên tất cả các đánh giá => Model phù hợp
```

In [38]:
```
# Ghi model
```

In [39]:
```
import pickle
pkl_filename = "ham_spam_model.pkl"
with open(pkl_filename, 'wb') as file:
    pickle.dump(model, file)
```

In [40]:
```
# Đọc model
with open(pkl_filename, 'rb') as file:
    ham_spam_model = pickle.load(file)
```

In [41]:
```
x_new = np.array(['Dear Ms. Phuong. I will come on time.',
                  'URGENT! We are trying to contact you. Today is the last day of
x_new = count.transform(x_new)
```

In [42]:
```
y_pred_new = ham_spam_model.predict(x_new)
y_pred_new
```

Out[42]:
```
array([0, 1], dtype=int64)
```