BUỔI 7 – PHÂN LOẠI SPAM SMS – BÀI 1

1. Giới thiệu bài toán

Tin nhắn rác (spam) thực sự là một vấn đề khó chịu đối với người sử dụng điện thoại di động. Thuật toán Naïve Bayes được sẽ được xây dựng và áp dụng cho việc phân loại tin nhắn rác với:

- Input: Tập văn bản mail tiếng anh và có gán nhãn spam or ham (non-spam~good mail), khoảng hơn 5000 tin nhắn
- Output: Với mỗi văn bản phải xác định loại của văn bản đó là ham or spam

Bài toán đặt ra ở đây là xác định xác suất để một điểm dữ liệu **x** bất kì rơi vào các class 1, 2, 3, ... C. Hay chính xác là đi tính p(y = c|x).

Mục tiêu ở đây, ví dụ với một string là "Tối nay có ăn tối không". Chúng ta sẽ tách string này thành từng từ riêng biệt: "Tối", "nay", "có", ăn", "tối", "không". Làm tương tự vậy với tất cả string. Và tính xác xuất trên tất cả các string với mỗi từ. Cứ hiểu đơn giản ở đây, là từ nào xuất hiện càng nhiều trong các tin nhắn được gán nhãn "ham" thì khi một tin nhắn chưa được gán nhãn chứa từ đấy→ Xác suất nó là tin nhắn "ham" càng cao. Tương tự vậy với các từ trong sms "spam".

2. Giải quyết bài toán

a. Tiền xử lý dữ liệu

```
import pandas as pd
import numpy as np

# đọc dữ liệu từ file
file_path = 'D:/data/SMSSpamCollection.txt'
df = pd.read_csv(file_path, delimiter='\t', header=None, skipinitialspace=True, names=['label', 'msg'])
print(df)
```

Kết quả đọc file

```
     label                                                msg
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
```

b. Giải thuật Machine Learning chỉ làm việc được với số, nên chúng ta sẽ convert "ham", "spam" và cả các sms về định dạng số. Bắt đầu với "ham" (tương ứng với số 0) và "spam" (tương ứng với số 1)

```
# chuyển label thành giá trị nhị phân 0: ham; 1: spam
df['label'] = df.label.map({'ham':0, 'spam':1})
df.head()
```

Tìm hiểu thêm tại:

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.head.html

Kết quả

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

c. Chia tập dữ liệu thành 2 tập: training (70%) và testing (30%)

```
# chia tập dữ liệu thành 2 tập: training data và test data theo tỉ lệ 7: 3
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df.iloc[:,1], df.iloc[:,0], test_size=0.3, random_state=50)
print (X_train.head(10))
print (y_train.head(10))
```

Kết quả

```
2696    And whenever you and i see we can still hook u...
1659    RGENT! This is the 2nd attempt to contact U!U ...
4829    Lol no. Just trying to make your day a little ...
5319                        Kothi print out marandratha.
1394                                             Oh ok..
3251                             Babe, I need your advice
2988    I'm there and I can see you, but you can't see...
298     Hurt me... Tease me... Make me cry... But in t...
3525    Yeah that'd pretty much be the best case scenario
2867                        Smith waste da.i wanna gayle.
Name: msg, dtype: object
2696    0
1659    1
4829    0
5319    0
1394    0
3251    0
2988    0
298     0
3525    0
2867    0
Name: label, dtype: int64
```

d. Chuyển đổi (transform) sms messages thành dạng số. Module mà scikit learn cung cấp cho phép chuyển đổi định dạng text thành vector.

Chúng ta import CountVectorizer và transform text thành vector. Cách transform như sau: có một mảng các string corpus, công cụ này sẽ transform mảng này sao mỗi string sẽ chuyển đổi thành 1 vector có độ dài d (số từ xuất hiện ít nhất 1 lần trong corpus), giá trị của thành phần thứ *i* trong vector chính là số lần từ đó xuất hiện trong string.

Ví dụ:

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
corpus = [
    'This is the first document.',
    'This is the second second document.',
    'And the third one.',
    'Is this the first document?',
]
X = vectorizer.fit_transform(corpus)
print(vectorizer.get_feature_names())
print(X.toarray())
```

Kết quả:

```
['and', 'document', 'first', 'is', 'one', 'second', 'the', 'third', 'this']
[[0 1 1 1 0 0 1 0 1]
 [0 1 0 1 0 2 1 0 1]
 [1 0 0 0 1 0 1 1 0]
 [0 1 1 1 0 0 1 0 1]]
```

+ vectorizer.get_feature_names() đã trả lại kết quả. Đó chính là các **từ** xuất hiện ít nhất 1 lần trong tất cả các string của corpus. Còn phần tử thứ 2 là mảng corpus sau khi transform. Nhìn vào mảng này và so sánh với với result thu được từ vectorizer.get_feature_names(), ta sẽ biết được là string đầu tiên ('This is the first document.') có một từ 'document, 'first', 'is', 'the', 'this'.

```
from sklearn.feature_extraction.text import CountVectorizer
# khởi tạo vectorizer
vect = CountVectorizer()

# đếm các từ trong tập dữ liệu train và chuyển thành ma trận từ
#(learn vocabulary and create document-term matrix in a single step)
train_dtm = vect.fit_transform(X_train)
train_dtm

# chuyển tập dữ liệu text thành ma trận từ (transform testing data into a document-term matrix)
test_dtm = vect.transform(X_test)
test_dtm

# chuyển tập train_dtm thành array
train_arr = train_dtm.toarray()
train_arr
```

e. Sử dụng Naive Bayes để huấn luyện và kiểm tra độ chính xác của Naïve Bayes khi phân loại tin nhắn trong tập test là spam hay không spam

```
from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()
nb.fit(train_dtm, y_train)

# dự đoán dữ liệu trong test (make predictions on test data using test_dtm)
y_pred = nb.predict(test_dtm)
y_pred

# so sánh độ chính xác với các label trong test có sẵn để biết độ chính xác phân loại (compare predictions to true labels)
from sklearn import metrics
print ('accuracy = ',metrics.accuracy_score(y_test, y_pred))
```

Kết quả phân loại sms spam là

```
accuracy =  0.986244019138756
```

***** Toàn bộ code:***

```python
import pandas as pd
import numpy as np

# đọc dữ liệu từ file
file_path = 'F:/data/SMSSpamCollection.txt'
df = pd.read_csv(file_path, delimiter='\t', header=None, skipinitialspace=True, names=['label', 'msg'])
#print(df)

# chuyển label thành giá trị nhị phân 0: ham; 1: spam
df['label'] = df.label.map({'ham':0, 'spam':1})
df.head()

# chia tập dữ liệu thành 2 tập: training data và test data theo tỉ lệ 7: 3
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df.iloc[:,1], df.iloc[:,0], test_size=0.3, random_state=50)
print (X_train.head(10))
print (y_train.head(10))

from sklearn.feature_extraction.text import CountVectorizer
# khởi tạo vectorizer
vect = CountVectorizer()

# đếm các từ trong tập dữ liệu train và chuyển thành ma trận từ
#(learn vocabulary and create document-term matrix in a single step)
train_dtm = vect.fit_transform(X_train)
train_dtm

# chuyển tập dữ liệu text thành ma trận từ (transform testing data into a document-term matrix)
test_dtm = vect.transform(X_test)
test_dtm

# chuyển tập train_dtm thành array
train_arr = train_dtm.toarray()
train_arr

from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()
nb.fit(train_dtm, y_train)

# dự đoán dữ liệu trong test (make predictions on test data using test_dtm)
y_pred = nb.predict(test_dtm)
y_pred

# so sánh độ chính xác với các label trong test có sẵn để biết độ chính xác phân loại (compare predictions to true labels)
from sklearn import metrics
print ('accuracy = ',metrics.accuracy_score(y_test, y_pred))
```