Chapter 10 - Ex3: Spam vs Ham

- Cho dữ liệu spam.csv chứa thông tin là nội dung các email. Bộ dữ liệu này có thể được sử dụng để dự đoán một email gửi đến là ham hay spam.

Yêu cầu:

- Đọc dữ liệu, tìm hiểu sơ bộ về dữ liệu
- Chọn phương pháp để chuẩn hóa dữ liệu và thực hiện việc chuẩn hóa.
- Áp dụng Logistic Regression để xác định một email được gửi đến là ham hay spam
- Với nội dung là: ["Hi, I have received your email. I will send my assigment on time", "Valid 12 hours only."] thì sẽ là ham hay spam?

In [1]:
```
# from google.colab import drive
# drive.mount("/content/gdrive", force_remount=True)
# %cd '/content/gdrive/My Drive/MDS5_2022/Practice_2022/Chapter10/'
```

```
Mounted at /content/gdrive
/content/gdrive/My Drive/MDS5_2022/Practice_2022/Chapter10
```

In [2]:
```
# Load libraries
import numpy as np
import pandas as pd
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
```

In [3]:
```
# import some data to play with
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):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

In [4]:
```
data.head()
```

Out[4]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

In [5]:
```
data = data[["v1", "v2"]]
data.head()
```

Out[5]:

	v1	v2
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...

In [6]:
```
# Count the occurrences of ham and spam print them
occ = data['v1'].value_counts()
print(occ)
```

```
ham     4825
spam     747
Name: v1, dtype: int64
```

In [7]:
```
# Có sự chênh lệch giữa ham và spam
# cần resample dữ liệu
```

In [8]:
```
data.head()
```

Out[8]:

	v1	v2
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...

In [9]:
```
source = data['v2']
type(source)
```

Out[9]:
```
pandas.core.series.Series
```

In [10]:
```
source.head()
```

Out[10]:
```
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 [11]:
```
target = data['v1']
type(target)
```

Out[11]:
```
pandas.core.series.Series
```

In [12]:
```
target.head()
```

Out[12]:
```
0     ham
1     ham
2    spam
3     ham
4     ham
Name: v1, dtype: object
```

In [13]:
```
# 0: ham, 1:spam
target = pd.get_dummies(target, drop_first=True)
target.head()
```

Out[13]:

	spam
0	0
1	0
2	1
3	0
4	0

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
X_train, X_test, y_train, y_test = train_test_split(source, target, test_size=0.
```

In [16]:
```
# Import CountVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# Instantiate CountVectorizer
cv = CountVectorizer(stop_words='english')
cv
```

Out[16]:
```
CountVectorizer(stop_words='english')
```

In [17]:
```
# Fit the vectorizer
cv.fit(source)
```

Out[17]:
```
CountVectorizer(stop_words='english')
```

In [18]:
```
cv.vocabulary_
```

Out[18]:
```
{'jurong': 4224,
 'point': 5741,
 'crazy': 2271,
 'available': 1271,
 'bugis': 1703,
 'great': 3534,
 'world': 8227,
 'la': 4349,
 'buffet': 1701,
 'cine': 1994,
 'got': 3494,
 'amore': 1051,
 'wat': 8026,
 'ok': 5343,
 'lar': 4385,
 'joking': 4192,
 'wif': 8134,
 'oni': 5369,
 'free': 3265,
```

In [19]:
```
# Apply the vectorizer
X_train_transformed = cv.transform(X_train)
# Print the full array
#cv_array = cv_transformed.toarray()
```

In [20]:
```
# resample X_train, y_train
# resample
from imblearn.over_sampling import SMOTE
method = SMOTE()
```

In [21]:
```
# Apply resampling to the training data only
X_resampled, y_resampled = method.fit_resample(X_train_transformed, y_train)
```

In [22]:
```
# Count the occurrences of ham and spam and print them
occ_no_ham = y_resampled[y_resampled==0].size
print("Ham:", occ_no_ham)
occ_no_spam = y_resampled[y_resampled==1].size
print("Spam:", occ_no_spam)
```

```
Ham: 7752
Spam: 7752
```

In [23]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, average_precision_score
from sklearn.metrics import confusion_matrix, precision_recall_curve
from sklearn. metrics import classification_report, roc_auc_score
```

In [24]:
```
# Continue fitting the model and obtain predictions
model = LogisticRegression()
model.fit(X_resampled, y_resampled)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:993: DataCon
versionWarning: A column-vector y was passed when a 1d array was expected. Plea
se change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[24]:
```
LogisticRegression()
```

In [25]:
```
# training score
model.score(X_resampled, y_resampled)
```

Out[25]:
```
0.983488132094943 2
```

In [26]:
```
# testing score
X_test_transformed = cv.transform(X_test)
model.score(X_test_transformed, y_test)
```

Out[26]:
```
0.8995515695067264
```

In [27]:
```
# Get your performance metrics
yhat_test = model.predict(X_test_transformed)
```

In [28]:
```
conf_mat = confusion_matrix(y_true=y_test, y_pred=yhat_test)
print('Confusion matrix:\n', conf_mat)
```

```
Confusion matrix:
 [[851  98]
 [ 14 152]]
```

In [29]:
```
# Calculate average precision and the PR curve
average_precision = average_precision_score(y_test, yhat_test)
average_precision
```

Out[29]:
```
0.5692789453779242
```

In [30]:
```
# Obtain precision and recall
precision, recall, _ = precision_recall_curve(y_test, yhat_test)
precision, recall, _
```

Out[30]:
```
(array([0.14887892, 0.608     , 1.        ]),
 array([1.        , 0.91566265, 0.        ]),
 array([0, 1], dtype=uint8))
```

In [31]:
```
# Obtain model probabilities
probs = model. predict_proba(X_test_transformed)
probs
```

Out[31]:
```
array([[9.99838060e-01, 1.61940185e-04],
       [3.17490737e-01, 6.82509263e-01],
       [9.51294666e-01, 4.87053344e-02],
       ...,
       [9.46113174e-01, 5.38868255e-02],
       [9.92572675e-01, 7.42732474e-03],
       [8.30008980e-01, 1.69991020e-01]])
```

In [32]:
```
# Print ROC_AUC score using probabilities
print(roc_auc_score(y_test, probs[: , 1] ) )
```

```
0.97133952035751
```

In [33]:
```
# Print classification report using predictions
print(classification_report(y_test, yhat_test))
```

```
              precision    recall  f1-score   support

           0       0.98      0.90      0.94       949
           1       0.61      0.92      0.73       166

    accuracy                           0.90      1115
   macro avg       0.80      0.91      0.83      1115
weighted avg       0.93      0.90      0.91      1115
```

In [34]:
```
new_data = pd.Series(["Hi, I have received your email. I will send my assigment (
                      "Valid 12 hours only."])
```

In [35]:
```
new_data_transformed = cv.transform(new_data)
```

In [36]:
```
new_data_transformed
```

Out[36]:
```
<2x8404 sparse matrix of type '<class 'numpy.int64'>'
        with 8 stored elements in Compressed Sparse Row format>
```

In [37]:
```
yhat_new = model.predict(new_data_transformed)
yhat_new
```

Out[37]:
```
array([0, 1], dtype=uint8)
```

In [37]:
```

```