Spam Classification using scikit-learn

November 30, 2019

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

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import recall_score, precision_score, f1_score, confusion_matrix
```

In [2]:
```
DATA_JSON_FILE = 'SpamData/01_Processing/email-text-data.json'
```

In [3]:
```
data = pd.read_json(DATA_JSON_FILE)
```

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

Out[4]:

	CATEGORY	MESSAGE	FILE_NAME
995	1	This is a multi-part message in MIME format.\n...	00208.c9e30fc9044cdc50682c2e2d2be4c466
996	1	I have been receiving emails saying that I'm c...	00237.0faf46ae2bfab24f0464c4a1a0425659
997	1	This is a multi-part message in MIME format.\n...	00224.1b3430b101a8a8b22493c4948fcbe9cc
998	1	<html>\n\n<head>\n\n<title>mailv05a.gif</title...	01381.7f1f9f4b8ea24fee6b87dc1172177eaf
999	1	Dear,sir\n\n \n\nI am Chief Tony Anenih,the mi...	00612.cd362b97ee34d41e72a66ed5199dd62e

In [5]:
```
data.shape
```

Out[5]:
```
(5796, 3)
```

In [6]:
```
data.sort_index(inplace=True)
```

```
In [7]: data.tail()
```

```
Out[7]:       CATEGORY                                            MESSAGE  \
        5791         0  On Tue, Jul 30, 2002 at 11:28:11AM +0200, Davi...
        5792         0  Paul, my apologies for being irritable on the ...
        5793         0  Yes indeed - there should be an agents directo...
        5794         0  There's been some discussion just now on the I...
        5795         0  On Fri, 2002-07-26 at 12:49, Ian Andrew Bell w...

                                          FILE_NAME
        5791  00154.7bda4738681c601e0fd93f3c6d1ae4a1
        5792  00363.2c66a99268facef9c5dab8c1f7b81190
        5793  00449.9272eb34ed6d02f46e34bd7300d9e7d8
        5794  00062.43847c613a539ca9c47b4593ee34bd6d
        5795  00912.74b435accaf4e65a948c7769b6380f01
```

```
In [8]: vectorizer = CountVectorizer(stop_words='english')

In [9]: all_features = vectorizer.fit_transform(data.MESSAGE)

In [10]: all_features.shape

Out[10]: (5796, 102694)

In [11]: type(vectorizer.vocabulary_)

Out[11]: dict

In [12]: X_train, X_test, y_train, y_test = train_test_split(all_features, data.CATEGORY, test_

In [13]: X_train.shape

Out[13]: (4057, 102694)

In [14]: X_test.shape

Out[14]: (1739, 102694)

In [15]: classifiers = {'Naive Bayes': MultinomialNB(), 'Decision Tree':DecisionTreeClassifier

In [16]: for key, value in classifiers.items():
             classifier = value
             classifier.fit(X_train, y_train)
             mean_accuracy = classifier.score(X_test, y_test)
             recall = recall_score(y_test, classifier.predict(X_test))
             precision = precision_score(y_test, classifier.predict(X_test))
             f1 = f1_score(y_test, classifier.predict(X_test))
             print()
             print('{} classifier'.format(key))
             print('Score: {}'.format(mean_accuracy))
```

```
                print('Recall: {}'.format(recall))
                print('Precision score: {}'.format(precision))
                print('F1 score: {}'.format(f1))
                cf = confusion_matrix(y_test, classifier.predict(X_test))
                print(cf)
```

```
Naive Bayes classifier
Score: 0.9361702127659575
Recall: 0.8068592057761733
Precision score: 0.991130820399113
F1 score: 0.8895522388059702
[[1181    4]
 [ 107  447]]

Decision Tree classifier
Score: 0.9620471535365153
Recall: 0.9494584837545126
Precision score: 0.9326241134751773
F1 score: 0.9409660107334527
[[1147   38]
 [  28  526]]

Support vector Machine classifier
Score: 0.8320874065554916
Recall: 0.47653429602888087
Precision score: 0.9924812030075187
F1 score: 0.6439024390243903
[[1183    2]
 [ 290  264]]
```