Université Paul Sabatier
EMMAB2C1 : Algorithmes de classification, data mining et text mining

Enseignant : José G. Moreno

24/2/2017

TP 3. Détection de courrier indésirable

Dans ce TP de détection d'anomalie, vous êtes chargé de trouver du courrier indésirable dans un corpus de courrier électronique.

Nous devons prédire pour 1500 mails (le jeu de *test*) s'ils sont du courrier indésirable (*spam*) ou du courrier désirable (*ham*). Nous avons 2827 mails étiquetés pour entraîner un modèle (le jeu de *training*). Nous avons à la fois l'objet et le contenu du courrier électronique. Les données sont stockées au format csv:
La première colonne est un numéro identifiant id
La deuxième colonne est l'objet du courrier électronique
La troisième colonne est le texte du courrier électronique
La quatrième colonne est l'étiquette du mail (0 pour *ham* et 1 pour *spam*)

Nous avons deux fichiers *training_info_tp3.csv* et *test_info_tp3.csv*. Comme leur nom l'indique, le premier contient les données de *training* et le deuxième les données de *test*. La quatrième colonne n'est pas incluse dans les données de test.

Un classifieur très simple est fourni, le fichier *simpleclassifier.py.*

```
import csv,sys
from sklearn.svm import SVC
from sklearn.feature_extraction.text import TfidfVectorizer

csv.field_size_limit(sys.maxsize)

#Read the training data from csv file
f = open('training_info_tp3.csv', 'r')

reader = csv.reader(f, delimiter=',', quotechar='"')
#X is used to represent the training data
X=[]
#Y is used to save the training labels
y=[]
for row in reader:
  X.append(row[1])
  y.append(row[3])

#A tfidf representation fo text documents is used
tfidf_v = TfidfVectorizer(max_df=0.95, min_df=2, max_features=100000,
stop_words='english')
tfidf = tfidf_v.fit_transform(X)

#A very simple classifier
clf = SVC()
```

```
clf.fit(tfidf, y)


#Read the test data
f = open('test_info_tp3.csv', 'r')
reader = csv.reader(f, delimiter=',', quotechar='"')
Xtest=[]
for row in reader:
  Xtest.append(row[1])

#The same tfidf representation is used
tfidf_test = tfidf_v.transform(Xtest)

#Labels are predicted to the tfidf data
pred_test=clf.predict(tfidf_test)
```

Ce classifieur n'est pas en mesure de faire un bon travail. Pour mesurer la performance de l'algorithme, nous utiliserons la métrique *f1_score*. La performance pour l'implémentation fournie est *f1_score [micro] = 0.6813*.

Si vous voulez connaître la performance de votre classifieur, envoyez-le à l'aide de moodle et demandez à votre enseignant pour l'évaluer. Avec une modification de moins de 5 lignes de code, vous pouvez obtenir f1_score [micro] = 0.9653. Pouvez-vous les trouver ? La performance optimale devrait être de 0,99. Essayez d'obtenir un performance optimale.

On peut utiliser la cross validation sur le jeu de données de training pour éviter le suraprentissage. Utilise *GridSearchCV* ou *cross_val_score* pour trouver les meilleurs paramètres avant de faire un soumission.

Comparez les résultats obtenues avec *Scikitlearn* contre ce de *Weka* avec la cross validation.