TD1 - Introduction au TAL
TF-IDF et classification automatique

1. Introduction

Ce TP porte sur un problème de détection de sentiments dans des Tweets. L'analyse des sentiments est un domaine de recherche extrêmement actif en Traitement automatique du langage (TAL). En effet, ces dernières années ont vu se multiplier les sources de données textuelles porteuses d'opinion disponibles sur le Web (Twitter, Facebook, Instagram, WhatsApp...). Devant cette abondance de données et de sources, l'automatisation de la synthèse des multiples avis devient cruciale pour obtenir efficacement une vue d'ensemble des opinions sur un sujet donné.

L'objectif de ce TP est de classifier les tweets selon l'opinion/sentiment/émotion exprimé par son auteur, en l'occurrence ici : objectif, positif, négatif ou mixte (si le tweet contient à la fois des opinions positives et des opinions négatives).

Vous devez créer un système de détection automatique de sentiment en appliquant la modélisation par sac-de-mots (ici, TF-IDF) et un classificateur de type SVM (Séparateur à Vaste Marge) à un corpus de tweets. Nous vous invitons à travailler ici soit sur votre environnement Linux ou bien sur la plateforme (https://colab.research.google.com/).

2. Mise en oeuvre

Données. Pour réaliser ce TP vous avez à votre disposition un corpus de tweets en français (disponible dans l'espace de cours sur madoc.univ-nantes.fr). Ce corpus, qui provient de la conférence DEFT 2017, est séparé en deux parties :

- Un jeu d'apprentissage : task1-train.csv
- Un jeu de test : task1-testGold.csv

Les deux fichiers sont structurés de la même façon, en trois colonnes séparées par des tabulations : numéro du tweet dans le jeu, contenu textuel du tweet, et sentiment associé au tweet.

Attention : le classifieur doit être entraîné sur le corpus d'apprentissage et évalué sur le corpus de test. Les données de test ne doivent surtout pas être intégrées dans les données d'apprentissage.

Etapes. De façon classique, les différentes étapes à réaliser sont les suivantes :

1. Chargez les deux fichiers correspondant au jeu d'apprentissage et au jeu de test.
2. Extrayez le lexique du corpus entier (apprentissage et test), c'est-à-dire la liste de tous les mots apparaissant au moins une fois dans un tweet.
3. Assignez un numéro unique à chaque mot du lexique.
4. Décomptez, pour chaque message du corpus entier, le score TF.IDF de chaque mot qu'il contient.

5. Convertissez séparément les deux jeux de données (apprentissage et test) de façon à obtenir deux fichiers au format supporté par le SVM.

6. Appliquez le classifieur aux fichiers obtenus.

Ces différentes étapes peuvent être simplifiées au moyen de la librairie *sklearn* (https://scikit-learn.org/). Vous aurez notamment besoin des outils suivants, que l'on peut résumer par les imports suivants :

- *from sklearn.feature_extraction.text import TfidfVectorizer* : permet de calculer les scores TF.IDF dans un tableau de données.

- *from sklearn.svm import SVC* : permet d'appliquer l'apprentissage et le test d'un classifieur sur un tableau de données et le tableau de classes associées.

- *from sklearn.metrics import accuracy_score, classification_report* : permet de calculer la performance des prédictions (i.e. valeurs prédites par le classifieur).

Il sera également nécessaire d'utiliser la librairie *pandas* :

- *import pandas as pd* : permet de traiter/charger les fichiers .csv

3. Améliorations

Quelques pistes possibles pour améliorer votre système :

- Normaliser les données :
  - Casse : supprimer la distinction entre minuscules et majuscules.
  - Stemming (racinisation) : ne garder que la racine des mots.
- Supprimer les mots-outils[1] : prépositions, conjonctions, pronoms, etc

[1] Par exemple : https://github.com/stopwords-iso/stopwords-fr/tree/master/raw