Dpt Informatique

Année universitaire	**2023-2024**		
Filière	**Data Science**	**Année**	**M2**
Matière	**Machine Learning 2**		
Enseignant	**Haytham Elghazel**		
Intitulé TD/TP :	**Atelier 1 : Traitement de données Textuelles**		
Contenu	• **Préparation de données textuelles** • **Vectorisation de données textuelles** • **TFIDF, LSA** • **Word2vec** • **Apprentissage sur des données textuelles**		

Dans cet atelier pratique, l'objectif dans cette partie est de faire une étude comparative entre plusieurs algorithmes d'apprentissage supervisé sur différents jeux de données textuelles avec le langage **Python**.

Pour lancer le notebook Python, il faut taper la commande **jupyter notebook** dans votre dossier de travail. Une fenêtre va se lancer dans votre navigateur pour ouvrir l'application Jupyter. Créer un nouveau notebook Python et taper le code suivant dans une nouvelle cellule :

```
import numpy as np
np.set_printoptions(threshold=10000, suppress = True)
import pandas as pd
import warnings
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
```

L'objectif dans cette partie est d'apprendre sur un jeu de données de *commentaires* du fichier **"yelp-text-by-stars.csv"** en suivant les étapes suivantes. *Votre code doit bien être factoriser en proposant plusieurs mini-fonctions pour chaque traitement proposé.*

- **Importer** ce jeu de données avec la librairie pandas (**c.f. read_csv**)
- **Analyser** votre jeu de données, essentiellement la **target**.
- **Modéliser** le problème d'apprentissage supervisé sur ces données.
- **Traiter** vos données textuelles en supprimant les bruits dans les textes et en les normalisant. Vous pouvez vous inspirer par exemple du code par ici[1] (en l'améliorant s'il le faut) si vous utiliserez la librairie **NLTK**. Vous pouvez aussi utiliser d'autres librairies (A vos recherches). N'oubliez pas que cette **étape de pré-traitement (preprocessing)** dépend de vos données et du problème traité. Quelques traitements à faire sont :
  - Suppression des ponctuations comme **. , ! $( ) * % @**
  - Suppression des URLs
  - Suppression des Stop words
  - Transformation de tout le texte en minuscule.
  - Tokenisation de vos textes
  - Racinisation (Stemming)
  - Lemmatisation (lemmatization)
  - Etc.

- **Séparer** les données en jeu de données d'**apprentissage** et jeu de données de **test**.
- **Proposer** une fonction run_models permettant de comparer plusieurs modèles d'apprentissage (en fonction de votre modélisation) sur ces données dont une **forêt aléatoire**, un **réseau de neurones**

[1] https://www.kdnuggets.com/2018/03/text-data-preprocessing-walkthrough-python.html

à deux couches et un **k-plus proche voisin**.

- **Proposer** une première **vectorisation** de vos données textuelles par une représentation **bag-of-words**.
- **Exécuter** ensuite votre fonction **run_models** sur vos données et interpréter les résultats obtenus. Il est nécessaire d'appliquer votre fonction sur les **données pré-traitées** et les **données non pré-traitées** afin d'analyser l'apport de la partie pré-traitement de données.
- **Améliorer** la vectorisation proposée par la technique **Tf–idf**. Exécuter ensuite à nouveau votre fonction **run_models** sur vos données et interpréter les résultats obtenus en les comparant à ceux obtenus à l'étape précédente. Il est nécessaire toujours d'appliquer votre fonction sur les **données pré-traitées** et les **données non pré-traitées** afin d'analyser l'apport de la partie pré-traitement de données.
- **Proposer** une approche de **sélection de mots clés pertinents** (***sélection de variables***). Tester l'apport de votre sélection de variables.
- **Appliquer** la méthode **SVD** de **réduction de dimensions** (**TruncatedSVD**) afin de construire des "**concepts**" liés aux documents et aux termes. Elle permettra entre autres de résoudre les problèmes de **synonymie** (plusieurs mots avec un seul sens) et de **polysémie** (un seul mot avec plusieurs sens).
- **Dans un nouveau notebook**, **proposer** un code qui permettra d'apprendre votre propre modèle de plongement lexical **Word2Vec** sur vos données textuelles. **Évaluer visuellement et numériquement** sur quelques mots clés votre nouveau modèle de vectorisation (*Embedding*).
- Exploiter votre modèle **Word2Vec** pour la vectorisation de vos textes (avec deux méthodes utilisant ou non le TF-IDF des mots). Exécuter ensuite à nouveau votre fonction **run_models** sur vos données vectorisées par **Word2Vec** et interpréter les résultats obtenus en les comparant à ceux obtenus aux étapes précédentes.
- Idem en utilisant le **modèle Word2Vec pré-entrainé de Google ou un autre**.
- ***Pipeline* :** Automatiser l'enchainement de votre meilleur traitement dans un pipeline