PROJET DE FIN DU MODULE

« NATURAL LANGUAGE PROCESSING »

MASTER SPECIALISE EN « INGENIERIE DE DONNEES ET DEVELOPPEMENT LOGICIEL »

MINI PROJET SUR SENTIMENT ANALYSIS

Realisé par :

El Aissi Mohamed

Abarkane Issam

Encadre par :

Pr.Mahmoudi Abdelhak

Année universitaire : ***2020-2021***

I. Introduction :

L'analyse des sentiments fait référence à l'identification et à la classification des sentiments exprimés dans la source du texte. Les tweets sont souvent utiles pour générer une grande quantité de données de sentiment lors de l'analyse. Ces données sont utiles pour comprendre l'opinion des gens sur une variété de sujets. Par conséquent, nous devons développer un modèle d'analyse des sentiments d'apprentissage automatique afin de calculer la perception du client. En raison de la présence de caractères non utiles (collectivement appelés bruit) ainsi que de données utiles, il devient difficile de mettre en œuvre des modèles sur eux. Dans cet article, nous visons à analyser le sentiment des tweets fournis à partir de l'ensemble de données Sentiment140 en développant un pipeline d'apprentissage automatique impliquant l'utilisation de trois classificateurs (Régression logistique, Bernoulli Naive Bayes et SVM) ainsi que l'utilisation de Term Frequency-Inverse Document Frequency (TF-IDF) . Les performances de ces classificateurs sont ensuite évaluées à l'aide de la précision et des scores F1.

II. Problématique :

Dans ce projet, nous essayons de mettre en œuvre un **modèle d'analyse des sentiments Twitter** qui aide à surmonter les défis d'identification des sentiments des tweets. Les détails nécessaires concernant l'ensemble de données sont :

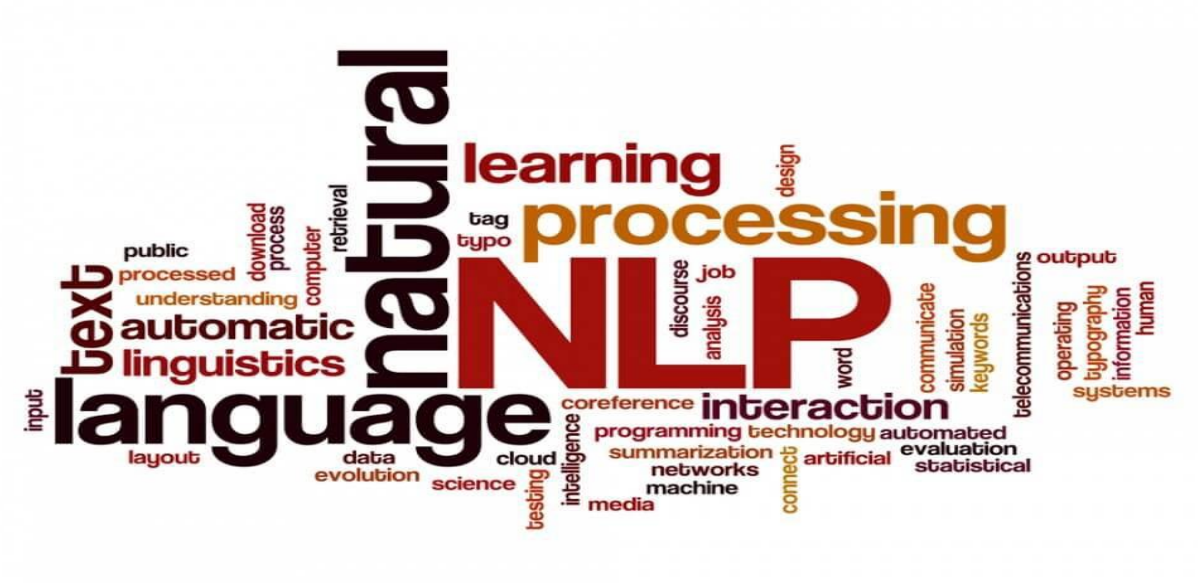
L'ensemble de données fourni est l'ensemble de données **Sentiment140** qui se compose de **1 600 000 tweets** extraits à l'aide de l'API Twitter. Les différentes colonnes présentes dans le dataset sont :

- **cible :** la polarité du tweet (positive ou négative)
- **ids : identifiant** unique du tweet
- **date :** la date du tweet
- **flag :** Il fait référence à la requête. Si aucune requête de ce type n'existe, il s'agit de NO QUERY.
- **user :** Il s'agit du nom de l'utilisateur qui a tweeté
- **text :** il fait référence au texte du tweet

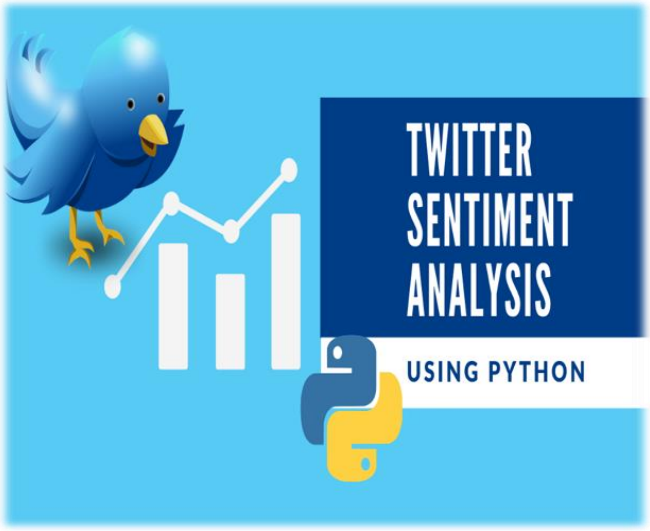
Qu'est-ce que le traitement du langage naturel :

Le traitement du langage naturel (TAL) fait référence à la branche de l'informatique, et plus précisément à la branche de l' intelligence artificielle ou IA, qui vise à donner aux ordinateurs la capacité de comprendre du texte et des mots prononcés de la même manière que les êtres humains.
La NLP combine la linguistique informatique (modélisation du langage humain basée sur des règles) avec des modèles statistiques, d'apprentissage automatique et d'apprentissage en profondeur. Ensemble, ces technologies permettent aux ordinateurs de traiter le langage humain sous forme de texte ou de données vocales et de « comprendre » sa pleine signification, avec l'intention et le sentiment de l'orateur ou de l'écrivain.

La NLP pilote des programmes informatiques qui traduisent le texte d'une langue à une autre, répondent aux commandes vocales et résument rapidement de gros volumes de texte, même en temps réel. Il y a de fortes chances que vous ayez interagi avec la NLP sous la forme de systèmes GPS à commande vocale, d'assistants numériques, de logiciels de dictée vocale, de chatbots de service client et d'autres commodités pour les consommateurs. Mais la NLP joue également un rôle croissant dans les solutions d'entreprise qui aident à rationaliser les opérations commerciales, à augmenter la productivité des employés et à simplifier les processus métier critiques.



- **<u>Sentiment Analysis (analyse des sentiments) :</u>**



L’analyse de sentiment est une tâche de traitement automatique des langues et d’extraction d’information. Pour un texte donné, il faut identifier la polarité du texte comme étant soit positif, soit négatif. Pang et Lee (Pang, Bo, and Lillian Lee, 2008) indiquent plusieurs méthodes, plusieurs références de performance et ressources pour réaliser cette tâche. La polarité d’un sentiment peut être calculée selon plusieurs seuils et peut être vue comme plusieurs différentes classes. Dans notre projet, nous considérons les textes comme pouvant appartenir à seulement deux classes (classification binaire) : soit le texte est positif, soit le texte est négatif

III. Les outils :

a. *Langage de programmation :*

Python est connu depuis longtemps comme un langage de programmation simple à maîtriser, du point de vue de la syntaxe. Il possède une communauté active et un vaste choix de bibliothèques et de ressources. On dispose donc d'une plate-forme de programmation capable d'être utilisée avec les technologies émergentes telles que l'apprentissage automatique et la Data Science.

b) *Notebook :*

La Data Science est itérative : il faut souvent tenter plusieurs approches et étudier les résultats avant de décider de la bonne façon de traiter un problème. C'est la raison pour laquelle les notebooks sont parfaitement adaptés à cette particularité. Un **notebook** est une interface web dans laquelle on peut taper du code Python, l'exécuter et voir directement les résultats, y compris une visualisation à l'aide de graphiques. Dans notre cas, nous avons choisi de travailler avec **Jupyter Notebook**.

c) *Librairies utilisées :*

Parmi les librairies Python que nous avons utilisées, on trouve :

• **NLTK :** C'est une librairie fondamentale pour la construction de programmes Python pour travailler avec des données de langage humain. Elle offre des

interfaces faciles à utiliser sur des corpus ou ressources lexicales telles que WordNet, ainsi que des outils pour le traitement de texte, la classification, la tokenisation, le stemming, le balisage, l'analyse et le raisonnement sémantique.

• **SKLEARN :** C'est une librairie incontournable en Machine Learning et très bien documentée, destinée à **l'apprentissage automatique**. Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification ainsi que les machines à vecteurs de support.

• **Numpy :** C'est une librairie fondamentale pour effectuer des **calculs numériques** avec
Python. Elle facilite grandement la **gestion des tableaux de données** et met à disposition également tout un **arsenal de fonctions pour effectuer des calculs mathématiques complexes** comme les fonctions trigonométriques ou encore les fonctions exponentielles et
logarithmes.

• **Pandas :** C'est une **librairie très utilisée en data science** qui permet la manipulation et
l'analyse des données. Elle propose en particulier des structures de données et des opérations
de manipulation de tableaux numériques et de séries temporelles.

• **WordCloud.**

IV. Implémentation :

Étape 1 : Importer les dépendances nécessaires :

```
# utilities
import re
import numpy as np
import pandas as pd
# plotting
import seaborn as sns
from wordcloud import WordCloud
import matplotlib.pyplot as plt
# nltk
from nltk.stem import WordNetLemmatizer
# sklearn
from sklearn.svm import LinearSVC
```

```
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, classification_report
```

Étape 2 : lire et charger l'ensemble de données

```
# Importing the dataset
DATASET_COLUMNS=['target','ids','date','flag','user','text']
DATASET_ENCODING = "ISO-8859-1"
df = pd.read_csv('Project_Data.csv', encoding=DATASET_ENCODING, names=DATASET_
COLUMNS)
df.sample(5)
```

Résultat :

Out[2]:

	target	ids	date	flag	user	text
232471	0	1979174451	Sun May 31 01:56:21 PDT 2009	NO_QUERY	fall79	yup the crying has started. Finished part I an...
1246014	4	1995212719	Mon Jun 01 13:07:45 PDT 2009	NO_QUERY	psdrea	Its a fact you can have your cake & eat it...
596382	0	2219010405	Wed Jun 17 23:36:19 PDT 2009	NO_QUERY	butifful	@0mGiiTzRee BFF U HAVE 190 N I HAVE 109 (FOLLO...
953768	4	1824712045	Sun May 17 02:35:42 PDT 2009	NO_QUERY	p0ssumman	@modom hi modom, welcome to Yorkshire forums ...
1076594	4	1967471968	Fri May 29 19:38:35 PDT 2009	NO_QUERY	OrionPR	Listening to old-school New Orleans jazz on my...

Étape 3 : Analyse exploratoire des données

```
df.head()
```

Résultat :

Out[2]:

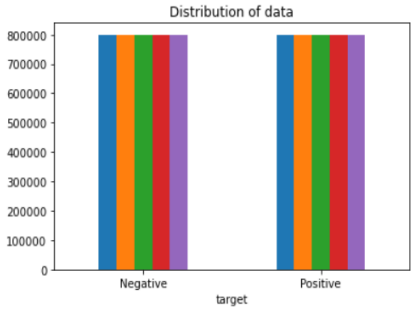
	target	ids	date	flag	user	text
232471	0	1979174451	Sun May 31 01:56:21 PDT 2009	NO_QUERY	fall79	yup the crying has started. Finished part I an...
1246014	4	1995212719	Mon Jun 01 13:07:45 PDT 2009	NO_QUERY	psdrea	Its a fact you can have your cake & eat it...
596382	0	2219010405	Wed Jun 17 23:36:19 PDT 2009	NO_QUERY	butifful	@0mGiiTzRee BFF U HAVE 190 N I HAVE 109 (FOLLO...
953768	4	1824712045	Sun May 17 02:35:42 PDT 2009	NO_QUERY	p0ssumman	@modom hi modom, welcome to Yorkshire forums ...
1076594	4	1967471968	Fri May 29 19:38:35 PDT 2009	NO_QUERY	OrionPR	Listening to old-school New Orleans jazz on my...

Étape 4 : Visualisation des données des variables cibles

```
# Plotting the distribution for dataset.
ax = df.groupby('target').count().plot(kind='bar', title='Distribution of data
',legend=False)
ax.set_xticklabels(['Negative','Positive'], rotation=0)
# Storing data in lists.
text, sentiment = list(df['text']), list(df['target'])
```

Entrée [13]:

```
# Plotting the distribution for dataset.
ax = df.groupby('target').count().plot(kind='bar', title='Distribution of data',legend=False)
ax.set_xticklabels(['Negative','Positive'], rotation=0)
# Storing data in lists.
text, sentiment = list(df['text']), list(df['target'])
```



Étape 5 : Prétraitement des données :

Dans l'énoncé du problème ci-dessus avant d'entraîner le modèle, nous avons effectué diverses étapes de pré-traitement sur l'ensemble de données qui traitaient principalement de la suppression des mots vides, de la suppression des emojis. Le document texte est ensuite converti en minuscules pour une meilleure généralisation.

Par la suite, les ponctuations ont été nettoyées et supprimées, réduisant ainsi le bruit inutile de l'ensemble de données. Après cela, nous avons également supprimé les caractères répétitifs des mots ainsi que les URL car elles n'ont pas d'importance significative.

Enfin, nous avons ensuite effectué Stemming (réduire les mots à leurs racines dérivées) et Lemmatisation (réduire les mots dérivés à leur forme racine connue sous le nom de lemme) pour de meilleurs résultats.

Étape 6 : Répartir nos données en sous-ensembles d'entraînement et de test

```
# Séparer les données de 95% pour les données d'entraînement et 5% pour les do
nnées de test
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.05, rand
om_state =26105111)
```

Étape 7 : Transformation de l'ensemble de données à l'aide de TF-IDF Vectorizer

7.1 : Installer le vectoriseur TF-IDF

```
vectoriser = TfidfVectorizer(ngram_range=(1,2), max_features=500000)
vectoriser.fit(X_train)
print('No. of feature_words: ', len(vectoriser.get_feature_names()))
```

7.2 : Transformer les données à l'aide de TF-IDF Vectorizer

```
X_train = vectoriser.transform(X_train)
X_test = vectoriser.transform(X_test)
```

Étape 8 : fonction pour l'évaluation du modèle

Après avoir formé le modèle, nous appliquons ensuite les mesures d'évaluation pour vérifier les performances du modèle. En conséquence, nous utilisons les paramètres d'évaluation suivants pour vérifier les performances des modèles respectivement :

- Score de précision
- Matrice de confusion avec intrigue
- Courbe ROC-AUC

Étape 9 : Construire un modèle

Dans l'énoncé du problème, nous avons utilisé respectivement trois modèles différents :

Bernoulli Naïf Bayes

SVM (Machine à vecteurs de soutien)

Régression logistique

L'idée derrière le choix de ces modèles est que nous voulons essayer tous les classificateurs de l'ensemble de données allant des modèles simples aux modèles complexes, puis essayer de trouver celui qui donne les meilleures performances parmi eux.

Étape 10 : Conclusion

Après avoir évalué tous les modèles, nous pouvons conclure les détails suivants, c'est-à-dire

Précision : En ce qui concerne la précision du modèle, la régression logistique est plus performante que SVM, qui à son tour est plus performante que Bernoulli Naive Bayes.

F1-score : Les scores F1 pour la classe 0 et la classe 1 sont :
(a) Pour la classe 0 : Bernoulli Naive Bayes (précision = 0,90) < SVM (précision = 0,91) < Régression logistique (précision = 0,92)
(b) Pour la classe 1 : Bernoulli Naive Bayes (précision = 0,66) < SVM (précision = 0,68) < Régression logistique (précision = 0,69)

Score AUC : les trois modèles ont le même score ROC-AUC.

Nous concluons donc que la régression logistique est le meilleur modèle pour l'ensemble de données ci-dessus.

Dans notre énoncé de problème, la **régression logistique** suit le principe du **rasoir** d' **Occam** qui définit que pour un énoncé de problème particulier, si les données n'ont pas d'hypothèse, alors le modèle le plus simple fonctionne le mieux. Étant donné que notre ensemble de données ne comporte aucune hypothèse et que la régression logistique est un modèle simple, le concept est donc vrai pour l'ensemble de données mentionné ci-dessus.