# NLP TD n°1

## Partie 1 :

**Les a priori :** Tout d’abord j’ai déjà travaillé sur un sujet de Natural Langage Processing, c’était une compétition Kaggle sur un corpus de reviews provenant du site Goodreads.

Je sais donc à peu près, ce que je dois faire en pré-traitement de mes données et quels modèles utilisés.

Pré-traitements :

* Supprimer les stopwords (« le », « les », « ce », …) tous les mots qui ne contribuent pas au sens d’une phrase.
* Faire du Stemming changer tous les mots qui sont dans un même lexique par la racine de ceux-ci.
* Suppression de la ponctuation

Modèles :

* Residual Neural Network : réseau de neurone sont beaucoup utilisés dans des tâches de NLP, donc je pense l’utiliser
* Transformers : pareil que pour les RNN, ce sont des modèles qui sont les plus utilisés dans les tâches de traitements du langage.

**Partie train :**Ensuite pour la partie train, j’ai changé le retour de la fonction, j’utilise la library joblib pour créer un dump de mon model que je passe en paramètre dans ma commande.

Dans « make\_model » j’utilise le model avec ce même paramètre pour que ça soit plus dynamique.

Les modèles que nous avons utilisés et leurs résultats sont les suivants avec le pré-traitement que nous avons évoqué dans la partie a priori :

Logistic Regression : 91.19497487437187%

Random Forest Classifier : 91.29346733668342%

SVM : 91.49547738693468%

Multinomial Naive Bayes : 91.49396984924623%

XGBoost : 93.39597989949749%

Nous pouvons remarquer que le model XGBoost est le modèle avec la meilleure accuracy.

J’ai utilisé que des modèles faciles à implémenter avec la library sklearn c’est pour cela que je n’ai pas utilisé le modèle transformer.

## Partie 2 :

**Les a priori :** Pour moi les deux features "is\_final\_word" et "is\_starting\_word" sont très utiles dans un titre de vidéo, souvent le nom de l’auteur du contenu dans le titre de celui-ci est à la fin de la phrase ou au début, ce sont donc pour moi deux features porteuses d’informations pour nos modèles.

Ensuite quant à la feature "is\_capitalized", celle-ci est aussi très importante, même si un peu « tricky » puisque les mots qui comment la phrases ont une majuscule. Donc elle peut provoquer un biais dans l’apprentissage de nos modèles.

Pour les balises, j’avoue n’avoir pas compris l’intérêts de celles-ci et comment les appliqués à notre pré-traitement.

Pour le traitement de la ponctuation, j’ai décider de l’enlever tout simplement.

Evaluate sur Logistic Regression :

* Sans ponctuation : Accuracy 96.29%
* Avec ponctuation : Accuracy 96.14%

Evaluate sur SVM :

* Sans ponctuation : Accuracy 96.25%
* Avec ponctuation : Accuracy 96.12%

J’aurai pu ajouter deux nouvelles features, **preceded\_by\_comma** et **followed\_by\_comma** qui auraient pu avoir du sens.

Passons maintenant aux tests des features "is\_final\_word","is\_starting\_word" et "is\_capitalized". Sur la Logistic Regression en guise de baseline:

Evaluate sur Logistic Régression :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Is\_final\_word Active | Is\_starting\_word Active | Is\_capitalized Active | Is\_final\_word Not Active | Is\_starting\_word Not Active | Is\_capitalized Not Active |
| Is\_final\_word Not Active |  | 96.2849% | 96.2849% | 96.2849% | 96.2849% | 96.2849% |
| Is\_starting\_word Not Active | 96.2849% |  | 96.2849% | 96.2849% | 96.2849% | 96.2849% |
| Is\_capitalized Not Active | 96.2849% | 96.2849% |  | 96.2849% | 96.2849% | 96.2849% |
| Is\_final\_word Active | 96.2942% | 96.2942% | 96.2942% |  | 96.2849% | 96.2849% |
| Is\_starting\_word Active | 96.2942% | 96.2942% | 96.2942% | 96.2849% |  | 96.2849% |
| Is\_capitalized Active | 96.2942% | 96.2942% | 96.2942% | 96.2849% | 96.2849% |  |

Nous pouvons voir qu’il n’y a pas une grande différence entre chacun de nos tests.

Mais quand toutes les features sont activées nous avons les meilleures performances de peu.