Rapport de projet
Extraction de connaissance dans les données

Mehdi BENNIS – Thibaut ETIENNE – Fabien FERAUD
Florian NOVELLON – Arnaud SOULIER

26 avril 2018

Table des matières

Chapitre 1

Pré-traitements

1.1 Tokenisation

La première étape que nous avons choisie pour pré-traiter les textes est d'en séparer chacun des mots. Pour cela, nous avons décidé d'utiliser la fonction tokenize de la librairie ntlk. Elle retourne une liste de mots à partir d'une chaine de caractère.

Avec cette liste de mots nous pourrons facilement appliquer chaque traitements sur les mots indépendament les uns des autres.

Durant cette phase, nous corrigeons également des erreurs de syntaxe telles que les points de fin de phrase qui restent collé aux mots. Nous les recherchons et les séparons pour éviter d'avoir deux mots dans un token.

1.2 Conversion minuscules

Certains de nos traitements ont besoin de comparer les mots pour les identifier. Pour simplifier ces comparaisons, nous avons décidé de convertir tout notre texte en minuscules. Ainsi, les mots de début de phrase par exemple ne seront pas oubliés dans les comparaisons.

1.3 TreeTagger

Lorem ipsum.

1.4 Suppression des stop-words

Certains mots ne seront pas intéressant à garder pour la classification. Afin d'éviter de surcharger notre texte de ces mots unitils, nous avons construit une liste de stop-words dont les occurences dans le texte seront supprimer. Grace à la conversion en minuscules et les corrections de syntaxe décrits précédement, aucun mot ne sera oublié.

1.5 Lemmatisation

La lemmatisation est une opération permettant de simplifier la syntaxe des mots en échangeant, par exemple, les verbes conjugués par leur infinitif ou les noms pluriels par leur singulier.

Cette opération permet de limiter la diversité des mots et garder un mot précis pour exprimer un avis précis.

Chapitre 2

Apprentissage

2.1 Naive Bayes

Dans un premier temps nous avons décider d'utiliser l'algorithme de Naïve Bayes pour avoir un premier résultat rapidement. Nous savons que cet algorithme n'est pas très optimisé mais il sera une base de travail.

2.1.1 Fonctionnement

L'algorithme Naïve Bayes effectue une apprentissage par probabilité sur les données qu'on lui fourni. Il calcule, selon les classes possible, les moyennes et les variances de chaque type de données et cherche ensuite quelle classe est la plus probable compte tenu des donnée qu'on lui donne en prédiction.

2.1.2 Résultats

Les résultats du Naïve Bayes sur le jeux de données test sont présenté dans le tableau suivant :

	Données de test	Données de challenge	Données challenge modifiées	Observation
Précision	0.89	? ? ?	? ? ?	c bo put1
Rappel	? ? ?	? ? ?	? ? ?	
F-mesure	? ? ?	0.49	? ? ?	c moch put1

Les résultats avec un apprentissage et une prédiction sur une même jeu de données sont plutôt bons mais lorsque l'on passe sur un apprentissage et une prédiction sur deux jeux de données différents, l'algorithme n'est plus capable de correctement prédire les classes. Il apprend correctement sur un type de texte mais lorsqu'il passe à un second type de texte il ne reconnait plus correctement les données. Il apprend trop sur le premier type pour être efficace sur le deuxième. C'est le surapprentissage.

2.2 K-Nearest Neighbors

2.2.1 Fonctionnement

2.2.2 Résultats

Chapitre 3

Conclusion

3.1 Comparaison

3.2 Résultats