Classification de documents d'opinions

Introduction

On retrouve sur internet un grand nombre d'informations qui peuvent être de type numérique mais aussi, et surtout, textuelle qui se divise en deux principaux domaines : faits et opinions. D'un côté nous avons les faits qui reposent sur l'objectivité des données et de l'autre, les opinions qui représentent le sentiment de son auteur.

De plus en plus de sites web offrent la possibilité aux clients de laisser un avis sur un produit par exemple. Dans ces avis, nous avons aussi une note qui représente la satisfaction du client qui est une donnée facilement représentable et interprétable. Cependant, une fouille des avis subjectifs nous apportent des informations supplmentaires sur le sentiment des clients où l'on peut en tirer les défauts et les avantages du produit. Ceci permet aux vendeurs d'avoir une idée beaucoup plus précise des attentes de leur clientèle et de mieux y répondrent.

L'analyse du sentiment de l'avis d'un internaute se fait de manière naturelle pour un être humain. Cependant, le nombre abondant de données qui doit être analyser ne peut être fait par ce dernier. Pour résoudre ce problème, un apprentissage automatique du langage naturel à une machine est inévitable car elle peut traiter une très grande quantitée de données par seconde. Ces techniques permettent de récolter des statistiques ou bien de trouver des schémas dans le texte qui révèlent son orientation.

L'objectif de ce projet est de créer une intelligence artificielle qui permettrait, à partir d'une phrase, de reconnaître la polarité de celle-ci : est-ce un avis positif ou bien négatif ? Comment lui apprendre à reconnaître la négation ou bien encore plus complexe, le sarcasme ? Nous essaierons d'y répondre par la suite.

1. Pré-traitements des données

1.1 Préparation à la tokenisation

- Contractions
- HTML (balise, url...)

`screenshot des fonctions`

1.2 Normalisation

- Tokenisation
- Transformation des nombres en lettres
- Ponctuation
- Stopwords

`screenshot avant/après tokenisation`

1.3 Lemmatisation

- BOW
- Lemmatisation
- Wordnet

`screenshot avant/après lemmatisation`

---

2. Apprentissage du modèle

- https://www.scss.tcd.ie/Khurshid.Ahmad/Research/Sentiments/K_Teams_Buchraest/mvie review review.pdf
- https://nlpforhackers.io/sentiment-analysis-intro/
- https://www.learndatasci.com/tutorials/predicting-reddit-news-sentiment-naive-bayes-text-classifiers/
-

2.1 Vectorisation

- TFIDF
- Paramètre (min_df, ngram pour le traitement des négations)

2.1-BIS Feature engineering

- https://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/

2.2 Cross Validation

- Evaluation des résultats

`screenshot du résultat de la Cross Validation`

- Optimisation des paramètres **(GridSearch)**

`screenshot du résultat du calibrage`

2.3 Sauveguarde du modèle

- Pipeline

---

3. Optimisation

3.1 WordCloud

2 interprétations possibles à cette visualisation :

- ironie
- avis à plusieurs phrases dont la polarité est différente

=> A prendre en compte sinon faussage des résultats (perte de précision, fiabilité?)

`screenshot des wordcloud` + **commentaires**

3.2 Traitement de l'ironie

- https://github.com/MirunaPislar/Sarcasm-Detection

Conclusion
