Compte-rendu TP5 : Analyse des sentiments

1. Prétraitement

Dans un premier temps je me suis occupé de nettoyer au maximum les données. Les tweets à analyser sont extrêmement parasités par des caractères inutiles, des fautes d'orthographes et des abréviations. Il s'agit donc de rendre ces mots exploitables pour les analyser par la suite.

J'ai décidé pour cela de créer plusieurs fonctions avec des expressions régulières permettant de matcher avec des patterns contenus dans les tweets. Par exemple la suppression des mots qui correspondent à des "@", supprimer les mots dièses en gardant le mot qui donne du sens la plupart du temps. J'ai également créé des fonctions permettant de supprimer les caractères spéciaux qui semblent ne pas être des émoticônes.

Enfin, pour la suite de l'analyse j'ai fait un split sur espace entre les mots permettant de créer une liste de liste. La liste contient les tweets et une liste à l'intérieur contient les mots du tweet.

2. Etiquetage grammatical

La création de la fonction "Part of Speech" permet d'identifier la catégorie grammaticale de chaque mot des tweets. Ainsi, nous pourrons par la suite faire un filtrage sur les mots qui ne nous intéressent pas.

Cette fonction permet d'identifier 1027 verbes au sein du corpus. Ce nombre n'est sans doute pas exact car cela dépend la manière dont le filtrage est fait en amont.

3. Algorithme V1

NB:

Précision(i) = (nombre de documents correctement attribués à la classe i) / (nombre de documents attribués à la classe i)

Rappel(i) = (nombre de documents correctement attribués à la classe i) / (nombre de documents appartenant à la classe i)

L'algorithme V1 permet de donner une note positive et négative sur chaque mot du corpus. On regarde en suite l'ensemble des notes des mots d'un tweet et on donne une note globale sur les mots de ce dernier.

Il y a 118 tweets qui ont été prédit positifs et qui sont correctement prédits.

<u>9ème tweet :</u>

[('not', 'RB'), ('love', 'VB'), ('Obama', 'NNP'), ('makes', 'VBZ'), ('jokes', 'NNS'), **1.5, 0.625, 4**]

4. Algorithme V2

Dans cette seconde version de l'algorithme nous allons pondérer les mots importants. Nous importons une liste de mots négatifs qui permettent de déterminer avec une forte probabilité que le mot qui le suit sera fortement négatif.

On obtient une précision de 53% et un rappel de 50%. Ces chiffres ne sont pas satisfaisants de toute évidence. C'est pourquoi nous allons améliorer l'algorithme afin d'augmenter la précision.

9ème tweet :

[('not', 'RB'), ('love', 'VB'), ('Obama', 'NNP'), ('makes', 'VBZ'), ('jokes', 'NNS'), **0.875, 1.25, 0**]

On voit que le sentiment a changé à juste titre sur le tweet 9. Cela prouve que le sentiment du tweet est mieux identifié.

3ème tweet :

[("You'll", 'NNP'), ('love', 'VB'), ('Kindle2', 'NNP'), ("I've", 'NNP'), ('had', 'VBD'), ('mine', 'NN'), ('few', 'JJ'), ('months', 'NNS'), ('never', 'RB'), ('looked', 'VBD'), ('new', 'JJ'), ('big', 'JJ'), ('one', 'NN'), ('is', 'VBZ'), ('huge', 'JJ'), ('No', 'NNP'), ('need', 'NN'), ('remorse', 'NN'), (':)', 'NN'), **2.0, 1.875, 4**]

5. Algorithme V3

Pour ce troisième algorithme nous prenons en compte les smileys contenus dans les tweets qui influencent également le sentiment général du message. 269 tweets ont été correctement prédits.

On obtient ainsi une précision de 56% et un rappel de 52%. Nous avons légèrement amélioré l'algorithme.

3ème tweet :

[("You'll", 'NNP'), ('love', 'VB'), ('Kindle2', 'NNP'), ("I've", 'NNP'), ('had', 'VBD'), ('mine', 'NN'), ('few', 'JJ'), ('months', 'NNS'), ('never', 'RB'), ('looked', 'VBD'), ('new', 'JJ'), ('big', 'JJ'), ('one', 'NN'), ('is', 'VBZ'), ('huge', 'JJ'), ('No', 'NNP'), ('need', 'NN'), ('remorse', 'NN'), (':)', 'NN'), **3.0, 1.875, 4**]