CR_tweet_AQV

June 6, 2017

1 ANALYSE DES OPINIONS SOUS TWITTER - Compte Rendu

Audrey Quessada INF344

Dans un premier temps on effectue un pre-processing simple des tweets.

On extrait les destinataires, les urls, les hashtags et on nettoie les tweets:
Pour ce faire, on crée un dictionnaire qui va contenir les adresses url, les destinataires des tweets et les hashtags répertoriés.
On va donc utiliser plusieurs regex nous permettant de récupérer ces informations. On va aussi "nettoyer" les tweets, à savoir, une fois qu'on a enlever les champs sus-mentionnés, on met tous les tweets en minuscule ou non (cela dépend du choix fait par l'utilisateur) et on enlève la ponctuation ou certains caractères spéciaux.
Il faut également enlever RT qui veut dire retweet (ou on peut compter le nombre de retweet).

La démarche a été de tester les différentes regex sur des échantillons de tweets.

Une fois ce premier processing effectué, on tokenize les tweets (`get_token_from_tweet`) et on enlève les tokens qui font partie du dictionnaire de slang qui a été complété à la main (`clean_from_dico`). Ces tokens sont remplacés par leurs équivalents qui vont permettre de récupérer le POS-tag du token (`getPOS`).

On crée également une fonction qui permet de récupérer le nombre de verbes (ayant les tags 'VB', 'VBZ' , 'VBP', 'VBG', 'VBN', 'VBD','MD'). On en compte **1170**. Ce nombre peut varier selon qu'on a mis tous les tweets en minuscules ou non.

1.1 Premier alogrithme de détection avec Sentiword

Une première fonction récupérée sur le site permet d'associer le tag utilisé par SentiWordnet au POS-tag. Ce qui nous permettra de récupérer les scores positifs et négatifs d'un mot.

Pour chaque tweet, on calcule un score global positif et un score global négatif, si le score positif est plus grand que le score négéatif alors la note du tweet est 1, 0 si les 2 scores sont égaux et -1 dans le cas ou le score négatif est plus grand.

On compare cette note globale à celle attribuée à la main par les utilisateurs.
Sur 498 tweets, on a :
* le nombre de tweets positifs vrais est: **182**

- le nombre de tweets neutres vrais est: **139**
- le nombre de tweets négatifs vrais est: **177**

Le rapport de classification donne:
precision recall f1-score support

-1	0.67	0.44	0.53	177
0	0.49	0.36	0.41	139
1	0.47	0.73	0.57	182

avg / total 0.55 0.52 0.51 498

- le nombre de tweets positifs prédits correctement: **132**
- le nombre de tweets negatifs prédits correctement: **78**
- le nombre de tweets mal prédits est: **238**

On voit que si les tweets positifs sont relativement bien prédits, ce n'est pas le cas pour les tweets négatifs ou neutres.

1.2 Algorithme de détection v2 : gestion de la négation et des modifieurs

On utilise pour cela également des fichiers qui ont été enrichis à la main.
Le score est calculé en fonction de l'algorithme précisé dans l'énoncé.
Le rapport de classification est le suivant:
precision recall f1-score support

-1	0.62	0.40	0.49	177
0	0.51	0.45	0.48	139
1	0.48	0.68	0.56	182

avg / total 0.54 0.52 0.51 498

- le nombre de tweets positifs prédits correctement est de: **124**

- le nombre de tweets negatifs prédits correctement est de: **71**
- le nombre de tweets mal prédits est: **240**

Cette méthode semble moins efficace que la méthode précédente, bien que la gestion des tweets neutres semble meilleure.

1.3 Algorithme de détection v3 : gestion des emoticons

On utilise le dictionnaire fourni et on calcule un score préalable si le tweet contient un émoticon ou non.
Une fois ce score initial calculé, on remplace l'émoticon par un espace et on recommence la chaine de traitement utilisée précédemment.
Ce qui est rassurant c'est qu'on retrouve le même nombre de verbes que précédemment.

Le rapport de classification obtenu est le suivant:
precision recall f1-score support

-1	0.61	0.42	0.50	177
0	0.54	0.45	0.49	139
1	0.48	0.69	0.57	182

avg / total 0.55 0.53 0.52 498

- le nombre de tweets positifs prédits correctement est de : **126**
- le nombre de tweets négatifs prédits correctement est de : **74**
- le nombre de tweets mal prédits est: **235**

On retrouve une légère amélioration avec la gestion des émoticons.

1.4 Test d'un autre algorithme: le Naïve Bayes vu en MDI343

Pour cette partie, j'ai repris le travail effectué en MDI 343 avec l'utilisation d'un CountVectorizer et d'un Naïve Bayes.

J'ai divisé les tweets en train et test et j'ai effectué une validation croisée.
Le résultat est meilleure qu'avec les méthodes précédentes:

precision recall f1-score support

-1	0.47	0.65	0.55	26
0	0.73	0.32	0.45	34
1	0.63	0.78	0.70	40

avg / total 0.63 0.59 0.57 100

1.5 Conclusion:

Ici, un simple Naïve Bayes est bien meilleur pour classer les tweets par rapport aux méthodes vues précédemment.
On aurait pu également utiliser TFIDF dans ce dernier cas.
Pour les premières methodes avec Sentiword, il serait possible d'utiliser la distance de Levenshtein pour choisir correctement le bon score d'un mot (par exemple pour I, il est proposé en premier choix Iodine et non le pronom).