**Rapport TAL : Prédictions des sentiments pour des revues de film**

**(opinions positives vs négatives)**

***Auteurs****: Anne-Cécile Toulemonde, Noémie Jacquet*

***Date****: décembre 2022*

**1. Données :**

On dispose:

* d’un set d’entrainement de 2000 revues positives et 2000 revues négatives
* d’un fichier Testsentiment contenant 25000 revues dont il faut prédire le sentiment positif ou négatif.

On choisit les labels : « 1.0 » pour la classe positive et « 0 » pour la classe négative. Avant prédiction on étiquette « -1 » pour les 25000 revues du fichier Testsentiment.

**2. prétraitements, split des données**

On effectue un prétraitement sur le texte (le texte d’entrainement et le texte que l’on testera) :

enlever majuscules, ponctuation et chiffres, et gérer le type d’encodage.

On partage ensuite les données d’entrainement en 2 ensembles : un ensemble test et un ensemble test pour la validation et l’évaluation de la performance de notre modèle (training :70% des données, test 30% des données):

Une image contenant texte

Description générée automatiquement

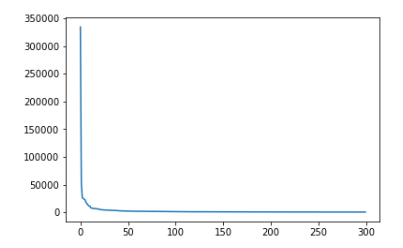
La taille du dictionnaire initial est de 33793 mots.

Les 100 mots les plus fréquents (identifiés avec « Counter ») et leur fréquence sont les suivants :

Une image contenant texte

Description générée automatiquement

On observe que « the » mot le plus fréquent apparaît 53422 fois, alors que le 100e mot le plus fréquent (plot) apparait « seulement » 1035 fois.

****

Cette liste de 100 mots les plus fréquents sera utilisée pour constituer la liste des stop words dont on va tester le retrait et son impact sur la performance.

**3. Vectorisation par Countvectorizer (Bag of words), entrainement de 3 classifieurs (NB, LR, SVM) et évaluation de l’impact de la taille du dictionnaire sur les performances des 3 classifieurs**

On va tester initialement 3 classifieurs : Naive Bayes Multinomial (NB), Logistic Regression (LR), SVM linéaire.

On teste :

* le fait de prétraiter les données comme décrit ci-dessus (vs ne pas le faire)
* le fait d’appliquer « countvectorizer » avec les options par défaut
* ou le fait d’appliquer « countvectorizer » en enlevant comme stopwords les 100 premiers mots les plus fréquents

Une image contenant texte

Description générée automatiquement

* ou le fait d’appliquer « countvectorizer » en faisant varier la taille du dictionnaire en testant plusieurs valeurs de max\_df et min\_df Une image contenant texte

  Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

* **Impact des stopwords et de la taille du dictionnaire définis dans Countvectorizer sur la performance (accuracy, précision recall, f1 score) des 3 classifieurs :**

Le fait de prétraiter les données (enlever majuscules, punctuation, chiffres…) n’a pas d’impact sur les performances des classifieurs en termes d’accuracy, précision, recall, f1 (regardé sur les 2 classes).

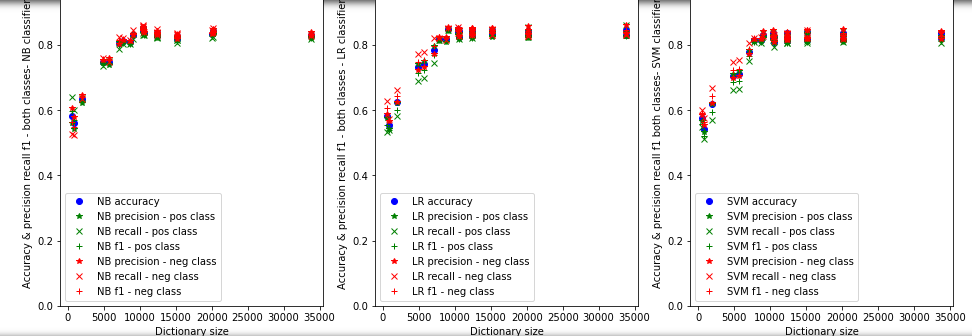
Sur les données prétraitées, le fait d’enlever les stopwords n’a pas d’impact n’a pas d’impact sur les performances des classifieurs (regardé sur les 2 classes).

C’est surtout la taille du dictionnaire qui a un impact (en jouant sur df\_max, df\_min). On observe surtout un décrochage de la performance (accuracy, et recall, précision f1 scores pour les 2 classes, cf. figure ci-dessous), pour une taille de dictionnaire < 9000 mots, quel que soit le classifieur.

Au-delà de 10000 mots, la performance est globalement stable quel que soit le classifieur.

*Figure : Accuracy (en bleu), précision, recall et f1 score pour les 2 classes (vert et rouge)*

*pour chacun des 3 classifieurs NB, LR et SVM, en fonction de la taille du dictionnaire modifiée grâce aux options de Countvectorizer max df et min\_df)\**

****

***Légende :*** *à gauche : classifieur NB, au milieu : classifieur LR, à droite : classifieur SVM*

*\*après prétraitement qui consistait à enlever majuscules, ponctuation et chiffres, et gérer le type d’encodage*

Par conséquent on décide de sélectionner les paramètres max-df=0.3 et min\_df=5 pour le traitement par Countvectorizer, soit un vocabulaire de 10379 mots, puisque accuracy précision, recall, f1 n’augmentent plus significativement lorsque la taille du vocabulaire augmente au-delà jusqu’ aux 33973 mots (i.e. le vocabulaire initial du set de training).

**4.Test des hyperparamètres des 3 classifieurs, une fois la taille du dictionnaire fixée dans Countvectorizer:**

La taille du vocabulaire et le type de prétraitement étant fixés (Countvectorizer max\_def=0.3, min\_def=5, vocabulaire 10379 mots), on teste désormais différentes valeurs d’hyperparamètres à la main, en vérifiant les scores d’accuracy, précision, recall et f1:

On teste 3 classifieurs :

* NB Multinomial
* Logistic regression : différentes valeurs de C (0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000) et différents solvers et penalty (liblinear + penalty l2, liblinear + penalty l1, lbfgs + penalty l2)
* SVM linéaire : différentes valeurs de C (0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000)

Une image contenant texte

Description générée automatiquementUne image contenant texte

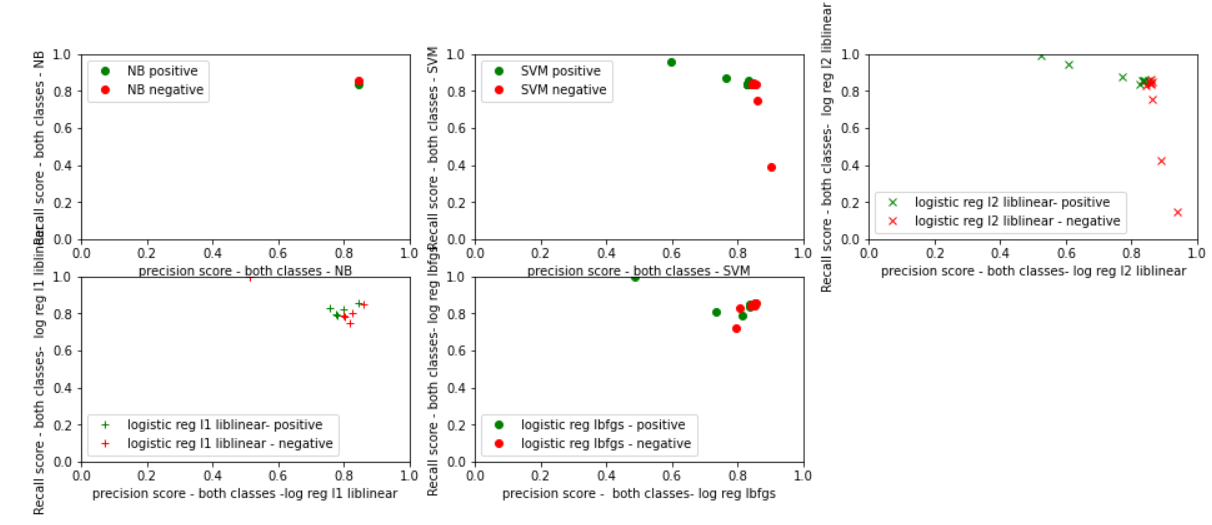
Description générée automatiquement

On fait varier la valeur de C et pour SVM et Logistic Regression et la penalty pour Logistic Regression.

Pour simplifier on a représenté seulement recall en fonction de précision pour différentes valeurs de C, pour chaque classifieur.

La valeur de C pour laquelle les performances sont optimales varie selon le classifieur (cf. notebook), et on observe de mauvaises performances en dessous de certaines valeurs testées de C. On a représenté sur la figure suivante recall en fonction de précision pour les 2 classes (positive vert, négative rouge) pour les différents classifieurs testés, et pour les 9 valeurs de C testées.

*Figure : Recall score (%) en fonction de precision score (%) pour 5 cas de figure :*

****

*Recall score en fonction de precision score:*

*(1) classifieur NB*

*(2) classifieur SVM\**

*(3) logistic regression (LR) penalty l1 liblinear\**

*(4) logistic regression (LR) penalty l2 liblinear\*,*

*(5) logistic regression (LR) penalty l2 lbfgs\**

*\*pour 9 valeurs de C testées pour LR et SVM (1.e-05, 1.e-04, 1.e-03, 1.e-02, 1.e-01, 1 , 1, 1.e+02, 1.e+03)*

* **Meilleur modèle :**

**Le meilleur modèle avec ses paramètres optimaux semble être logistic regression liblinear, penalty l2, C=0,1 où l’on obtient des valeurs de précision recall accuracy f1 au moins égales à 0,85 (cf. valeurs surlignées en Jaune pour C=0,1). La performance est bonne pour les 2 classes (précision et recall de chaque classe)**

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

**5. Test de TF-IDF et classifieur SVM, comme identifié dans la littérature**

La publication de Naeem et al. (2022)[[1]](#footnote-1) indique que les meilleures performances de prédictions d’opinion pour des revues de film avec sont obtenues avec l’emploi de TF-IDF et un classifieur SVM (par rapport à Countvectorizer i.e. l’approche « Bags of words »).

On teste donc une nouvelle approche TF-IDF et un classifier SVM linéaire C=1. On teste :

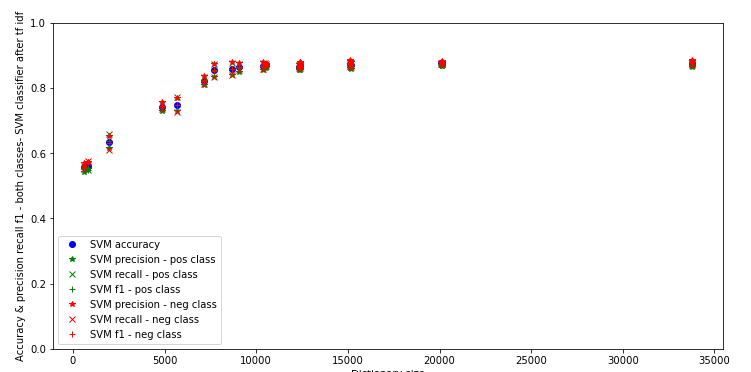
* l’impact des N-grams (1 à 3) : la meilleure performance est obtenue pour N-grams=1
* l’impact de faire varier la taille du dictionnaire dans TF-IDF, avec différentes combinaisons de max\_df et min\_df

On teste l’impact sur accuracy, et sur precision, recall f1 score pour les 2 classes.

**Une image contenant texte

Description générée automatiquement**

*Figure : Accuracy et précision, recall f1 score pour chaque classe (rouge :négative, vert : positive) du classifieur SVM testé en fonction de la taille du dictionnaire modifiée grâce aux options de TF-IDF (max df et min\_df) (N-gram=1)\**

****

*\*après prétraitement qui consistait à enlever majuscules, ponctuation et chiffres, et gérer le type d’encodage*

**De même qu’avec Countvectorizer, au-delà de 10000 mots, la performance est globalement stable.**

**On définit max\_df=à.8 et min\_df=2 (dictionnaire 20104 mots), comme valeurs pour TF-IDF, car associées à de bonnes performances de classification (cf. valeurs surlignées).**

*Metrics précision et recall sur les 2 classes en fonction de max\_df et min\_df :*

Une image contenant table

Description générée automatiquement

*Metrics accuracy et metrics precision recall f1 pour la classe positive en fonction de max\_df et min\_df*

Une image contenant table

Description générée automatiquement

**Avec TD-IDF (max\_df=0.8 et min\_df=2, N-grams=1) appliqué, on teste les hyperparamètres du classifieur SVM et la valeur optimale est C=1. C’est le classifieur que l’on sélectionne pour faire les prédictions finales demandées (cf. section suivante).**   
Les performances sont donc en effet un peu plus élevées avec TF-IDF que Counvectorizer (sur SVM).

*Metrics accurcay et metrics précision, recall f1 sur la classe positive:*

**Une image contenant texte

Description générée automatiquement**

*Metrics précision et recall sur les 2 classes:*

**Une image contenant texte

Description générée automatiquement**

* **Résumé des performances du classifieur séléctionné :**

**Une image contenant texte

Description générée automatiquement**

**6. Application du classifieur sélectionné au fichier « Testsentiment » pour prédiction du sentiment de l’opinion (positive / négative) pour chaque revue**

Avec le prétraitement (enlever majuscule, ponctuation et chiffres), l’application de TF-IDF (max\_df=0.8, min d\_f=2), l’entrainement du classifieur sélectionné (SVM linéraire C=1) sur le training set,

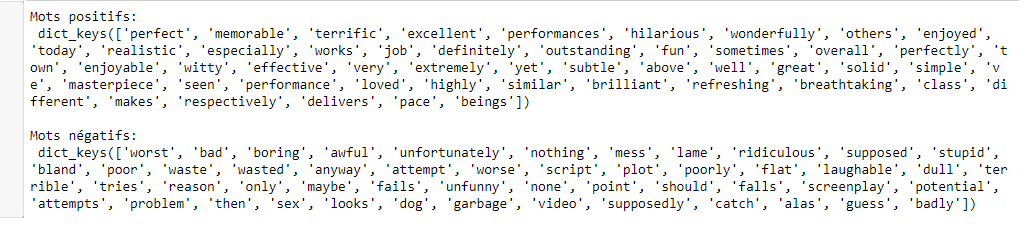
**on obtient sur le fichier Testsentiment  le fichier de prédictions suivant : test\_movies\_prediction.txt, avec 11157 revues positives et 13843 revues négatives.**

Une image contenant texte

Description générée automatiquement

**7. Regard sur les features associées aux coefficients les plus élevés dans SVM (classifieur sélectionné):**

On peut identifier les 50 mots du dictionnaire ayant les coefficients les plus élevés, donc les mots les plus « positifs » et les 50 ayant les coefficients les plus faibles dans la fonction de décision, donc les mots les plus négatifs :



1. Naeem MZ, Rustam F, Mehmood A, M-z-d, Ashraf I, Choi GS. 2022. Classification of movie reviews using term frequency-inverse document frequency and optimized machine learning algorithms. PeerJ Comput. Sci. 8:e914 DOI 10.7717/peerj-cs.914 [↑](#footnote-ref-1)