Faculté des sciences et Ingénierie

Sorbonne Université

Traitement Automatique de la Langue

Mouhammad Ikhlef

Wenshan WU

6 mai 2019

Table des matières

Chapitre 1

Introduction

L'objectif de nos travaux réside dans la classification de documents et se scinde en trois parties, la première se base sur une campagne d'expériences portée sur la détection d'auteur entre des discours de Chirac et Mitterrand puis ensuite, une analyse de sentiments sur des revues de films. Nous allons dans une première partie vous présenter les résultats des modèles étudiés concernant la détection d'auteur puis dans la deuxième, les résultats concernant l'analyse de sentiments. Et enfin dans la dernière, les résultats concernant la classification non supervisée sur divers sujets.

Chapitre 2

Classification de documents

1 Détection d'auteur

Nous avons à notre disposition deux fichiers : un fichier train qui contient les discours labelisés qui servira à entraîner notre modèle puis un fichier test contenant les discours sans labels et qui servira de base de tests.

Pour information le fichier train contient 49890 discours provenant de Chirac et 7523 pour Mitterrand.

1.1 Preprocessing

Nous allons nous focaliser sur les paramètres suivants dans le cadre du traitement de texte de notre corpus.

Codage

Nous nous intéressons à trois types de codage : le premier est celui avec les sacs de mots, il s'agit de définir une matrice (terme x document) et donc de regrouper les mots de chaque document avec leur fréquences d'apparition. Le deuxième se base sur un codage TF-IDF où les scores des mots fréquents seraient pénalisés et le troisième sur le codage présentiel où il s'agit d'indiquer si le mot est présent ou non dans le document. Comme nous sommes en présence de discours, il ne serait peut-être pas préférable de choisir le codage TF-IDF car certains mots peuvent être récurrents ce qui caractérise l'élocution d'une personne.

Stemming

La racinisation permet de réduire le nombre de mots présent dans notre corpus et donc de réduire la dimension pour un meilleur apprentissage. Il est difficile de savoir si cela aura un réel impact sur notre corpus sans avoir fait des tests au préalable.

Stopwords

Les stopwords permettent d'éliminer plusieurs termes communs et récurrents, de ce fait l'apprentissage devient plus rapide et la prédiction plus précise en supprimant le "bruit".

Lowercase

La fonctionnalité lowercase permet de convertir toutes les majuscules en minuscules.

Tokenization

La tokenization est un moyen de "casser" une séquence de mots en plusieurs mots. Pour cela, deux paramètres à définir s'offrent à nous, la première concerne le ou les délimiteurs utilisé(s) et donc on fait appel à une expression régulière ou encore regex, celle utilisée est la suivante : `r"\b[^\d\W]+\b"` qui nous permet de supprimer les chiffres ainsi que les caractères spéciaux. Et le deuxième choix réside dans les N-Gram qui consiste à prendre une suite de N mots dans notre apprentissage, ici nous allons nous limiter aux 1-Gram et 2-Gram pour ne pas trop augmenter la complexité.

1.2 Métrique d'évaluation

Concernant la métrique à utiliser, nous allons nous intéresser à celle du F1 score dont la formule est la suivante :

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Les raisons de ce choix portent sur le fait de pouvoir trouver un équilibre entre le rappel et la précision, mais surtout par la présence de classes déséquilibrées comme nous l'avons pu le constater, il y a beaucoup plus de discours prononcés par Chirac que Mitterrand.

1.3 Modèles d'apprentissage

Nous disposons des modèles suivant : SVM, Naive Bayes et régression logistique. Nous allons étudier ces trois modèles sur plusieurs expérimentations en faisant varier les pramètres issus du préprocessing afin de pouvoir évaluer leur impacts sur notre corpus.

1.4 Résultats

Meilleurs paramètres

Les résultats sont obtenus via la méthode *train_test_split* de la librairie **Sklearn**. Les graphes suivants représentent le score F1 obtenu par rapport au pourcentage de test, par exemple si le pourcentage de test vaut 30% alors cela signifie qu'on a séparé notre fichier train en 70% de train et 30% de test.

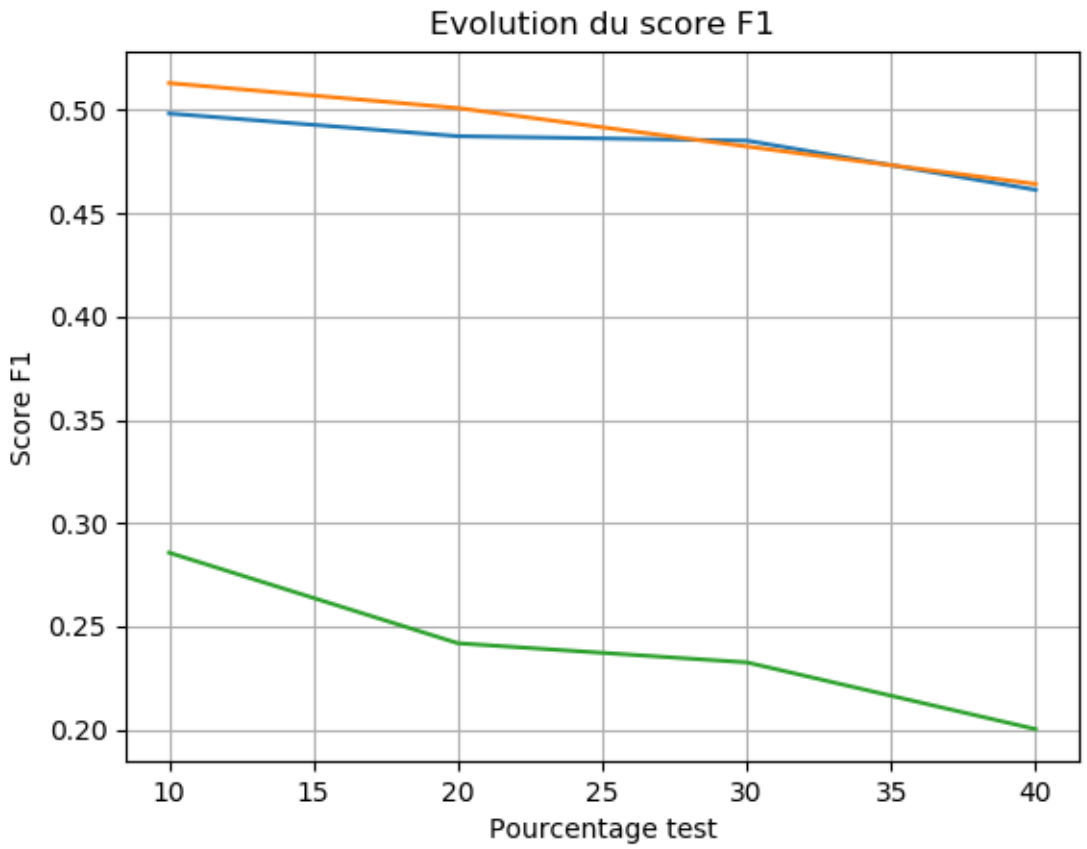


FIGURE 2.1 – Évolution par rapport aux stopwords, stemming, lowercase et N-grams

Ce graphe représente l'évolution du score par rapport aux stopwords, stemming, lowercase et N-grams, ces derniers sont mis à True donc appliqués. Les courbes bleu, verte et orange mettent en avant respectivement les unigrams, bigrams et à la fois les uni et bigrams. On observe que les scores présentés par la courbe verte sont bien plus inférieurs à ceux du reste donc on en conclut que les bigrams n'est pas un bon choix. On aimerait donc privilégier les 1-2 grams étant donné le score atteint lorsque le test est à 10% (environ 0.525).

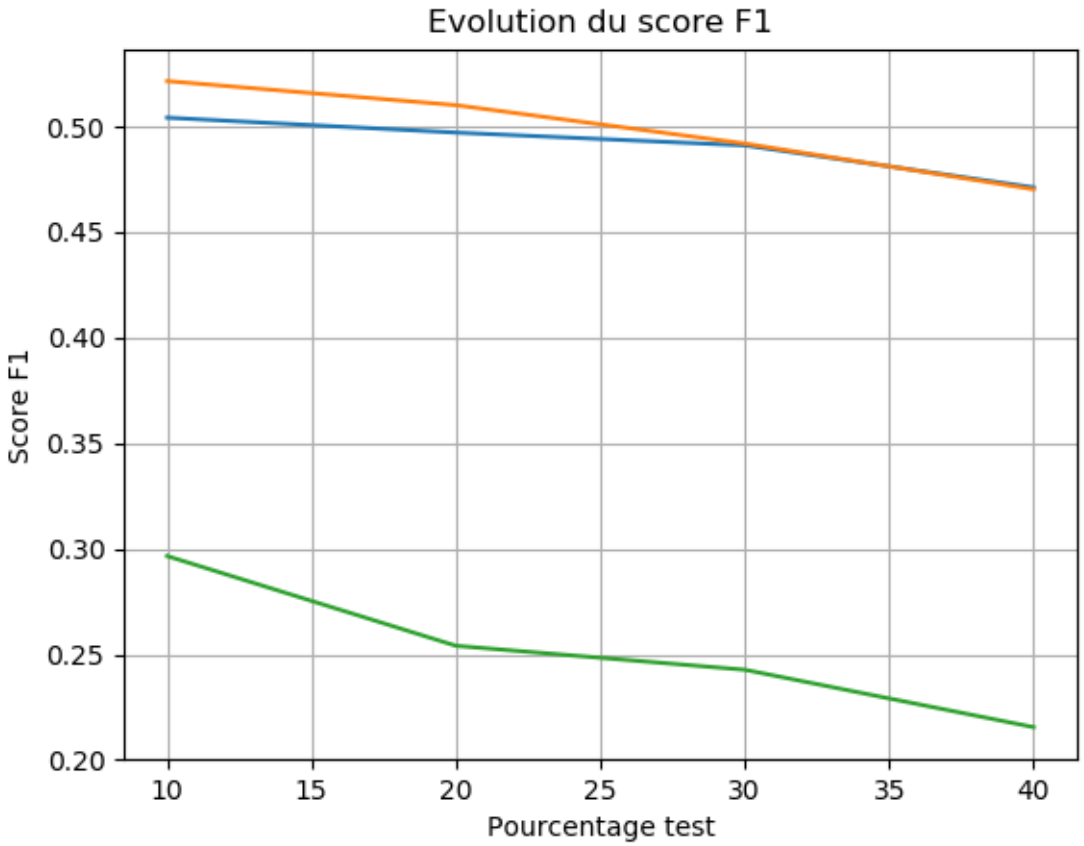


FIGURE 2.2 – Évolution par rapport aux stopwords, stemming et N-grams

Ce graphe représente l'évolution du score par rapport aux stopwords, stemming et N-grams, en comparaison avec le graphe précédent, nous avons ici éliminé le lowercase. On remarque que les scores des courbes bleu et orange sont plus élevés sur l'ensemble du test, ainsi il est préférable de ne pas prendre en compte le lowercase.

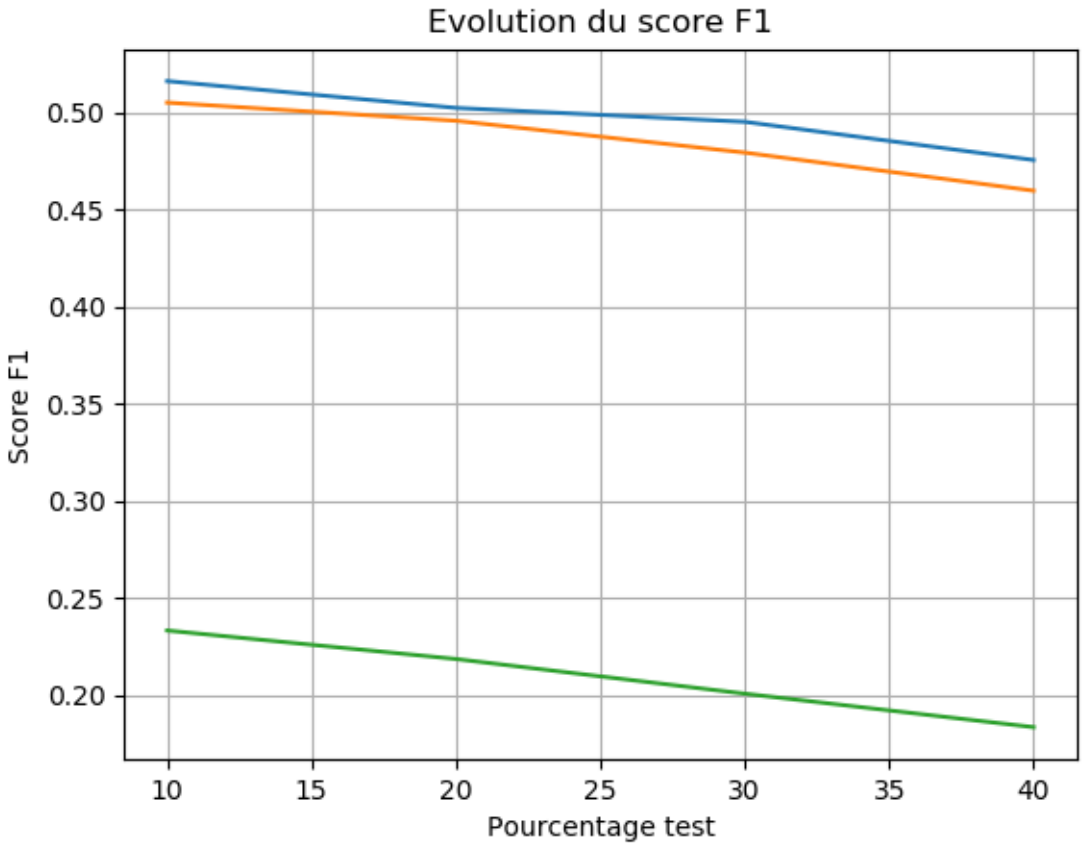


FIGURE 2.3 – Évolution par rapport aux stopwords, lowercase et N-grams

Ce graphe représente l'évolution du score par rapport aux stopwords, lowercase et N-grams, nous avons donc exclu la stemmatisation. On observe cette fois-ci une hausse du score des unigram (courbe bleu) vers 0.525 lorsque le test est à 10%. L'inversement des courbes bleu et orange semble conforter un impact positif sur les unigram mais défavorise les 1-2 grams cependant le score de ce dernier était supérieur à la valeur max obtenue sur ce graphe, donc cela laisse à penser que la stemmatisation n'est pas une bonne idée.

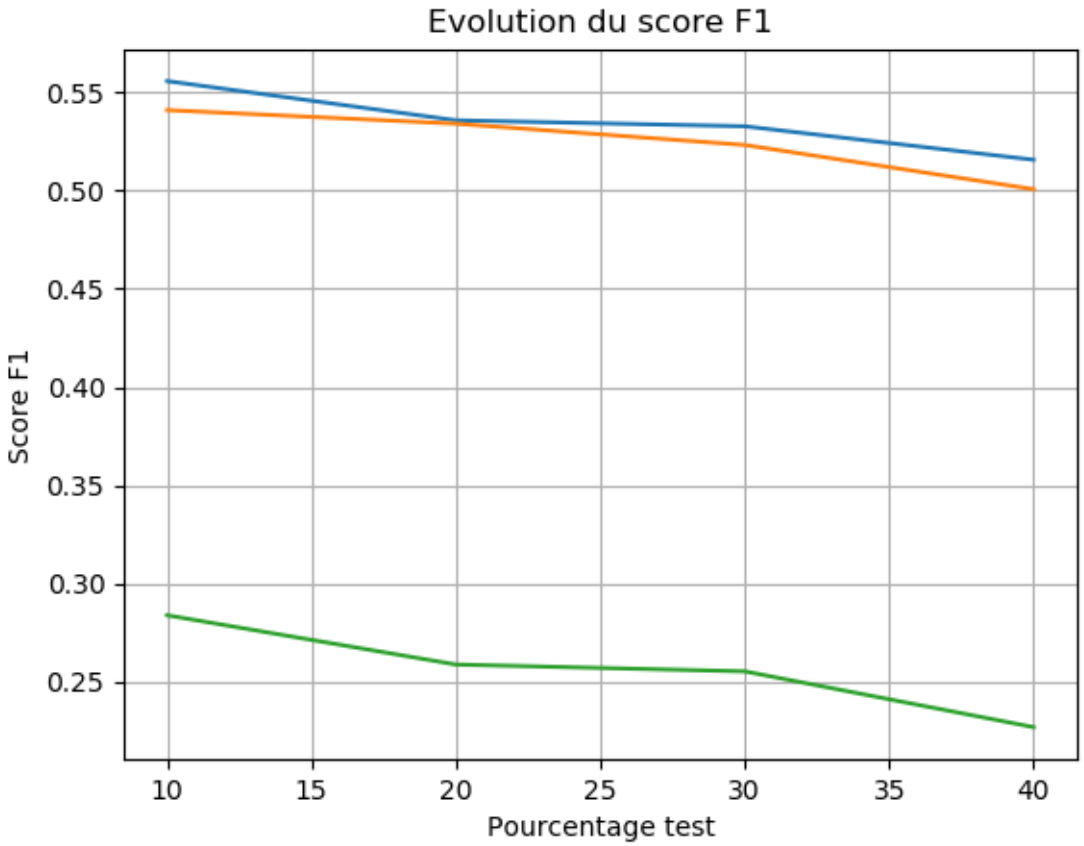


FIGURE 2.4 – Évolution par rapport au stopwords et N-grams

Ce représente l'évolution du score par rapport aux stopwords et N-gram, ici il n'y a pas de stemming et lowercase. On observe un score d'environ 0.56 pour les unigram (courbe bleu) et 0.54 pour les 1-2 grams (courbe orange), cela nous conforte dans les observations précédentes, ces deux paramètres ne seront pas à prendre en compte.

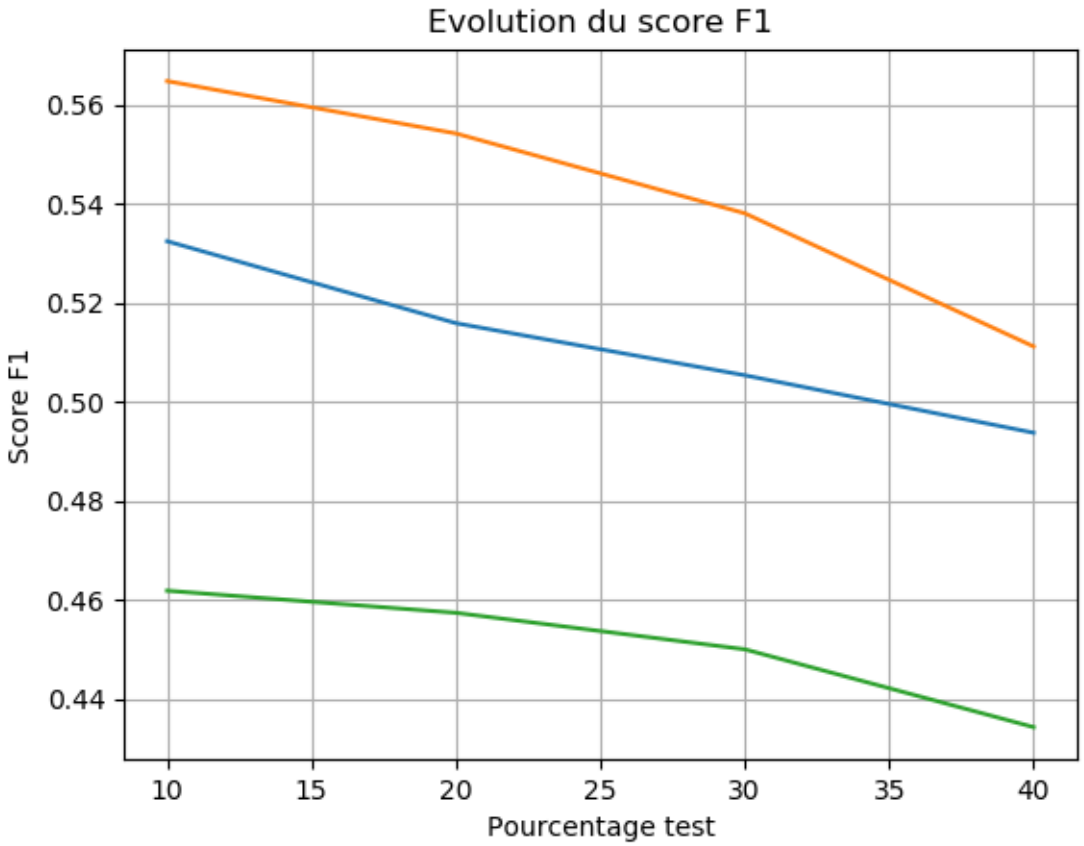


FIGURE 2.5 – Évolution par rapport aux stemming, lowercase et N-grams

Ce graphe représente l'évolution du score par rapport aux stemming, lowercase et N-gram. Nous avons retiré les stopwords et remarquons avec nos résultats précédents que le score des 1-2 grams (courbe orange) a augmenté avec 10% de test (0.54 à 0.58) et celui des unigram (courbe bleu) a diminué (0.56 à 0.53). Les stopwords non plus ne donnent pas l'impression de contribuer à un bon apprentissage, donc ce paramètre serait à écarter.

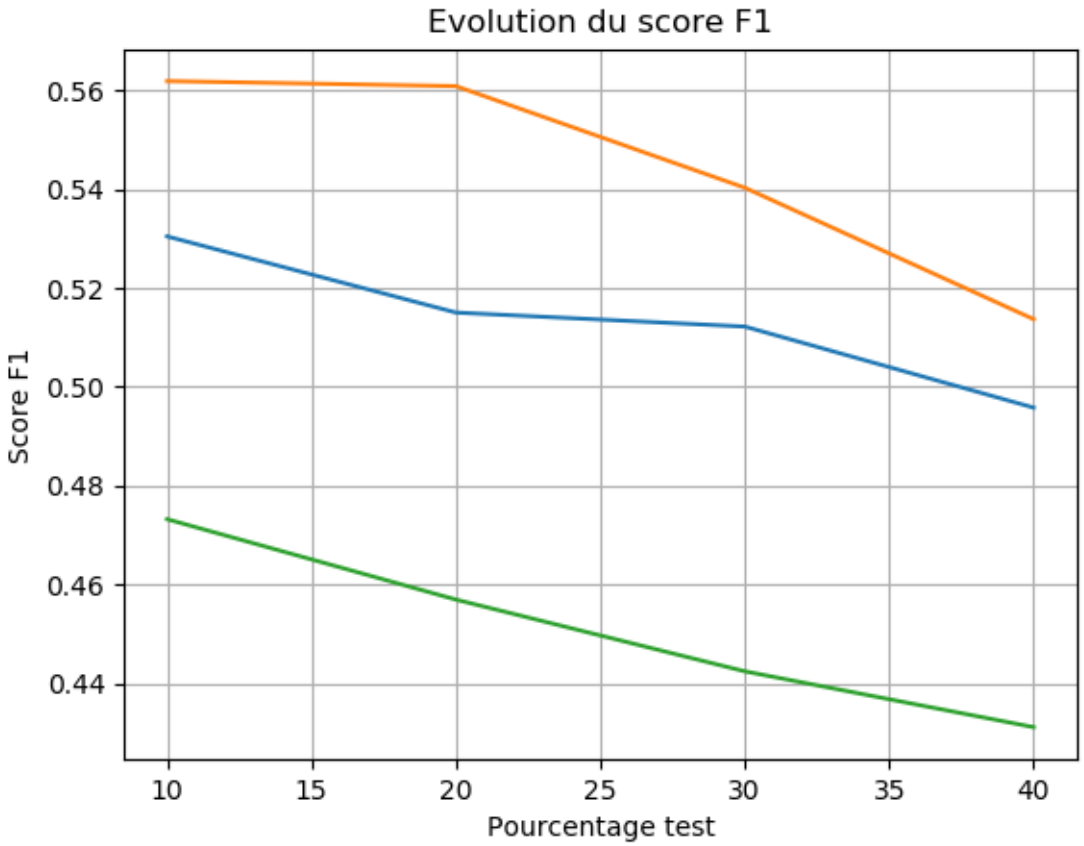


FIGURE 2.6 – Évolution par rapport aux stemming et N-grams

Ce graphe représente l'évolution du score par rapport aux stemming et N-grams. Nous avons retiré les stopwords et lowercase et observons une légère baisse du score concernant les 1-2 grams (courbe orange) sur l'ensemble du test, cela confirme une nouvelle fois que la stemmatisation n'est pas une bonne chose.

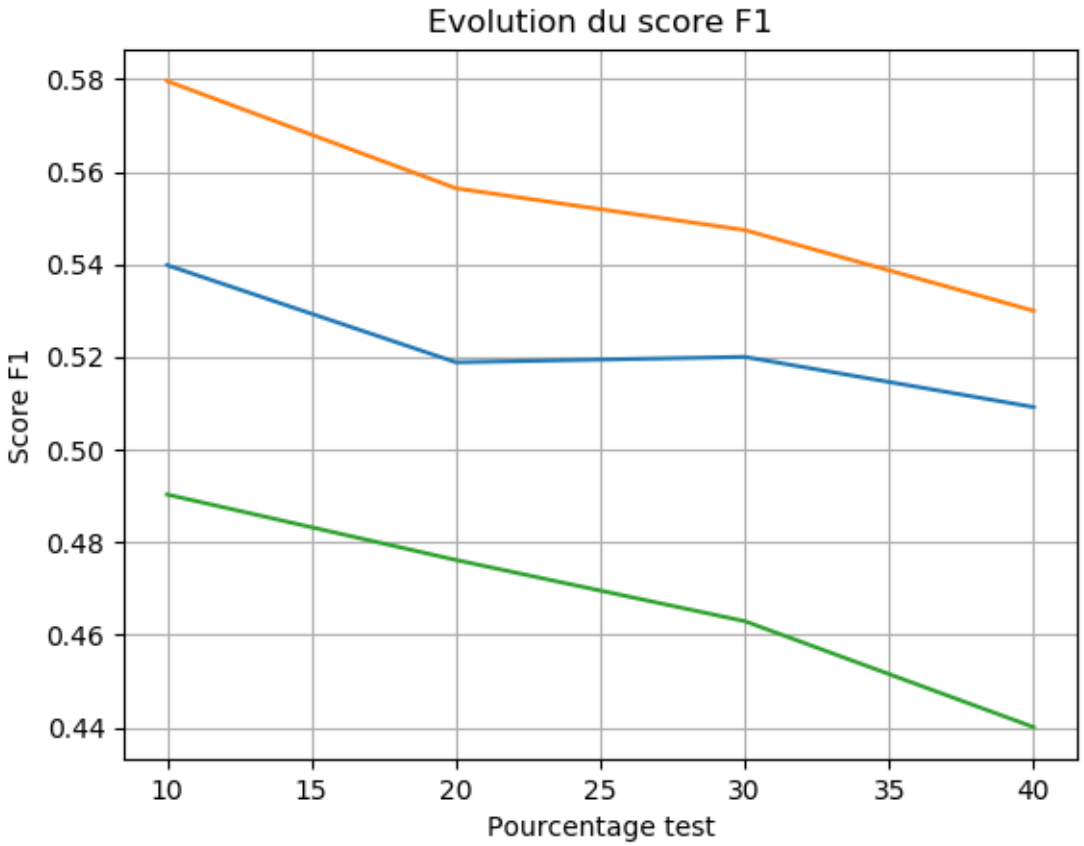


FIGURE 2.7 – Évolution par rapport aux lowercase et N-grams

Ce graphe représente l'évolution du score par rapport aux lowercase et N-grams. Ici, il n'y a pas de stopwords et lowercase et nous observons une augmentation du score pour les 1-2 grams (courbe orange) qui atteint cette fois un score de 0.58 avec 10% de test.

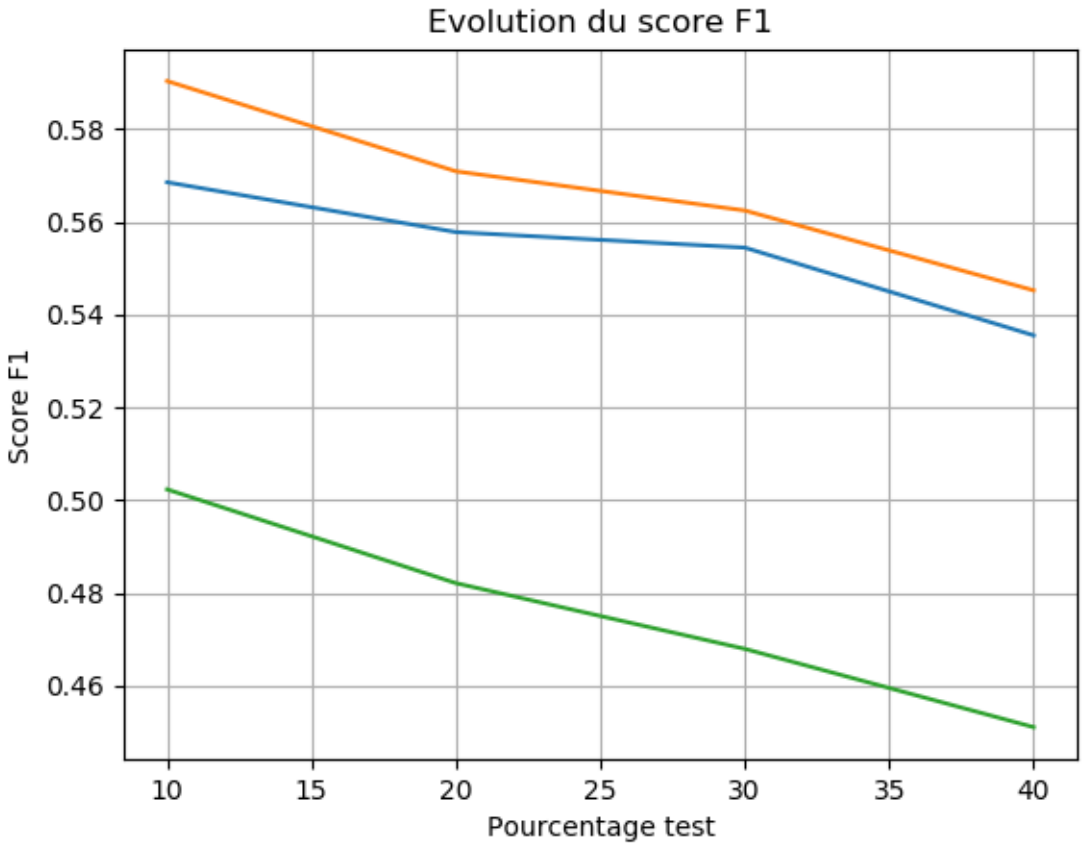


FIGURE 2.8 – Évolution par rapport aux N-grams

Ce dernier graphe représente l'évolution uniquement par rapport aux N-grams, tous les autres paramètres ont été omis et on remarque qu'on a de nouveau une augmentation du score pour les unigram (courbe orange) et 1-2 grams (courbe bleu) avec notamment un score très proche de 0.6 pour les unigram.

Suite à nos résultats obtenus, nous pouvons en déduire que les meilleurs paramètres à prendre en compte sont les 1-2 grams et qu'il n'est pas préférable d'appliquer les stopwords, stemming et lowercase.

Cross validation

Nous allons procéder à une cross validation afin de déterminer quel modèle est le plus adapté pour notre apprentissage

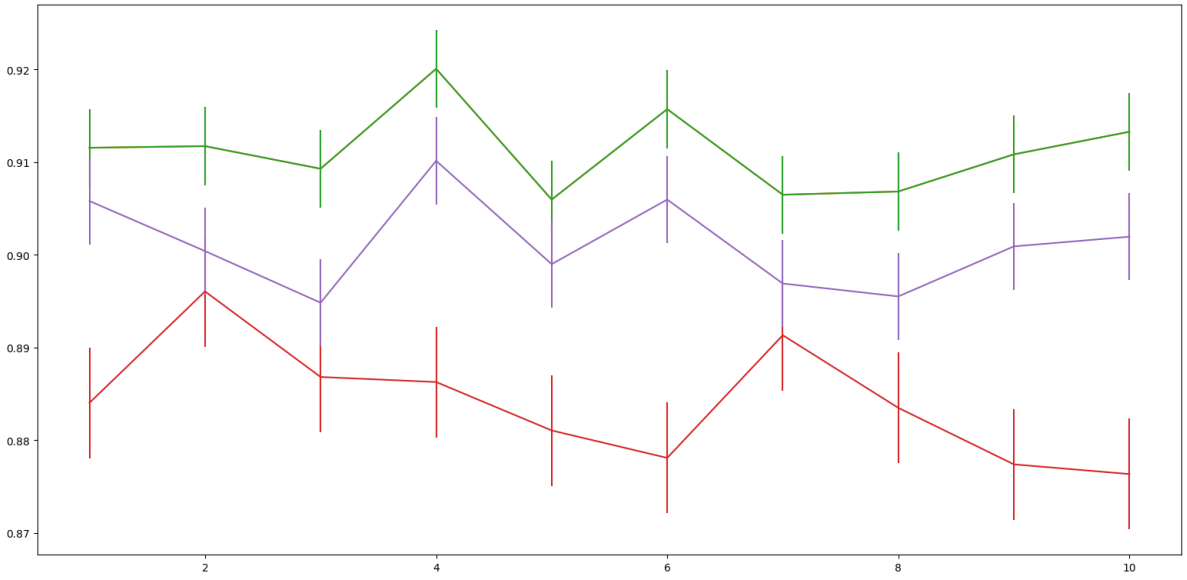


FIGURE 2.9 – Évolution de la cross validation sur différents modèles

Ce graphe illustre l'évolution de la cross validation sur une fenêtre de 10-fold, c'est à dire qu'on a partitionné notre fichier train en 10 parties et nous avons pris un dixième comme étant du test. Nous avons également représenté l'écart-type sur chaque valeur pour les comparer entre elles. L'axe des abscisses représente la i-ième partie prise en tant que test et l'axe des ordonnées représente le score obtenu par cross validation. La courbe rouge représente le modèle Naive Bayes, la courbe violette représente le modèle SVM et la courbe verte celle de la régression logistique. Par l'allure des courbes, on remarque que le meilleur modèle se trouve être celui de la régression logistique.

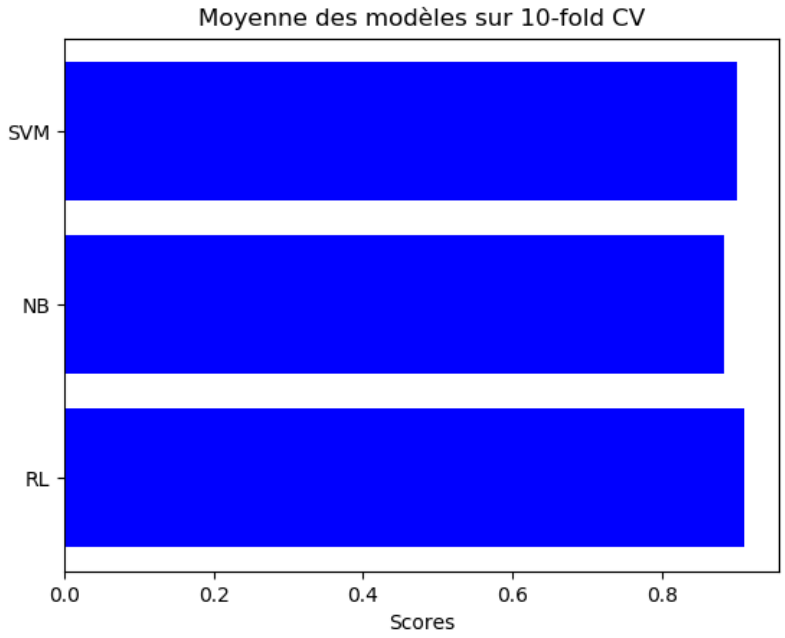


FIGURE 2.10 – Moyenne de la cross validation sur différents modèles

On observe sur cet histogramme que la moyenne la plus élevée est celle de la régression logistique avec environ 0.91 suivi par celle du SVM avec 0.90 puis de celle de Naive Bayes avec 0.88.

1.5 PostProcessing

Il serait dommage de passer à côté d'une certaine astuce que l'on peut mettre en place pour avoir un bon score à la fin, il s'agit de faire un lissage sur les labels prédits avec la méthode fenêtrée, cette dernière consiste à prendre les k voisins les plus proches et de calculer la moyenne pondérée avec des poids que nous avons attribué au préalable à nos classes afin de les équilibrer. Ensuite il s'agit de reconstituer les labels en se basant sur les moyennes pondérées calculées, à chaque fois qu'on rencontre un label (M ou C) on ajoute autant de C ou de M équivalent à la moyenne. Pour finir, comme il s'agit d'un discours il est très peu probable de retrouver des labels seuls, c'est à dire qui ne soient pas en bloc donc on pourrait simplement les transformer par le label inverse, par exemple si on avait la séquence : $M\ M\ M\ C\ M\ M\ M$ cette dernière deviendrait $M\ M\ M\ M\ M\ M\ M$

2 Analyse de sentiments

La tâche consiste à prédire si une revue de film est positif ou négatif, pour cela on dispose d'une base de 2000 revues étiquetées avec la moitié positive et l'autre moitié négative.

2.1 Preprocessing

Codage

Encore une fois nous allons nous intéresser à trois codages différents : TF, binaire et TF-IDF. Il serait intéressant d'accorder un poids moins important aux termes récurrents dans le cadre du codage TF-IDF, mais cela reste difficile à prouver, une étude sera faite dans ce cadre.

Stopwords

Retirer les stopwords ici semble une bonne chose car en général, ils n'apportent aucune information concrète concernant leur classification.

Stemming

Comme pour le travail précédent, difficile de savoir l'impact de la racinisation.

Lowercase

Il peut être intéressant de considérer les majuscules dans le cadre d'une revue, on a plus tendance à écrire en majuscule des mots importants et donc considérer comme discriminants.

N-grams

Il serait préférable de ne pas considérer que les unigram, les bigrams semblent être intéressants si on prend par exemple "Not Good" qui ne veut plus dire la même chose que "Good", donc encore une fois nous allons considérer les unigram, bigrams et 1-2 grams

Tokenization

La regex prise en compte sera la même que précédemment : `r"\b[^\d\W]+\b"`

2.2 Métrique d'évaluation

Nous sommes en présence de classes équilibrées avec 1000 labels positifs et 1000 autres négatifs donc on peut utiliser le taux de bonne classification comme métrique d'évaluation.

2.3 Modèles d'apprentissage

Les modèles restent les mêmes que pour les présidents.

2.4 Résultats

Nos résultats ont été obtenus à la suite d'une cross validation effectuée sur CV 5-fold, nos pramètres à varier sont les suivants : stemming, codage, stopwords, lowercase, N-grams, model.

Codage	Stopwords	Stemming	Lowercase	N-Grams	LR (CV 5-fold)
TFIDF	True	True	True	1	0.8344999999999999
TFIDF	True	True	True	1-2	0.821
TFIDF	True	True	True	2	0.8029999999999999
TFIDF	True	True	False	1	0.8344999999999999
TFIDF	True	True	False	1-2	0.821
TFIDF	True	True	False	2	0.8029999999999999
TFIDF	True	False	True	1	0.8370000000000001
TFIDF	True	False	True	1-2	0.8390000000000001
TFIDF	True	False	True	2	0.797
TFIDF	True	False	False	1	0.8370000000000001
TFIDF	True	False	False	1-2	0.8390000000000001
TFIDF	True	False	False	2	0.797
TFIDF	False	True	True	1	0.8230000000000001
TFIDF	False	True	True	1-2	0.8115
TFIDF	False	True	True	2	0.8320000000000001
TFIDF	False	True	False	1	0.8230000000000001
TFIDF	False	True	False	1-2	0.8115
TFIDF	False	True	False	2	0.8320000000000001
TFIDF	False	False	True	1	0.8155000000000001
TFIDF	False	False	True	1-2	0.8059999999999998
TFIDF	False	False	True	2	0.8244999999999999
TFIDF	False	False	False	1	0.8155000000000001
TFIDF	False	False	False	1-2	0.8059999999999998
TFIDF	False	False	False	2	0.8244999999999999

FIGURE 2.11 – Tableau de la moyenne du CV 5-fold en codage TFIDF

Codage	Stopwords	Stemming	Lowercase	N-Grams	LR (CV 5-fold)
TF	True	True	True	1	0.8425
TF	True	True	True	1-2	0.8494999999999999
TF	True	True	True	2	0.789
TF	True	True	False	1	0.8425
TF	True	True	False	1-2	0.8494999999999999
TF	True	True	False	2	0.789
TF	True	False	True	1	0.8385
TF	True	False	True	1-2	0.8445
TF	True	False	True	2	0.7795
TF	True	False	False	1	0.8385
TF	True	False	False	1-2	0.8445
TF	True	False	False	2	0.7795
TF	False	True	True	1	0.8435
TF	False	True	True	1-2	0.85
TF	False	True	True	2	0.8215
TF	False	True	False	1	0.8435
TF	False	True	False	1-2	0.85
TF	False	True	False	2	0.8215
TF	False	False	True	1	0.8425
TF	False	False	True	1-2	0.8565000000000002
TF	False	False	True	2	0.8240000000000001
TF	False	False	False	1	0.8425
TF	False	False	False	1-2	0.8565000000000002
TF	False	False	False	2	0.8240000000000001

FIGURE 2.12 – Tableau de la moyenne du CV 5-fold en codage TF

Codage	Stopwords	Stemming	Lowercase	N-Grams	LR (CV 5-fold)
Binary	True	True	True	1	0.8504999999999999
Binary	True	True	True	1-2	0.8595
Binary	True	True	True	2	0.8005000000000001
Binary	True	True	False	1	0.8504999999999999
Binary	True	True	False	1-2	0.8595
Binary	True	True	False	2	0.8005000000000001
Binary	True	False	True	1	0.861
Binary	True	False	True	1-2	0.8695
Binary	True	False	True	2	0.7859999999999999
Binary	True	False	False	1	0.861
Binary	True	False	False	1-2	0.8695
Binary	True	False	False	2	0.7859999999999999
Binary	False	True	True	1	0.857
Binary	False	True	True	1-2	0.8655000000000002
Binary	False	True	True	2	0.828
Binary	False	True	False	1	0.857
Binary	False	True	False	1-2	0.8655000000000002
Binary	False	True	False	2	0.828
Binary	False	False	True	1	0.8654999999999999
Binary	False	False	True	1-2	0.874
Binary	False	False	True	2	0.8275
Binary	False	False	False	1	0.8654999999999999
Binary	False	False	False	1-2	0.874
Binary	False	False	False	2	0.8275

FIGURE 2.13 – Tableau de la moyenne du CV 5-fold en codage binaire

Ces trois derniers tableaux représentent la moyenne obtenue par CV 5-fold sur le modèle de la régression logistique. On observe que le meilleur score obtenu est issu d'un codage binaire donc présentiel avec un score de 0.874 en n'appliquant pas les stopwords et stemming.

Codage	Stopwords	Stemming	Lowercase	N-Grams	NB (CV 5-fold)
TFIDF	True	True	True	1	0.8130000000000001
TFIDF	True	True	True	1-2	0.8240000000000001
TFIDF	True	True	True	2-2	0.805
TFIDF	True	True	False	1	0.8130000000000001
TFIDF	True	True	False	1-2	0.8240000000000001
TFIDF	True	True	False	2-2	0.805
TFIDF	True	False	True	1	0.8194999999999999
TFIDF	True	False	True	1-2	0.829
TFIDF	True	False	True	2-2	0.7929999999999999
TFIDF	True	False	False	1	0.8194999999999999
TFIDF	True	False	False	1-2	0.829
TFIDF	True	False	False	2-2	0.7929999999999999
TFIDF	False	True	True	1	0.8030000000000002
TFIDF	False	True	True	1-2	0.8355
TFIDF	False	True	True	2-2	0.8305
TFIDF	False	True	False	1	0.8030000000000002
TFIDF	False	True	False	1-2	0.8355
TFIDF	False	True	False	2-2	0.8305
TFIDF	False	False	True	1	0.8084999999999999
TFIDF	False	False	True	1-2	0.8345
TFIDF	False	False	True	2-2	0.829
TFIDF	False	False	False	1	0.8084999999999999
TFIDF	False	False	False	1-2	0.8345
TFIDF	False	False	False	2-2	0.829

FIGURE 2.14 – Tableau de la moyenne du CV 5-fold en codage TFIDF

Codage	Stopwords	Stemming	Lowercase	N-Grams	NB (CV 5-fold)
TF	True	True	True	1	0.8055
TF	True	True	True	1-2	0.8225000000000001
TF	True	True	True	2-2	0.7795
TF	True	True	False	1	0.8055
TF	True	True	False	1-2	0.8225000000000001
TF	True	True	False	2-2	0.7795
TF	True	False	True	1	0.8085000000000001
TF	True	False	True	1-2	0.8160000000000001
TF	True	False	True	2-2	0.7535000000000001
TF	True	False	False	1	0.8085000000000001
TF	True	False	False	1-2	0.8160000000000001
TF	True	False	False	2-2	0.7535000000000001
TF	False	True	True	1	0.8109999999999999
TF	False	True	True	1-2	0.8325000000000001
TF	False	True	True	2-2	0.837
TF	False	True	False	1	0.8109999999999999
TF	False	True	False	1-2	0.8325000000000001
TF	False	True	False	2-2	0.837
TF	False	False	True	1	0.8135000000000001
TF	False	False	True	1-2	0.836
TF	False	False	True	2-2	0.8375
TF	False	False	False	1	0.8135000000000001
TF	False	False	False	1-2	0.836
TF	False	False	False	2-2	0.8375

FIGURE 2.15 – Tableau de la moyenne du CV 5-fold en codage TF

Codage	Stopwords	Stemming	Lowercase	N-Grams	NB (CV 5-fold)
Binary	True	True	True	1	0.8285
Binary	True	True	True	1-2	0.8390000000000001
Binary	True	True	True	2-2	0.7825
Binary	True	True	False	1	0.8285
Binary	True	True	False	1-2	0.8390000000000001
Binary	True	True	False	2-2	0.7825
Binary	True	False	True	1	0.8324999999999999
Binary	True	False	True	1-2	0.8314999999999999
Binary	True	False	True	2-2	0.7415
Binary	True	False	False	1	0.8324999999999999
Binary	True	False	False	1-2	0.8314999999999999
Binary	True	False	False	2-2	0.7415
Binary	False	True	True	1	0.825
Binary	False	True	True	1-2	0.852
Binary	False	True	True	2-2	0.8540000000000001
Binary	False	True	False	1	0.825
Binary	False	True	False	1-2	0.852
Binary	False	True	False	2-2	0.8540000000000001
Binary	False	False	True	1	0.8285
Binary	False	False	True	1-2	0.8564999999999999
Binary	False	False	True	2-2	0.851
Binary	False	False	False	1	0.8285
Binary	False	False	False	1-2	0.8564999999999999
Binary	False	False	False	2-2	0.851

FIGURE 2.16 – Tableau de la moyenne du CV 5-fold en codage binaire

Les trois derniers tableaux représentent la moyenne obtenue par cross validation 5-fold avec le modèle Naive Bayes en faisant varier les paramètres et en ne gardant que le codage binaire à chaque fois. On observe des valeurs similaires pour les différents paramètres, la plus grande valeur se rapproche de 0.86 sans stopwords, stemming et lowercase.

Codage	Stopwords	Stemming	Lowercase	N-Grams	SVM (CV 5-fold)
TFIDF	True	True	True	1	0.8495000000000001
TFIDF	True	True	True	1-2	0.842
TFIDF	True	True	True	2	0.8049999999999999
TFIDF	True	True	False	1	0.8495000000000001
TFIDF	True	True	False	1-2	0.842
TFIDF	True	True	False	2	0.8049999999999999
TFIDF	True	False	True	1	0.853
TFIDF	True	False	True	1-2	0.8540000000000001
TFIDF	True	False	True	2	0.7959999999999999
TFIDF	True	False	False	1	0.853
TFIDF	True	False	False	1-2	0.8540000000000001
TFIDF	True	False	False	2	0.7959999999999999
TFIDF	False	True	True	1	0.8525
TFIDF	False	True	True	1-2	0.8525
TFIDF	False	True	True	2	0.8405000000000001
TFIDF	False	True	False	1	0.8525
TFIDF	False	True	False	1-2	0.8525
TFIDF	False	True	False	2	0.8405000000000001
TFIDF	False	False	True	1	0.857
TFIDF	False	False	True	1-2	0.8504999999999999
TFIDF	False	False	True	2	0.8425
TFIDF	False	False	False	1	0.857
TFIDF	False	False	False	1-2	0.8504999999999999
TFIDF	False	False	False	2	0.8425

FIGURE 2.17 – Tableau de la moyenne du CV 5-fold en codage TFIDF

Codage	Stopwords	Stemming	Lowercase	N-Grams	SVM (CV 5-fold)
TF	True	True	True	1	0.8245000000000001
TF	True	True	True	1-2	0.85
TF	True	True	True	2	0.7905
TF	True	True	False	1	0.8245000000000001
TF	True	True	False	1-2	0.85
TF	True	True	False	2	0.7905
TF	True	False	True	1	0.827
TF	True	False	True	1-2	0.8394999999999999
TF	True	False	True	2	0.764
TF	True	False	False	1	0.827
TF	True	False	False	1-2	0.8394999999999999
TF	True	False	False	2	0.764
TF	False	True	True	1	0.8295
TF	False	True	True	1-2	0.8460000000000001
TF	False	True	True	2	0.825
TF	False	True	False	1	0.8295
TF	False	True	False	1-2	0.8460000000000001
TF	False	True	False	2	0.825
TF	False	False	True	1	0.8315000000000001
TF	False	False	True	1-2	0.85
TF	False	False	True	2	0.8244999999999999
TF	False	False	False	1	0.8315000000000001
TF	False	False	False	1-2	0.85
TF	False	False	False	2	0.8244999999999999

FIGURE 2.18 – Tableau de la moyenne du CV 5-fold en codage TF

Codage	Stopwords	Stemming	Lowercase	N-Grams	SVM
Binary	True	True	True	1	0.833
Binary	True	True	True	1-2	0.857
Binary	True	True	True	2	0.7945
Binary	True	True	False	1	0.833
Binary	True	True	False	1-2	0.857
Binary	True	True	False	2	0.7945
Binary	True	False	True	1	0.8445
Binary	True	False	True	1-2	0.865
Binary	True	False	True	2	0.7795
Binary	True	False	False	1	0.8445
Binary	True	False	False	1-2	0.865
Binary	True	False	False	2	0.7795
Binary	False	True	True	1	0.8404999999999999
Binary	False	True	True	1-2	0.8640000000000001
Binary	False	True	True	2	0.8280000000000001
Binary	False	True	False	1	0.8404999999999999
Binary	False	True	False	1-2	0.8640000000000001
Binary	False	True	False	2	0.8280000000000001
Binary	False	False	True	1	0.8540000000000001
Binary	False	False	True	1-2	0.8714999999999999
Binary	False	False	True	2	0.828
Binary	False	False	False	1	0.8540000000000001
Binary	False	False	False	1-2	0.8714999999999999
Binary	False	False	False	2	0.828

FIGURE 2.19 – Tableau de la moyenne du CV 5-fold en codage binaire

Même schéma que dernièrement avec cette fois-ci le modèle SVM, les valeurs restent homogènes avec plus de 0.8, et la meilleur vaut 0.87 avec le codage binaire sans stopwords, stemming et lowercase.

3 Clusters par thème

Nous avons à notre disposition un jeu de données *20newsgroups* contenant environ 20 000 documents dont 11 314 pour le train et 7532 pour le test, ce jeu de données est chargé depuis la librairie Sklearn. Notre objectif sera de diviser nos données en différents "paquets" présentant des caractéristiques communes (exemple proximité) donc constituer des ensembles homogènes.

3.1 Preprocessing

Entêtes, footers et quotes

Ce jeu de données contient plusieurs entêtes, pieds de page et citations qui ne sont pas pertinents à évaluer, il faut donc impérativement les supprimer.

Caractères spéciaux

Lorsqu'on jette un coup d'oeil à nos données, on remarque une présence de nombreux caractères spéciaux qu'on n'aimerait pas clusteriser, nous notons par exemple la présence d'adresses mail, d'URI, ponctuations ou encore de plusieurs tabulations, retours à la ligne, d'espaces, etc. De ce fait, nous décidons de les retirer en utilisant plusieurs regex spécifiques comme par exemple `www[a-z0-9\.\/:%_+.#?!@&=-]+` pour supprimer les URI.

Chiffres et mots

Afin d'éviter d'obtenir des "clusters poubelles", c'est à dire dénuer de sens, on décide de retirer les chiffres ainsi que les mots dont la longueur est inférieur ou égal à 3.

Stopwords

Les stopwords semblent être une bonne chose car on ne souhaite pas clusteriser des mots sans intérêts et donc obtenir des "clusters poubelles"

Stemming

Il est difficile d'estimer l'impact de la racinisation sur notre corpus pour l'instant.

Lowercase

Cela semble conseiller d'appliquer le lowercase et donc de traiter chaque mot de la même façon.

Codage

De nouveau, les trois types de codage présentés précédemment, celui du TF-IDF semble intéressant dans le cas où plusieurs mots reviennent souvent cepedant le TF semble également une bonne approche et reste très classique. Difficile d'avoir un à priori pour l'instant.

3.2 Modèles d'apprentissage

Plusieurs modèles s'offrent à nous, nous allons présenter LDA, NMF et Kmeans. Latent Dirichlet Distribution (LDA) est un modèle probabiliste et permet de réduire le nombre de dimensions. Non-negative matrix factorization (NMF) est une technique de réduction de dimension appliquée aux matrices creuses ce qui semble adapter pour un codage TF. Kmeans permet de partitionner les données en k groupes en faisant converger différents points.

3.3 Métriques d'évaluation

Dans notre jeu de données, les classes ne semblent pas être déséquilibrées, ainsi on peut faire valoir le taux de bonne classification. On peut aussi faire une cross-validation concernant le kmeans pour trouver le nombre de classes adéquat.

3.4 Résultats

Clusters

('Topic ', 0)	('Topic ', 1)	('Topic ', 2)	('Topic ', 3)	('Topic ', 4)	('Topic ', 5)	('Topic ', 6)	('Topic ', 7)	('Topic ', 8)	('Topic ', 9)
game	scsi	peopl	armenian	drive	peopl	imag	file	nrhj	inform
team	medic	govern	israel	problem	believ	entri	window	ethernet	mail
play	patient	state	isra	work	becaus	output	program	wwiz	includ
year	diseas	encrypt	turkish	time	onli	convert	server	gizw	program
player	health	presid	arab	onli	time	input	graphic	bhjn	space
season	food	secur	muslim	card	thing	jpeg	display	bxom	list
hockey	doctor	year	greek	good	christian	orbit	applic	pnei	number
leagu	condit	american	kill	veri	mani	printer	widget	pmfq	post
score	sale	time	turkey	thing	question	build	code	nriz	avail
period	treatment	kill	nazi	anyon	point	valu	color	ffff	data
goal	gordon	public	turk	driver	good	ground	motif	wmbxn	send
defens	cover	weapon	peopl	control	exist	defin	softwar	bxlt	version
shot	skeptic	hous	villag	someth	reason	function	font	tbxn	comput
pitch	bank	protect	jewish	hard	veri	current	avail	nkjz	work
chicago	medicin	crime	armenia	realli	jesu	print	screen	wwhj	pleas
basebal	quadra	stephanopoulo	attack	speed	someth	circuit	directori	mbxn	gener
point	caus	polic	soldier	power	differ	program	unix	jpwu	provid
divis	clinic	becaus	land	becaus	read	info	user	eqtm	address
good	shame	happen	german	sinc	person	check	includ	chzv	year
pittsburgh	surrend	nation	genocid	littl	fact	draw	sourc	gizwt	develop

FIGURE 2.20 – Clusters obtenus avec la LDA

	0	1	2	3	4	5	6	7	8	9
Pureté	1.0	1.0	0.95	0.95	1.0	1.0	1.0	1.0	1.0	1.0

Table 2.1 – Tableau de la pureté pour les 10 topics

La figure 2.17 représente les clusters que l'on a obtenu avec le modèle LDA, ces clusters sont au nombre de 10 avec une vingtaine de mots par cluster. Le codage utilisé est celui du TF avec tous les preprocessing appliqués.

Le tableau 2.1 représente la pureté de chaque cluster obtenu avec la classe dominante divisée par le nombre total de mots dans le cluster en question. On observe que la majorité des clusters atteignent une pureté de 1, montrant une bonne classification.

('Cluster ', 0)	('Cluster ', 1)	('Cluster ', 2)	('Cluster ', 3)	('Cluster ', 4)	('Cluster ', 5)	('Cluster ', 6)	('Cluster ', 7)	('Cluster ', 8)	('Cluster ', 9)
work	window	drive	game	chip	card	pleas	christian	file	peopl
time	program	scsi	team	encrypt	driver	mail	jesu	format	armenian
good	applic	disk	player	clipper	video	post	believ	program	israel
problem	font	hard	play	secur	monitor	anyon	bibl	imag	govern
veri	version	floppi	year	escrow	color	email	peopl	directori	state
year	manag	control	season	phone	mode	list	faith	convert	isra
thing	driver	boot	score	govern	port	address	exist	disk	kill
bike	server	format	hockey	algorithm	slot	send	religion	read	arab
onli	display	cabl	leagu	number	graphic	repli	belief	copi	turkish
space	screen	jumper	basebal	data	memori	info	christ	color	countri
realli	problem	switch	playoff	privaci	board	advanc	church	graphic	attack
power	mous	tape	pitch	public	diamond	appreci	truth	util	crime
someth	microsoft	problem	goal	devic	anyon	inform	life	site	weapon
engin	motif	extern	detroit	enforc	modem	someon	word	creat	live
littl	memori	intern	defens	serial	vesa	group	becaus	tiff	forc
cost	softwar	comput	toronto	secret	problem	sale	atheist	write	becaus
long	printer	instal	leaf	agenc	nubu	book	true	avail	palestinian
differ	xterm	quantum	point	technolog	acceler	offer	claim	cview	mani
sound	advanc	seagat	espn	scheme	appl	contact	moral	swap	murder
test	user	power	pick	wiretap	svga	question	question	data	muslim

Figure 2.21 – Clusters obtenus avec la NMF

	0	1	2	3	4	5	6	7	8	9
Pureté	1.0	1.0	1.0	1.0	0.95	1.0	1.0	1.0	1.0	1.0

Table 2.2 – Tableau de la pureté pour les 10 topics

La figure 2.18 représente les clusters que l'on a obtenu avec le modèle NMF, ces clusters sont au nombre de 10 avec une vingtaine de mots par cluster. Le codage utilisé est celui du TF-IDF avec tous les preprocessing appliqués.

Le tableau 2.2 représente la pureté de chaque cluster obtenu avec la classe dominante divisée par le nombre total de mots dans le cluster en question. On observe également que la majorité des clusters atteignent une pureté de 1, il n'y a qu'une valeur en dessous de 1 montrant des résultats plus concluants que ceux précédemment.

('Topic ', 0)	('Topic ', 1)	('Topic ', 2)	('Topic ', 3)	('Topic ', 4)	('Topic ', 5)	('Topic ', 6)	('Topic ', 7)	('Topic ', 8)	('Topic ', 9)
projector	perijov	chastiti	teenag	rushdi	sutcliff	nrhj	bentsen	peopl	inguiri
phig	tappara	njxp	chevrolet	defrag	bubblejet	wwiz	aloud	time	wovi
altima	majorov	surrend	clair	imak	pneumonia	triangul	thud	onli	dusseldorf
drawabl	bacitracin	intellect	indi	zoroastrian	nambla	gizw	whitten	anyon	melittin
myhint	precompil	shame	sevr	vanbiesbrouck	rickey	bhjn	uhhhh	work	omran
hexagon	hpxx	skeptic	rectum	xmosaic	bonilla	delaunay	batavia	window	quoth
trinomi	alexi	gordon	tesrt	lxmu	slaught	ffff	pistrix	good	nigel
calibra	bonn	bank	yammi	humanist	henderson	pmfq	behnk	problem	wfwg
xsizehint	vaselin	tyre	cranston	fatwa	spanki	bxom	bensen	year	netbeui
xclrp	ointment	xman	aantal	popupshel	hite	pnei	nicht	pleas	baxxx
myscreen	espoo	hypercard	snijpunten	indo	cindi	watchman	macaloon	post	rohm
scand	nilsson	methanol	frode	ucblib	guzzi	nriz	mire	drive	muzzah
decoupl	modo	timmon	fervour	libtermcap	sander	whoi	angeben	veri	turku
neurotic	noseble	ezekiel	schoolteach	speedisk	critu	wwhj	polkadot	thing	cologn
phlegmat	oakley	refil	benzodiazepin	walli	sosa	wmbxn	prizm	file	captainci
damico	raitanen	mening	surreal	fixabl	fwiw	bxlt	fnal	becaus	giveth
deali	sandiego	potassium	spectacl	vonnegut	irwin	tbxn	sirri	game	taketh
xwid	scalper	specint	deltabox	schirra	bilinski	mailread	siperian	card	winc
ywid	nettl	tylenol	sriniva	wgep	erythromycin	xelm	doco	mail	rigidli
sanguin	graig	chiropractor	niet	asimov	gonzal	networld	slimi	question	averi

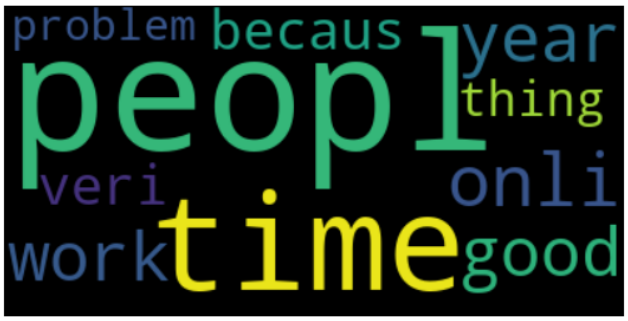
Figure 2.22 – Clusters obtenus avec la NMF

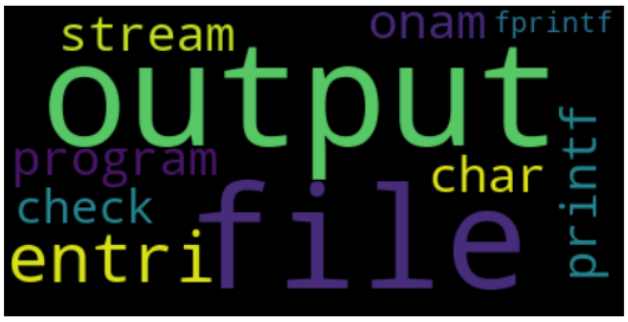
	0	1	2	3	4	5	6	7	8	9
Pureté	0.5	0.5	0.75	0.35	0.4	0.6	0.65	0.3	1.0	0.25

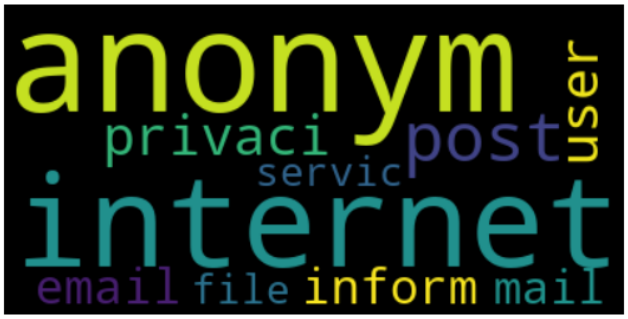
TABLE 2.3 – Tableau de la pureté pour les 10 topics

La figure 2.19 représente les clusters que l'on a obtenu avec le modèle LDA, ces clusters sont au nombre de 10 avec une vingtaine de mots par cluster. Le codage utilisé est celui du TF-IDF avec tous les preprocessing appliqués.

Le tableau 2.3 représente la pureté de chaque cluster obtenu avec la classe dominante divisée par le nombre total de mots dans le cluster en question. On observe cette fois ci que les résultats sont assez bas, avec par ex 0.3 et 0.35 pour les topics 3 et 7 donc le modèle LDA n'est pas compatible avec le codage TF-IDF.







Sur les 4 dernières images générées par la libraire *WorkCloud*, sont illustrés les clusters obtenus par Kmeans avec k=4, preprocessing et codage TF. On remarque que les résultats obtenus sont très similaires aux autres modèles et donc nous sommes parvenus à une cohérence dans les clusters et donc une bonne classification.