Rapport: traitement automatique du langage

Etudiant :
Hakim CHEKIROU
M1 DAC

Mai 2020

Classification de discours

Nous tentons ici de classifier des extraits de discours de deux anciens présidents français, Francois Mitterrand et Jaques Chirac. Notre base de données compte 57413 extraits, 86,8% sont des citations de J. Chirac et 13, 2% de F. Mitterrand.

Pré-traitement

Avant de pouvoir entraîner des modèles sur ces données, nous avons appliquer plusieurs techniques de pré-traitement présentées ci dessous.

Tokenization

Cette étape consiste à découper le texte afin d'obtenir le dictionnaire. Nous utilisons le tokenizer inclus dans la bibliothèque nltk. Nous avons examiné comment les performances changent si on utilisent des mono-grammes ou des bi-grammes.

Stémmistation ou racinisation

Il s'agit de réduire les mots à leur racine, cela pourrait être utile dans le cas ou seul le mot utilisé importe. On utilise la classe frenchStemmer du module snowball de nltk.

Lemmatisation

Similaire à la stemmisation, sauf que les mots sont réduit à leur entrée lexicale commune ("forme canonique" enregistrée dans les dictionnaires de la langue, le plus couramment), que l'on désigne sous le terme de lemme. Pour cette tache, Spacy est utilisé.

Suppression des mots vides (Stop Words)

Un mot vide (ou stop word, en anglais) est un mot qui est tellement commun qu'il est inutile de l'indexer ou de l'utiliser pour la classification. On génère 3 versions du texte sans les mots vides, une version avec le texte brut, une avec le texte lemmatisé et une dernière avec le texte stemmisé. Ici aussi, on utilise Sapcy.

Étiquetage morpho-syntaxique (Pos Tag)

Nous distinguons ici la nature grammaticale des discours. Pour chaque type de mot, on compte ses occurrences dans le texte. Spacy inclus un pos tagger.

Tests

Pour tout nos tests, nous appliquons une validation croisée sur 5 échantillons.

Codage fréquentiel

Nous avons choisi d'utiliser le classifieur NaiveBayesMultinomial, car les données étant trop volumineuses, c'est le seul algorithme qui est assez rapide et performant pour cette tache.

Mono-grammes :

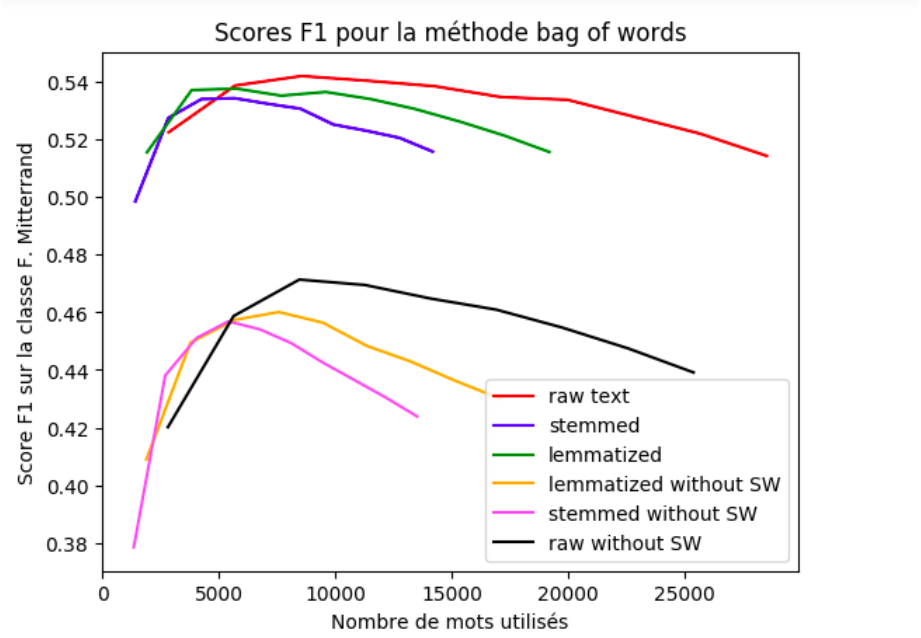

Figure 1: Comparaison des performances des différentes techniques de prétraitement

Dans la figure 1: Nous comparons les scores des différentes méthodes de prétraitement en fonction du nombre de mots pris en considération. Il n'y a pas de différences apparente entre la stémmisation, la lémmatisation ou l'utilisation des textes bruts. Cependant les performances diminuent de près de 10% en éliminant les mots vides. Le nombre optimal de mots à utiliser est entre 5000 et 10000 mots, soit l'équivalent de 30% des mots disponibles pour chaque prétraitement.

Bi-grammes :

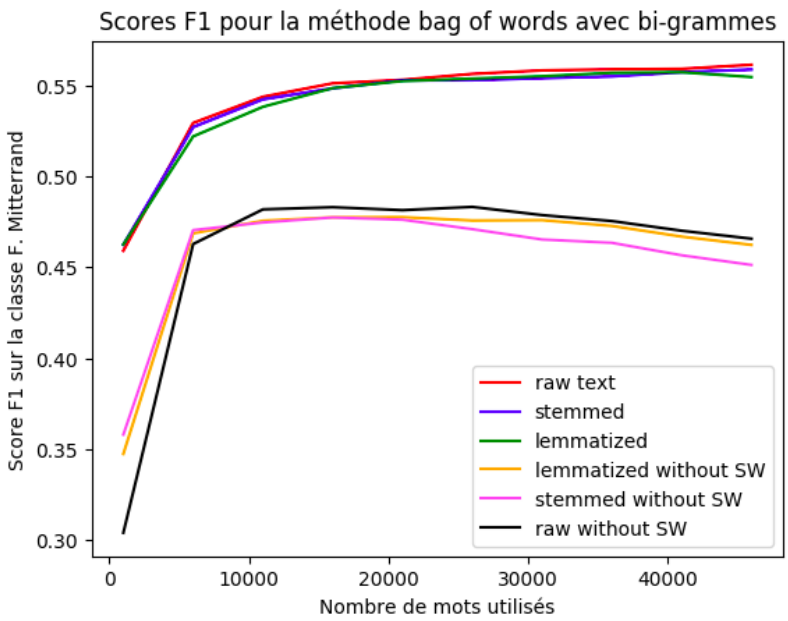

Figure 2: Comparaison des performances des différentes techniques de prétraitement en utilisant des bi-grammes

En utilisant des bi-grammes, les performances augmentent légèrement et ne semblent pas diminuer avec l'augmentation du nombre de mots, ils se stabilisent à près de 55% pour les version avec les mots vides. Mais nous gardons la même observation que précédemment, la suppression des mots vides diminue les performances et la lémmatisation/racinisation n'améliore pas le classifieur.

TF-IDF

Dans cette partie, Nous utilisons la pondération TF-IDF. Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus. Pour l'expérience suivante, nous n'avons considéré que des mono-grammes

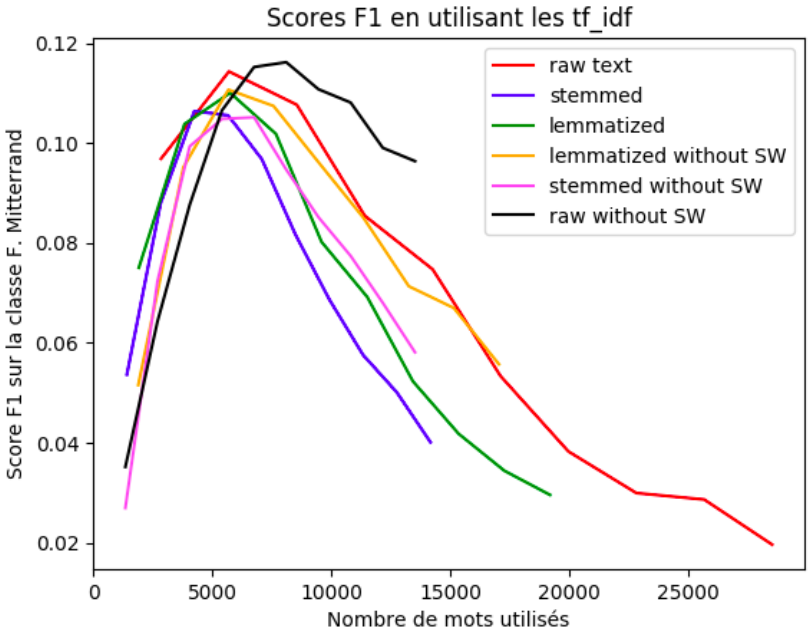

Figure 3: Performances en utilisant la pondération TF-IDF

Cette technique ne donne pas plus de 12% pour la mesure F1. Toutes les techniques de pré-traitement donnent des résultats similaires. La baisse de performance n'est pas surprenante puisque la pondération TF-IDF vise à donner un poids moins important aux termes les plus fréquents, considérés comme moins discriminants. Cependant, les mots fréquents utilisés de différentes manières peuvent permettre une bonne classification.

POS tagging

Ici nous explorons l'étiquetage morpho-syntaxique. Puisque nous ne travaillons plus avec du texte mais les occurrences de chaque type de mots (verbe, nom, etc), nous pouvons nous permettre d'utiliser des algorithmes plus lourds.

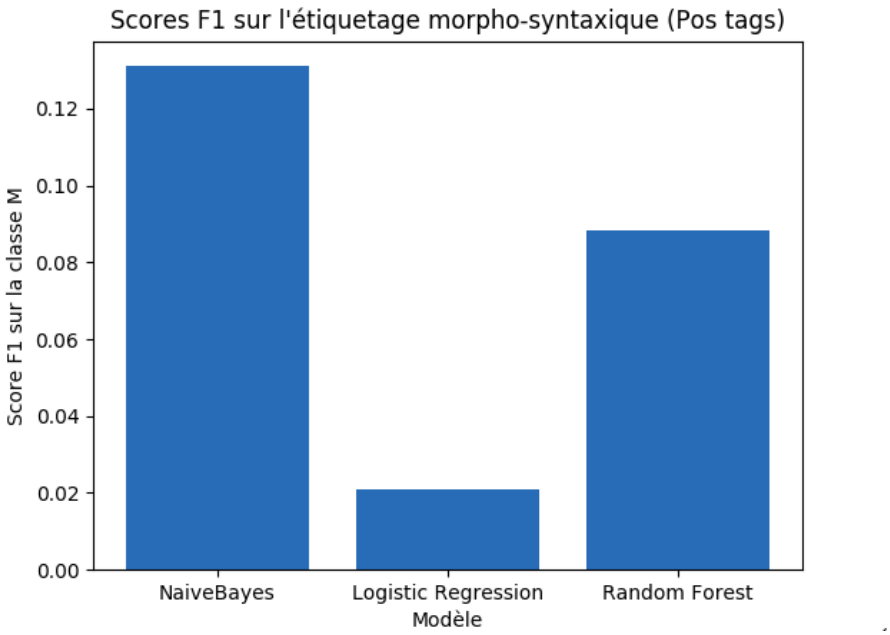

Figure 4: Classification en utilisant l'étiquetage morpho-syntaxique

Ici aussi, les performances sont très basses. On peut en conclure que les deux orateurs utilisent des phrases avec des structures similaires. L'étiquetage morpho-syntaxique est donc inutile pour cette tache.

Post-traitement

Les données d'entraînements sont structurées en blocs, c'est a dire qu'il y a un certain nombre de citation de F. Mitterrand puis un bloc de citation de J. Chirac et ainsi de suite. Nous lissant donc les prédictions données par notre meilleur modèle: NaiveBayesMultinomial. Pour chaque prédiction, nous regardons les prédictions pour les 5 citations juste avant et les 5 prédictions juste après, et nous lui attribuons la classes dominantes. Les résultats sont représentés dans le figure 5.

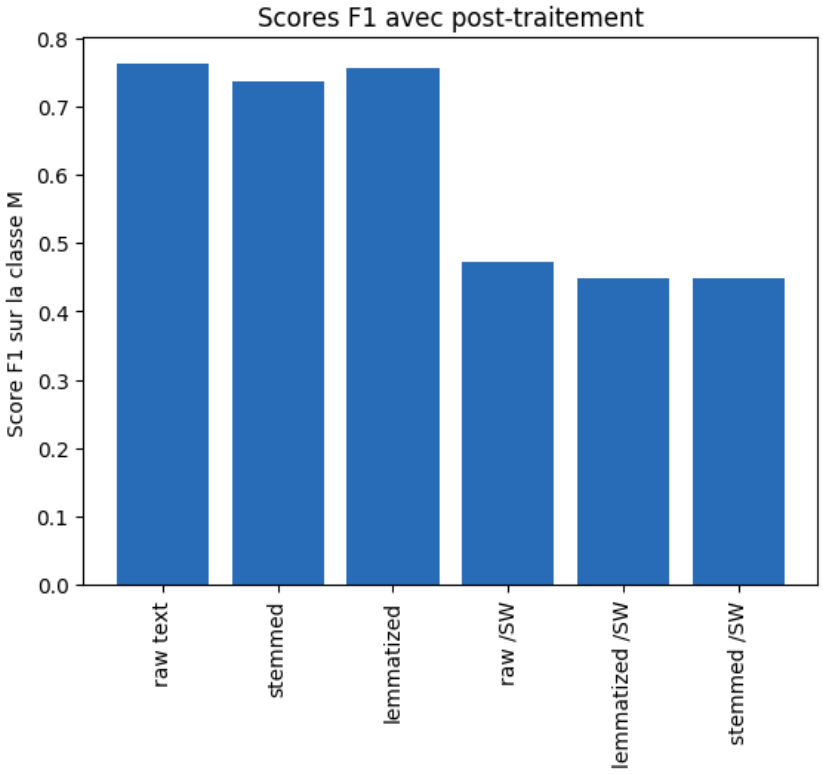

Figure 5: Post-traitement sur les prédiction du modèle Bayesien

La mesure F1 a augmenté de presque 20%. Comme précédemment, nous n'observons pas de différences entre la lemmatisation/stemmisation et les textes utilisés directement. Le post-processing n'améliore pas les performances des données sans les mots vides.

Motifs de parole des deux présidents

Nous allons regarder les termes que notre meilleur modèle utilise pour différencier entre les deux orateurs. Nous aurions pu regarder pour chaque classe, les termes les plus probables sachant cette classe $P(x/y)$, mais cela donne les meme mots pour les deux classes et ce sont des mots vides. Nous choisissons plutôt de regardes les $P(y|x)$. Dans la figure suivante, on a affiché pour chaque classe, les 10 termes avec la plus grande probabilité de cette classe sachant le terme $P(y|x)$.

Mitterrand	Chirac
quelle façon	votre capitale
quoi que	ultimes
On va	Naturellement il
voilà que	ancré
société industrielle	économies de
madame	Nation tout
Eurêka	pour évoquer
Morvan	grande exposition
convenait de	Météor
CEE	système juridique

Figure 6: 10 termes avec la plus grande probabilité $P(y/x)$ pour chaque classe.

Ce ne sont pas les termes ou bi-grammes les plus fréquents, mais quand ils apparaissent, ils sont presque exclusivement utilisé par un des deux orateurs. Il y a certains mots comme "Morvan" ou "CEE" relatifs à des dossiers qu'un président a traité et pas l'autre, mais pour le reste, ce sont des mots communs. Ces termes, quand ils apparaissent, sont de bons indicateurs de leur orateur.

Topic Modeling : Allocation de Dirichlet latente (LDA)

Un Topic model ou modèle de sujet est un type de modélisation statistique pour la découverte de sujets abstraits dans une collection de documents. L'allocation de Dirichlet latente (LDA) est un exemple de topic modeling qui est utilisé pour classifier des textes dans un document à un sujet particulier. La LDA construit un sujet par modèle de document et des mots par modèle de sujet, modélisé par une distribution de Dirichlet.
Nous allons utiliser la LDA pour déterminer les sujets les plus évoqués. Nous utilisons la bibliothèque Gensim. Nous avons tenté plusieurs valeurs différentes pour le nombre de sujets, on garde les résultats pour 8 sujets. La liste des mots par sujet est donnée ci-dessous.

- Sujet 1 : date, année, falloir, engager, oeuvre, y, réforme, mettre, devoir et aller.
- Sujet 2 : droit, social, démocratie, responsabilité, sécurité, devoir, solidarité, politique, liberté et respect.
- Sujet 3 : européen, France, pays, Union, coopération, Europe, politique, international, relation et partenaire.
- Sujet 4 : nom, Monsieur, président, vouloir, avoir, monsieur, ici, français, y et date.

- Sujet 5 : falloir, entreprise, jeune, service, vie, public, français, nouveau, travail et pouvoir.
- Sujet 6 : pays, monde, grand, problème, Europe, mondial, développement, économie, marché et France.
- Sujet 7 : France, peuple, histoire, français, grand, pays, être, homme, Europe et culture.
- Sujet 8 : faire, avenir, vivre, savoir, monde, enfant, temps, France, confiance et société.

On peut deviner le sujet en utilisant les mots, par exemple le sujet 3 parle clairement de relation internationales et de l'union européenne. Les sujets 5 et 6 parlent d'économie, le 7 de l'histoire de France, etc. On peut utiliser ces sujets pour appliquer une clusterisation sur les textes. Pour chaque président, on applique le clustering de ses textes pour voir ses sujets de prédilection. Sur les figures suivantes, on représente les resultats pour chaque président.

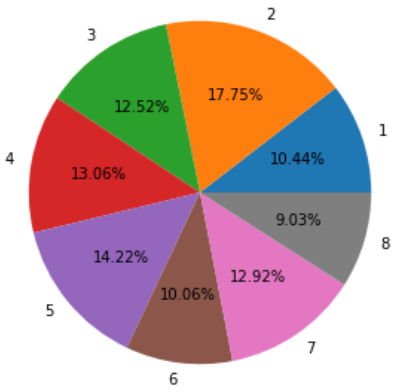

Figure 7: Distributions des sujets pour le président J.Chirac

Les sujets semblent distribués presque uniformément, J. Chirac à tendance a parler plus du sujet 2, qui pourrait être intitulé "valeurs et devoirs".

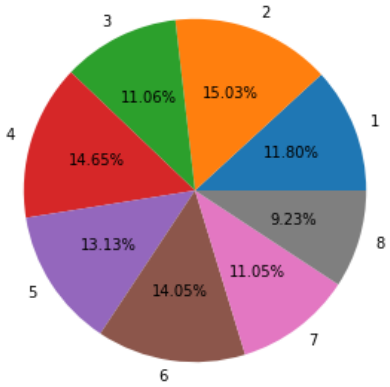

Figure 8: Distributions des sujets pour le président F. Mitterrand

La distribution des sujets dont parle F. Mitterrand est très similaire à celle de J. Chirac. Une différence notable est que F. Mitterrand parle moins du sujet 2 et plus du sujet 6 "monde et économie".
Mais les deux distributions reste très similaires et c'est assez logique, les deux présidents travaillent sur les mêmes dossiers: économie, relations internationales, etc. On ne peut pas s'attendre à ce que les deux présidents parlent de sujets complètement différents, c'est pour cette raison que des techniques plus poussées, pouvant déterminer les motifs de paroles étaient nécessaires.

Analyse de sentiments

Nous tentons ici d'analyser les sentiments des critiques de films. Pour cela, nous utilisons une base d'entraînement de 1000 avis positifs et 1000 avis négatifs. Le but étant de discriminer entre les avis positifs et négatifs dans un fichier de tests de 25000 avis.

Pré-traitement

Comme pour la tache précédente, nous allons générer des données d'entraînement en racinisant, lémmatisant, en enlevant les mots vides et en faisant un étiquetage morpho-syntaxique sur les textes bruts. Nous avons utilisé les mêmes modules que précédemment, mais en chargeant les modèles anglais vu que les avis sont en anglais.

Codage fréquentiel

Nous réalisons ici des tests pour le codage fréquentiel avec deux algorithmes, NaiveBayes multinomiale et Machine à vecteur de support linéaire (LinearSVC). Pour les tests, une validation croisée en 5-folds est réalisée. Les résultats sont résumés dans le tableau suivant.

	Naive Bayes			SVM		
	Mono-gramme	Bi-gramme	Tri-gramme	Mono-gramme	Bi-gramme	Tri-gramme
Textes bruts	82.5%	83.9%	84.1%	82.7%	84.9%	85.2%
Stémmatisation	80.8%	83.5%	83.7%	82.8%	85.2%	85.4%
Lémmatisation	81.6%	83.5%	84.1%	83.7%	85.60%	85.55%
Stop Words	82.2%	81.8%	81.5%	82.7%	83.1%	83.4%
Stop Words: lemmatisé	80.7%	81.0%	80.7%	81.4%	81.6%	81.6%
Stop Words: stemmisé	80.7%	81.3%	81.4%	82.4%	82.9%	82.9%

L'exactitude des modèles ne varie pas énormément en changeant le type de modèle ou en utilisant des mono, bi, ou tri-grammes. Cependant, il y a une légère amélioration en passant à des bi-grammes et l'utilisation des SVMs rajoute entre 1% et 2%. Le meilleur modèle pour ce codage est de $85,6\%$ et il est obtenu en utilisant des bi-grammes et un SVM en appliquant la lémmatisation.

Codage TF-IDF

Nous tentons ici l'indexation TF-IDF. Les résultats sont résumés dans le tableau ci-dessous.

	Naive Bayes			SVM		
	Mono-gramme	Bi-gramme	Tri-gramme	Mono-gramme	Bi-gramme	Tri-gramme
Textes bruts	82.25%	83.15%	83.50%	85.50%	86.85%	86.84%
Stémmatisation	81.0%	83.10%	83.65%	84.0%	86.65%	86.50 %
Lémmatisation	82.05%	83.7%	84.35%	85.4%	87.20%	87.25%
Stop Words	80.7%	80.75%	80.8%	84.4%	84.3%	84.35%
Stop Words: lemmatisé	81.15%	80.89%	80.6%	83.95%	83.75%	83.70%
Stop Words: stemmisé	81.60%	81.70%	81.55%	83.95%	84.5%	84.55%

Ici aussi les scores ne varient pas, mais on arrive à atteindre 87.25%. la suppression des mots vides semble diminuer les performances. Nous gardons alors le modèle SVM en utilisant des tri-grammes sur les données lémmatisés.

Part of speech tagging

Nous avons testé plusieurs algorithmes sur les données étiquetés morpho-syntaxiquement.

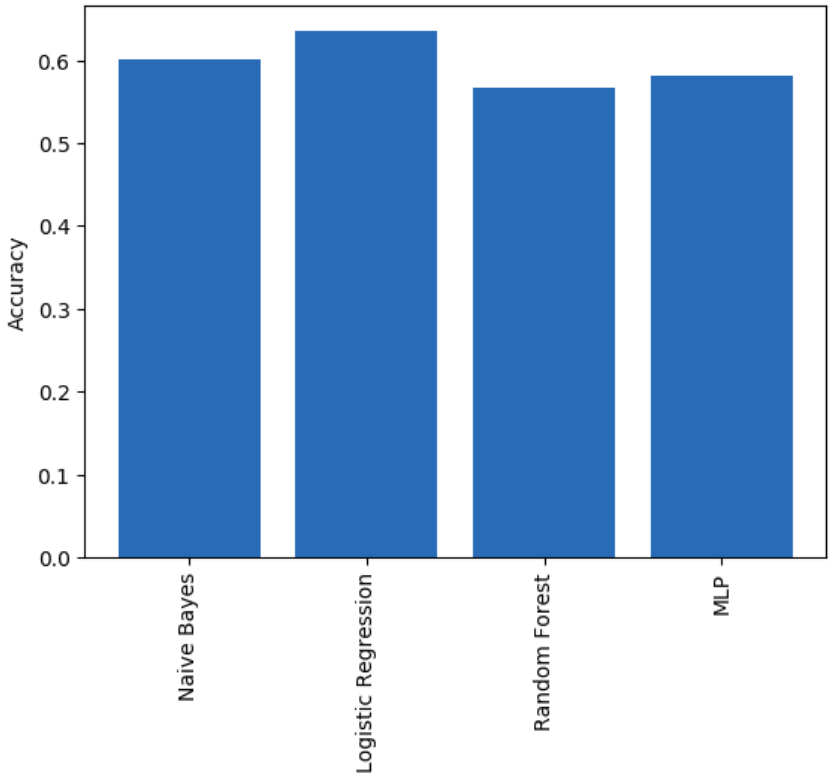

Figure 9: Accuracy sur les pos tags

Cette technique donne des résultats inférieurs de 20% au meilleur modèle précèdent. Nous n'allons donc pas l'utiliser.

Word2Vec

Word2vec est l'une des techniques les plus populaires pour apprendre les word embeddings. C'est un réseau de neurones à deux couches qui prend en entrée un corpus et retourne en sortie un ensemble de vecteurs. Les Word embeddings

obtenus avec word2vec peuvent rendre le langage naturel compréhensible par un ordinateur, puis des méthodes plus poussés peuvent être utilisées pour trouver les similarité entre les mots. Un ensemble bien entraîné de vecteur-mots placera des mots similaires proche dans l'espace de représentation. Par exemple les mots homme, femme et humain formeront un groupe et les mots jaune ,rouge et bleu seront dans un autre groupe séparé.
Pour réaliser cette partie, nous utiliserons l'implementation de word2vec incluse dans la bibliothèque Gensim, plus particulièrement, la méthode d'entraînement continuous bag of words(CBOW).
Nous commençons d'abord par essayer de déterminer la taille de vecteur qui convient le mieux. Pour 4 tailles différentes de vecteur, nous lançons 3 algorithmes diffèrent et on test le taux de bonne classification. La taille de la fenêtre a été fixée à 10 pour ce test. La figure suivante résume l'expérience.

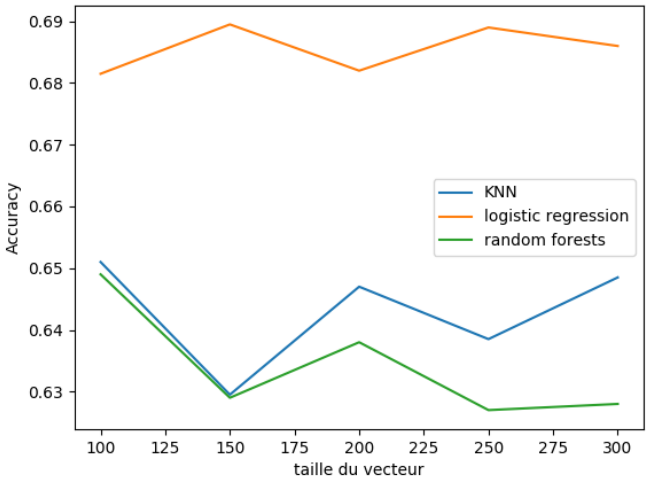

Figure 10: Taux de bonne classification pour la méthode Word2Vec selon la taille du vecteur

Même si on peut voir que la régression logistique classifie un peu mieux les données, la classification n'est pas très précise.
Pour déterminer la taille de la fenêtre optimale, nous testons plusieurs valeurs de fenêtre. La figure 11 résume l'expérience.

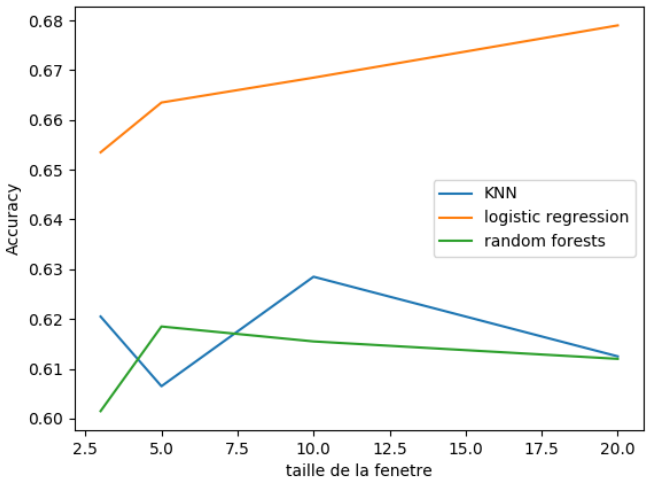

Figure 11: Taux de bonne classification pour la méthode Word2Vec selon la taille de la fenêtre

Comme précédemment les performances sont basses et il n'y a pas de lien apparent entre la taille de la fenêtre et le taux de bonne classification.
La raison pour laquelle notre modèle est si peu performant vient du fait que l'ensemble d'apprentissage ne contient que 2000 documents et que ce modèle n'a pas été pré-entraîné. On ne peut pas s'attendre à avoir de bon scores avec seulement 2000 instances d'apprentissage.

Termes discriminants

Pour conclure cette partie, nous allons regarder les termes discriminants utilisés par notre meilleur modèle, la version avec la lémmatisation en tri-grammes codé en TF-IDF en appliquant un SVM.

Negatif	positif
bad	great
waste	fun
plot	life
boring	performance
nothing	very
attempt	be also
unfortunately	excellent
the bad	overall
suppose	quite
poor	many
any	perfectly
should have	hilarious
harry	truman
why	bulworth
ridiculous	cameron

Figure 12: Les termes discriminants

On peut voir que les adjectifs positifs et négatifs sont les termes les plus discriminants, mais il y a certains mots, comme "cameron" qui font référence au réalisateur "james cameron", dont les films sont bien accueillis. Il y a aussi des termes qui font références à ce qu'un film aurait du être comme "should have", "unfortunately" ou "attempt" et qui a été décevant. Ce modèle semble donc bien modéliser le langage utilisé pour exprimer le sentiment envers un film.

Conclusion

A travers les deux taches qui constituent ce projet, nous avons couvert une variété de techniques en traitement du langage naturel, que ce soit en supervisé ou en non-supervisé. Nous avons comparé les codages possibles, les méthodes utilisant la nature grammaticale des mots, les prolongements de mots "embeddings" et nous avons lancé plusieurs modèles d'apprentissage. Notre travail est sujet à amélioration, nous pourrions tenter d'utiliser un modèle de Word2Vec pré-entraîné ou bien d'utiliser un réseau de neurones récurrents avec LSTM. Le sujet reste donc ouvert.