# Traitement Automatique des Langues Naturelles

## Devoir 1 - Classification de blogs


### Abstract

Dans ce document nous nous intéressons à la création d'un classifieur permettant de prédire la tranche d'âge de l'auteur d'un blog. Nous avons commencé par créer un classifieur très simple, permettant d'obtenir une accuracy de référence. Puis, nous avons séléctionné trois classifieurs que nous avons pu comparer entre eux et à la référence. Ces classifieurs nous ont mené à établir des statistiques sur le corpus d'entrainement. Enfin, à l'aide de ces résultats et en ajoutant une étape de préprocessing, nous avons pu améliorer l'accuracy des classifieurs.


## 1 Baseline

Nous commençons par étudier un modèle de classifieur simple, ce qui nous permettra d'avoir un score de référence pour l'étude des classifieurs plus complexes que l'on exposera par la suite.
Le modèle que nous avons codé prédit une catégorie parmi les trois possibles avec comme probabilité la distribution des classes dans le corpus. Les distributions des catégories étant définies comme suit :

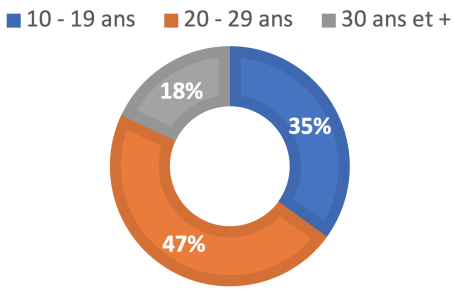


Figure 1: Distribution des 3 catégories dans le corpus d'entraînement

En prenant une distribution de 10.000 points basée sur ces probabilités et en comparant avec les labels de *test_split01*, on a une accuracy de **37.7%**. En prenant DummyClassifier de scikit-learn, avec comme stratégie de prendre la classe la plus fréquente, nous obtenons une accuracy de **46.8%**. Ce résultat est cohérent puisqu'il est le reflet de la distribution de la classe "20-29 ans", classe ayant la même répartition dans le corpus d'entrainement et de test. Dorénavant nous utiliserons donc ces scores comme référence, afin d'évaluer la qualité de nos modèles.

## 2 Choix des classifieurs

Pour mener à bien la tache de classification des blogs, nous avons choisi trois algorithmes différents : Naive Bayes (version adaptée au traitement de texte), Random Forest et une régression logistique. Ils prennent chacun pour entraînement une matrice dite de *bag of words* calculée sur une partie du corpus d'entrainement sans pré-traitement. Chaque colonne de la matrice représente le nombre d'occurence d'un mot de vocabulaire pour chaque blog du corpus. Nous évaluerons l'importance du pré-traitement sur les prédictions dans un second temps (cf. section 4).
La régression logistique et Random Forest que nous utilisons proviennent de la librairie Scikit-Learn.
L'algorithme de Naïve Bayes que nous avons utilisé a été créé pour un projet de notre cours de Machine Learning. Son fonctionnement est le suivant :

l'algorithme repose sur trois fonctions d'etimation de probabilité. Chacune indique la probabilité qu'une phrase appartienne à une classe d'âge. Ces fonctions s'entrainent sur les blogs de la classe qui leur est dédiée (i.e. la fonction indicatrice de la classe 1 s'entraine sur toutes la phrases de la classe 1) en récupérant le nombre total d'occurences de chaque mot. Puis, elles additionnent la fréquence observée pendant l'entrainement de chaque mot de la phrase test.

Cette somme correspond à la probabilité qu'une phrase soit effectivement dans la classe qu'elles qualifient. Enfin, après multiplication de ces mots par des coefficients d'a priori, l'algorithme choisit la classe ayant la plus grande probabilité. Afin de pouvoir appliquer les différents algorithmes de classification, nous avons décidé de ne travailler que sur une partie du corpus d'entrainement. Nous limitons ainsi la mémoire réquisitionnée par l'entrainement (la conversion en objet array des matrices de dimension trop élévée étant particulièrement coûteuse). Nous avons déterminé un vocabulaire de 33.000 mots : il s'agit des mots présents plus d'un nombre n (choisi empiriquement) de fois dans l'ensemble du corpus d'entrainement. Pour tester ces classifieurs, nous travaillons donc avec une matrice *bag of words* calculée sur 33.000 mots (uni-grammes) et avec un corpus d'entraînement de différentes tailles. De plus, nous avons ajouté à ce vocabulaire les 70 mots de chaque classe ayant les poids les plus importants dans le cadre de la regression logistique. En effet, ces mots représentent des features fortement discriminants.
Nous avons obtenu les résultats suivants:

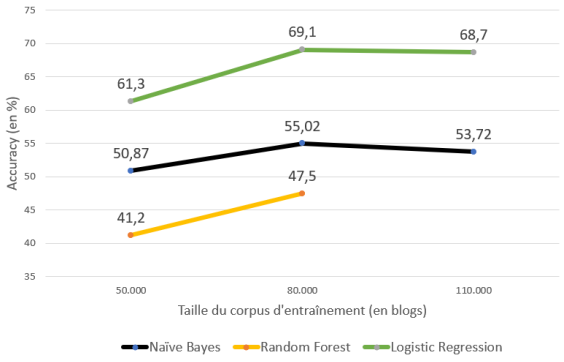


Figure 2: Accuracy des différents classifieurs en fonction de la taille du corpus d'entraînement avant pré-traitement

Nous remarquons que le classifieur Random Forest est largement sur-entraîné : il prédit très souvent la deuxième catégorie (20-29 ans), atteignant une accuracy de **47,5%** donc dépassant à peine DummyClassifier. (Suite à des problèmes de taille de mémoire, nos ordinateurs n'ont pas réussi à obtenir de résultats pour les 110.000 blogs avec ce classifieur.)
Aussi, la régression logistique se présente comme étant le meilleur classifieur pour notre étude, présentant une accuracy de **69,1%** pour un set d'entraînement de 80.000 blogs.

Nous avons également calculé le temps d'exécution des différents algorithmes testés, présenté à la Figure 3.

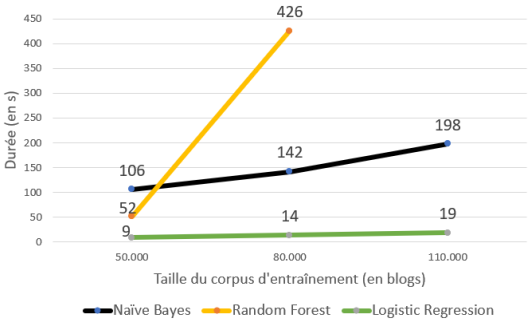


Figure 3: Temps d'éxecution des différents algorithmes avant pré-traitement

En plus de présenter les meilleurs accuracy, la régression linéaire est le classifieur qui s'exécute le plus rapidement. Il est à noter que l'accuracy varie de manière très notable en fonction du set d'entraînement pris en compte. On observe notamment à la Figure 2 que prendre les 80.000 premiers blogs du corpus train comme entraînement permet au classifieur de faire de meilleures prédictions que lorsque l'on prend les 110.000 premiers blogs. Nous allons donc analyser le corpus d'entraînement dans la section suivante afin de comprendre ce résultat.

Pour conclure, il nous est apparu au cours de ces premières expérimentations que deux algorithmes étaient capables régulièrement (dans toutes nos expérimentations) de battre la baseline établie en première partie : la regression logistique et Naive Bayes. D'autre part, nous avons remarqué que nos courbes d'apprentissage ont une forme de cloche, avec un maximum aux alentours de 80.000 blogs appris. Ce phénomène sera détaillé dans la section 3.

## 3 Analyse statistique du corpus d'entrainement

En entrainant nos classifieurs sur différents extraits du corpus d'entrainement, il nous est apparu une forte disparité dans les résultats obtenus. Ainsi la regression logistique nous présentait des accuracy variant de $56.8\%$ à $70,3\%$ selon qu'on l'entraine sur les 80.000 premiers blogs, ou les 80.000 derniers du corpus d'entrainement.
Afin d'améliorer la création des bag of words, et

dans un but d'optimisation des capacités des classifieurs décrits dans la section précédente, il nous a dès lors paru essentiel de réaliser quelques statistiques sur le corpus d'entrainement.
Dans un premier temps, nous avons vérifié l'équirépartition des blogs en fonction des catégories de tranche d'âge au sein du corpus :

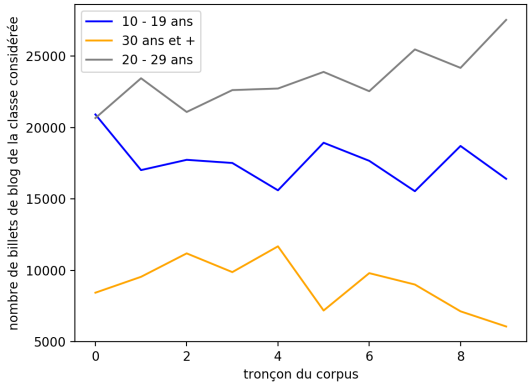


Figure 4: Répartition du nombre de blogs des différentes classes au sein du corpus d'entrainement.

La Figure 4 nous permet de remarquer une linéarité relative tout au long du corpus d'entrainement de la répartition des billets de blog de chacune des catégories. Dès lors, la différence notifiée auparavant entre les parties du corpus nous a parue porteuse d'intérêt en ce qu'elle permet de mettre en évidence une inégalité de l'information portée par les différents tronçons du corpus. Nous avons alors réalisé une analyse statistique de ces deux tronçons. Pour commencer, nous nous sommes proposés de comparer stucturellement les deux parties du corpus :

Table 1: Distribution des 3 catégories

| 80.000 billets du | début | fin |
|---|---|---|
| moyenne de mots | 222 | 237 |
| moyenne de caractères | 992 | 1.063 |

La table ci-dessus permet de mettre en évidence la répartition équitable des billets dans le corpus.
Ensuite, utilisant les mots ayant le plus de poids dans le modèle de régression logistique tel que décrit dans la section 2, nous avons essayé d'observer leur répartition moyenne au sein du corpus d'entrainement. En utilisant les cinquante mots ayant le poids le plus important au sein de chaque classe, nous avons obtenu les résultats suivants :

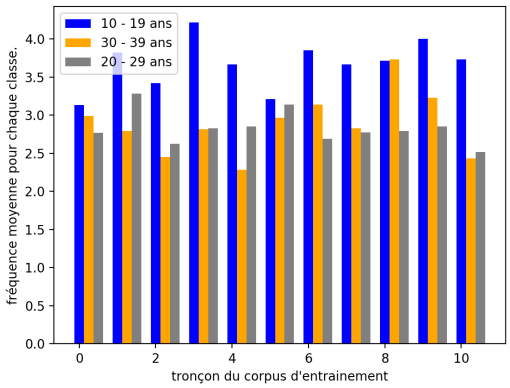


Figure 5: Répartition des mots avec les poids les plus importants au sein du corpus d'entrainement.

Si l'on ne remarque pas immédiatement de disparité dans la répartition, on peut à l'inverse s'intéresser à la figure 6:

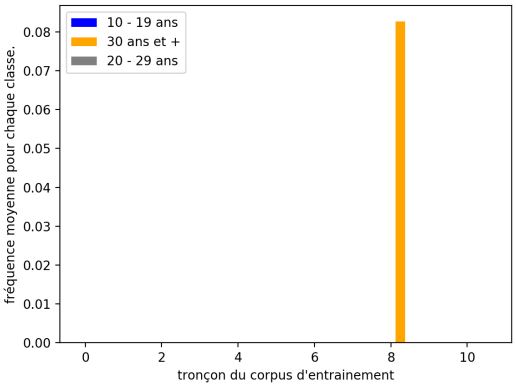


Figure 6: Répartition du mot 'Corsair' dans le corpus d'entrainement.

On remarque une forte disparité de la répartition du mot *Corsair* au sein du corpus. En fait, il s'avère que le corpus d'entrainement est assez inégal dans la répartition du vocabulaire ayant un poids assez important.
Nous avons donc décidé de nous intéresser aux modifications sur le vocabulaire des textes, c'est à dire que nous avons étudié l'influnce de la normalisation des textes.

## 4 Influence du pre-traitement

Pour pouvoir comparer avec les trois classifieurs de la section 2, il faut entraîner ces classifieurs dans un environnement semblable à celui présenté à la section 2. Nous les avons donc une nouvelle fois entraîné sur les 50.000, 80.000 et 110.000

premiers blogs du corpus. La taille du vocabulaire pris en compte est désormais plus faible car de nombreux mots ont été normalisé suite au pré-traitement, travaillant ainsi avec un vocabulaire de 8.412 mots. De plus, pour rester cohérent avec la section 2, nous avons gardé le même seuil de fréquence d'apparition en dessous duquel le mot n'est pas pris en compte dans le vocabulaire. Nous ajoutons à cette liste de vocabulaire les mots qui ont le plus de poids lors de la régression logistique. C'est le même procédé mais calculé cette fois sur les documents pré-traités.
Les résultats obtenus sont présentés à la Figure 7.

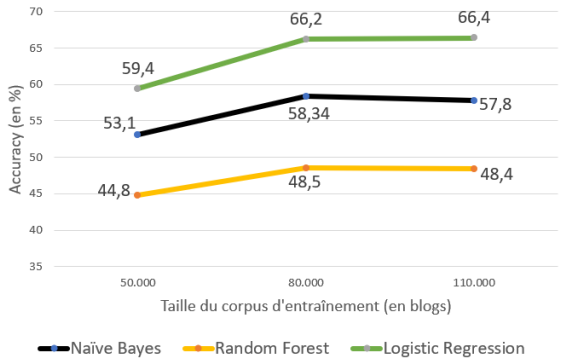


Figure 7: Accuracy des différents classifieurs en fonction de la taille du corpus d'entraînement après pré-traitement.

Nous observons que la régression logistique présente toujours les meilleurs résultats, tandis que Random Forest est encore le moins bon des trois classifieurs, dépassant à peine DummyClassifier présenté à la section 1.
Cependant, il est assez surprenant de noter que sur les 80.000 premiers blogs du set d'entraînement, la régression logistique présente une accuracy de **69,1%** avant le pré-traitement contre **66,2%** après le pré-traitement.
Les accuracy obtenues avec notre Naïve Bayes et avec Random Forest ont quant à elles augmenté après le pré-traitement.
Afin de vérifier que nous n'avons pas introduit un biais en ne calculant nos résultats que sur un vocabulaire réduit, nous avons décidé d'augmenter la taille du vocabulaire. Nous avons donc refait les calculs, cette fois avec un vocabulaire faisant approximativement la même taille que celui de la partie 2. Les résultats obtenus sont présentés dans le tableau 2.

On observe qu'en prenant un vocabulaire d'une taille similaire à celle de la section 2,

Table 2: Accuracy de la régression logistique en fonction de la taille du vocabulaire

| mots de vocabulaire : | 8.412 | 33.340 |
|---|---|---|
| après pré-traitement (en %) | 66,2 | 67,7 |

l'accuracy de la régression linéaire est de **67,7%** contre **69,1%** sans pré-traitement. La régression logistique reste donc moins efficace sur un corpus avec pré-traitement.

Ainsi, nous observons tout de même que deux des trois classifieurs utilisés, à savoir Naïve Bayes et Random Forest, voient leurs accuracy s'améliorer suite au pré-traitement. Nous émettons donc l'hypothèse que la normalisation du texte a malgré tout eu un apport positif quant à la performance de nos algorithmes. Dans ce cas comment expliquer que la régression logistique ne présente pas de meilleurs résultats ?

Il a été démontré dans le TP2 que les occurences des éléments normalisés dans le corpus étaient relativement faibles. D'autre part, nous nous sommes aperçus que lors du calcul de la matrice *bag of words* en prenant tout le vocabulaire présent dans le set d'entraînement et avec tous les blogs, les features (donc les mots) ayant les poids les plus élevés, i.e. les mots les plus influents, ne sont pas pas les mêmes selon que le corpus était pré-traîté ou non.

Table 3: Répartition du poids de la regression logistique dans l'ensemble du vocabulaire

| | 70 plus influents | Reste du vocabulaire |
|---|---|---|
| Poids cumulé | 432 | 124.560 |
| Poids moyen d'un mot | 6,17 | 0,24 |

En observant le tableau 3, nous remarquons que le poids moyen des mots les plus influents est 26 fois plus important que celui des autres mots. Cette différence d'influence montre que le classifieur s'appuie fortement sur un groupe restreint de mots pour établir ses prédictions.

Nous proposons de conclure que ce classifieur est un classifieur assez souple quant à la présence de mots non normalisés : leur présence n'affecte pas ou peu ses prédictions. Notre processus de pré-traitement quant à lui affecte l'écriture de certains mots, peu présents. Dès lors, il n'a plus

pour seul effet que de modifier le poids accordé aux mots, et donc par conséquent la liste de ceux ayant les poids les plus importants.
Ainsi, les mots normalisés du corpus auraient pu permettre d'améliorer les prédictions du classifieur, mais ces mots ayant un poids relativement faible, cette amélioration est moins importante que les erreurs entraînées par le nouveau choix des mots les plus influents par le classifieur.

Par conséquent nous pouvons affirmer que la normalisation du texte peut avoir un apport positif sur la tâche : deux de nos classifieurs ont vu leur accuracy augmenter.
La regression logistique étant l'algorithme ayant la plus forte accuracy, le pré-traitement n'améliore pas la précision de nos meilleures prédictions.

## 5 Résultats des différents modèles de langue

Nous nous intéressons maintenant au classifieur régression logistique, dont nous avons vu dans les parties précédentes qu'il était le plus efficace. Dans cette partie, nous le testons en l'entraînant sur différents modèles de langue, c'est à dire en utilisant un vocabulaire prenant en compte des unigrammes, des bigrammes et trigrammes. Le vocabulaire pourra être composé de différentes associations : seulement d'uni-grammes, d'unigrammes et de bi-grammes et autres.
Nous entraînons ce classifieur sur tout le corpus d'entraînement et en faisant varier le modèle de langue avec lequel on calcule le vocabulaire (et par conséquent le bag of words).
Les résultats sont exposés dans le tableau 4.

Table 4: Performance de la régression logistique en fonction des modèles de langue

| | Accuracy (%) | Précision (%) | Recall (%) |
|---|---|---|---|
| (1) | 70,3 | 71 | 70 |
| (2) | 68,1 | 68 | 68 |
| (3) | 63,1 | 64 | 63 |
| (1) et (2) | 72,8 | 73 | 73 |
| (1), (2) et (3) | 71,9 | 72 | 72 |
| Légende: | (1): uni-gramme | (2): bi-gramme, | (3): tri-gramme |

On observe ainsi qu'un modèle de langue composé à la fois d'unigrammes et de bi-grammes permet d'obtenir le meilleur taux de prédiction avec une régression logistique, atteignant une accuracy de **72,8%**.
Il est à noter qu'utiliser conjointement unigrammes, bi-grammes et tri-grammes ne permet pas d'améliorer ce résultat, puisque l'on a une accuracy de **71,9%**. Ainsi, l'utilisation d'un maximum de modèles de langue pour déterminer le vocabulaire et calculer la matrice *bag of words* n'augmente pas l'efficacité des classifieurs de manière automatique. Nous avons finalement un modèle capable de prévoir la classe d'âge de l'auteur d'un blog avec une précision de **72.8%**, ce qui est nettement supérieur aux algorithmes de référence exposés dans la section 1.

## 6 Conclusion

Au cours de nos recherches, nous avons calculé l'efficacité de trois différents classifieurs sur un corpus avec et sans pré-traitement. En étudiant l'influence de ce pré-traitement, des caractéristiques propres au corpus d'entraînement ont été analysées, notamment la répartition des mots les plus influents au sein de ce corpus. Afin de compléter nos recherches, nous nous sommes intéressés à différents modèles de langues permettant de calculer le vocabulaire.
Nous avons conclu que le classifieur le plus efficace était la regression logistique. Nous avons ensuite montré qu'un vocabulaire composé d'unigrammes et de bi-grammes était la meilleure option et qu'ajouter des tri-grammes à ce vocabulaire diminuait l'accuracy du classifieur.
Enfin, nous avons noté que le pré-traitement ne permet pas forcément d'améliorer le taux de prédictions ; son efficacité varie selon le classifieur utilisé : l'accuracy de Random Forest et Naïve Bayes progressant, mais celle de la régression logistique (classifieur le plus adapté à cette étude) diminuant.
Finalement, ce devoir a été pour nous l'occasion de pratiquer les modèles de langues étudiés en cours, et de comprendre que l'augmentation de l'accuracy d'un algorithme passait par les modifications de ses meta-paramètres, mais aussi pas des modifications du contexte, c'est à dire une observation du modèle de langue, et des particularités des textes étudiés.