Comparaison de plusieurs méthodologies permettant de constuire un filtre à spam

Pol Labarbarie et Mehdy Hounkonnou [1]

[1]Master 2 Modélisation Statistique et Stochastique, Université de Bordeaux

8 janvier 2023

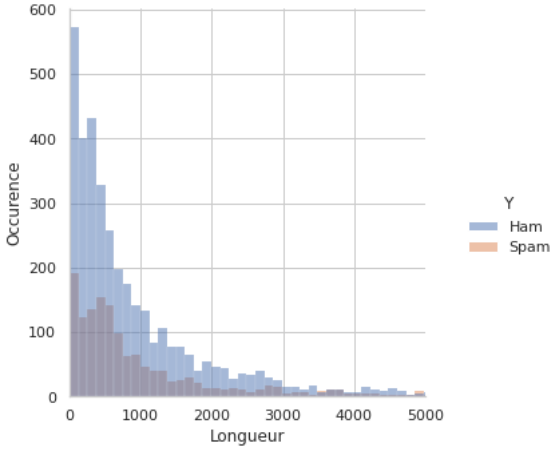
Table des matières

I. Introduction

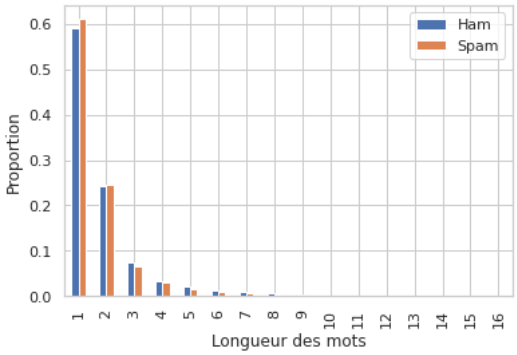
Le développement d'internet dans les années 60-70 a permis l'émergence de diverses applications comme par exemple la messagerie électronique. Bien que cette technologie a permis une augmentation majeure de la capacité de communication, certains utilisateurs, souvent des entreprises, se sont servis de ce moyen de communication instantané pour effectuer de la publicité massive. Ce type de communication électronique non sollicitée, souvent via courrier électronique se nomme spam ou courriel indésirable. Ces 15 dernières années, le taux moyen de courrier électronique étant un spam fluctuait entre 50 et 60 %. Ce fort taux n'est en réalité pas visible par les utilisateurs étant donné que les prestataires de services informatiques qui stockent les courriers électroniques effectuent au préalable un traitement de classification de courrier en spam ou non. Nous appellerons hams les courriers électroniques qui ne sont pas des spams. La problématique posée par les spams est de trouver une méthodologie, un algorithme qui permet non pas de classifier correctement les spams et les hams, mais qui permet d'éviter au maximum qu'un ham se retrouve dans la boite à spam. Dans ce rapport, nous allons proposer divers méthodes d'apprentissages statistiques permettant de filtrer au mieux les spams sans laisser passer dans les spams des messages qui n'en sont pas. La classe d'intérêt est donc la classe ham. Notre étude est basée sur un jeu de données issu de la plateforme Kaggle. Dans la première partie, nous effectuerons des statistiques descriptives afin de mieux comprendre les difficultés posées par les spams et nous décrirons deux normalisations que nous effectuerons sur notre jeu de données initial. Puis, dans la seconde partie, nous décrirons les méthodes que nous avons mis en place. Enfin, dans la dernière partie, nous présenterons les résultats obtenus en utilisant les différentes méthodes décrites au préalable.

II. Statistiques descriptives

Dans cette partie, nous allons effectuer des statistiques descriptives de notre jeu de données. Notre jeu de données est une matrice $X \in \mathbb{N}^{n \times p}$ où $n = 5172$ est le nombre de courriels à classer, $p = 3000$ est le nombre de mot unique potentiellement contenu dans notre courriel. La matrice X est une matrice de comptage, c'est à dire que pour tout $1 \leq i \leq n$, pour tout $1 \leq j \leq p$, X_{ij} correspond au nombre de fois que le j-ème mot apparaît dans le i-ème courriel. Tout d'abord, avant de présenter des résultats fins, exposons pourquoi une approche statistique basique ne permettrait pas de détecter si un courriel est un spam. Les figures ci-dessous présentent quelques statistiques descriptives de notre jeu de données.



(a) Histogramme de la longueur des courriels



(b) Diagramme en barre de la proportion de la longueur des mots utilisés dans les courriels

Figure 1 – Graphiques de statistiques descriptives en fonction de la classe du courriel

Au vue des graphiques 1a, 1b et **??**, la distribution de la longueur des courriels, la distribution de la proportion de la longueur des mots utilisés dans les courriels et la distribution du nombre de mots différents utilisés dans un courriel sont les mêmes selon si le courriel est un spam ou un ham. Il est donc clair que des approches triviales de statistiques descriptives ne vont pas permettre de

discriminer nos courriels. Néanmoins, au vue de la figure **??**, il apparaît que la fréquence d'utilisation d'un mot dans un courriel semble être déterminant pour détecter si le courriel est un spam ou non. Dans la prochaine partie, nous allons utiliser des méthodes d'apprentissage supervisé afin de construire notre filtre à spam.

III. Méthodes

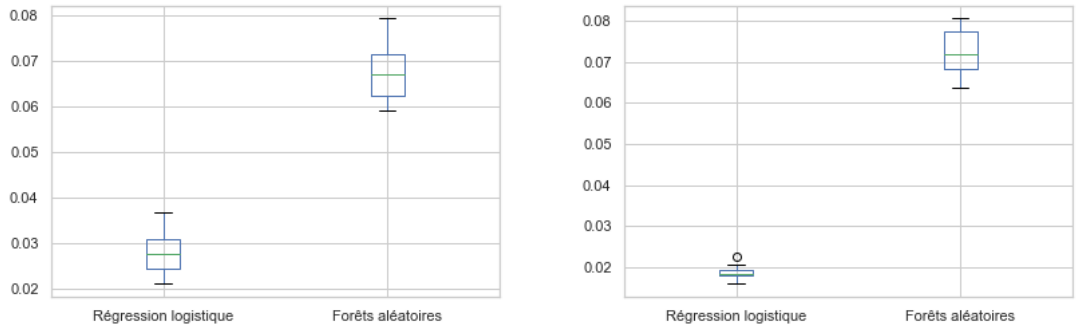
Dans cette partie, nous allons détailler la méthodologie que nous avons appliqué afin de construire notre filtre à spam optimal au sens donné dans l'introduction. Dans un premier temps, nous avons réalisé l'application brute de plusieurs algorithmes d'apprentissage supervisé. Nous n'avons retenu que deux algorithmes que nous avons jugé comme étant les plus pertinents pour cette étude. Il s'agit de la régression logistique avec une pénalité Ridge et les forêts aléatoires. Avant de comparer ces deux méthodes, nous avons défini et utilisé deux méthodes de "normalisation" différentes. La première consiste à normaliser pour chaque courriel la fréquence d'apparition d'un mot par le nombre total de mots du courriel. La seconde normalisation est une normalisation très utilisée en théorie de l'information, c'est la normalisation TF-IDF. La seconde normalisation contrairement à la première permet de tenir compte de l'occurrence d'un mot non pas que dans le courriel en question mais dans l'ensemble des courriels disponibles. Décrivons maintenant notre méthodologie. Ce sont les étapes suivantes :

1. Nous créons deux jeux de données correspondant aux deux normalisations décrites précédemment.
2. Nous confrontons la régression logistique avec pénalisation Ridge aux forêts aléatoires sur les différents jeux de données. Pour ces deux méthodes, nous effectuons B découpages apprentissage-test, puis nous calibrons les paramètres sur un échantillon de validation construit à partir du b-ème découpage en question. Pour la régression logistique Ridge, nous optimisons le paramètre de régularisation par validation croisée 5-folds. Pour les forêts aléatoires, nous optimisons le paramètre *mtry* en créant une grille de paramètre *mtry* et en effectuant un découpage apprentissage-validation. Nous retenons le modèle avec le taux de mauvaise classification médian le plus bas.
3. Nous sélectionnons la normalisation qui minimise l'erreur médiane en effectuant de nouveau B découpages apprentissage-test avec la méthode sélectionnée précédemment. Une fois le modèle et la normalisation fixés, nous introduirons une matrice de coût et des variations de seuil afin de maximiser le taux de vrais positifs et ainsi répondre à la problématique de l'introduction.
4. Enfin, nous créerons un second modèle, plus parcimonieux en réduisant le nombre de variables. D'une part, nous testerons la sélection de variables par une méthode ElasticNet et d'autre part grâce à une d'ACP non normée. Nous sélectionnerons le modèle qui minimise l'erreur de prédiction médiane sur de nouveau B découpages apprentissage-test. De nouveau, nous introduirons une matrice de coût et des variations de seuil afin de maximiser le taux de vrais positifs.

A la fin de cette méthodologie, nous obtiendrons donc deux modèles. Un modèle performant mais difficilement interprétable et un second modèle un peu moins performant mais plus parcimonieux. Ces deux modèles utiliseront le même prédicteur et la même normalisation.

IV. Résultats

Dans cette partie, nous allons présenter les différents résultats de l'application de la méthodologie que nous avons introduite précédemment. Tout d'abord, nous avons confronté la régression logistique avec pénalité Ridge aux forêts aléatoires en suivant la méthodologie précédente. Nous avons effectué $B = 20$ découpages apprentissage-test.
Nous constatons sur la figure 2 que les deux méthodes donnent des erreurs de prédiction convenables. Cependant, nous remarquons que pour les deux normalisations, la régression logistique avec pénalisation Ridge apporte de meilleurs résultats. La suite de notre étude se base donc sur la méthode de la régression logistique avec pénalisation Ridge. De plus, il semblerait que la normalisation avec



(a) DEM des erreurs de prédiction des données normalisés

(b) DEM des erreurs de prédiction des données TF-IDF

FIGURE 2 – Diagrammes en moustaches des erreurs de prédiction des méthodes sur les deux jeux de données pour $B = 20$ découpages

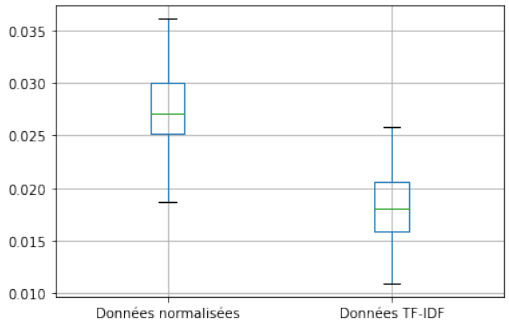


FIGURE 3 – DEM des erreurs de prédiction des données normalisés et des données TF-IDF pour $B = 20$ découpages

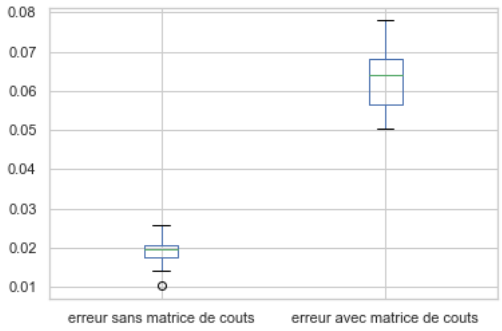
la méthode TF-IDF apporte également de meilleurs résultats. C'est ce que nous allons vérifier. Dans toute la suite de notre étude, le paramètre devant le terme de régularisation sera optimisé par validation croisée 5-folds.

A partir de la figure 3, nous remarquons que les données normalisées avec la méthode TF-IDF sont celles qui conduisent à des meilleurs résultats en moyenne. Nous adoptons donc cette normalisation pour la suite de notre étude. Nous introduisons une matrice de coût afin de mieux prédire notre classe d'intérêt, la classe ham. Nous avons choisi une matrice de coût tel qu'il est 10 fois plus risqué de prédire ham en spam que inversement. Les graphiques ci-dessous présentent les résultats de l'erreur de prédiction, le taux de vrais positifs et le taux de faux positifs pour $B = 20$ découpages apprentissage-test pour notre modèle avec et sans matrice de coût.

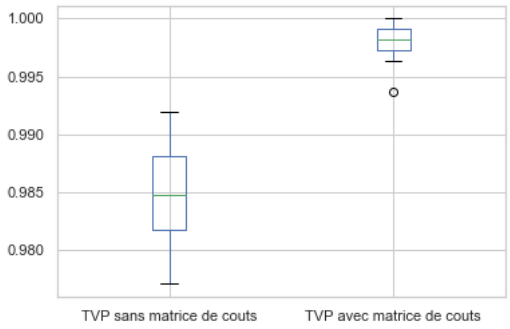
Au vu des graphiques de la figure 4, l'introduction d'une matrice de coût a eu l'effet escompté. En effet, nous avons augmenté le taux de vrais positifs c'est à dire nous avons diminué le nombre de ham qui tombait dans la boite à spam. Cependant, comme prévu, nous observons une dégradation du taux de bonne prédiction et une augmentation du taux de faux positifs. De façon générale, les résultats obtenus sont plus que satisfaisants, le taux d'erreur est très très bas et le taux de faux positifs est très proche de 1. Seulement quelques hams tombent encore dans la boite à spam.

Maintenant, afin de construire un modèle plus parcimonieux, nous avons de nouveau effectué $B = 20$ découpages apprentissage-test et nous avons sélectionné la méthode qui minimise le taux de mauvaise classfication médian. La coefficient de régularisation de la pénalité ElasticNet est choisi par validation croisée 5-folds et le choix du nombre de composantes principales par critère empirique. Le graphique ci-dessous présente ce résultat.

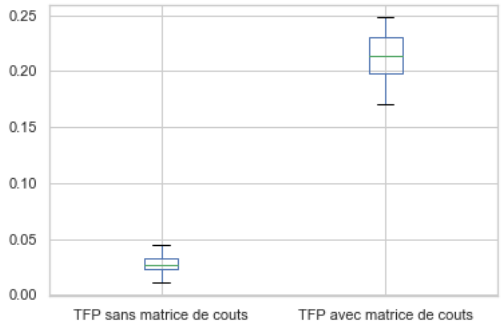
Sur le graphique 5, nous observons que la réduction de dimensionalité par ACP non normée est celle qui permet de minimiser le taux de mauvaise classification médian. Nous conservons donc



(a) DEM du taux d'erreur de prédiction



(b) DEM du taux de vrais positifs



(c) DEM du taux de faux positifs

FIGURE 4 – DEM de l'erreur de prédiction et des taux de vrais et faux positifs pour $B = 20$ découpages apprentissage-test de la méthode RL Ridge sur données TF-IDF

cette méthode. De même que pour le modèle plus complexe, le graphique ci-dessous présentent les résultats de l'erreur de prédiction, le taux de vrais positifs et le taux de faux positifs pour $B = 20$ découpages apprentissage-test pour notre modèle réduit par ACP avec et sans matrice de coût.

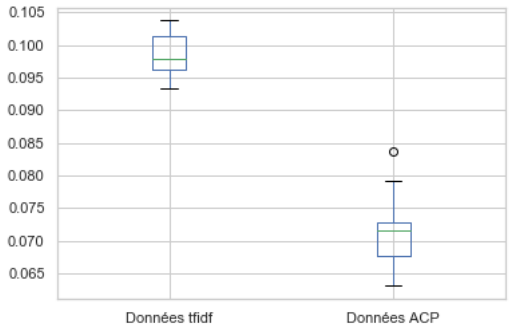
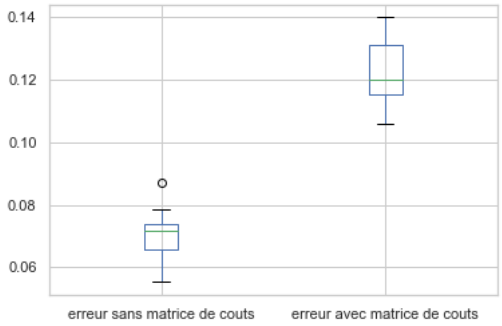
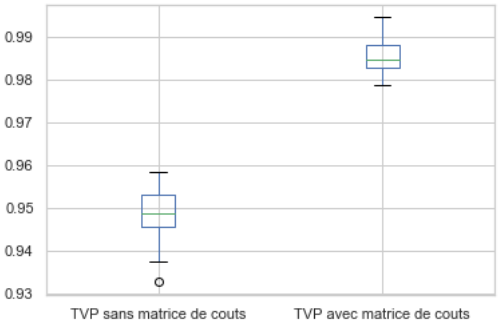


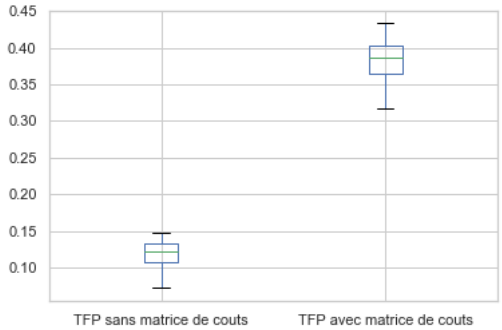
FIGURE 5 – DEM des erreurs de prédiction des données TF-IDF après réduction de dimensionalité



(a) DEM du taux d'erreur de prédiction



(b) DEM du taux de vrais positifs



(c) DEM du taux de faux positifs

FIGURE 6 – DEM de l'erreur de prédiction et des taux de vrais et faux positifs pour $B = 20$ découpages apprentissage-test de la méthode RL Ridge sur données TF-IDF réduites par ACP

De même que pour le modèle complet, au vu des graphiques de la figure 6, l'introduction d'une matrice de coût a eu l'effet escompté. Contrairement au modèle complet, l'introduction de la matrice de coût dégrade d'avantage le taux de bonne classification. Cependant, le taux de vrais positifs quant à lui atteint presque 1.

V. Conclusion

Nous avons vu à travers ce rapport comment il était difficile de construire un filtre à spam de façon triviale. En utilisant une normalisation TF-IDF et une régression logistique Ridge nous avons obtenu des résultats d'une très grande précision. L'introduction d'une matrice de coût permet de classer correctement presque la totalité des hams. Seulement quelques hams tombent encore dans la boite à spam après l'introduction de cette matrice de coût. L'introduction de cette matrice de coût dégrade le taux de bonne classification mais les résultats restent excellents. A partir d'une ACP non normée, nous avons construit un modèle beaucoup plus parcimonieux avec 30 fois moins de variables que le modèle initial. Malgré cela, ce modèle conserve de très bons résultats. Quelques hams supplémentaires tombent dans la boite à spam mais cela reste tout à fait convenable.