Sentimental analysis

Romain Lesauvage

Nous souhaitons désormais utiliser les *word-embeddings* obtenus afin d'implémenter un modèle de *sentimental analysis*. Nous disposons pour cela d'une nouvelle base de données, constituée de tweets anglais traduits en français et annotés, avec 0 pour les tweets « négatifs »et 4 pour les tweets « positifs ». Elle est découpée en deux fichiers : un d'entraînement avec plus d'1,5 millions de tweets et une de test avec 4000 de tweets. Il a été retenu de garder 50000 tweets du fichier d'entraînement pour entraîner nos modèles, en faisant bien attention à garder un fichier équilibré, c'est-à-dire avec autant de tweets positifs que négatifs. Nous avons également transformé la valeur 0 en -1 et la valeur 4 en 1 pour les sentiments.

1 Description du modèle

Afin d'exploiter nos vecteurs, l'idée est de réaliser une régression logistique sur la base d'entraînement, en considérant comme prédicteurs chacune des dimensions des vecteurs. Pour réaliser cela, il faut toutefois pouvoir attribuer à chaque tweet un vecteur, réaliser une forme de *sentence embedding*. Dans une première approche, nous avons décidé d'attribuer à chaque tweet la moyenne des vecteurs de l'ensemble des mots. Nous pouvons ensuite réaliser la régression sur ce vecteur.

Si on note Y_i le sentiment du tweet (noté *polarity* dans la base), et $X_{i,1}, \dots, X_{i,n}$ chacune des coordonnées du vecteur de la phrase, alors nous allons chercher à estimer :

$$\log\left(\frac{\mathbb{P}(Y_i = 1)}{1 - \mathbb{P}(Y_i = 1)}\right) = \beta_0 + \sum_{i=0}^{n} \beta_i X_{i,j} \tag{1}$$

Cependant, une fois ce modèle estimé, il faudra pouvoir estimer son efficacité. Pour cela, on décide donc de s'intéresser à deux modèles plus simples, ne faisant pas appel aux *word-embeddings*, qui nous serviront de référence.

2 Modèles de référence

2.1 Modèle aléatoire

Le premier modèle de référence auquel on peut penser est le modèle aléatoire, c'est-à-dire celui qui attribue à chaque tweet le sentiment 1 ou -1 avec la probabilité $\frac{1}{2}$. On obtient avec ce modèle une précision de **50,15 %**.

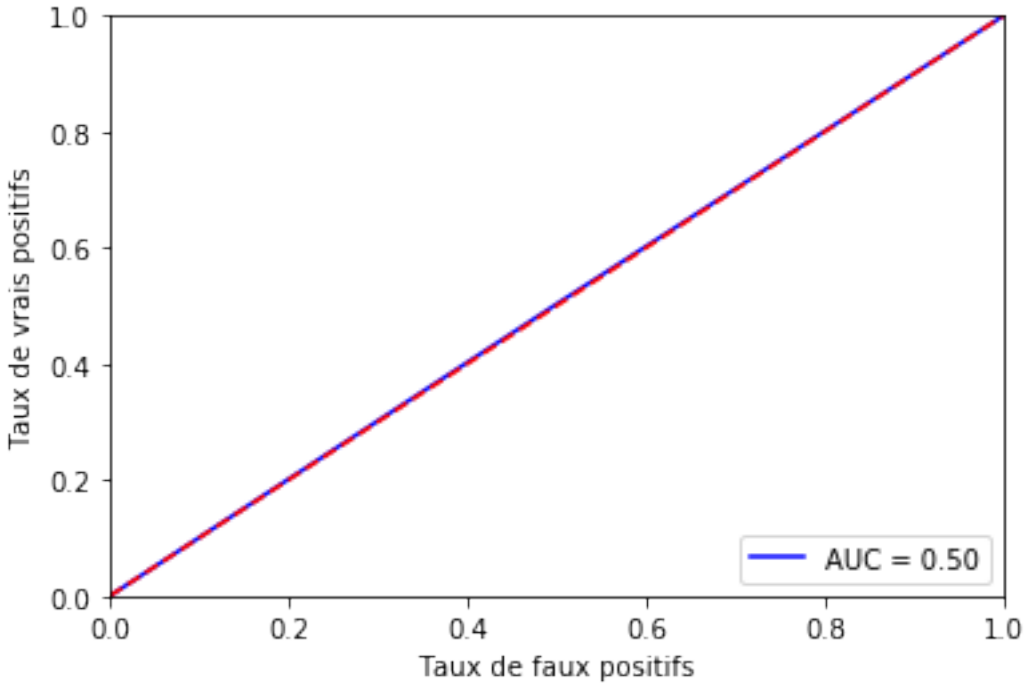

FIGURE 1 – Courbe ROC pour modèle aléatoire

2.2 Modèle avec sentiment moyen des mots

Le modèle aléatoire n'est évidemment pas satisfaisant. Il peut cependant être amélioré car il ne prend pas en compte les informations disponibles dans les données, à savoir les tweets annotés. Nous pouvons donc penser à un autre modèle qui utilisera cette information. Pour cela, on raisonne en trois temps :

- On commence par calculer un sentiment moyen par mot, en prenant la moyenne des sentiments des tweets dans lesquels le mot intervient ;
- On calcule ensuite pour chaque tweet le sentiment moyen des mots de la phrase (s);
- On définit ensuite une réglère de décision du type $\mathbb{I}(s \geq \alpha)$ avec un paramètre α à optimiser.

On implémente donc cela et on cherche à optimiser la valeur de α. En représentant la précision sur les bases de *train* et *test* pour toutes les valeurs de α entre -1 et 1 (avec un pas de 0,1), on trouve que $\alpha_{opti} = -0.05$.

Avec cette valeur de α, on obtient alors une précision de **87,33 %** sur le *train* et **72,90 %** sur le *test*.

A ce stade, nous avons donc notre seuil pour évaluer notre modèle. En effet, nous aimerions idéalement que notre modèle basé sur les *word-embeddings* fasse mieux que le modèle avec sentiment moyen des mots, donc **nous aimerions une précision supérieure à 72,9 % sur le test**.

2.3 Modèle basé sur word-embeddings

On utilise maintenant les vecteurs obtenus grâce au modèle *Word2Vec*. Comme indiqué, on effectue la régression logistique du sentiment sur le vecteur moyen de la phrase. Puisque nous avons des résultats pour différentes dimensions (50, 100 et 300), nous allons tester les 3. Par ailleurs, nous avons à disposition 6 seeds par dimension, nous utiliserons donc les vecteurs moyens de ces 6 seeds.

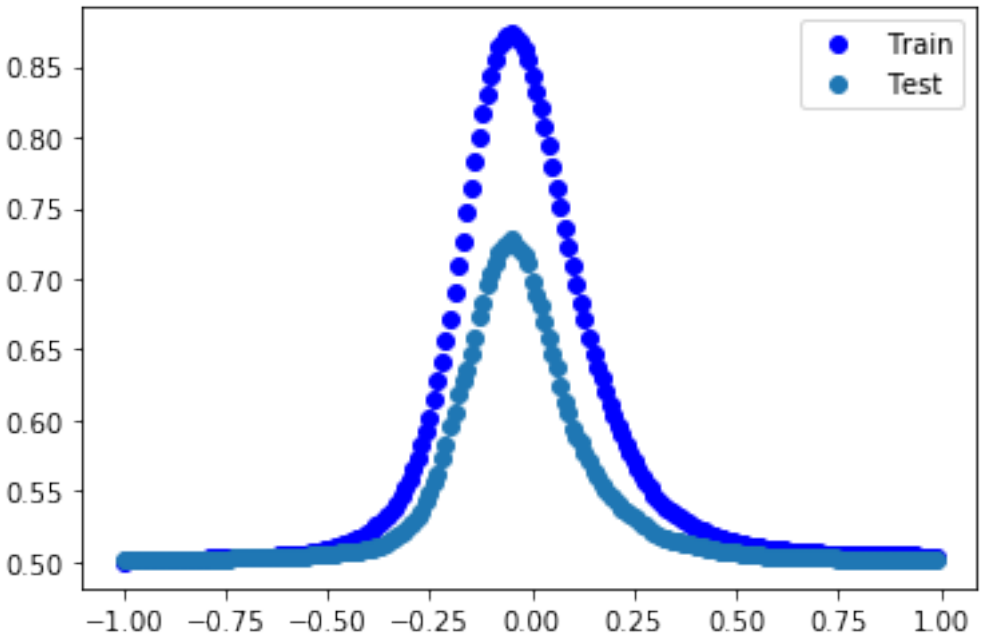

FIGURE 2 – Choix du meilleur alpha

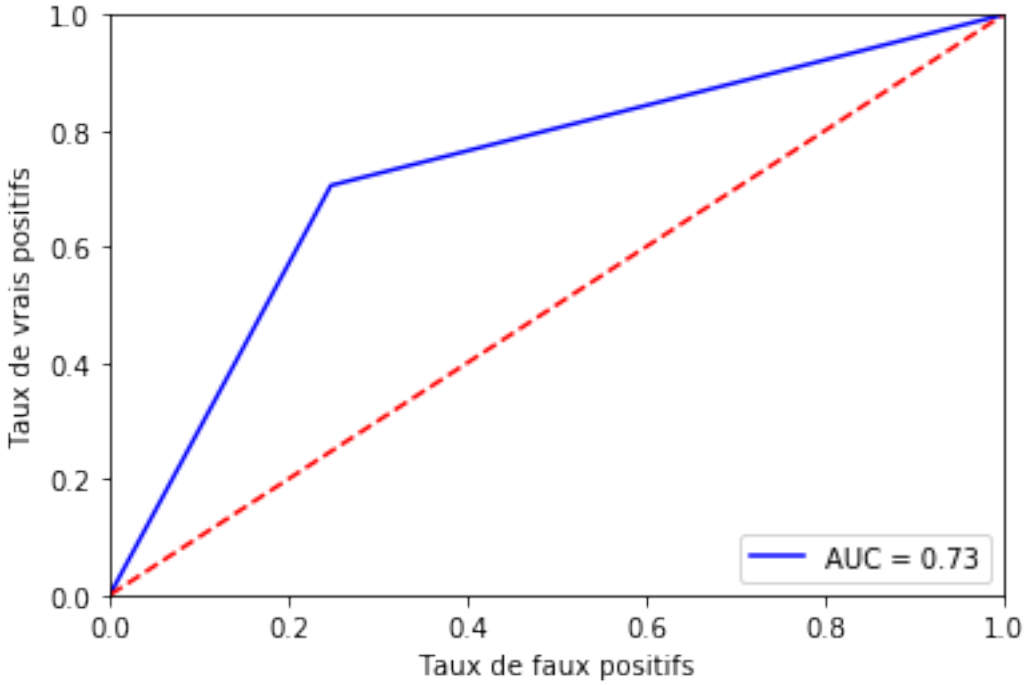

FIGURE 3 – Courbe ROC pour modèle sentiment moyen des mots

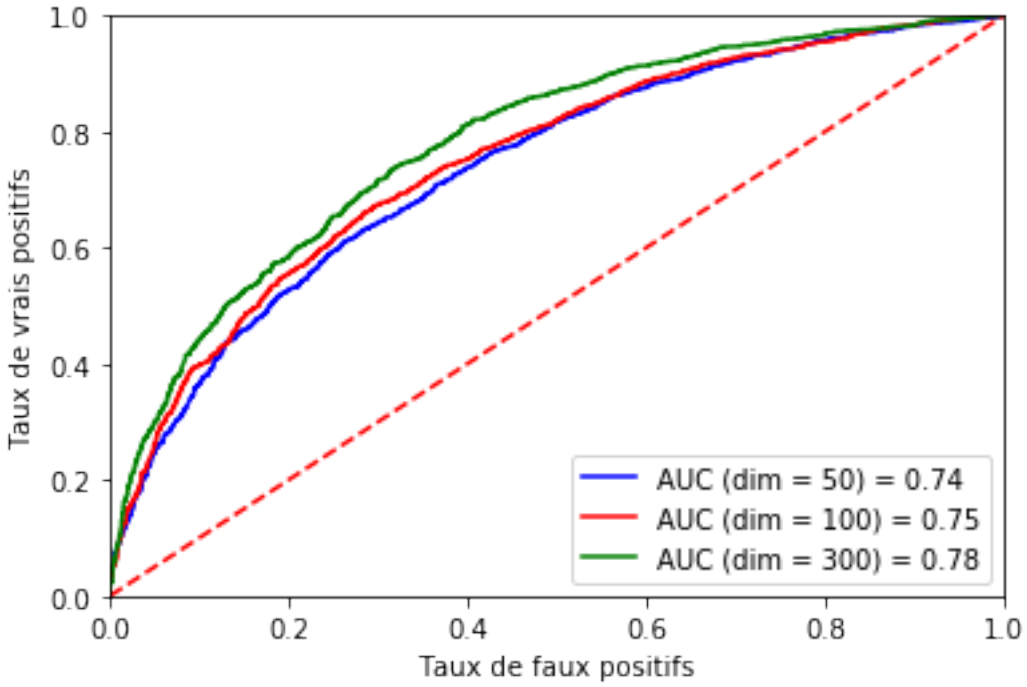

FIGURE 4 – Courbe ROC pour modèle sur *word-embedding*

On obtient alors les résultats suivants :

Dimension	Précision - Train (%))	Précision - Test (%))
50	67,98	67,05
100	69,19	68,71
300	71,64	70,39

On remarque donc, comme attendu, que le résultat est moins bien sur la base test que celle d'entraînement, mais de très peu. Plus la dimension augmente, meilleurs sont les résultats. Toutefois, même en dimension 300, on se rapproche des résultats du modèle sur le sentiment moyen des mots mais reste légèrement moins bien.

Afin d'améliorer les résultats, nous tentons d'enlever les *stop words* qui, a priori, n'apportent pas d'informations sur le sentiment d'un tweet. On obtient les résultats suivants :

Dimension	Précision - Train (%))	Précision - Test (%))
50	66,42	65,85
100	67,64	66,88
300	70,08	69,54

Les résultats ne sont pas meilleurs, ils sont même légèrement moins bon, donc cette approche n'est pas validée. Nous tentons alors de voir désormais si la normalisation des vecteurs peut avoir un impact. Nous réalisons les mêmes expériences :

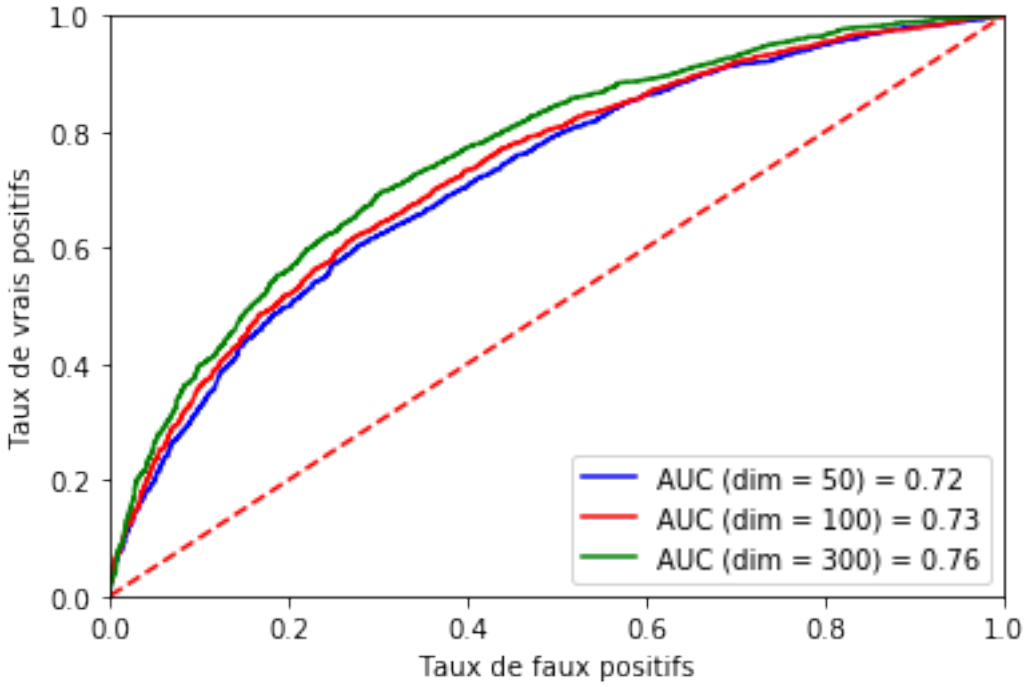

FIGURE 5 – Courbe ROC pour modèle sur *word-embedding* en enlevant les *stop words*

Dimension	Précision - Train (%))	Précision - Test (%))
50	68,32	67,23
100	69,15	69,09
300	71,49	70,49

Encore une fois, on obtient des résultats similaires, bien que très légèrement meilleurs au modèle non normalisé.
Afin d'améliorer les résultats, une dernière idée pourrait être d'utiliser dans la régression logistique les sentiments moyens calculés pour chaque mot, en pondérant chaque mot dans une phrase par son sentiment dans le calcul de l'*embedding* moyen de la phrase. On obtient alors les résultats qui suivent :

Dimension	Précision - Train (%))	Précision - Test (%))
50	78,83	68,96
100	79,72	69,01
300	81,42	71,02

Les résultats s'améliorent un peu, mais on est encore légèrement en-dessous du modèle de référence.

3 Conclusion

En conclusion, quelque soit le modèle logistique utilisé, les résultats sont très proches et légèrement moins bien que le modèle de référence. Le meilleur résultat est toutefois obtenu pour le modèle pondéré par les sentiments des mots et pour une dimension de 300. On peut donc décider d'utiliser cette dimension pour la suite. Même si notre modèle ne semble pas améliorer les performances par

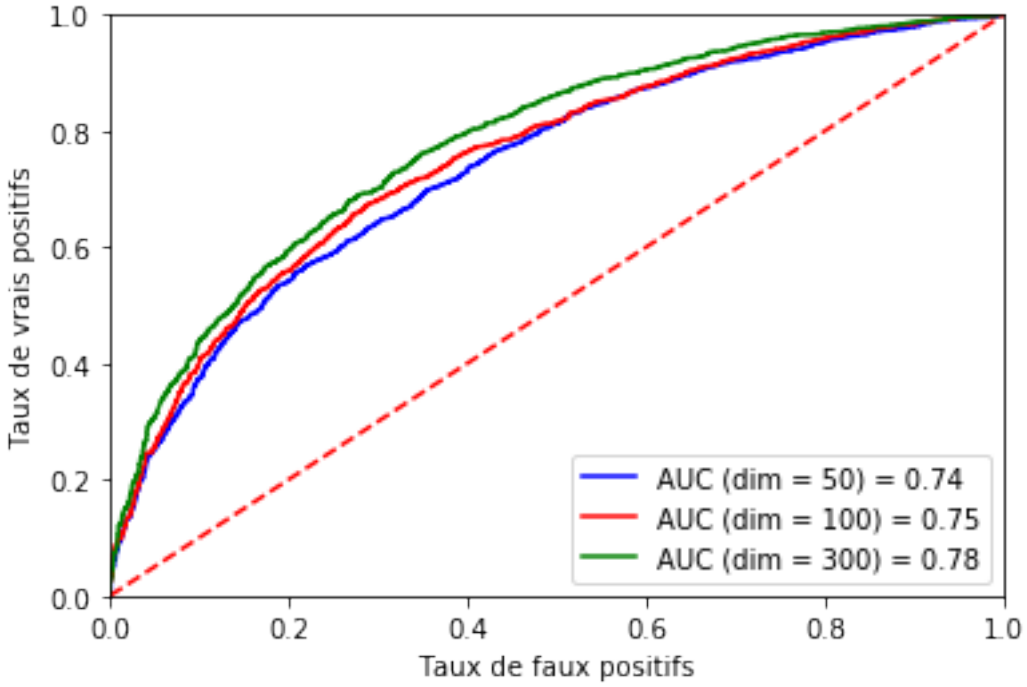

FIGURE 6 – Courbe ROC pour modèle sur *word-embedding* en normalisant les vecteurs

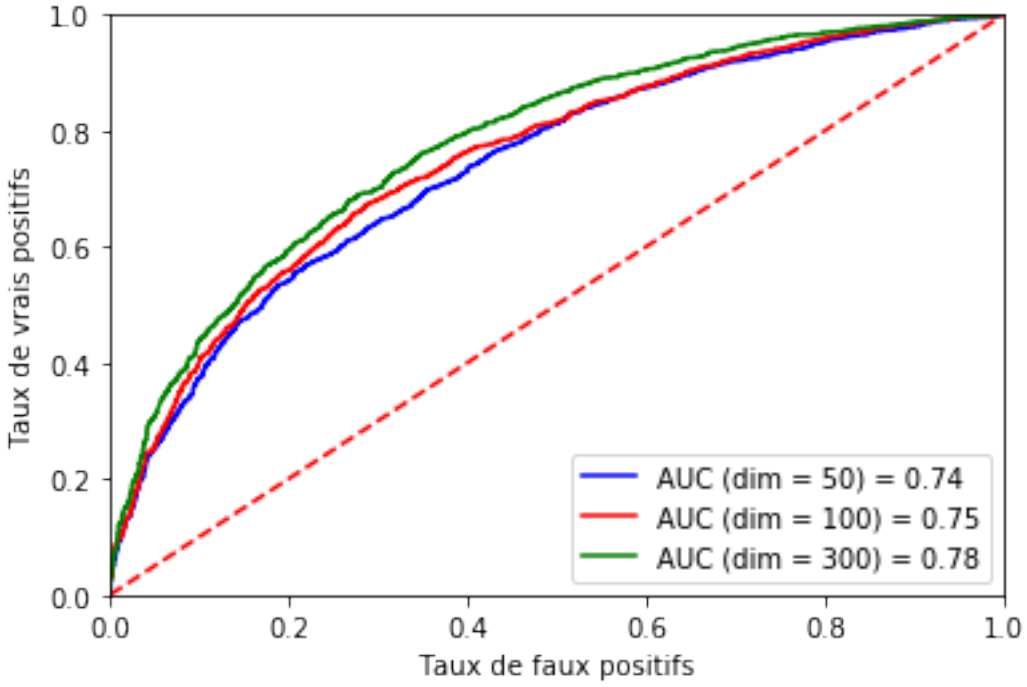

FIGURE 7 – Courbe ROC pour modèle sur *word-embedding* en pondérant par sentiment des mots

rapport au modèle sans *word-embedding*, il faudrait pouvoir le tester sur une autre base annotée pour mesurer cet écart, car dans le corpus actuel environ un mot sur deux n'a pas été entraîné par *Word2Vec*. Afin d'améliorer également les performances, il faut améliorer le préprocessing de la base.