CLUSTERINGS DE TWEETS D'ACTUALITÉ

A PREPRINT

William Harvey
Matricule 1851388
william.harvey@polymtl.ca

Jérémie Miglierina
Matricule 1856272
jeremie.miglierina@polymtl.ca

Claudia Onorato
Matricule 1845448
claudia.onorato@polymtl.ca

November 12, 2019

1 Introduction

Le présent document vise à décrire une méthode de regroupement de tweets portant sur des sujets d'actualité divers. L'objectif est de regrouper les tweets portant sur la même actualité au sein de catégories communes. Ce genre de classification non-supervisé est importante, car elle permet de faire une analyse automatique de l'activité sur Twitter. Ainsi il est possible de connaître de façon automatique les sujets qui sont les plus populaires et même d'étudier la l'importance d'une actualité à travers le temps.

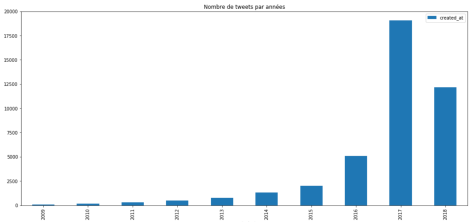
Dans le but de parvenir à ce genre de *clustering*, nous avons appliqué deux méthodes différentes dont les résultats sont drastiquement différents. La première méthode de classification est basée sur une variante de l'algorithme d'apprentissage non-supervisé K-Mean. Celle-ci permet de créer un nombre fixe de *clusters* et classe l'ensemble de nos tweets en leur sein. La définition du *cluster* se raffine au fur et à mesure de l'exécution de l'algorithme. La deuxième méthode qui a été utilisé, afin de créer nos regroupements de sujet d'actualité est l'algorithme de regroupement automatique DBScan. Celle-ci permet de créer une quantité de groupe qui est inconnue d'avance tant que des tweets sont suffisément proche d'un *cluster*. Finalement, l'intérêt et la popularité de chaque catégorie d'actualité a été étudié dans le temps. Nous avons aussi créé des nuages de mots afin de mieux visualiser le contenu de chaque catégorie créée. Ceux-ci reflète bien le sujet qu'essaie de représenter chaque catégorie.

2 Présentation du dataset

Comme il été demandé dans l'énoncé de laboratoire, nous avons utilisé le dataset "News Outlet Tweet Ids" [1]. Celui-ci est constitué d'une base de données de plus de 39 millions de tweets provenant de divers comptes Twitter associé à des entreprises médiatiques. Cette base de données ne fait qu'offrir les *tweets IDs* qui sont des identifiants uniques vers des tweets conformément à la politique de Twitter au sujet du partage de tweets. Il a donc été nécessaire *d'hydrater* notre base de données à l'aide de multiples requêtes vers l'API de Twitter. Ces appels ont pu être effectué à l'aide de la librairie libre de droit Twython.

Dans le but d'obtenir des résultats plus facilement et plus rapidement, nous avons décider d'utiliser seulement un petit sous-ensemble de la base de données de tweets, soit 41 412 tweets sélectionnés selon une méthode d'échantillonage par réservoir [3]. Ceci correspond à 0,104% des tweets disponibles dans la base de données. De ces tweets, nous avons extrait les features de «text», «lang» et «created_at». Le feature «lang» a surtout permis de s'assurer que la langue du tweet est l'anglais, «created_at» a permis l'analyse de nos tweets dans le temps et le feature «text» nous a permis d'extraire les mots du tweet, afin de le preprocess et d'en faire l'analyse.

Figure 1: Nombre de tweets en fonction de l'année de publication des tweets.



Cette figure montre la répartition des tweets de notre échantillon en fonction des années. Les principaux constats sont que la très grande majorité des tweets proviennent des 3 dernières années. Ainsi, cela montre que de 41 412 tweets sont plutôt répartis dans le temps. Voilà pourquoi nous pensons que nos *clusters* vont mieux illustrer la catégorie d'actualité que le sujet d'actualité exact. Il n'y a pas assez de tweets condensés dans le temps pour créer des *clusters* qui traitent d'un seul et même sujet préçis.

3 Preprocessing

Les étapes de preprocessing servent à nettoyer les messages extraits de l'API de Twitter, puis à les transformer en *features* pouvant être utilisés par nos algorithmes d'apprentissage. Notre pipeline de preprocessing est constitué des étapes suivantes:

- un preprocessing agit sur le contenu des tweets originaux;
- une étape de *tokenization* et de *lemmatization* transforme les chaîne de caractère en un vecteur où chaque élément est une représentation du nombre d'occurence d'un mot;
- TF-IDF est appliqué sur chaque élément de notre vecteur.

À noter que les deux dernières étapes se font par le biais de la classe TfidfVectorizer.

3.1 Preprocessing appliqué sur les tweets

Premièrement, il est important de filtrer les messages, afin de conserver seulement les mots qui peuvent définir la catégorie d'actualité. La majorité de ces filtres consistent à appliquer une expression régulière sur chaque message. Tout d'abord, les messages qui ont été partagés sont nommés «retweets», et possèdent toujours le préfixe «RT» qu'il faut filtrer. Nous filtrons également tous mots qui débutent par "http" ou "https" et toutes mentions à d'autre utilisateur. D'autre part, nous avons conserver les mentions, mais enlevé le caractère «#» qui les préfixaient. Par la suite, nous supprimons tous les espaces blancs (soit des tabulations, des nouvelles lignes, etc.) et nous insérons un espace afin de n'avoir que des phrases. Nous nous assurons que les caractères qui ne sont pas des espaces appartiennent aux lettres de l'alphabet. Les symboles de ponctualités sont donc également filtrés.

Un désavantage de cette méthode est que les contractions (par exemple, "don't" ou "it'll") seront plus difficiles à interpréter. Comme nous voulons faire de l'analyse de texte, nous n'avons pas conservé de chiffres, ceux-ci étant généralement trop spécifique au sujet d'actualité et aux statistique qui s'y rattache par exemple.

3.2 *Tokenization* et *lemmatization*

La deuxième étape de notre pipeline de preprocessing, consiste à convertir chaque message en jeton, en suivant la méthode *bag of words*. Pour appliquer nos algorithmes de clustering, il faut que chaque tweet s'exprime sous la forme d'un vecteur, où chaque élément correspond au nombre d'occurrences d'un mot. La *lemmatization* consiste à trouver la racine de chacun de ces mots. Ainsi, plusieurs mots de même famille seront représentés par un jeton commun.

3.3 Méthode TF-IDF

La dernière étape consiste à appliquer la méthode TF-IDF. Celle-ci permet de remplacer la fréquence de chaque mot dans un tweet par un indicateur d'importance du mot. Cet indicateur correspond à la fréquence d'un mot dans un tweet donné (TF) multiplié par le logarithme du nombre de tweet sur le nombre de tweet qui contiennent un mot donné (IDF). On accorde donc plus d'importance à un jeton généralement peu commun, mais qui revient souvent dans un tweet donné.

3.4 Date de création

La date de création des tweets était sous la forme de String de type "Mon apr 09 23:30:00 +0000 2018". Ce string a été facilement transformé en objet de type DateTime qui se manipule mieux pour avec les librairies qui gèrent les graphiques. Le nouveau format DateTime permet d'accéder a chaque parti de la date avec des attribut différent. Pour simplifier et grouper des dates lors de la création de graphique nous avons pris la décision de simplifier la Date et de garder seulement l'année et le mois.

4 Méthodologie

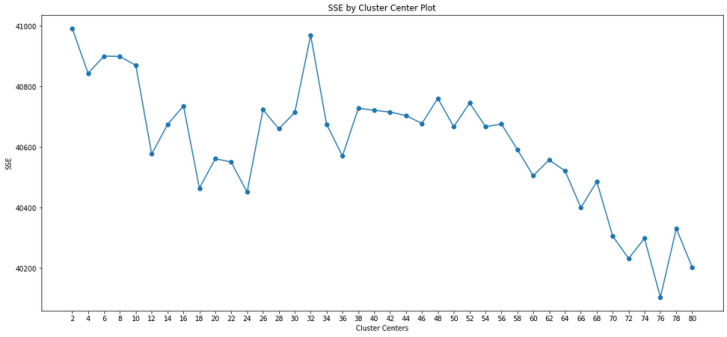
Nous avons décidé d'appliquer deux algorithmes de *clusterings*, l'un d'eux, K-Mean, est basé sur l'étude des centres et l'autre, DBScan, est basé sur la répartition et la densité des tweets. Toutes les implantations des algorithmes de *clusterings* utilisés proviennent de la librairie scikit-learn.

4.1 Clustering par K-Means

Nous avons choisis d'utiliser l'algorithme K-Means. Les mots utilisés devraient appartenir à un même sous-dictionnnaire qui se répètent de tweets en tweets. Nous pouvons également espérer avoir des clusters uniformes, comme l'algorithme tente de réduire la variance parmi les regroupements. Cet algorithme est très intéressant, car il oblige l'entièreté de nos tweets à faire partie d'un certain nombre de catégorie qui peut être paramétré. Ainsi, nous pensons que ce genre de méthode permet d'obtenir des cluster dont la définition est plutôt large, ce qui concorde avec notre objectif, car nous souhaitons connaître les catégories d'actualités qui transcende le temps (politique, sport, économie, etc.).

Le premier paramètre à définir pour l'algorithme K-means est le nombre de clusters que nous voulons former. Nous avons d'abord posé un nombre arbitraire de groupes afin de pouvoir rapidement obtenir des résultats. Il est également à noter que nous avons utilisé une variante de l'algorithme K-Mean, soit MiniBatchKMean, car celui-ci est plus performant, et les résultats sont très similaires. La méthode d'initialisation pour les centres est faites de manière à ce qu'on minimise le temps avant que l'algorithme converge.

Figure 2: Somme des erreurs au carré selon le nombre de centres utilisé par K-Mean



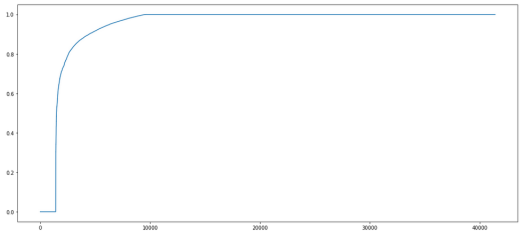
Nous avons amélioré nos résultats en choisissant le nombre de clusters qui minimisent les erreurs au carré au sein d'un même cluster, ce qui est communément appelé la méthode du coude. En observant la figure 2, nous voyons que le choix optimal, pour un nombre raisonnable de regroupements, est de 78 *clusters*. Il est également possible d'ignorer les clusters qui contiennent un nombre de tweets inférieurs à un certain nombre, ce qui facilite l'analyse des résultats.

4.2 Clustering par DBSCAN

DBScan est un algorithme très intéressant pour ce genre de situation, puisqu'il regroupe les tweets dont la distance au regroupement est inférieur à un seuil donné. Nous pensons que ce type d'algorithme va permettre de créer des catégories qui sont très cohérentes, puisque le seuillage oblige le contenu des tweets qui les composent à être très similaire. Un des problèmes qui peut être lié à cette technique est l'obtention de catégorie trop précise. Dans ce cas, nous passerions à côté de notre objectif et nous obtiendrons beaucoup de tweets de *bruit*, c'est-à-dire qui appartiennent à aucune catégorie.

Les hyperparamètres à fixer pour cet algorithme sont la distance maximale entre deux échantillons pour que l'on considère qu'elles sont voisins et le nombre d'échantillons minimal pour former un regroupement. La figure suivante montre la distance entre chaque tweet et son plus proche voisin:

Figure 3: Distance entre chaque tweet et son plus proche voisin



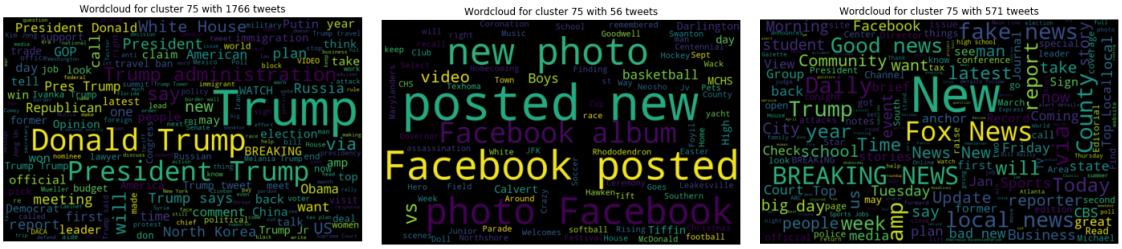
La figure précédente montre qu'une très grande majorité de tweets possède une distance euclidienne qui est supérieur à $\epsilon = 1.0$. Nous pensons qu'il s'agit d'un problème, car selon nos tests empirique, un paramètre ϵ significativement plus grand ($\epsilon = 1.5$) tend à créer un seul et unique cluster. Après plusieurs tests, nous avons trouvé que $\epsilon = 0.99$ était un des meilleurs paramètres, puisqu'il nous permettait d'obtenir des nuages de mots qui nous semblait plus pertinents qu'avec des ϵ plus petits ou plus grands.

5 Résultats

5.1 Clustering par K-Means

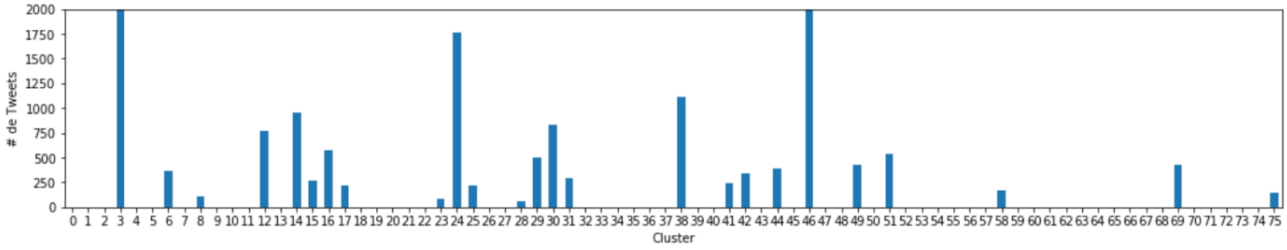
Les résultats obtenus par le biais de l'algorithme K-Means sont généralement concluants. En effet, en filtrant les clusters afin de n'avoir que ceux qui regroupent au moins 15 tweets, nous en conservons finalement que 25. Voici le contenu principales de quelques-uns de ces principaux clusters (la taille des mots est proportionnelle à leur nombre d'occurences):

Figure 4: Exemples de nuage de mots obtenus à l'aide de K-Mean.



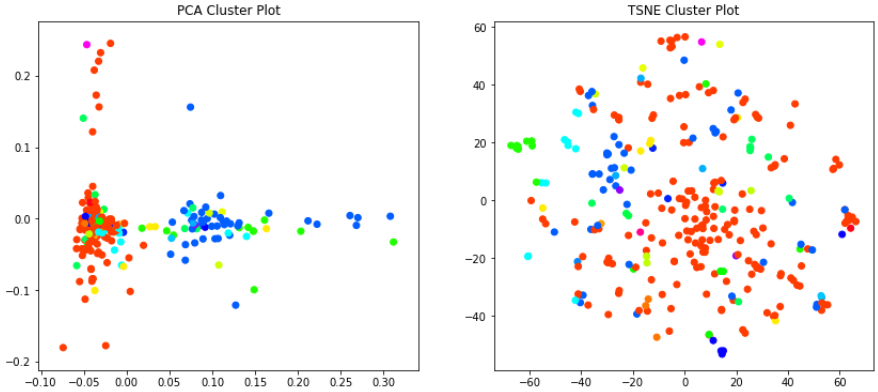
Nous y retrouvons des sujets d'actualités liés au président américain et aux actualités internationales (China, North Korea). Un autre cluster regroupe les tweets traitant de sur la criminalité et la police, par exemple (voir le calepin). D'un autre côté, un cluster regroupent des tweets ayant un contenu très similaire et ne semble pas traité d'un sujet d'actualité en particulier (Voir le deuxième nuage de points ci-dessous). Cela peut être dû au fait que certains tweets d'actualité sont liés à des tweets générés par Facebook. Finalement, les nouvelles de dernière heures (avec les mots "Breaking news" ou "Report") ont été regroupées ensemble, alors qu'on aurait plutôt voulu conserver le sujet d'actualité qui y était relié. Au niveau du preprocessing, nous aurions pu filtrer ces mots, car ils n'apportent pas d'information discriminante par rapport au sujet auquel se rapporte le tweet. En somme, alors que certains clusters porte sur des sujets d'actualité très précis, tel que le président américain, d'autres sont liés à des mots clés qui n'ont pas de liens avec des sujets d'actualité en particulier.

Figure 5: Nombre de tweets par cluster.



L'histogramme présenté à la figure 5 limite le nombre de tweets à 2000. La majorité des tweets ont été regroupés dans un certain nombre de groupes. On peut aussi voir que plusieurs tweets sont seul ou presque dans leur catégorie. Cela est d au fait que ces tweets parle de sujets très précis et nous pouvons considérer ces tweets comme des données aberrantes que nous n'avons pas filtré.

Figure 6: Analyse en composante principal à gauche et algorithme TSNE à droite appliquée sur les données de tweets. Les différents clusters sont affichés en différentes couleurs.

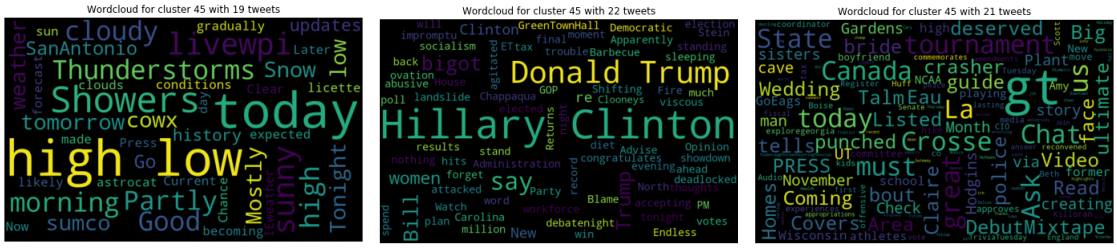


Finalement, nous avons affiché de deux manières différentes les clusters. D'abord, nous avons appliqué la décomposition en composantes principales et nous avons affiché seulement les deux composantes les plus significatives. Cela nous a permis de voir que la variance parmi les catégories comptant un grand nombre de tweet est assez grande. D'une autre façon, la visualisation par t-sne semble montrer que les points qui composent chacun de nos catégories sont concentrés un peu plus autour du centre de masse, mais qu'il existe beaucoup de points plutôt éloignés. Cela résulte en un graphique de catégories dont les points les plus éloignés d'un cluster se superposent au centre de masse d'un autre cluster.

5.2 Clustering par DBSCAN

Les résultats obtenus par dbscan se distinguent de ceux issus de la méthode K-Mean. Nous nous retrouvons avec des clusters comportant un plus petit nombre de tweets, mais dont le sujet d'actualité est plus évident et plus précis. Voici quelques exemples de ces résultats:

Figure 7: Exemples de nuage de mots obtenus à l'aide de DBScan.



À la figure 7, nous voyons clairement un regroupement lié aux annonces météo et un lié à la rivalité entre Clinton et Trump. Par contre, il existe tout de même des clusters dont l'interprétation est plus ardue, comme le dernier nuage de mot le montre. Probablement que celui-ci comporte des sujets d'actualités biens précis, mais différents, tout en ayant quelques mots en commun.

Figure 8: Nombre de tweets par cluster.

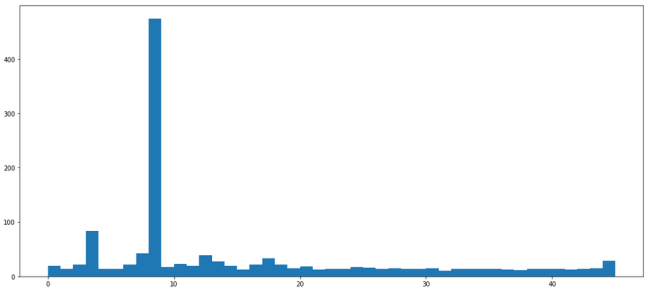
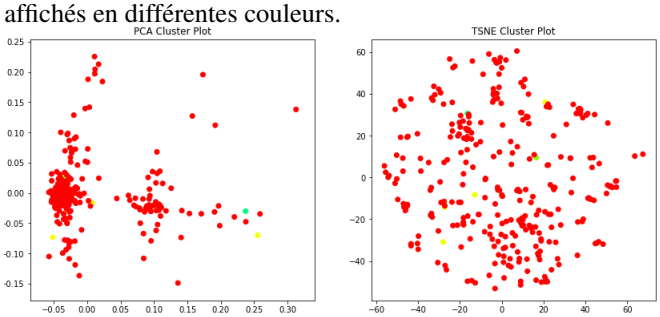


Figure 9: Analyse en composante principal à gauche et algorithme TSNE à droite appliquée sur les données de tweets. Les différents clusters sont affichés en différentes couleurs.



Nous pouvons confirmer à la figure 8 que la majorité des catégories regroupent peu de tweets. Ceci est dû à notre définition de epsilon, qui aurait pu être plus élevée afin d'inclure plus de données classées présentement comme du bruit. La catégorie la plus peuplée et celle qui est anormalement élevée est celle qui regroupe principalement les mots liés au président américain. Nous pouvons d'ailleurs voir que la majorité des données sont classées comme du bruit (soit en rouge) à la figure 9 (et on ne voit d'ailleurs pas grand chose d'autre...).

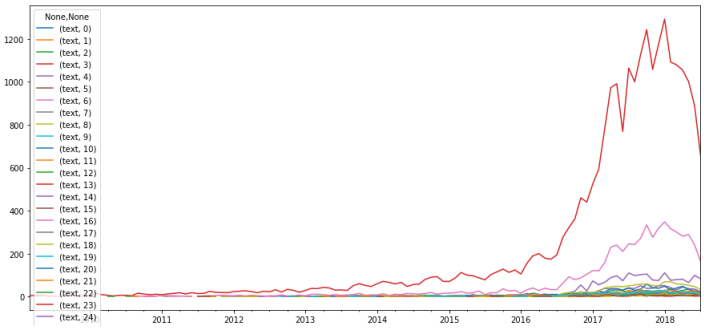
6 Discussion

6.1 Avantages et inconvénient de notre approche

6.2 Futures idées d'explorations et améliorations

Si ce projet avait été de plus grande envergure nous aurions pu tester plusieurs techniques de preprocessing pour les comparer et voir quelle serait la meilleure. De plus, nous aurions pu prendre le hardware de Google pour traiter une plus grande partie des tweets de la base de données comme cela aurait pu aider à la classification. Une autre méthode telle que la méthode de mélange de Gaussienne aurait pu être utiliser pour classifier les tweets, celle-ci aurait pu être comparé au 2 autres techniques pour s'assurer d'utiliser celle avec les plus grande performance. Finalement, nous aurions pu prendre plus d'attributs des tweets lors de notre analyse, car l'API de Twitter nous permet de recueillir beaucoup d'attributs et nous en avons utilisés peu.

Figure 10: Nombre de tweets par cluster en fonction de l'année pour la méthode K-Mean.



La figure 10 ci-haut présente le nombre de tweet par cluster au fil du temps. On peut voir que les premiers tweet sont en 2009 et les dernier en 2008. Selon le database cela est plus que normal. Ce qui est étonnant, est que la plupart des tweets ont été produit entre 2016 et 2018. Tout les clusters suivent cette distribution au fil du temps. Alors nous pouvons dire que les sujets dominant dans les cluster ne sont pas influencer par le temps, car sinon un cluster pourrait avoir un très grand nombre de tweet dans un autre période de temps précédant la période de 2016. la distribution des tweets pour chaque cluster est environ la meme pour tout les clusters cependant seulement leur amplitude (nombre de tweet pour la période) va différer. Bref, aucun cluster n'a révélé une nouvelle ou une catégorie de nouvelle qui était relié au temps mais le faible nombre de tweets dans les période hors 2016-2018 ne nous mets pas dans une bonne position pour trouver ce genre de regroupements.

References

[1] Littman, Justin and Wrubel, Laura and Kerchner, Daniel and Bromberg Gaber, Yonah. News Outlet Tweet Ids. 2017

[2] John B. Clustering documents with TFIDF and KMeans., 2018. `https://www.kaggle.com/jbencina/clustering-documents-with-tfidf-and-kmeans#Overview`

[3] Massoudn Seifi Random Sampling From Very Large Files. 2014 `http://metadatascience.com/2014/02/27/random-sampling-from-very-large-files/`

[4] Scikit-learn Comparison of the K-Means and MiniBatchKMeans clustering algorithms `https://scikit-learn.org/stable/auto_examples/cluster/plot_mini_batch_kmeans.html`