HEC MONTRÉAL

MATH 80629 – Apprentissage Automatique I

Projet de Session – Analyse de Sentiment

Présenté à

Laurent Charlin

Par

Christelle George - 11288106

Quentin Tabourin - 11290690

Le 11 décembre 2020

À Montréal

Table des matières

Liste des figures

Liste des tableaux

Introduction

Dans le cadre des élections présidentielles américaines de 2016, une série de 12 débats républicains ont pris place entre les 17 candidats, entre le 6 août 2015 et le 10 mars 2016. Le premier débat, organisé par Fox News et Facebook, modéré par Bret Baier, Megyn Kelly et Chris Wallace, eut lieu à l'arène « Quicken Loans » à Cleveland, Ohio. Les dix candidats les plus populaires dans les sondages y étaient invités : Donald Trump, Jeb Bush, Scott Walker, Mike Huckabee, Ben Carson, Ted Cruz, Marco Rubio, Rand Paul, Chris Christie et John Kasich.

Très rapidement, le réseau social Twitter s'est enflammé puisque plusieurs milliers d'utilisateurs se sont exprimés sur ce débat. Quelles ont été les réactions des citoyens américains ? Est-il possible de discerner une certaine tendance d'avis compte tenu du vaste étendu d'information à disposition ?

Le but de ce projet est de faire une analyse de sentiment. En d'autres termes, il s'agit d'analyser une grande quantité de données afin de déduire les différents sentiments exprimés.

Analyse exploratoire des données

La base de données choisie provient de la plateforme « Kaggle » et contient un ensemble de tweets des citoyens américains suite au premier débat républicain du 6 août 2015. Elle contient 21 variables et un total de 13871 observations (soit 13871 tweets). Dans le cadre de ce projet, deux variables seront considérées et ne possèdent pas de données manquantes :

- Le « text », soit le tweet écrit : il s'agit du prédicteur X de notre modèle.
- Le « sentiment », soit le label associé au texte : il s'agit de la variable d'intérêt Y que l'on cherche ultimement à prédire.

Tableau 1 : Variables à l'étude

	text	sentiment
0	RT @NancyLeeGrahn: How did everyone feel about...	Neutral
1	RT @ScottWalker: Didn't catch the full #GOPdeb...	Positive
2	RT @TJMShow: No mention of Tamir Rice and the ...	Neutral
3	RT @RobGeorge: That Carly Fiorina is trending ...	Positive
4	RT @DanScavino: #GOPDebate w/ @realDonaldTrump...	Positive

Ceci s'inscrit dans l'optique d'un apprentissage supervisé : le modèle possède un ensemble d'exemples avec leur label : $(\{x_i\ ; y_i\})_{i=1}^{n=13871}$. La difficulté de ce projet réside dans le fait que le tweet est un texte court et emploie un langage informel (emojis, majuscules, répétition de lettres – happyyyyy).

La variable sentiment possède 3 classes : « Positive », « Négative » et « Neutre ». La majorité des observations sont de sentiment négatif (61%).

La base de données comporte 8493 tweets avec une opinion négative, 3412 tweets avec une opinion neutre et 2236 tweets avec une opinion positive. Ce déséquilibre entre les classes n'est pas à négliger, nous y reviendrons plus tard plus en détails.

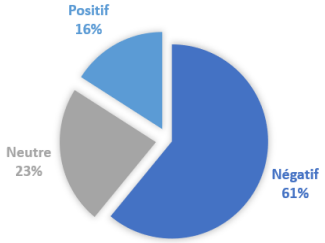

Figure 1 : Diagramme circulaire du sentiment

Revue de littérature

L'analyse de sentiment repose sur le traitement automatique du langage naturel (NLP) qui est également utilisé en autre pour la classification de documents, la recherche d'information et la traduction de textes. Dans le cas présent de l'analyse de sentiment à partir de tweets, les principales difficultés proviennent de la concision du message (moins de 140 caractères) qui réduit considérablement les termes qui expriment le sentiment, et de l'utilisation d'un langage informel comme les abréviations [1].

Le prétraitement des tweets représente ainsi une part importante du travail et consiste à retirer les mentions de retweets, les mots d'arrêts, les hashtags, etc. L'identification d'éléments informels intensifiant le sentiment comme les mots en capitales (e.g. « I LOVE ») et la répétition des caractères d'un mot (e.g. « hapyyyyy ») [1] ,ainsi l'utilisation d'un lexique (« lexicon ») déjà existant de mots exprimant les sentiments [2] sont deux techniques utilisées afin d'améliorer la performance des modèles de classification utilisés.

Plusieurs modèles permettent de réaliser la classification des sentiments. Une étude de 2014 a réalisé une comparaison des modèles SVM linéaire, Random Forest, Régression logistique, Naïve Bayes Multinomial et d'un modèle d'ensemble, combinant ces différents modèles sur quatre jeux de données de Twitter [2]. La meilleure performance a été obtenue par le modèle d'ensemble, avec une « accuracy » globale moyenne de 81.06% sur les jeux de données test. Il est à noter que l'utilisation d'un lexique en plus du « bag-of-words » dans le prétraitement fournit de meilleures performances.

Le débalancement des classes entre les sentiments nuit également à l'analyse de sentiment. Dans un article sur l'analyse de sentiment sur Twitter [3], sur trois jeux de données débalancées avec plus de 50% des tweets ayant un sentiment neutre, les outils populaires de classification produisent un faible « macro recall », inférieur à 50%. Pour contrer les effets du large débalancement des classes, une méthode d'ensemble de bootstrapping (BPEF), regroupant plusieurs techniques, est privilégiée. Cette méthode utilise plusieurs jeux de données de tweets différents lors de l'entraînement. Elle multiplie le nombre de features utilisés par le modèle en combinant les mots simples, les « parts-of-speech » et les traits sémantiques des mots. Finalement, la classification est réalisée sur plusieurs modèles à la fois pour former un ensemble.

La méthode d'ensemble est toutefois plus complexe du fait de la multiplication des jeux de données, des features et des modèles utilisés. Une approche plus simple consiste à équilibrer les classes soit en augmentant le poids des classes minoritaires (suréchantillonnage), soit en réduisant le poids des classes majoritaires (sous-échantillonnage). Bien que ces deux méthodes améliorent en moyenne la performance des modèles de classification [4] [5], le sous-échantillonnage a le désavantage d'enlever de l'information en supprimant des observations de la classe majoritaire.

Finalement, comme la base de données provient du site Kaggle, un bref survol des travaux déjà réalisés sur cette base de données a permis de relever que presque tous les travaux observés n'ont considéré qu'uniquement la classe positive et la classe négative pour réaliser une classification binaire au lieu d'une classification multi-classe. Le fait d'enlever les sentiments neutres aide probablement les modèles à mieux performer. Il est possible de remarquer que les différents modèles choisis (Naïve Bayes, K-Nearest Neighbors et Neural Networks) classifient très bien la classe négative avec un recall de plus de 90% mais très mal la classe positive variant de 30% pour certains modèles à près de 60% pour d'autres.

Enfin, il est à noter que la méthode de suréchantillonnage de la classe positive a permis d'augmenter le recall de cette classe de 54% à 79% [6].

Pré-traitement des données

Dans un premier temps, la variable « text » nécessite un prétraitement afin de pouvoir la manipuler dans un contexte de prédiction. Pour se faire, la librairie NLTK est utilisée. Trois étapes sont nécessaires :

1- La Tokenisation : ce processus permet de décomposer la chaine de caractère du tweet en « pièces détachées », soit les « jetons ». Un « jeton » est donc une instance d'une séquence de caractères dans un document particulier qui, une fois rassemblé, permet le traitement de la variable en question.
2- Le retrait des « stop words » : les « stop words » sont les mots qui ne rajoutent pas de sentiment à une phrase. Sans ces mots, la phrase aurait encore tout son sens en terme de sentiment. Dans le cas de notre projet, ceci inclus également le « RT @username » propre au tweets.
3- La lemmatisation : ce traitement permet de réduire toutes les dérivées d'un mot à son noyau. Les différents temps d'un verbe sont ramenés à un seul, le pluriel au singulier etc.

Somme toute, après être passé aux travers de ces étapes, les tweets ont cette forme :

```
0    RT @NancyLeeGrahn: How did everyone feel about...
1    RT @ScottWalker: Didn't catch the full #GOPdeb...
2    RT @TJMShow: No mention of Tamir Rice and the ...
3    RT @RobGeorge: That Carly Fiorina is trending ...
4    RT @DanScavino: #GOPDebate w/ @realDonaldTrump...
```

```
0        everyone feel climate change question last nig...
1          catch full last night scott best line 90 second
2                    mention tamir rice held cleveland wow
3        carly fiorina trending hour debate men complet...
4        w delivered highest rating history presidentia...
```

Figure 2 : Tweets originaux (GAUCHE) et tweets obtenus après le prétraitement (DROITE)

Ensuite, les données sont converties en numériques au moyen de l'approche TF-IDF (Vectorizer). Cette approche est privilégiée face à l'approche CountVectorizer car cette dernière ne retourne que le nombre d'occurrences d'un mot dans le tweet. Le bénéfice de l'approche TF-IDF Vectorizer est qu'elle prend également en compte la présence de ce mot dans les autres tweets. Plus un mot est présent dans les autres tweets, plus son poids sera réduit. Un total de 9506 mots forment le dictionnaire généré à partir des 13871 tweets de notre jeu de données.

D'autre part, la variable cible « sentiment » est traitée brièvement : le « positif » vaut 1, le « neutre » vaut 0 et le « négatif » vaut -1. Désormais, les variables à l'études sont prêtes à être manipulées.

Méthodologie

Grâce à l'approche du train-test-split de la librairie Scikit-Learn, le jeu de données est divisé en ensemble d'entraînement (80%) et de test (20%). Dans un premier temps, un modèle de référence sera testé : celui du Naïve Bayes Multinomial. Il s'agit d'un modèle assez simple qui ne nécessite pas de définir de paramètres. Ensuite, deux modèles linéaires, la régression logistique et le Support Vector Classifier (SVC), ainsi que deux modèles non-linéaires, le K-Nearest Neighbor (KNN) et les forêts aléatoires (RF), sont également testés. Étant donné qu'au vu de la revue de littérature le modèle de réseaux de neurones produit des performances similaires aux autres modèles, ce modèle n'est pas considéré. Les hyperparamètres des modèles sont optimisés pour maximiser le score F1 macro. Une fois optimisés, ces modèles sont comparés au moyen du score F1 macro calculé sur l'échantillon test.

L'ensemble des performances est mesuré au moyen de l'indicateur « F1 score macro average » . Le score F1 est une mesure de performance calculée à partir des mesures de précision et de recall. L'utilisation du score F1 macro permet donc de mieux évaluer la performance du modèle, dans le cadre de classes déséquilibrées, que la mesure d'accuracy. $F1 = \frac{2*Recall*Precision}{Recall+Precision}$, $Recall = \frac{TP}{TP+FN}$, $Precision = \frac{TP}{TP+FP}$

Finalement, étant donné que le jeu de données est déséquilibré, deux méthodes de suréchantillonnage des données minoritaires, méthode « Random Oversampler » et « SMOTE » sont testées.

Analyse des résultats

Naïve Bayes Multinomial

Le modèle du Naïve Bayes Multinomial représente notre modèle de référence. Il se base sur une formule simple. Pour un tweet *t* et une classe *c* : $P(c \mid t) = \frac{P(t|c)*P(c)}{P(t)}$

Ce modèle assume l'hypothèse d'indépendance entre les attributs d'une classe. Cependant, ceci n'est pas souvent vrai. Parmi les avantages de ce modèle, il requiert un très petit nombre de paramètres et sa complexité temporelle est linéaire, soit $O(n)$. Dans notre cas, le modèle obtient un score F1 macro de 50%. Les résultats de ce premier modèle sont regroupés dans les tableaux suivants :

Tableau 2 : Rapport de classification (GAUCHE) et matrice de confusion (DROITE) sur l'échantillon test du modèle Naïve Bayes Multinomial

	precision	recall	f1-score	support
-1	0.67	0.94	0.78	1670
0	0.54	0.21	0.30	642
1	0.69	0.30	0.42	463
accuracy			0.66	2775
macro avg	0.64	0.48	0.50	2775
weighted avg	0.65	0.66	0.61	2775

Prédictions \ Observations	Negative	Neutral	Positive
Negative	1563	76	31
Neutral	476	134	32
Positive	285	37	141

Le modèle prédit très bien les tweets « négative -1 » puisqu'il obtient un score F1 de 78% dans cette catégorie : 1563 sur les 1670 sont bien classés, c'est excellent. Le modèle performe moyennement quant à la prédiction des tweets « positive 1 » avec un score F1 de 42% : parmi les 463 tweets positifs, 141 sont bien classés et 285 sont classés en tant que négatifs. Finalement, le modèle ne parvient pas à prédire les tweets « neutre 0 ». Cette catégorie obtient un score de 30%, soit le pire score : parmi les 642, seulement 134 sont correctement classés et plus de 476 sont attribués à la catégorie négative. Ces deux catégories manquent de données pour être en mesure de prédire correctement ces classes.

Régression logistique

Un modèle de régression logistique est ajusté par la suite puisque la revue de littérature indique que cette méthode est adaptée dans le contexte d'analyse de sentiment, pour une variable cible $Sentiment = \{-1; 0; 1\}$. La régression logistique réalise une classification binaire 1 et 0. Ainsi dans le cas présent d'une classification à trois classes, le problème initial est divisé en trois problèmes de classification binaire « one-vs-rest »(OVR). Pour chaque classe de sentiment, un modèle de régression logistique différent estime la valeur des coefficients de chaque mot qui prédisent au mieux le sentiment donné.

Tableau 3 : Rapport de classification sur l'échantillon test du modèle de régression linéaire

	precision	recall	f1-score
Negative	0.70	0.93	0.80
Neutral	0.59	0.26	0.35
Positive	0.67	0.37	0.48
macro avg	0.65	0.52	0.54

Ce tableau permet de tirer quasiment les mêmes observations que celles de la méthode précédente : le score F1 associé à la classe majoritaire « Négative » est très bon et vaut 80%. À contrario, les classes « Neutre » et « Positive » sont mal prédites avec un score F1 respectif de 35% et 48%. Au final, le score F1 macro du modèle vaut 54%.

SVC

La méthode SVC est similaire à la régression logistique dans le sens où elle ne permet que de faire une classification binaire. L'approche « one-vs-rest » est donc également employé pour le SVC. La méthode SVC possède plusieurs kernels différents qui sont chacun adaptés pour des jeux de données différents. Le kernel linéaire est le plus adapté à notre problème de classification de texte car l'échantillon d'entraînement possède de milliers de features (mots). Les librairies LibSVM (SVM) et LibLinear (LinearSVC) permettent de faire de la classification SVM linéaire. La complexité de LibSVM est $O(n^2)$ ou $O(n^3)$ alors que la complexité de LibLinear n'est que $O(n)$. De ce fait, comme notre échantillon d'entraînement est grand (près de 10000), la fonction LibLinear a été choisi.

Optimisation des paramètres

Le coût « C » correspond à un paramètre de tuning choisi par validation croisée, et représente la tolérance aux observations mal classifiées. Plus « C » augmente, plus cette tolérance augmente et pose un risque de sous-apprentissage. À contrario, si « C » est trop faible, il y a un risque de sur-apprentissage.

La figure 3 indique un score F1 macro maximisé pour un coût de 0,4. Lorsque le coût augmente d'avantage, le score F1 diminue graduellement.

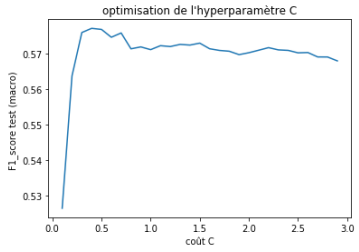

Figure 3 : Score F1 en fonction du coût

Résultats

La matrice de confusion de ce modèle ainsi que le rapport de classification sur l'échantillon test sont les suivants :

Tableau 4 : Rapport de classification (GAUCHE) et matrice de confusion (DROITE) sur l'échantillon test du modèle SVC

	precision	recall	f1-score
Negative	0.71	0.89	0.79
Neutral	0.54	0.31	0.39
Positive	0.67	0.46	0.55
macro avg	0.64	0.55	0.58

Prédictions \ Observations	Negative	Neutral	Positive
Negative	1492	119	59
Neutral	398	197	47
Positive	200	49	214

Le modèle SVC prédit très bien les sentiments négatifs (recall de 89%), de façon satisfaisante les sentiments positifs (recall de 46%) mais mal les sentiments neutres (recall de 31%). Bien que le score F1

soit encore faible pour le sentiment neutre, le modèle SVC a un score de F1 de 55% pour le sentiment positif ! Comparé aux modèles précédents, celui-ci prédit mieux les classes minoritaires et cette prédiction se fait au profit de la classe majoritaire. Pour le moment, c'est le modèle avec le score F1 macro le plus élevé de 58%.

KNN

Le modèle des KNN se base sur la ressemblance entre les observations pour réaliser la classification. Elle utilise les K voisins les plus proches de chaque observation pour l'assigner à une classe de sentiment. L'optimisation du paramètre K est donc essentielle pour maximiser la performance du modèle.

Optimisation des paramètres

Les deux graphiques de la figure 4 indiquent que la performance optimale du modèle (accuracy et score F1) est atteinte en utilisant les 150 voisins les plus proches de chaque observation.

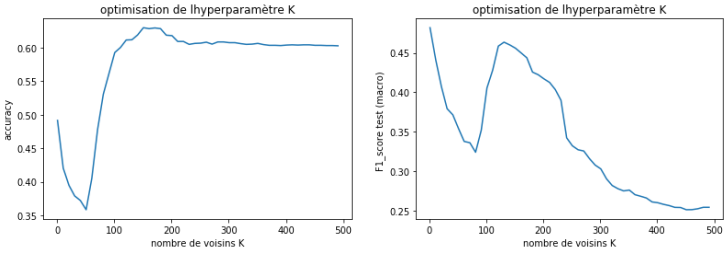

Figure 4 : Accuracy (GAUCHE) et Score F1 (DROITE) en fonction du nombre de voisins K pour le modèle KNN

Résultats

La matrice de confusion de ce modèle ainsi que le rapport de classification sur l'échantillon test sont les suivants :

Tableau 5 : Rapport de classification (GAUCHE) et matrice de confusion (DROITE) sur l'échantillon test du modèle K-NN

	precision	recall	f1-score
Negative	0.67	0.90	0.77
Neutral	0.41	0.24	0.31
Positive	0.57	0.21	0.31
macro avg	0.55	0.45	0.46

Prédictions \ Observations	Negative	Neutral	Positive
Negative	1495	130	45
Neutral	459	155	28
Positive	277	89	97

Le modèle KNN prédit correctement presque tous les tweets de sentiment négatif, soit 1495 sur 1670. Par contre, c'est l'un des pires modèles en ce qui concerne la prédiction des sentiments positif et neutre, avec un score-f1 de 31% pour chacun de ses sentiments.

Forêt aléatoire

Optimisation des paramètres

Afin de trouver le modèle de forêt aléatoire optimal pour le jeu de données, plusieurs graphiques sont tracés afin d'obtenir les meilleurs hyperparamètres. L'étude porte sur $n\_estimators$, max $\_depth$, min $\_\_samples\_split$, max $\_leaf\_nodes$ et max $\_features$.

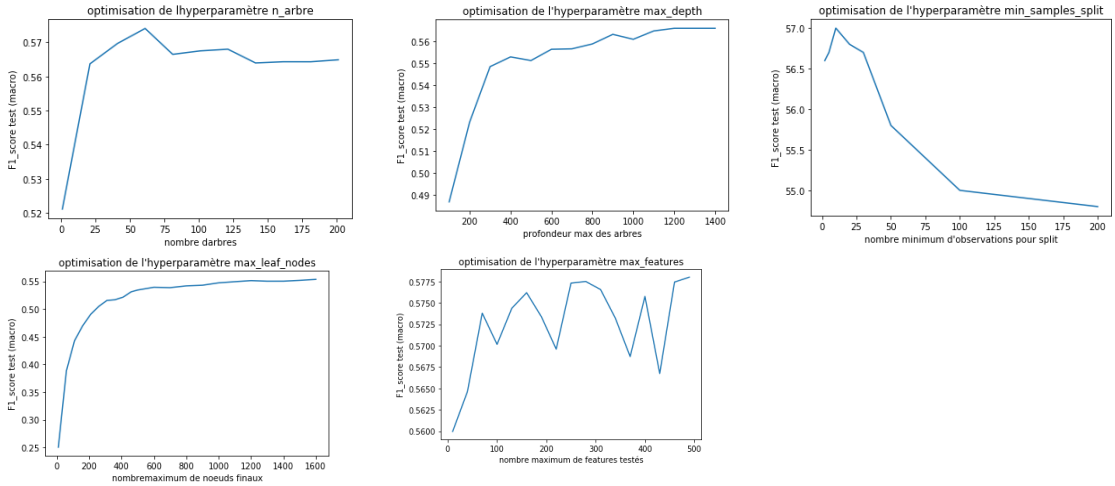

Figure 5 : Graphique du score F1 en fonction du nombre d'arbres de la forêt (1), de la profondeur de l'arbre (2), du nombre d'observations minimum pour split (3), du nombre maximum de nœuds terminaux (4) et du nombre maximal de features (5)

La lecture de l'ensemble de ces graphiques de la figure 5 indique qu'un nombre de 60 arbres avec une profondeur de 1200 serait idéal pour notre modèle. On peut noter qu'à partir d'une profondeur de 600, l'augmentation de la performance est marginale. Il faudrait également un minimum de 10 échantillons par split et 450 nœuds terminaux. Finalement, le dernier graphique de la figure 6 indique que quel que soit le nombre de features testés, le score F1 varie très peu. Il s'agit d'un choix stratégie à faire pour trouver le bon équilibre entre le nombre choisi et la vitesse de l'algorithme.

Résultats

La matrice de confusion de ce modèle ainsi que le rapport de classification sur l'échantillon test sont les suivants :

Tableau 6 : Rapport de classification (GAUCHE) et matrice de confusion (DROITE) sur l'échantillon test du modèle des forêts aléatoires

	precision	recall	f1-score	support
Negative	0.68	0.94	0.79	1670
Neutral	0.63	0.23	0.34	642
Positive	0.70	0.37	0.48	463
macro avg	0.67	0.51	0.54	2775

Prédictions \ Observations	Negative	Neutral	Positive
Negative	1570	60	40
Neutral	460	147	35
Positive	266	25	172

Encore une fois, la classe majoritaire « négative » est très bien prédite (Score F1 de 79%), les deux autres classes ne sont pas satisfaisantes. « Neutre » obtient un score de 34% et « positive » un score de 48%. Ceci dit, contrairement aux autres modèles qui obtiennent des résultats sensiblement similaires entre les échantillons train et test, la méthode des forêts aléatoires permet d'obtenir d'excellents résultats sur le train, et ce, quelle que soit la classe ! La mesure de accuracy globale vaut 93%, et reflète un score F1 de 95% pour la catégorie « négative », 87% pour la catégorie « neutre » et 90% pour la catégorie « positive ». En fait, ceci est probablement lié au hyperparamètre de profondeur choisi de ce modèle : la valeur de 1200 est sans doute trop élevée puisqu'une valeur de 600, soit la moitié, semble tout aussi convenable. Ce modèle effectue du surapprentissage, ce qui se traduit par de mauvaises performances avec de nouvelles observations (échantillon test).

Tableau 7 : Rapport de classification sur l'échantillon train du modèle des forêts aléatoires

	precision	recall	f1-score	support
-1	0.94	0.96	0.95	6823
0	0.90	0.84	0.87	2500
1	0.90	0.90	0.90	1773
accuracy			0.93	11096
macro avg	0.91	0.90	0.91	11096
weighted avg	0.93	0.93	0.93	11096

Les forêts aléatoires se basent sur le principe du bagging qui permet d'améliorer significativement la stabilité et la précision des algorithmes. Ceci dit, il s'agit d'une méthode très complexe qui requiert de générer beaucoup d'arbres de décision, ce qui nécessite une grande puissance de calcul. Aussi, c'est une méthode qui demande une longue période d'entrainement comparée aux autres. Au final, dans le cadre de ce projet, ce n'est pas une méthode très rentable.

Suréchantillonnage

La méthode de suréchantillonnage des classes minoritaires permet d'équilibrer les observations. Ceci s'effectue au moyen de la libraire « imbalances-learn » de Scikit Learn. Plusieurs techniques sont disponibles, parmi elles, la méthode du « Radom OverSampler » et le « SMOTE » sont celles considérées, au moyen de la validation croisée.

La première technique du suréchantillonnage aléatoire permet de répéter des observations de la classe minoritaire afin d'équilibrer les observations entre les classes. L'approche « SMOTE », quant à elle, suréchantillonne la/les classes minoritaires en créant des exemples « synthétiques ». Cette approche offre une plus grande diversité des observations. Le modèle de la régression logistique est employé dans ce contexte car d'après la revue de la littérature, il semble produire de bons résultats lorsqu'il est utilisé avec l'approche de TF-IDF de Vectorizer.

Résultats

Le tableau 8, présente les rapports de classification pour les approches « RandomOverSampler » et « SMOTE » avec un modèle de régression logistique.

Tableau 8 : Rapports de classification avec l'approche « RandomOverSampler » (GAUCHE) et « SMOTE » (DROITE)

	precision	recall	f1-score
Negative	0.79	0.74	0.76
Neutral	0.46	0.46	0.46
Positive	0.51	0.64	0.57
macro avg	0.58	0.60	0.59

	precision	recall	f1-score
Negative	0.77	0.78	0.78
Neutral	0.49	0.38	0.43
Positive	0.49	0.65	0.56
macro avg	0.57	0.59	0.59

On voit que les deux approches étudiées présentent des résultats sensiblement similaires. Le score F1 macro avec les deux méthodes de suréchantillonnage sur la régression logistique est identique et supérieur au score F1 en gardant les classes déséquilibrées, respectivement 59% versus 54%. Ceci montre que ces techniques sont nécessaires dans le cas où les classes ne sont pas équilibrées et permettent d'obtenir de bien meilleurs résultats. En observant précisément la performance de chaque classe, on peut noter que le suréchantillonnage « Random OverSampler» permet au modèle de nettement mieux prédire les classes minoritaires. En effet, le score F1 pour le sentiment neutre augmente de 35% à 46%, et de 37% à 64% pour le sentiment positif. Toutefois, il prédit moins bien la classe majoritaire avec le score F1 du sentiment négatif qui diminue de 80% à environ 74%. Ceci dit, le score reste très convenable. Ces observations sont logiques puisque le suréchantillonnage augmente les observations des classes

minoritaires, ce qui permet au modèle de s'entrainer sur un jeu plus complet de données. Entre les deux méthodes utilisées, il est difficile à ce stade de parvenir à les départager.

Interprétation des résultats

La figure 6 présente le Recall de chaque méthode sur l'échantillon test, tandis que le tableau 9 plus bas regroupe le score F1 global de ces méthodes. L'ensemble des modèles performe très bien lorsqu'il s'agit de prédire la classe majoritaire négative : le Recall de cette classe dépasse 90% pour les modèles étudiés sans suréchantillonnage et se situe autour de 75% avec suréchantillonnage. Concernant les classes minoritaires, il est étonnant de noter que, bien que la classe neutre contienne plus de données que la classe positive (respectivement 21% et 16% des observations), c'est la classe positive qui est toujours la mieux prédite.

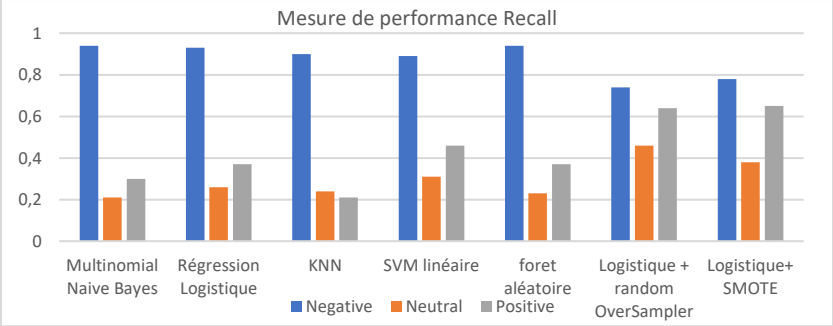

Figure 6 : Graphique du Recall des méthodes étudiés pour chaque catégorie de sentiment

Il apparait clair que quelle que soit la méthode employée, travailler avec des classes déséquilibrées empêche d'obtenir une performance optimale car les modèles prédisent mal les sentiments minoritaires neutre et positif. Utiliser une méthode de suréchantillonnage pour équilibrer les classes permet d'augmenter la performance des modèles : comme l'indique la figure 6, le Recall des classes neutre et positive augmente significativement et atteint environ 40% pour la classe neutre et plus de 60% pour la classe positive ! Avant cela, ces valeurs plafonnaient approximativement à 30% (neutre) et 40% (positive).

Tableau 9 : Scores F1 « Macro Average »

Modèle	F1score test
MultinomialNB	50.2 %
Régression logistique	54.2%
K-NNs	45.9%
SVC	57.7%
Random Forest	53.8%
Logistique + Oversampling	59.0%

Cette différence de performance entre la classe neutre et positive, toutes d'elles minoritaires, reflète une problématique sous-jacente. Le suréchantillonnage permet de résoudre le problème de classification de la classe positive, puisque sa performance vient concurrencer celle de la classe négative majoritaire. Il permet d'autre part d'améliorer la performance de la classe neutre mais celle-ci reste tout de même non convenable et demeure un bémol non résolu. En regardant les 10 mots les plus importants pour la classification de chaque sentiment, on remarque que, pour l'ensemble des méthodes, les mots des sentiments positif et négatif expriment réellement le sentiment associé et sont très discriminants. Cependant, les mots les plus importants pour prédire le sentiment neutre n'expriment pas vraiment de sentiment. Les mots n'ont pas de signification particulière, comme par exemple le nombre 24.

😡 Négatif : *Hate, joke, police, deserve, idiot, lie, fox, shit, disappointed*
😐 Neutre : *Reporter, transcript, Cleveland, 8pm, 24, retweeted, tie, yesterday*
😄 Positif : *Awesome, excited, thank, impressed, best, great, fantastic, enjoyed, loved*

Ces mots associées à la catégorie neutre sont dénués de sentiment, et ce quelque soit le modèle étudié (par exemple, le modèle Multinomial à la Figure 7).

Si beaucoup de travaux dénigrent cette catégorie neutre pour miser sur une classification binaire (Négatif/Positif), ce travail tente de proposer une classification adéquate de cette catégorie longtemps délaissée. Au final, il s'agit d'une catégorie importante puisqu'elle traduit, justement, une absence de sentiment. Et c'est bien là que réside la difficulté de la classification.

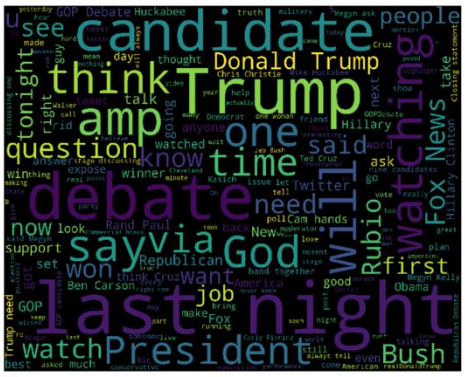

Figure 7 : Mots récurrents dans la catégorie neutre du modèle Naïve Bayes Multinomial

Au final, parmi l'ensemble de ces méthodes, c'est le modèle de régression logistique avec suréchantillonnage qui performe le mieux sur l'échantillon test avec un score F1 de 59%. Sans suréchantillonnage, c'est le modèle de classification des vecteurs C-Support qui semble être le plus adapté avec un score F1 de 57,7% obtenu sur l'échantillon test. Ceci est dû au fait que la frontière de séparation entre les classes positive et négative est très claire et discriminante, ce qui constitue une arme de force majeure pour la méthode SVC. Elle permet d'ailleurs d'obtenir le meilleur score F1 de la classe positive de 55%, associé à un Recall de 46%.

Conclusion et recommandations

Dans le contexte d'analyse de sentiment de tweets portant sur le premier débat de la primaire républicaine de 2016, l'analyse exploratoire a mis en évidence un déséquilibre de la variable réponse sentiment, avec 61% des tweets qui expriment un sentiment négatif, 23% un sentiment neutre et 16% un sentiment positif. L'analyse comparative de cinq modèles de classification (sans suréchantillonnage) selon le critère du score F1 macro a montré que le modèle SVC (57%) est le plus performant suivi du modèle de régression logistique (54%) et du modèle de Random Forest (54%). L'utilisation d'une méthode de suréchantillonnage a prouvé le bénéfice du rééquilibrage des classes en augmentant la performance du modèle de régression logistique de 5%, soit un score F1 macro de 59%. Finalement, la classification du sentiment neutre représente la tâche la plus difficile pour nos modèles, puisque son vocabulaire n'est pas discriminant et traduit une absence de sentiment.

Pour augmenter la performance des modèles, plusieurs améliorations peuvent être apportées :

1. Tester les autres modèles avec une méthode de suréchantillonnage
2. Utiliser un lexique prédéfini de mots associés aux sentiments dans le positif, neutre et négatif
3. Prendre en compte l'identification d'éléments informels intensifiant tels que les emojis, les majuscules et la répétition de lettres (happyyyy) etc.
4. Envisager une approche différente au problème telle que dans un premier temps, déterminer la neutralité du tweet pour ensuite déterminer la polarité du tweet le cas échéant (Négatif/Positif)

In fine, une limitation de l'utilisation de l'apprentissage automatique dans le cadre de l'analyse de sentiment réside dans la détection du sarcasme et des expressions de la langue (« he killed it ! ») qui aboutira à de mauvaises classifications. De plus, il est important de noter que le sentiment attribué aux tweets dans la base de données à l'étude a été effectué de manière subjective par un tiers parti, ce qui représente un biais intrinsèque aux données dans le cadre d'un apprentissage supervisé.

Bibliographie

[1] E. Kouloumpis, T. Wilson et J. Moore, «Twitter sentiment analysis: The good the bad and the omg!,» chez *Fifth International AAAI conference on weblogs and social media.*, 2011.

[2] N. F. Da Silva, E. R. Hruschka et E. R. Hruschka Jr., «Tweet sentiment analysis with classifier ensembles,» chez *Decision Support Systems*, vol. 6, 2014, pp. 170-179.

[3] A. Hassan, A. Abbasi et D. Zeng, «Twitter Sentiment Analysis: A Bootstrap Ensemble Framework,» chez *International Conference on Social Computing*, 2013.

[4] W. Satriaji et R. Kusumaningrum, «Effect of Synthetic Minority Oversampling Technique (SMOTE), Feature Representation, and Classification Algorithm on Imbalanced Sentiment Analysis,» chez *2nd International Conference on Informatics and Computational Sciences (ICICoS)*, Semarang, Indonesia, 2018.

[5] J. Prusa, T. Khoshgoftaar, D. Dittman et A. Napolitano, «2015, 197-,» chez *IEEE International Conference on Information Reuse and Integration*, San Francisco, 2015.

[6] S. Kumar, «Twitter tweets classification (+/-) (RNN)|Keras,» 2018. [En ligne]. Available: https://www.kaggle.com/shivam001/twitter-tweets-classification-rnn-keras. [Accès le 07 décembre 2020].