Traitement des données

présenté par
Alex Sirois (111 238 323) - 2e cycle
Guillaume D. De Grandpré (111 022 887) - 1er cycle
Xavier Lindsay (111 135 621) - 2e cycle

6 mai 2020

1 Description des algorithmes implantés

1.1 Algorithmes d'extraction d'attributs

Afin d'avoir des données utilisables dans la forêt d'arbres, une multitude d'attributs doivent d'abord être extraits des données. La libraire d'apprentissage machine utilisée pour le projet est *scikit-learn*, ce qui impose l'utilisation d'attributs strictement numériques. Ainsi, en plus de l'extraction d'attributs, plusieurs transformations sont nécessaires afin de traduire de l'information textuelle en information numérique.

1.1.1 Préparation des scores

L'indicateur de succès d'un article est son score. Ce score doit être calculé à l'aide des événements de vues disponibles dans les données. Comme mentionné dans le rapport précédent, la procédure pour attribuer un score à un article en fonction de ses événements de vues est la suivante:

Figure 1: Attribution d'un score à un article en fonction de ses événements de vues

Exemple : Supposons un article avec les évènements suivants :

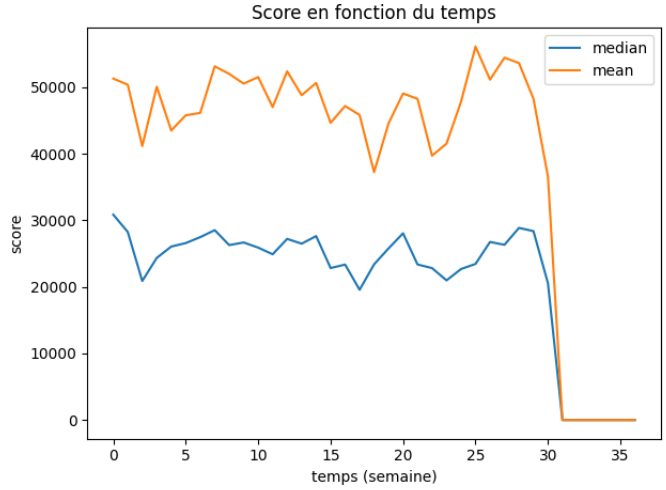
évènement	compte	pondération
view	500	1
view5	450	1
view10	300	2
view30	120	5
view60	10	10

La valeur pondérée de cet article est : $(10 \times 10) + (120 \times 5) + (300 \times 2) + (450 \times 1) + (500 \times 1) = 2\,250$.

L'équipe se demande par la suite si un travail de normalisation des scores doit être fait. Le raisonnement est qu'un article ayant eu plus de temps en ligne aura probablement un score plus élevé qu'un article ayant été publié récemment,

simplement car il aura eu plus de temps pour amasser des vues, et pas forcément en raison de sa popularité. Afin de valider cette supposition, l'équipe trace un graphe des scores des articles en fonction du temps:

Figure 2: Scores des articles en fonction du nombre de semaines depuis la publication



Le graphe invalide la supposition de l'équipe. Il ne semble pas y avoir de corrélation entre le temps en ligne et le score de l'article. Ainsi, les scores ne sont pas normalisés.

1.1.2 Extraction des données de publications

Un important travail d'extraction de données pertinentes des publications doit être fait en amont. En effet, un article peut être publié dans plusieurs des six journaux, et à plusieurs reprises dans un même journal (dans plusieurs catégories, dans la version site ou mobile, à plusieurs dates différentes). Cette étape de l'extraction d'attributs est l'une de celles qui prennent le plus de temps et d'essais-erreurs. En effet, il y a plusieurs façons d'extraire l'information pertinente d'un article. L'équipe se demande d'abord quelle date de publication doit être retenue. Une première idée est de ne garder que celle de la publication la plus ancienne, mais cette idée est rapidement écartée en raison de la perte évidente d'information. L'idée retenue est plutôt celle de découper les attributs en six journaux différents. Ainsi, pour chaque journal, sa date de publication est retenue. Cette date est découpée en quatre attributs: l'heure de la journée (en minutes depuis le débute de la journée), le jour de la semaine (1 pour lundi, 7 pour dimanche), la date et le mois. Le choix de ne pas retenir l'année de publication est fait délibérément, car en raison des données d'événements provenant uniquement de 2019, l'algorithme attribuerait une importance beaucoup trop forte à l'année.

Un travail similaire est fait pour distinguer le médium de publication. En effet, une publication peut être affichée sur la version site, mobile ou externe

(*external*) d'un journal. Après quelques tests sur l'attribut *external* (voir la section Tests pour la procédure utilisée), il est déterminé que celui-ci a une importance nulle. Ainsi, seuls les attributs site et mobile sont gardés. Pour chaque article et chaque journal, le nombre de publications par médium est l'attribut retenu.

Finalement, la catégorie de l'article est extraite de ses publications. Cette information, qui se trouve dans le champ *slug* des données, peut être multiple pour un seul article. En effet, un même article peut être publié dans différentes sections d'un même journal ou de journaux différents. Cependant, une seule catégorie est retenue: celle qui est la plus populaire parmi toutes les publications de la base de données. Afin de transformer la catégorie en une information numérique, une simple association est effectuée à l'aide d'une *map*: chaque catégorie est associée à un identifiant numérique. Ces identifiants sont préservés dans un fichier afin de préserver l'information.

1.1.3 Autres attributs

Quelques autres attributs simples sont extraits des données: *template name*, *visual* et *channel*. Les attributs *template name* et *visual* contiennent de l'information sur l'affichage du contenu et le visuel de l'en-tête. L'attribut *channel*, quant à lui, est un mot clé décrivant l'article. L'auteur d'un article est également un attribut retenu. La grande majorité des articles ont un seul auteur, mais certains d'entre eux en ont plus qu'un. Ainsi, un attribut *nombre d'auteurs* est ajouté aux données, et un des auteurs de l'article est sélectionné au hasard. Pour tous les attributs mentionnés ci-haut, une simple association d'un mot à un identifiant unique est utilisée afin de transformer l'information textuelle en attribut numérique.

1.1.4 Extraction des données textuelles

Un premier lancement de la forêt aléatoire avec uniquement les attributs mentionnés aux sections précédentes est effectué, afin de voir si des résultats significatifs peuvent être obtenus avec des attributs de base. Après tests (voir la section Tests pour la procédure utilisée), le coefficient de détermination R^2 stagne autour de 0,5 après 10 *epoch*. Il semble donc nécessaire d'extraire plus d'attributs. Pour le moment, aucun attribut contenant de l'information sur le contenu de l'article n'est utilisé. L'équipe se concentre donc sur l'extraction d'information à partir des données textuelles de l'article.

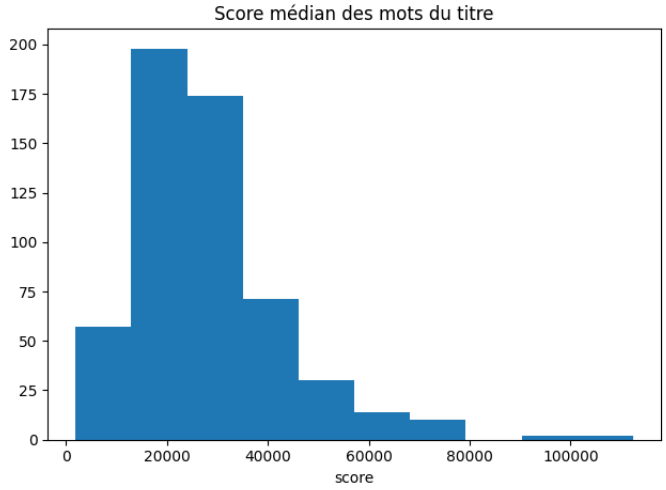
Les trois champs des données représentant le contenu de l'article sont: *title*, *lead* et *chapters*. *Title* est le titre de l'article. *Lead* est un résumé de l'article apparaissant au début. C'est typiquement la première chose que le lecteur lit, après le titre. *Chapters* est le corps de l'article, divisé par chapitres.

Pour ces trois champs, le nombre de mots est calculé et ajouté en tant qu'attribut. Pour le titre et le *lead*, un attribut supplémentaire contenant le nombre de caractères est également ajouté.

Ces attributs, bien qu'intéressants, ne semblent pas suffisants. Il est nécessaire de faire une analyse du contenu. L'équipe décide d'implanter un algorithme

permettant de déterminer les mots les plus *populaires* par champ, et d'ajouter un attribut par article contenant le nombre de mots populaires dans celui-ci. L'algorithme pour obtenir les mots populaires consiste à parcourir tous les articles, en extraire les mots d'un certain champ en prenant soin d'enlever les mots sans importance (appelés *stopwords*) grâce à une librairie de traitement de langage naturel *spacy*, et de remplir une *map* contenant, pour chaque mot, la liste des scores de tous les articles contenant ce mot. À noter que seuls les mots apparaissant plus d'un certain nombre N de fois sont retenus afin de filtrer les valeurs aberrantes. Une fois cette *map* créée, la médiane des scores pour chaque mot est calculée. Le P pourcentage des mots ayant les plus hautes médianes sont retenu et qualifié de mot populaire. Cet algorithme est roulé trois fois séparément: une fois pour le titre, une fois pour le *lead*, et une fois pour les chapitres. Un exemple de distribution des scores médians des mots des titres est présenté à la figure 3.

Figure 3: Distribution des scores médians des mots des titres pour $N = 50$



Une fois les listes de mots populaires générées, le nombre de mots populaires contenu dans les champs *title*, *lead* et *chapters* est ajouté en tant qu'attribut.

Après avoir extrait les mots populaires, l'équipe entreprend d'extraire les entités nommées. Roja Bandari et al. [1] ont tenté de prédire dans leur article *The Pulse of News in Social Media: Forecasting Popularity* la popularité d'un article qui circule à travers les réseaux sociaux. Avec seulement quatre caractéristiques d'un article, ils ont réussi à estimer avec une précision d'environ 84% la popularité de l'article. Ces quatre caractéristiques sont la source de l'article, sa catégorie, la subjectivité de la langue et les entités nommées. La catégorie et la source étant déjà faites, l'équipe se penche sur les entités nommées et la subjectivité.

Roja Bandari et al. ont fait leur recherche sur des articles provenant de réseaux sociaux. Ces articles ne provenant pas nécessairement de sources fiables, ils contiennent un niveau de subjectivité variable. Puisque le corpus de texte du projet provient d'articles de journaux, l'indice de subjectivité est plus

bas et donc, l'équipe décide de ne pas extraire d'attribut de subjectivité de la langue, compte tenu de ce fait et du temps requis d'implantation.

Il reste donc les entités nommées à extraire. Pour ce faire, l'équipe se tourne vers la libraire externe *spacy* qui permet de facile extraire des entités nommées d'un texte. Les entités sont par la suite enregistrées avec les scores des articles associés. Cependant, en observant les entités extraites avec leurs scores, l'équipe réalise rapidement que *spacy* crée beaucoup d'entités non valides. Par exemple, l'une des entités est un astérisque. L'équipe remarque également que toutes les entités non valides sont des entités n'apparaissant que dans un seul article. La décision est prise de ne tenir compte que des entités nommées se trouvant dans plus d'un article. Le score final de chaque entité est la moyenne des valeurs de toutes ses occurrences. Une fois chaque entité notée, l'algorithme de prétraitement des données parcours les articles, et trouve, dans chacun de ceux-ci, les entités nommées ayant les meilleurs scores (s'il y en a) dans le *lead* et le *title*. S'Il y en a beaucoup, uniquement 10 sont retenues pour le *lead* et 5 pour le *title*. Ainsi, chaque article se fait attribuer 15 scores, soit les scores des entités nommées les plus élevés présents dans le titre et le *lead*. S'il y en a moins que 15, la différence est remplie avec des 0.

1.2 Algorithme de prédiction des scores

Pour commencer, une forêt aléatoire d'arbres de régression générée par *bagging* est utilisée. L'idée est de s'assurer que les arbres de la forêt apprennent de manière différente. L'apprentissage différent va mener à une meilleure décision de groupe.

À chaque étape d'entraînement, 10 000 articles publiés en 2019 sont échantillonnés dans la base de données. De ces échantillons sont extraits les différents attributs mentionnés plus haut et assemblés dans un *array numpy*. La forêt entraîne 5 nouveaux arbres sur ces données. Chaque arbre ne reçoit que le tiers des attributs. Pour suivre la progression de l'apprentissage, un jeu de données de validation est échantillonné de la même façon que pour les données d'entraînement. Pour évaluer les performances de la forêt, le coefficient R^2 est utilisé et sera discuté plus loin dans la section 2.

Après chaque étape d'apprentissage, s' il y a amélioration sur le jeu de validation, la forêt est sauvée sur disque. De cette façon on s'assure de ne pas perdre de progrès s' il y a une erreur, et l'entraînement se veut un algorithme "anytime" où il dure aussi longtemps que désiré. Par contre, s' il n'y a pas de progrès sur le jeu de validation pendant 3 étapes de suite, les pires $log(N)$ arbres de la forêt sont coupés où N représente le nombre d'arbres total. Pour sélectionner les pires arbres, les données d'entraînement de l'étape courante sont utilisées.

Plus tard, lorsqu'est venu le temps de mesurer l'importance des attributs, la tâche s'est avérée assez complexe lorsque les différents arbres n'utilisent pas les mêmes attributs. De plus les performances ne sont pas aussi bonnes qu'espérées. Le *bagging* fonctionne bien en classification où les données aberrantes n'ont pas d'impact sur le résultat, mais en régression, les données aberrantes affectent les

prédictions. Il est donc préférable que la forêt n'ait pas trop de mauvais arbres. Pour ces raisons, le *bagging* est remplacé par une forêt aléatoire de régression standard.

1.2.1 Hyper paramètres

Il y a plusieurs hyper paramètres à l'algorithme. Le nombre de données d'entraînement utilisées à chaque étape fait varier la durée de chaque étape d'entraînement et la précision de chaque arbre individuel. Il pourrait être souhaitable de diminuer le nombre de données pour que chaque arbre soit plus différent que le prochain et augmenter le nombre d'arbres. Le nombre de données de validation joue sur le temps de chaque étape, mais de manière moins importante. Également, plus il y a de données de validation, plus les performances associées sont représentatives des performances générales de la forêt.
Le nombre d'arbres entraînés à chaque étape va encore une fois jouer sur la différence entre les arbres et le temps d'entraînement. En entraînant plus d'arbres par étape, le temps par étape augmente peu et la forêt améliore plus rapidement ses performances. Par contre, les performances maximales de la forêt peuvent être un peu inférieures si les arbres se ressemblent plus en ayant les mêmes données pour s'entraîner.

1.2.2 *Multi processing*

Au fil du temps, de plus en plus d'attributs s'ajoutent et la complexité de ceux-ci augmente. Par conséquent, le temps de chaque étape d'entraînement augmente de manière significative. Le parallélisme est vite apparu comme la solution pour pouvoir accélérer l'algorithme. Les forêts d'arbres aléatoires de *scikit-learn* supportent toutes le parallélisme. Par contre, le plus long reste l'extraction des attributs des articles. Pour paralléliser l'extraction, un pool de processeur est utilisé. Cependant, les associations entre attributs non numériques vers numériques sont générés de manière dynamique et ces associations ne sont donc pas partagées entre les différents processeurs. Pour remédier au problème, les associations sont placées en mémoire partagée avec des mutex pour les modifier. Le temps par étape diminue de moitié.

1.2.3 Temps d'exécution et complexité

Sur une machine très compétente avec 16 coeurs logiques, l'algorithme prend environ 6 millisecondes par données pour extraire les attributs et prédire un score. Pour l'entraînement, l'algorithme prend environ 90 secondes par étape pour échantillonner 10000 données, entrainer 2 arbres et ensuite calculer le coefficient R^2 sur 10000 données de validation. Ce temps augmente graduellement avec le nombre d'arbres dans la forêt, pouvant augmenter à environ 120 secondes par étapes après environ une centaine d'arbres.

Le plus long est l'extraction d'attributs qui dépend de la longueur du texte pour chaque article. L'algorithme gère bien la grande quantité de données.

2 Tests

Pour estimer les performances de la forêt, le coefficient R^2 est utilisé. Cette mesure permet d'estimer à quel point la forêt est capable de faire une bonne régression des données. Pour le calculer, on divise l'erreur au carré par la variance des données. Donc, plus les données sont éparpillées, plus les prédictions peuvent être loin de la valeur réelle. Le coefficient R^2 parfait de 1 n'est possible qu'en prédisant la valeur exacte de toutes les données. Bien entendu, l'équipe ne s'attend pas à de tels résultats. Plusieurs projets avec des buts semblables à celui-ci obtiennent des coefficients R^2 d'au moins 0,6. C'est donc un résultat attendu. Au début, le coefficient ne monte pas plus haut que 0,5. Il est donc clair que la forêt manque d'attributs. Une fois tous les attributs ajoutés, la forêt obtient un coefficient de 0,667 sur le jeu de données de validation.

Un autre test effectué est de valider l'importance des attributs. La forêt d'arbre permet de savoir quels attributs sont les plus utiles. Des attributs comme le nombre de publications externes peuvent être éliminés après constat de l'importance nulle de l'attribut. Dans d'autres cas, cela permet de trouver des erreurs dans l'algorithme quand un attribut tombe à 0 d'importance après qu'il ait été enlevé de la liste d'attributs à aller chercher dans la base de données. L'importance des attributs permet aussi de valider des suppositions sur l'efficacité d'un attribut. Elle permet d'orienter le temps de développement vers des attributs qui sont utiles à la régression.

3 Version finale

3.1 Valeurs prédictives des attributs utilisés

Les dix attributs les plus importants sont affichés dans le tableau 1. Le score constitue la métrique d'importance de Gini où la valeur indique la baisse d'impureté de Gini de l'attribut. Les attributs **NER** sont les entités nommées.

Attribut	Score
Premier score lead NER	11.95%
Premier score titre NER	6.15%
Nombre de mots du contenu	5.96%
Nombre de mots populaires du contenu	5.72%
Mois de publication dans Le Soleil	4.90%
Nom de l'auteur	4.41%
Date de publication dans Le Soleil	4.35%
Nombre de publications version site du Soleil	4.30%
Journée de la semaine dans Le Soleil	4.15%
Deuxième score Lead NER	3.29%

Table 1: Score des dix attributs les plus significatifs

L'attribut le plus significatif (voir tableau 1) est le premier score d'entité nommée

provenant du *lead* et le deuxième attribut le plus significatif est le premier score d'entité nommée provenant du titre. Ces attributs représentent le plus grand score d'une entité nommé présent dans leur section de l'article. La supposition initiale de l'équipe était que les entités nommées représentaient en soi des sujets d'article. Par exemple, si l'entité nommée Trump est présente alors il fait partie du sujet et le score de cet article sera similaire à celui des autres articles qui contiennent la mention de Trump. Cette supposition s'avère fort probablement exacte compte tenu de l'importance des attributs qui sont 7x et 4x supérieurs à la moyenne.

Les autres attributs importants proviennent du contenu, de l'auteur et du journal Le Soleil. Mise à part Le Soleil, qui doit avoir des tendances de lectures marquées par les saisons et les jours de la semaine, il n'est pas surprenant de voir le contenu et l'auteur au sommet des attributs. L'équipe a fait l'hypothèse lors du premier livrable que certaines personnes achètent le journal pour lire un certain journaliste. De plus, certains journalistes doivent être de meilleurs écrivains et être plus lus. L'équipe avait aussi mentionné dans le premier livrable que si un article est trop long ou trop court, cela allait jouer sur le temps de lecture. Un texte trop long peut intimider et un texte trop court n'a pas le temps de conserver le lecteur 60 secondes sur la page.

Une remarque intéressante est que dans le top 10 des attributs les plus importants, seul Le Soleil y retrouve des attributs spécifiques à un journal. Cela vient probablement du fait que Le Soleil est le journal le plus populaire parmi les 6 journaux concernés par ce projet. En effet, une enquête de Vividata [1] indique que Le Soleil fait partie des 5 journaux les plus lus au Québec. Aucun autre des 6 journaux de ce projet ne se retrouve dans ce top 5. Ainsi, il est logique que les articles du Soleil aient de meilleures cotes de popularité. Cette information est pratique pour la prédiction de la popularité d'un article en particulier, cependant, lors des recommandations aux auteurs, elle sera omise. En effet, il serait inconcevable de recommander à un auteur d'un autre quotidien de publier son article dans Le Soleil.

3.2 Recommandations

En regroupant les articles par catégorie, l'équipe obtient les 10% meilleurs et pires articles ainsi que leurs attributs. Ainsi, en se basant sur les attributs mentionnés dans la section précédente, l'équipe émet les recommandations suivantes aux auteurs.

3.2.1 Entités nommées

En premier lieu, les articles les plus faibles sont identifiés grâce aux attributs d'entités nommées provenant du *lead* et du titre ainsi qu'aux métadonnées des journaux. Chaque *lead* peut contenir jusqu'à 10 valeurs d'entités nommées et chaque titre peut en contenir jusqu'à 5, pourvu que l'algorithme les ait vue en entraînement.

[1] tel que rapporter par Infopresse: https://www.infopresse.com/article/2018/1/25/vividata-devoile-son-rapport-trimestriel-sur-le-lectorat-des-quotidiens-et-magazines

Il est facile de reconnaître les articles les plus faibles, car ceux-ci n'ont pratiquement jamais de valeur d'entités nommées. L'absence d'entité nommée implique que le système ne les reconnaît pas, car ce sont de nouveaux sujets.

Afin d'améliorer la popularité de leurs articles, les auteurs peuvent ajouter des entités nommées déjà comprises dans le système et qui sont en lien avec le sujet afin d'attirer l'attention du lecteur. En effet, les entités nommées reconnues par le système sont généralement des valeurs sûres dans leurs catégories, ce qui leur confère un certain gage de popularité. L'utilisation de ces entités nommées ou mots clés dans un article peut augmenter la visibilité et donc la popularité de cet article. Par exemple, si un auteur écrit un article sur la maison blanche, il peut être judicieux d'aller chercher les autres entités nommées en lien avec la maison blanche qui sont populaires (p.ex. *Trump, leader of the free world*). Cette technique doit cependant être utilisée avec parcimonie afin de ne pas surcharger le texte de mots clés inutiles.

3.2.2 Métadonnées des journaux

En deuxième lieu, les articles les moins populaires sont mis en ligne la nuit vers minuit. Ceci est en contraste avec les articles plus populaires qui sont mis en ligne tôt le matin. Il pourrait être avantageux de retarder la publication pour atteindre le 4h du matin qui semble idéal. Cependant, il ne faut pas oublier que les données de dates sont dans le fuseau horaire UTC. Le temps idéal dans le fuseau horaire québécois pour sortir un article selon l'algorithme serait donc entre 23h et 24h, et le pire entre 19h et 20h. À noter que ce résultat surprend l'équipe. il ne semble pas logique qu'un article performe mieux s'il est publié à minuit plutôt qu'à 19h. Ces résultats surprenants mènent à l'hypothèse que les heures de publications ont peut-être une valeur par défaut de minuit lorsque la vraie heure est non disponible. Dans tous les cas, cette information est donc à prendre avec garde.

La moyenne du nombre de mots dans le contenu varie aussi beaucoup entre le meilleur 10% et le pire 10%. Les articles les moins populaires ont en moyenne 321 mots dans leur section de contenu. En comparaison, la moyenne du nombre de mots du contenu des articles les plus populaires est de 707 mots. Ce qui implique en moyenne une différence de 385 mots. Le nombre de mots du contenu est le troisième attribut le plus important pour la forêt aléatoire avec une valeur d'importance 3.87 fois plus élevée que la moyenne des attributs. Ainsi, il semble important pour garder un lecteur sur la page longtemps que l'article soit assez long. Cela peut sembler logique, bien que la supposition initiale était qu'un article court susciterait plus l'intérêt des lecteurs. Un conseil pour améliorer la performance des articles non populaire est d'augmenter le nombre de mots de leur contenu afin d'atteindre une valeur similaire à celle des articles populaires.

4 Rétrospective

Les suppositions initiales de l'équipe étaient que les plus gros facteurs de prédiction de popularité seraient le nom de l'auteur, la longueur de l'article, la date de publication ainsi que les mots populaires dans le titre. Ces suppositions s'avèrent

vraies en partie. Le nom de l'auteur est le 6e attribut le plus significatif. La longueur du contenu est le 3e. Les mots populaires du titre ne font pas partie des 10 attributs les plus significatifs, mais ceux du *lead* le sont. La supposition qui ne semble pas être avérée est celle concernant la date de publication. En effet, seule la date des publications dans Le Soleil ressort comme un attribut fort, celles des autres journaux n'étant pas ou peu significatifs. Le raisonnement initial était que les lecteurs des journaux pourraient être plus enclins à lire des articles selon la période de l'année. Cependant, après réflexion, il semblerait plus judicieux de faire une association entre la date de publication et le thème de l'article. Ainsi, il serait possible de prédire, par exemple, la popularité d'un article sur le thème de Noël en hiver ou en été.

Une information qui a surpris l'équipe est l'importance des entités nommées. En effet, l'équipe avait prédit que le thème de l'article serait important, mais ne s'attendait pas à ce que le plus gros facteur de popularité soit la présence d'entités nommées fortes. En rétrospective, il semble logique que certains noms propres soient plus populaires auprès du public que d'autres. Ce qu'il y a d'intéressant est que la présence d'entité nommées dans le *lead* est plus importante que dans le titre. En général, il semble que le *lead* soit un plus gros indicateur de popularité que le titre. Ainsi, contrairement à ce que l'on pourrait penser, le lecteur est probablement plus accroché par le résumé de l'article au début de celui-ci que par son titre.

Finalement, certaines difficultés furent rencontrées au fil du projet. La plus grande difficulté fut d'extraire l'information des publications d'un article. En effet, dans le format actuel des données, un article peut avoir un nombre illimité de publications. Ainsi, il est impossible de construire un vecteur d'attributs de taille illimitée ou variable. Il a donc fallu trouver une solution pour extraire le plus d'attributs significatifs en perdant le moins d'information possible. La date de publication d'un article, par exemple, n'est pas si simple à déterminer. Si un article a été publié plusieurs fois, quelle date doit-on retenir? L'équipe a fait la supposition que la première date de publications est la plus intéressante, car c'est à partir de ce moment que l'article a commencé à amasser des vues. Cependant, lorsque certaines informations sont omises, il a toujours un risque de perdre des données intéressantes. Une autre solution aurait été de créer autant d'instances du même article que celui a de dates de publications. Ainsi, il y aurait une instance pour chaque date, ce qui réglerait le problème de perte d'information. Cependant, cette solution risque de biaiser le modèle en faveur des articles ayant plusieurs dates de publications si ceux-ci sont populaires, ce que l'équipe a jugé être un problème plus grave que la perte d'information.

En conclusion, le modèle proposé n'est pas parfait, mais semble être en mesure de distinguer les bons articles des mauvais dans l'ensemble. Les attributs les plus forts sont de bons indicateurs de popularité, et devraient être utilisés comme guide à la rédaction d'un article. Cependant, il est de l'avis de l'équipe que la recherche de popularité ne devrait jamais compromettre l'intégrité journalistique.

References

[1] Roja Bandari, Sitaram Asur, and Bernardo Huberman. The pulse of news in social media: Forecasting popularity. *ICWSM 2012 - Proceedings of the 6th International AAAI Conference on Weblogs and Social Media*, 02 2012.