Rapport 2

Équipe 21

Éric Cardinal	(111 227 625) 1er cycle (GLO-4027)
Frédéric Kassab	(111 258 828) 1er cycle (GLO-4027)
Denis Labrecque	(536 847 639) 2e cycle (GLO 7027)

Analyse et traitement de données massives

Travail présenté à
M. Richard Khoury

21 avril 2021

1 Introduction

L'objectif de ce projet est de fournir une aide à la rédaction afin d'identifier les articles qui obtiendront un bas pointage. Ce pointage est calculé via la formule fournie par CN2i, soit une pondération de : 1 point pour une vue de 5 secondes, 2 points pour une vue de 10 secondes, 5 points pour une vue de 30 secondes, et 10 points pour une vue de 60 secondes ou plus.

Ce rapport contient le détail du traitement de l'information, de l'extraction jusqu'à la prédiction. Un fichier texte est aussi joint à la remise, le 10% des articles avec le plus bas pointage pour chaque catégorie.

2 Algorithmes de préparation des données

2.1 Environnement et outils de développement

Le projet a été développé dans un environnement Python compte tenu des librairies disponibles et de sa prédominance pour le traitement des données. Les composantes suivantes ont été utilisées :

- *Pandas, Dask* : gestion et manipulation des données
- *Sklearn, Numpy* : algorithmes de classification (Random Forest, Naïve Bayes)
- *Seaborn, Mathplot* : production des graphiques
- *Spacy* : traitement de langage naturel (NLP)
- *TKinter :* création d'un outil graphique

2.2 Extraction des données

L'extraction des données a été effectuée à partir des fichiers de type JSON fournis dans cette étude. Les premières extractions ont été réalisées avec la librairie *Dask* afin de profiter de ses capacités de traitement parallèle. Une fois les ensembles de données réduits à une taille traitable en mémoire, elles ont été transférées dans différents tableaux (*Dataframes* Pandas*)* avec pour colonnes les attributs extraits.

Ce processus a permis de réaliser un gain d'espace, la taille des données passant du 120 Go initial à moins de 1 Go, seuls les attributs sélectionnés étant gardés. (À ce stade, l'équipe avait déjà procédé à l'élimination des attributs non-pertinents; le texte complet n'est pas conservé pour y appliquer le traitement de langage naturel.) Ce gain est très appréciable, car il permet de travailler avec les données en mémoire, accélère le traitement de l'information tel les modifications ou la jointure des tableaux, et facilite le processus d'apprentissage des modèles.

2.2.1 Métadonnées

Le premier livrable a mis en lumière une des difficultés du jeux de données. Les fichiers de publication contiennent un document JSON pour chaque journal et plateforme où un article a été publié. Puisque le pointage des articles est calculé au total des publications, il a été nécessaire d'en extraire la catégorie, qui varie d'un média à l'autre. L'attribut *canonical* avec la valeur *True* a été priorisée. On suppose que cet attribut fait référence au URL canoniques utilisés par les webmasters pour indiquer la réelle origine de l'article. La même chose est faite pour les métadonnées de la publication.

Ensuite, la date de publication est transformée en deux attributs : l'heure de publication, et le jour de la semaine. Le mois et le numéro de la semaine n'ont pas été sélectionnés, car les données analytiques

utilisées pour calculer le score d'un article ne sont disponibles que pour 7 mois seulement (de janvier à juillet inclusivement). Le classificateur serait ainsi faussement entraîné sur un intervalle de mois réduit dans l'année.

Puisque l'endroit où l'article est publié est important, douze attributs sont extraits pour signifier dans lesquels des six journaux un article est publié et s'il a été publié sur mobile et/ou le site (i.e : soleil-mobile, soleil-site, la-tribune-mobile etc.). Un 1 est stocké dans l'attribut si l'article est publié dans le journal en question et un 0 est laissé si ce n'est pas le cas.

Finalement, puisqu'un article peut avoir plus d'un auteur, seul le premier auteur a été retenu. Cette simplification a été choisie puisque la plupart des articles dans le jeu de données n'ont qu'un seul auteur (De Grandpré, Sirois, & Lindsay, 2019)

2.3 Préparation à l'entraînement

Un prétraitement important doit être effectué avant de fournir les données à un algorithme d'apprentissage machine.

Premièrement, une correction du bruit est effectuée. Un des attributs importants à corriger est la catégorie (*slug*). Plusieurs catégories n'apparaissent qu'à quelques occasions dans la base de données. Puisque le nombre de catégories uniques n'est pas élevé, un dictionnaire correctif a été créé manuellement. Cela permet de remplacer automatiquement une catégorie spécifique vers une catégorie plus générale. Voici quelques exemples :

- actualite, actualites => actualites
- autos, essais-routier => autos
- la-voix-de-saint-cesaire, la-voix-de-waterloo => regional,
- steve-turcotte, pyeonchang-2018 => sports
- Les catégories vides => n/a

Cela élimine les catégories avec moins d'occurrences tout en gardant un certain degré d'information par rapport à la catégorie initiale (i.e. des essais routiers plairont probablement à des amateurs de la catégorie *autos*).

De plus, le format actuel des attributs doit être homogénéisé. Certains attributs sont catégoriques (*slug*) et d'autres sont numériques. Tous les attributs sélectionnés doivent être passés sous la forme numérique pour les algorithmes d'apprentissage de la librairie *scikit-learn*.

Les attributs numériques sont simplement normalisés par la méthode centrer-réduire à l'aide de la classe *StandardScaler* de la librairie *scikit-learn.*

Les attributs catégoriques sont traités par un encodage du type *OneHotEncoder*. Ceci a pour effet de créer une matrice creuse (*sparse matrix*) où une colonne est replacée par autant de colonnes qu'il y a d'attributs uniques. Ainsi, une colonne *Ville* avec les données *Québec, Montréal* sera remplacée par deux colonnes, *Ville_Québec* et *Ville_Montréal*. Chacune recevra 0 ou 1 selon la présence de l'attribut. Ce traitement a été choisi au profit d'une simple association d'une catégorie à un chiffre puisque la magnitude du chiffre pourrait influencer la prédiction. Cette technique aurait été d'avantage appropriée pour des attributs

ordinaux. Il est important de noter que ce choix aura un impact sur les résultats et leur interprétation puisque le nombre d'attributs a été décuplé.

2.4 Données textuelles

Une première itération a été effectuée avec les attributs mentionnés précédemment. Le classificateur choisi pour cette première itération est un algorithme de forêt aléatoire de régression qui est décrit à la section 2.5. En mettant de côté 20% des données pour quantifier la performance de l'algorithme et avec une profondeur maximale d'arbre initiale de 10, un score R^2 de 0.50 a été obtenu. En interprétant ce score, l'équipe conclut que les attributs sélectionnés sont assez significatifs pour obtenir une prédiction correcte, mais qu'il y a place à l'amélioration. Aucun attribut provenant du contenu du texte n'a été utilisé pour la première itération.

Le contenu de l'article se résume surtout à ses *paragraphes*, à son *résumé* et à son *titre (chapters, lead,* et *title)*. Pour la deuxième itération, l'équipe s'est concentré sur l'extraction des lemmes du titre à l'aide de la librarie Spacy. Une fois les lemmes du titre extraits et les mots vides (*stop words*) enlevés, le score de l'article est associé à chaque mot. Ensuite, la moyenne des scores des articles contenant de ce mot et le nombre d'occurrences du mot sont stockés dans un dictionnaire.

2.5 Algorithme prédictif

2.5.1 Choix de l'algorithme prédictif

Deux algorithmes ont été considérés pour l'analyse finale. À la suite des notions vues en classe, un algorithme de forêt aléatoire de régression (*Random Forest*) et un classificateur Naïf de Bayes ont été mis en place avec un ensemble de critères restreints. Très rapidement, les résultats des coefficients de détermination, R^2, du *Random Forest* se sont démarqués en atteignant 0.5 à la première itération. Un classificateur Naïf de Bayes a également été entraîné sur les scores séparés en une centaine de classes (*bins*) de fréquence égale. L'algorithme de Naïve Bayes n'a pas offert les résultats escomptés en conservant une précision sous la barre de 0.2. L'algorithme de forêt aléatoire a donc été retenu.

2.5.2 Tests et recherche des paramètres pour l'entrainement du modèle

L'entraînement de la forêt aléatoire de régression nécessite moins d'une minute. L'entraînement s'effectue rapidement, car le nombre d'articles possédant un score est d'environ 35 000, et le prétraitement a déjà été effectué, comme expliqué à la section 2.1. Le temps d'extraction des données brutes avec la librairie *Dask* prend environ 5 heures avec quatre « *concurrent workers* ». Les opérations les plus coûteuses en temps de calcul sont le processus de lemmatisation du texte et la lecture sur disque des fichiers JSON.

La qualité de la prédiction faite par l'algorithme à chaque itération est déterminée à l'aide de la fonction *score* de l'objet *RandomForestRegressor*. Cette métrique est le coefficient de détermination R^2 mesuré sur le jeu de données mis de côté (20% des données). Une prédiction parfaite donnerait un R^2 de 1, ce qui n'est pas atteignable dans notre cas.

À l'itération 1, lorsque seulement les métadonnées de l'article étaient incluses dans les attributs d'entraînement, le R^2 grimpait à 0.50. Ce score était attendu, car les métadonnées d'un article permettent partiellement d'en prédire sa popularité. Par exemple, un article d'un auteur connu influence sa

popularité. Cependant, l'intuition demeure que le contenu d'un article et de son titre aura plus d'influence sur sa popularité.

L'ajout du score moyen des lemmes constituant le titre d'un article, ainsi que le nombre d'entités nommée a permis d'améliorer le R^2 à 0.68. Le score moyen des lemmes devient l'attribut le plus important, et de loin. L'intuition est que la forêt aléatoire fera du surapprentissage sur cet attribut. Le dictionnaire de la moyenne du score associé à chaque lemme est calculé à l'aide de l'ensemble des données. Lors de l'entraînement, les données d'entraînement et les données de test sont séparées (80%/20%) pour le calcul du R^2 sur les données de test. Or, les données de test ont tout de même contribué à établir le score moyen des mots du dictionnaire, ce qui biaise favorablement la qualité de la prédiction pour ces articles. Il faudrait calculer le dictionnaire d'après les données d'entraînement seulement (80% des données) pour avoir un calcul de R^2 non biaisé.

Guidé par les bons résultats de l'itération 2, le nombre et le score moyen des mots populaires du résumé (*lead*) de l'article ont été ajoutés aux attributs dans la troisième itération. De plus, les dictionnaires utilisés pour le calcul du score moyen des lemmes ont été filtrés pour ne contenir que les mots ayant plus de 10 occurrences et ayant un score supérieur à 50 000, à un écart-type au-dessus de la moyenne environ. Ce filtrage a été effectué pour ne garder que les mots populaires, réduisant ainsi les mots plus rares qui biaiseraient le score moyen des lemmes. Cela a légèrement réduit le coefficient de détermination; cependant, l'équipe fait l'hypothèse que cette itération obtiendra de meilleures prédictions sur de nouveaux articles.

2.5.3 Entraînement et hyperparamètres

Une fois la sélection de l'algorithme complété et les attributs choisis, les prochaines étapes ont été la sélection des hyperparamètres d'entrainement du modèle. Cette fois-ci, l'analyse des résultats s'est effectuée sur deux critères : le coefficient de détermination et le temps d'exécution requi, l'objectif de l'exercice étant de trouver une combinaison idéale qui maximise le score R^2 tout en minimisant le temps d'exécution.

Dans un premier temps, les hyperparamètres considérés ont été la profondeur maximale (*maxDepth*), le nombre d'arbres (*n_estimators*) dans la forêt, et le nombre maximum d'attributs à considérer à chaque *split* (*max_features*). Les données présentées à la Figure 1 et à la Figure 2 (Section 8 : Annexe) démontrent l'impact des changements. On constate les effets suivants :

1. L'augmentation de la profondeur augmente le coefficient de détermination, mais augmente aussi le temps d'exécution. Il faut éviter d'augmenter la profondeur de l'arbre indéfiniment, car le risque surapprentissage s'accroît dans un arbre trop profond.
2. L'augmentation du nombre d'attributs dans l'analyse augmente le coefficient de détermination; cependant, au-delà d'un certain nombre (près de 500 dans notre cas), le gain devient nul et on constate une simple augmentation du temps d'exécution.
3. Enfin, le nombre d'arbres semble avoir peu d'impact sur le coefficient de détermination. Cependant, le temps d'exécution augmente de manière importante.

À l'itération subséquente, de nouveaux hyperparamètres ont été ajoutés: le nombre minimum d'échantillons pour le partitionnement d'un nœud (*min_samples_split*), et le nombre minimum d'échantillons à la fin de la branche (*min_samples_leaf*). Finalement, avec les paramètres énumérés au Tableau 1, le coefficient de détermination obtenu a été de 0.65 en 9 secondes. Ces données représentent

une différence de 0.02 du coefficient de détermination original pour un temps de calcul deux fois plus long.

Tableau 1: Sélection finale des hyperparamètres

Hyperparamètres	Valeurs
maxDepth	18
n_estimators	100
max_features	750
min_samples_split	5
min_samples_leaf	2

Le temps de calcul a été obtenu au moyen d'un processeur i7-8770k, 6-cœurs/12-threads et 32 Go de mémoire. Au moment du calcul, *Python/scitkit-learn* a maximisé l'utilisation du processeur avec 12 travailleurs concurrents (*concurrent workers*).

3 Attributs importants

Le tableau ci-dessous affiche les résultats des facteurs d'importance de Gini provenant de la fonction *Feature_Importances_* de la librairie *scikit-learn*. Selon la documentation du cadriciel, le résultat de cette fonction représente la baisse d'impureté de l'attribut. Le Tableau 2 présente les dix premières positions.

Dans un premier temps, les données semblent confirmer une hypothèse posée lors du premier rapport: la plateforme de diffusion est très importante. Ainsi, le journal *Le Soleil* avait le plus grand lectorat[1] en 2019, soit 845 000; tandis que *Le Droit*, son plus proche rival, n'a que 217 000 lecteurs hebdomadaires. On constate donc plusieurs attributs de type Journal-Plateforme qui reçoivent un compte élevé. Il demeure cependant surprenant que des journaux avec un plus petit tirage semblent avoir un plus grand effet que *Le Droit*.

Tableau 2: Top 10 des attributs du modèle

Attribut	Facteur d'importance de Gini
lesoleil-mobile	0.196
art.lead.lemma_score	0.108
lesoleil-site	0.084
art.title.lemma_score	0.068
latribune-site	0.055
art.lead.lemma_populaire_count	0.046
lequotidien-mobile	0.037
p.publication.section_actualites	0.037

[1] Nombre de lecteurs des quotidiens québécois selon la propriété au Québec (2019). Il s'agit du nombre de personnes ayant feuilleté ou lu au moins une édition pendant une semaine de publication. https://www.cem.ulaval.ca/economie/propriete/presse-quotidienne/

art.paragraph_count	0.030
lenouvelliste-site	0.029
latribune-mobile	0.023

Une seconde hypothèse avancée dans le premier rapport qui semble aussi se confirmer est l'importance de l'analyse linguistique (NLP). On remarque trois attributs liés à la linguistique (art.lead.lemma_score, art.title.lemma_score, art.lead.lemma_populaire_count) sont présents dans cette liste. En effet, l'hypothèse que le contenu titre et le résumé d'un article permettent de différencier entre un article populaire et impopulaire est validée. Cependant, il n'est pas surprenant que les attributs linguistiques aient un facteur d'importance élevé, car ils sont calculés à partir du score des exemples d'entraînement.

Notez que plusieurs hypothèses n'ont pu être validées avec ce tableau dû à l'utilisation de l'encodage du type *OneHotEncoder*. Tel que décrit précédemment, les attributs catégoriques ont été transformés en autant d'attributs individuels avec un 0 ou un 1 pour marquer leur présence. Ainsi, l'attribut *section* (sports, actualites, etc.) a généré plusieurs nouveaux attributs (section_sports, section_actualites, etc.). Chaque nouvel attribut a donc son propre facteur d'importance.

Dans la liste complète des attributs importants, la présence de certains auteurs comme Marc Allard ou Richard Therrien a un facteur d'importance élevé. Cette observation valide le fait que certains auteurs ont un lectorat fidèle et ont un impact sur la popularité d'un article.

Finalement, le nombre de paragraphes de l'article fait partie des 20 attributs les plus importants. L'algorithme fait donc la distinction entre des entrefilets qui se lisent rapidement et les articles d'une longueur adéquate qui sont attirants à lire pendant plus de 60 secondes (View60). Puisque les données brutes contiennent des balises * * de paragraphes HTML, il faut d'abord lire tout le texte et nettoyer les balises pour en obtenir le compte exact.

4 Étude de cas

Le Tableau 3 montre les mesures statistiques du score réel et du score prédit pour les données de test (20% des données).

Tableau 3: Description statistique des score réels et prédits sur les données de test

	Score réel	Score prédit
Moyenne	20 680	20 388
Écart-type	26 666	18 837
Minimum	4	1 869
Maximum	481 983	202 365
Écart interquartile	21 433	24 099

À première vue l'algorithme semble surestimer les articles très impopulaires et sous-estimer les articles viraux puisque l'écart entre le score minimum et le score maximum est inférieur au réel. Cependant, la moyenne et l'écart interquartile est similaire entre le score réel et le score prédit. C'est donc dans la capacité de gérer les cas extrêmes (en bas du 25^{e} centile et en haut du 75^{e}) que l'algorithme éprouve des difficultés.

4.1 Bonnes prédictions

Après un survol des prédictions des données de tests, trois articles ont été sélectionnés pour démontrer de bonnes prédictions.

Tableau 4: Étude de cas des bonnes prédictions

Titre	Score réel	Score prédit	Titre score mots	Lead score mots	Lead nombre mots pop.	Section	Auteur
Fillette agressée au McDonald's: trois accusations déposées contre un Granbyen	211 551	202 366	104 273	98 642	23	actualites	Karine Blanchard
«La voix»: un premier blocage et Corey Hart	117 133	123 452	135 605	77 797	23	arts	Richard Therrien
Aidez-nous à vous offrir une infolettre encore plus à votre goût	2 067	2 179	53 690	56 399	15	la-vitrine	Groupe Capitales Médias

Les deux premiers articles du tableau sont des articles très populaires. Dans les deux cas, ils ont été publiés dans les six journaux, dont le journal *Le soleil*, celui avec le plus grand lectorat. De plus, ils ont été écrits par des auteurs qui attirent le lectorat. Finalement, leur popularité est caractérisée par la moyenne et le nombre élevé de mots populaires du lead et du titre.

L'algorithme démontre aussi que le dernier article, un sondage du Groupe Capitales Médias, n'est pas très populaire. L'auteur, Groupes Capitales Médias, n'est pas très populaire et ne publie pas fréquemment (moyenne de 18 420 sur 3 articles publiés). De plus, les mots utilisés dans le titre et le lead ont une moyenne de scores peu élevés.

4.2 Mauvaises Prédictions

Dans certains cas, l'algorithme fait des erreurs de grande magnitude. L'erreur s'accroît particulièrement pour les articles avec des scores élevés, comme le montre le premier exemple du Tableau 5.

Tableau 5: Étude de cas des mauvaises prédictions

Titre	Score réel	Score prédit	Titre score mots	Lead Score mots populaires	Lead nb de mots populaires	Section	Auteur
Une camionnette percute des motos au New Hampshire: 7 morts [PHOTOS]	481 983	173 459	144 797	109 314	10	actualites	David Sharp

Musée des plaines d'Abraham	202	55 186	70 639	77 666	11	mission	Capitales Studio

Dans le premier exemple, tous les attributs sont indicateurs d'un article populaire (score > 100 000). Cependant, l'auteur est inconnu de l'algorithme, car les données d'entraînement ne contiennent pas l'auteur David Sharp. Puisque cet auteur fait partie de l'agence *Associated Press*, il aurait été préférable de stocker le nom de l'agence au lieu de l'auteur pour l'entraînement. Le lead est également plutôt court, ce qui fait en sorte que le nombre de mots populaires dans le lead est bas et le score est sous-estimé. Une solution serait d'inclure la longueur du lead dans les attributs pour que l'algorithme prenne en considération cette variable. Malgré l'erreur de prédiction élevée, l'algorithme a quand même été en mesure de prédire que l'article atteindrait un bon degré de popularité. En effet, le score prédit se situe au-dessus du 99^{e} centile des scores de la base de données (118 000). La rare occurrence des articles viraux dans le jeu de données fait en sorte qu'il est difficile d'en prédire le score réel.

Pour le deuxième exemple, il s'agit d'un contenu sponsorisé inclus dans les 6 publications. Le *lead* possède une moyenne de mots populaires élevée. Puisque l'algorithme a tendance à prédire un score élevé pour les moyennes de mots populaires élevés et pour les articles publiés dans *Le Soleil*, le score réel est grandement surestimé. Une solution serait d'obtenir plus de données concernant le contenu sponsorisé dans la base de données, puisque le contenu sponsorisé traite des sujets différents que les sujets d'actualité, qui constituent la plupart des articles du jeu de données.

5 Recommandations pour l'aide à la rédaction

Puisque l'algorithme utilise un grand nombre d'attributs, et que ce nombre d'attributs a été décuplé par l'utilisation d'un encodeur du type *OneHotEncoder*, l'interprétation des arbres par un humain est très difficile. Cependant, il est possible d'en tirer des conclusions et émettre des recommandations à partir des attributs importants trouvés à la section 3.

5.1 Contenu de l'article

Le titre et le lead sont de bons indicateurs de la popularité d'un article, c'est pourquoi il faut porter une attention particulière aux mots clés inclus dans ces sections. Afin d'améliorer la popularité d'un article, le dictionnaire des mots populaires pourrait être mis à la disposition de l'auteur pour guider la rédaction du titre et du résumé. De plus, à première vue il peut être payant en termes de popularité d'inclure au moins une entité nommée telle qu'une localisation (LOC), une personne (PER), un organisme (ORG) ou autre (MISC) dans le titre pour attirer le lecteur. Cependant, les données présentées dans le Tableau 6 ne sont pas concluantes. Pour aider la rédaction, il aurait fallu effectuer une analyse sur les entités nommées similaires à l'analyse des mots populaires. Ainsi un dictionnaire d'entités nommées populaires serait fourni à l'auteur pour ne cibler que les entités qui ont un impact significatif sur le lectorat (tel le nom d'un ministre ou d'une célébrité).

Tableau 6: Score moyen par nombre d'entités nommées dans le titre

Nombre total d'entités nommées (titre)	Score réel moyen	Nombre d'articles
0	21 262	9 000
1	19 976	17 067
2	20 188	9 001

3	21 136	2 452
4	24 132	512
5	27 454	82
6	11 063	3
7	17 342	2

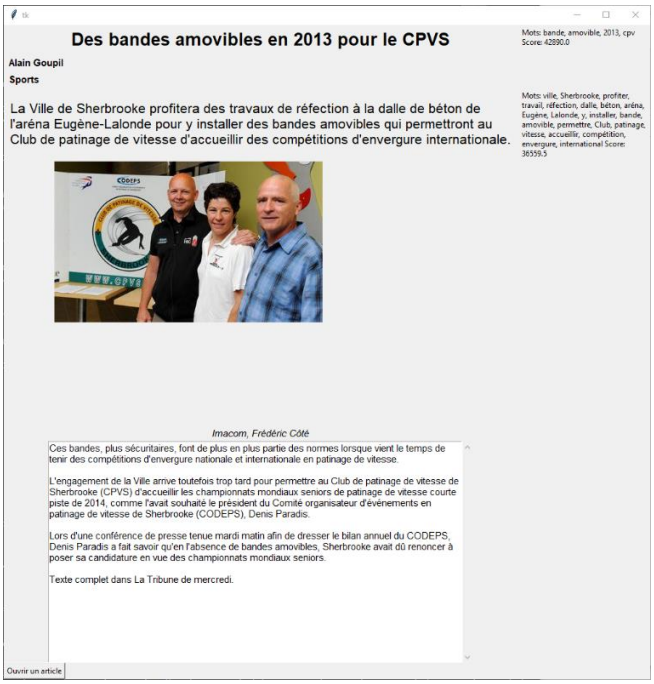
5.2 Métadonnées

Le jour de la semaine à laquelle un article est publié impacte légèrement le score. La tendance est que les articles publiés le lundi et le dimanche sont légèrement plus populaires par rapport à leur médiane, tandis que le samedi et le vendredi sont les journées les moins populaire auprès des lecteurs. En termes d'heures, un certain bruit dans les données est présent. Plusieurs articles possèdent une heure de publication de minuit (0:00), ce qui semble indiquer la présence d'une valeur par défaut plutôt qu'une vraie heure. De plus, l'équipe fait l'hypothèse que l'heure est stockée dans le fuseau horaire UTC. Le lundi et le dimanche tôt en matinée ainsi que le samedi tard en soirée semblent être le meilleur moment pour publier un article. Cela indique que les lecteurs font le tour des nouvelles de la fin de semaine, le dimanche matin. C'est pourquoi il est préférable de publier les articles dans une fenêtre entre le samedi soir et le dimanche matin afin de maximiser la popularité d'un article donnée.

La longueur de l'article peut aussi influencer sa popularité. En effet, les 10% des articles populaires contiennent une médiane de 13 paragraphes et les 10% pires ont une médiane de 7 paragraphes. L'hypothèse est qu'un article plus long garde le lecteur un peu plus longtemps que 60 secondes (événement View60), ce qui permet de faire augmenter le score.

5.3 Outil graphique

Un outil graphique permettant de visualiser un document d'article JSON a été créé pour visualiser les résultats prédits des articles.



Cet outil affiche les lemmes trouvés dans le titre et dans le *lead*, tout en affichant la moyenne des scores de ces mots. L'outil nettoie aussi les balises HTML pour mieux présenter les paragraphes. L'idée future serait de noter en détail les effets des divers attributs sur le score. En expérimentant avec différents mots, différentes longueurs du texte, différentes sections du journal, et ainsi de suite, l'auteur pourrait alors maximiser le score prévu de son article.

6 Rétrospective

6.1 Pistes d'amélioration

Lors de la réalisation de cette analyse, certaines avenues n'ont pas été considérées par manque de temps et certaines corrections auraient pu être apportées.

6.1.1 K-Fold Cross Validation

Le processus de validation croisée de k-échantillons (K-Fold Cross Validation) aurait été une méthode plus adéquate pour la sélection des hyperparamètres et les attributs de l'arbre. Cette méthode consiste à partitionner les données d'entraînement en k-partitions en testant sur une seule partition pendant k-itérations. En prenant la moyenne des résultats des k-itérations, cette méthodologie permet de sélectionner les hyperparamètres et les attributs qui minimisent le surapprentissage.

6.1.2 Catégorisation des attributs numériques

Puisque plusieurs valeurs possibles du nombre de paragraphes sont présentes dans le jeu de données, il y a lieu de croire que la différence de pointage entre deux textes ne pourrait être due qu'à la simple différence entre 6 ou 7 paragraphes. Il y aurait lieu de séparer le nombre de paragraphes en classes (*bins*) d'intervalle égal ([1 à 5][6 à 10][11 à 15] paragraphes) pour signifier un article court, moyen et long. L'hypothèse est qu'en classifiant de la sorte la longueur de l'article, le nombre de paragraphes deviendra un attribut plus important, permettant de réduire l'erreur sur la prédiction.

6.1.3 Augmentation des tâches de prétraitement

Certains attributs similaires à la catégorie (*author*, *channel*) auraient bénéficié d'un prétraitement afin d'en standardiser les données. Un traitement similaire à la catégorie (*slug*) aurait été bénéfique pour réduire le bruit de ces attributs, améliorant ainsi la prédiction.

6.1.4 Attributs omis

Certains attributs prometteurs, tels l'inclusion de points d'interrogation ou d'exclamation dans le titre et le sous-titre, ou bien l'analyse NLP complète du texte, n'ont pu être retenus à cause de contraintes en temps. Cependant, leur ajout améliorerait la prédiction.

6.2 Conclusion

Conformément aux hypothèses posées par l'équipe, les mots de l'article (lemmes), sa catégorie, le journal, et la présence de certains auteurs populaires sont des attributs importants dans la prédiction de la popularité d'un article. Par contre, l'équipe se serait attendu à voir un impact un peu plus élevé à propos de l'heure et le jour de la semaine auquel un article est publié. Finalement, l'impact de *l'attribut lesoleil-mobile* est beaucoup plus significatif que l'équipe aurait prévu, puisqu'il est l'attribut le plus significatif. Initialement, la gestion d'une taille massive des données a été un défi résolu par l'utilisation de librairies de parallélisation et par la pré-sélection d'attributs pertinents.

7 Bibliographie

De Grandpré, G. D., Sirois, A., & Lindsay, X. (2019). Analyse et prétraitement des données sur les journaux de capitales médias. *GLO-7027*, 12.

8 Annexe

Figure 1: Variation du coefficient de corrélation en fonction du nombre d'attributs et arbres

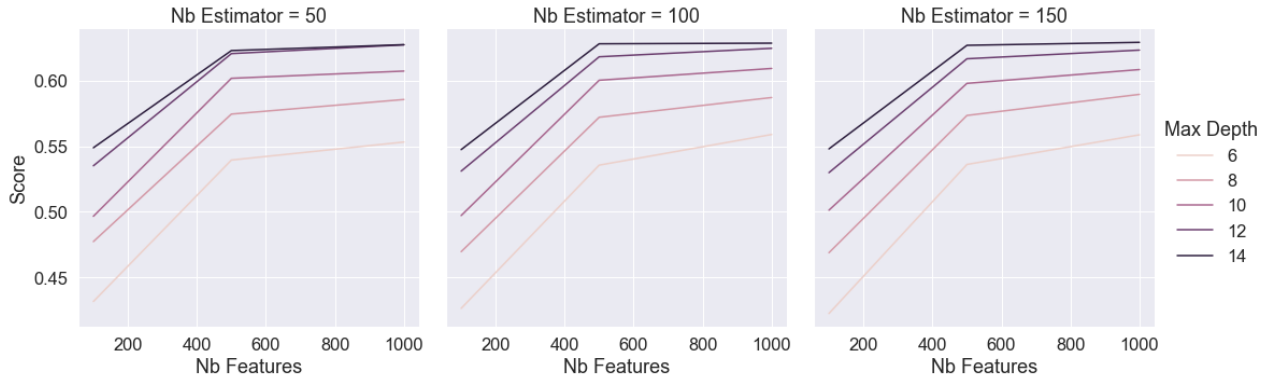
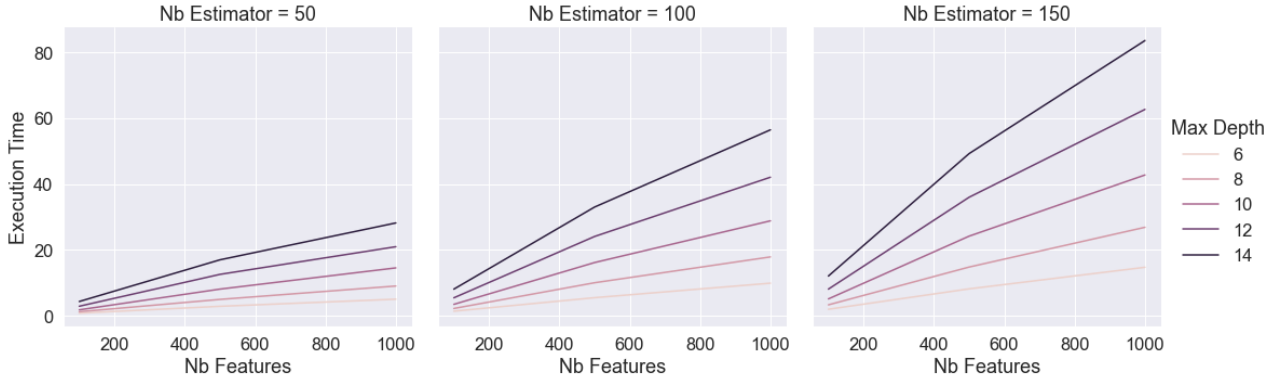


Figure 2: Variation du temps d'exécution en fonction du nombre d'attributs et arbres



8.1 Liste des attributs

Tableau 7: Liste des attributs utilisés pour entraîner la forêt aléatoire

Attribut	Type/ Encoder	Provenance
'art.mainAuthor',	Categorical oneHotEncoder	Article (JSON)
'p.publication.section',	Categorical oneHotEncoder	Publication (JSON) Extrait du *slug*
'art.templateName',	Categorical oneHotEncoder	Article (JSON)
'art.channel',	Categorical oneHotEncoder	Article (JSON)
'p.publications.weekday',	Categorical oneHotEncoder	Publication (JSON) Extrait de la date
'art.visual_type'	Categorical oneHotEncoder	Article (JSON)
'latribune-site', 'latribune-mobile', 'ledroit-site', 'ledroit-mobile', 'lavoixdelest-site', 'lavoixdelest-mobile', 'lequotidien-site', 'lequotidien-mobile', 'lenouvelliste-site', 'lenouvelliste-mobile', 'lesoleil-site', 'lesoleil-mobile'	Numeric/ StandardScaler	Publication (JSON) Traitement pour l'extraction des journaux et plateformes utilisés pour la distribution
'art.lead.lemma_score', 'art.lead.lemma_populaire_count',	Numeric/ StandardScaler	Articles (JSON) Score et compte basé sur les articles connus
'art.author_count', 'art.paragraph_count', 'art.quote_count', 'art.photo_count',	Numeric/ StandardScaler	Articles (JSON)
'art.title.ner_PER', 'art.title.ner_MISC', 'art.title.ner_ORG', 'art.title.ner_LOC', 'art.title.lemma_score'	Numeric/ StandardScaler	Articles (JSON) Traitement NLP pour l'extraction des mots