Université Paris-Saclay

Master 2 – Innovation, Marchés et

Sciences des Données (IMSD)

Projet Data Mining

Thierry GAMEIRO MARTINS Romain KRAWCZYK Samir LAZZALI

Daniel KHARSA Mohamed-Amine BOUREGA

Année universitaire : 2019 - 2020

Table des matières

1 Présentation du projet

1.1 Description du jeu de données initial

Pour notre projet de Data Mining, nous avons choisi d'utiliser la base « Online Popularity News », disponible en libre accès ici. La base de données contient un ensemble d'informations relatives aux articles publiés sur le site Mashable entre janvier 2013 et janvier 2015. La base contient 39644 observations et 61 colonnes. Voici un extrait du jeu de données (des 12 premières variables et des 6 dernières) :

	url	timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words
0	http ://mashable.com/2013/01/07/amazon-instant-...	731.0	12.0	219.0	0.663594	1.0
1	http ://mashable.com/2013/01/07/ap-samsung-spon...	731.0	9.0	255.0	0.604743	1.0
2	http ://mashable.com/2013/01/07/apple-40-billio...	731.0	9.0	211.0	0.575130	1.0
3	http ://mashable.com/2013/01/07/astronaut-notre...	731.0	9.0	531.0	0.503788	1.0
4	http ://mashable.com/2013/01/07/att-u-verse-apps/	731.0	13.0	1072.0	0.415646	1.0

	n_non_stop_unique_tokens	num_hrefs	num_self_hrefs	num_imgs	num_videos	average_token_length
0	0.815385	4.0	2.0	1.0	0.0	4.680365
1	0.791946	3.0	1.0	1.0	0.0	4.913725
2	0.663866	3.0	1.0	1.0	0.0	4.393365
3	0.665635	9.0	0.0	1.0	0.0	4.404896
4	0.540890	19.0	19.0	20.0	0.0	4.682836

	max_negative_polarity	title_subjectivity	title_sentiment_polarity	abs_title_subjectivity	abs_title_sentiment_polarity	shares
0	-0.200000	0.500000	-0.187500	0.000000	0.187500	593
1	-0.100000	0.000000	0.000000	0.500000	0.000000	711
2	-0.133333	0.000000	0.000000	0.500000	0.000000	1500
3	-0.166667	0.000000	0.000000	0.500000	0.000000	1200
4	-0.050000	0.454545	0.136364	0.045455	0.136364	505

Cette base de données a été construite dans le cadre de travaux cherchant à prédire la popularité d'un article à partir de ses caractéristiques. Les différentes variables contenues dans cette base ont été créées en utilisant des méthodes d'analyse naturelle du langage (NLP). La liste des variables et leur signification est donnée ci-dessous :

- **url** : URL of the article
- **timedelta** : Days between the article publication and the dataset acquisition
- **n_tokens_title** : Number of words in the title
- **n_tokens_content** : Number of words in the content
- **n_unique_tokens** : Rate of unique words in the content
- **n_non_stop_words** : Rate of non-stop words in the content
- **n_non_stop_unique_tokens** : Rate of unique non-stop words in the content
- **num_hrefs** : Number of links
- **num_self_hrefs** : Number of links to other articles published by Mashable
- **num_imgs** : Number of images
- **num_videos** : Number of videos
- **average_token_length** : Average length of the words in the content
- **num_keywords** : Number of keywords in the metadata

— **data_channel_is_lifestyle** : Is data channel 'Lifestyle' ?
— **data_channel_is_entertainment** : Is data channel 'Entertainment' ?
— **data_channel_is_bus** : Is data channel 'Business' ?
— **data_channel_is_socmed** : Is data channel 'Social Media' ?
— **data_channel_is_tech** : Is data channel 'Tech' ?
— **data_channel_is_world** : Is data channel 'World' ?
— **kw_min_min** : Worst keyword (min. shares)
— **kw_max_min** : Worst keyword (max. shares)
— **kw_avg_min** : Worst keyword (avg. shares)
— **kw_min_max** : Best keyword (min. shares)
— **kw_max_max** : Best keyword (max. shares)
— **kw_avg_max** : Best keyword (avg. shares)
— **kw_min_avg** : Avg. keyword (min. shares)
— **kw_max_avg** : Avg. keyword (max. shares)
— **kw_avg_avg** : Avg. keyword (avg. shares)
— **self_reference_min_shares** : Min. shares of referenced articles in Mashable
— **self_reference_max_shares** : Max. shares of referenced articles in Mashable
— **self_reference_avg_sharess** : Avg. shares of referenced articles in Mashable
— **weekday_is_monday** : Was the article published on a Monday ?
— **weekday_is_tuesday** : Was the article published on a Tuesday ?
— **weekday_is_wednesday** : Was the article published on a Wednesday ?
— **weekday_is_thursday** : Was the article published on a Thursday ?
— **weekday_is_friday** : Was the article published on a Friday ?
— **weekday_is_saturday** : Was the article published on a Saturday ?
— **weekday_is_sunday** : Was the article published on a Sunday ?
— **is_weekend** : Was the article published on the weekend ?
— **LDA_00** : Closeness to LDA topic 0
— **LDA_01** : Closeness to LDA topic 1
— **LDA_02** : Closeness to LDA topic 2
— **LDA_03** : Closeness to LDA topic 3
— **LDA_04** : Closeness to LDA topic 4
— **global_subjectivity** : Text subjectivity
— **global_sentiment_polarity** : Text sentiment polarity
— **global_rate_positive_words** : Rate of positive words in the content
— **global_rate_negative_words** : Rate of negative words in the content
— **rate_positive_words** : Rate of positive words among non-neutral tokens
— **rate_negative_words** : Rate of negative words among non-neutral tokens
— **avg_positive_polarity** : Avg. polarity of positive words
— **min_positive_polarity** : Min. polarity of positive words
— **max_positive_polarity** : Max. polarity of positive words
— **avg_negative_polarity** : Avg. polarity of negative words
— **min_negative_polarity** : Min. polarity of negative words

— **max_negative_polarity** : Max. polarity of negative words
— **title_subjectivity** : Title subjectivity
— **title_sentiment_polarity** : Title polarity
— **abs_title_subjectivity** : Absolute subjectivity level
— **abs_title_sentiment_polarity** : Absolute polarity level
— **shares** : Number of shares

La dernière variable (*shares*) correspond au nombre de partages sur les réseaux sociaux (Facebook, Twitter, Google+, LinkedIn, Stumble-Upon et Pinterest), c'est la variable que nous essaierons de prédire dans ce qui suit. La première variable (*url*) correspond à l'URL de chaque article. Cette variable nous permettra de reconstruire la base de données dans son intégralité. Sachant que les autres variables ont été construites en utilisant des méthodes d'analyse naturelle du langage, un certain nombre de remarques préalables doivent être énoncées :

— les stopwords correspondent à ce que l'on appelle des mots vides, c'est-à-dire des mots tellement communs que leur prise en compte n'est pas requise (les mots tels que « la » , « de », « un » en sont des exemples).
— Les créateurs de la base ont distingués sept catégories auxquelles les articles peuvent appartenir, ces catégories sont désignées par le mot channel (business, tech, lifestyle etc.) et sont représentées par les variables commençant par *data_channel*.
— un algorithme LDA (Latent Dirichlet Allocation) a été utilisé sur l'ensemble des articles afin de mettre en évidence cinq sujets récurrents et mesurer la proximité des chaque article à ces cinq thèmes. Les variables correspondantes commencent par *LDA_*.
— la polarité fait référence au degré de négativité ou de positivité d'un énoncé, tandis que la subjectivité fait référence au degré de jugement, d'émotion ou d'implication personnelle de l'auteur. Ces notions sont capturées par les variables se terminant par *_polarity* et *_subjectivity*.

Notre objectif est de construire une base de données similaire à celle que nous venons décrire. Nous recommencerons l'analyse précédente dans son intégralité et nous la comparerons avec le jeu de données initial lorsque cela est possible. Pour ce faire, nous collecterons les différentes informations (le contenu des articles et leur titre) grâce aux URL de la base de données initiale. Notons bien que l'objectif n'est pas de reproduire exactement la base de données déjà disponible en libre accès (cela aurait peu d'intérêt) mais plutôt de construire une base de données alternative en utilisant également des méthodes de NLP.

1.2 Problématique

Notre objectif pour ce projet consiste à créer un modèle de prédiction concernant la popularité d'un article. Pour cela, nous produirons d'abord une analyse descriptive pour déterminer quels types d'articles sont les plus populaires : les articles négatifs ou positifs ? Les articles subjectifs ou objectifs ? Est-ce qu'un thème particulier est plus populaire qu'un autre ? Dans une seconde partie, nous intégrerons une analyse prédictive. Celle-ci consiste d'abord en un problème de ré-

gression : tenter de prédire le nombre de partages d'un nouvel article. Ensuite, nous passerons à un problème de classification après avoir discrétisé notre variable cible en deux classes : « populaire » et « non populaire ». Avant d'en arriver là, il a été nécessaire de récupérer et nettoyer les données, puis de procéder à la création de thème et à l'analyse de sentiment.

2 Collecte, nettoyage et préparation des données

2.1 Récupération des articles et sélection des variables

La première étape de ce projet consiste donc à collecter les données nécessaires afin de construire notre propre base de données. Nous avons donc utiliser les URL des articles pour récupérer les variables suivantes : le contenu de l'article, le titre de l'article, le nombre d'images présentes dans l'article, le nom de l'auteur de l'article ainsi que la classification de l'article selon le site Mashable (Lifestyle, Entertainment, Business etc.). Voici un extrait du jeu de données collecté :

	url	chanel
0	http ://mashable.com/2013/01/07/ap-samsung-spon...	Business
1	http ://mashable.com/2013/01/07/apple-40-billio...	Business

	title	nb_images	author_name
0	AP's Twitter to Begin Displaying Sponsored Tweets	1	Seth Fiegerman
1	Apple's App Store Passes 40 Billion Downloads	1	Seth Fiegerman

	cleaned_article
0	The Associated Press is the latest news organ...
1	It looks like 2012 was a pretty good year for...

À ce nouveau jeu de données, nous ajoutons d'autres variables pertinentes déjà présentes dans le jeu de données initiales mais qui ne sont pas obtenues grâce à des méthodes de NLP. Ces variables sont : le nombre de partages (la variable que nous cherchons à prédire), le nombre de liens, le nombre de mot-clés et le nombre de vidéos de l'article.

2.2 Les valeurs manquantes

La seconde étape concerne le traitement des valeurs manquantes. Le tableau suivant présente les valeurs manquantes de chaque variable du nouveau jeu de données.

	Valeurs manquantes
url	0
shares	0
cleaned_article	191
author_name	428
title	189
nb_images	189
num_videos	0
num_hrefs	0
timedelta	0
num_keywords	0
chanel	997

On décide ici de retirer les observations pour lesquelles il existe des valeurs manquantes. En effet, après une recherche de leur raison d'apparition, on trouve les raisons suivantes :

— Des observations présentent le titre suivant : *The Bad News*. Après vérification, cela signifie que le lien associé est mort (les autres variables ont alors pris une valeur NA).
— Lorsque le titre est collecté mais pas le contenu, cela signifie que l'article ne contient pas de texte, seulement des images/vidéos.
— Lorsque le contenu et le titre sont manquants, c'est que la structure de l'article est différente et que le scraping n'a pas pu collecter l'information, ou qu'il s'agit d'un article de nature différente (galerie de photos par exemple).

On se retrouve avec maintenant 38405 observations, soit 1238 lignes en moins. Cela équivaut à 3% de NA (ce qui est donc négligeable comparé à la taille du jeu de données).

2.3 Création de nouvelles variables : la date

On peut remarquer que la date de l'article est incluse dans le lien. Plutôt que de les calculer en utilisant l'écart séparant la date du scrapping à la date de publication (ce qui est le cas dans la base de données en libre accès),nous décidons de directement récupérer la date à partir d'une regex. À partir de la date générée, on peut donc obtenir le jour de la semaine (Lundi, mardi, etc.) et inclure une variable binaire s'il s'agit d'un jour en semaine, ou du week-end.

2.4 Variables textuelles : le contenu des articles

Avant de commencer la partie NLP, il nous faut mettre en forme nos données textuelles. Pour cela, plusieurs étapes doivent être faites :

2.4.1 Étape 1 – La tokenisation

La tokenisation est l'acte de décomposer une séquence de chaînes de caractères en morceaux tels que des mots, des mots-clés, des phrases, des symboles et d'autres éléments appelés tokens.

Lors du processus de tokenisation, certains caractères comme les signes de ponctuation sont éliminés et le texte est transformé en minuscule. Pour notre cas, le processus de tokenisation, permet de créer une liste de chaque mot présent dans un article, de lui retirer sa ponctuation, ses stopwords, et considérer les mots composés comme un token unique.

2.4.2 Étape 2 – Le stemming ou la lemmatisation

Le stemming est le processus de réduction d'un mot à sa racine. L'avantage de ce procédé est que nous pouvons réduire le nombre total de mots uniques dans le dictionnaire. Par conséquent, le nombre de colonnes de la matrice documents–termes contiendra moins de colonnes.

La lemmatisation est une méthode de conversion d'un mot en sa forme de base. La différence avec le stemming est qu'elle tient compte du contexte et convertit le mot dans une forme compréhensible, alors que le stemming ne fait que supprimer les derniers caractères, ce qui entraîne souvent des significations incorrectes et pouvant rassembler des mots complètement différents.

Le sens des mots est important pour la détection des thèmes latents, nous décidons d'opter pour la lemmatisation.

3 Construction des thèmes

La première étape « NLP » de notre projet consiste à la construction des thèmes latents pour l'ensemble des article du jeu de données. Nous ajouterons les nouvelles variables créées à notre jeu de données pour les utiliser lors de notre modèle de prédiction. La procédure se fait en plusieurs étapes.

3.1 Bag of Words

Afin de pouvoir utiliser des algorithmes de recherche de thèmes latents, il est nécessaire au préalable d'utiliser une méthode de **Bag of Words** pour vectoriser le contenu des articles. Cette méthode consiste à calculer le nombre d'occurrences de chaque terme dans le document considéré (on parle souvent de « fréquence » par abus de langage). Plusieurs variantes de cette méthode existent : un choix simple, dit « binaire », est de mettre 1 si le terme apparaît dans le document et 0 sinon. À l'opposé, on peut normaliser logarithmiquement la fréquence brute pour amortir les écarts. Ici, on a choisi deux méthodes pour afficher la matrice des documents-termes :

— La première utilise la **fréquence brute** (nombre d'occurrences de chaque mot pour le texte considéré). Il s'agit donc d'un simple comptage des mots pour le document en question.
— La seconde utilise **la matrice TF-IDF**. Cela consiste à multiplier TF (nombre d'occurence sur le nombre total de mots) avec IDF : la fréquence inverse de document (*inverse document frequency*). Il s'agit d'une mesure de l'importance du terme dans l'ensemble du corpus. Dans le schéma TF-IDF, on donne un poids plus important aux termes un peu moins fréquents, considérés comme plus discriminants. Pour cela, on calcule le logarithme

de l'inverse de la proportion de documents du corpus qui contiennent ce terme.

Toutefois, le nombre de mots différents présent dans le corpus est beaucoup trop élevé (140 000 mots différents après élimination des mots stopwords) et l'utilisation du vocabulaire dans sa totalité n'est pas nécessaire à la création des thèmes. Pour pallier cela, plusieurs arguments introduits dans les méthodes de Bag of Words nous permettent de sélectionner les mots les plus importants.

— Une première solution est qu'il est possible de définir le nombre de *max feature*. Ceci permet de limiter le nombre mots à prendre en compte dans le vocabulaire. Par exemple, on peut prendre les 100 premiers mots les plus utilisés. Cette méthode nous sera utile si nous voulons afficher graphiquement les mots.
— Un autre choix serait de définir un maximum et un minimum de mots en %. le *max* est utilisé pour supprimer les termes qui apparaissent trop fréquemment. Le *min* est utilisé pour supprimer les termes qui n'apparaissent que trop rarement, également appelés « mots spécifiques au corpus ».

Pour notre cas, nous décidons de fixer un seuil minimal à 1% (si un mot n'apparaît que dans 1% des documents, il est retiré) et un seuil maximum à 95% (à l'inverse, s'il apparaît dans 95% des documents, il est retiré). Ceci nous donne donc une matrice de dimension : 38 000 x 1500. Maintenant nous possèdons le vocabulaire des mots, nous pouvons comparer les mots se trouvant à une distance proche et les représenter sous la forme d'un embedding. On se limitera à 100 mots différents pour cet exercice.

3.2 Représentation vectorielle : Word2vec

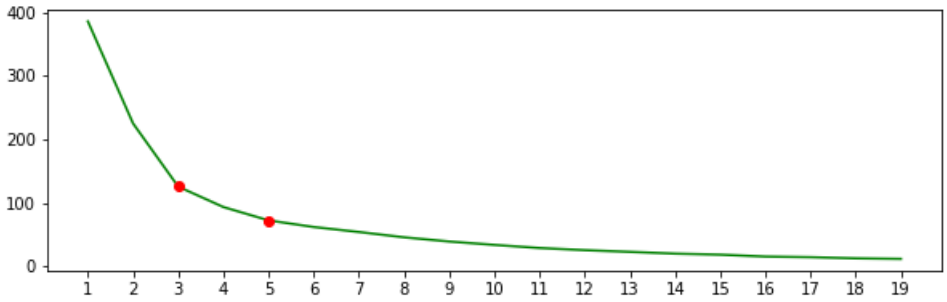
Word2vec est méthode de représentation vectorielle de mots dans n dimensions (généralement 300 et cela sera également notre cas pour ce projet). On appelle ceci l'*embedding* . Cet algorithme a pour but de capturer le contexte, la similarité sémantique et syntaxique (genre, synonymes, ...) d'un mot sous la forme d'un vecteur. À partir de ce vecteur, il sera facile de le représenter sous forme graphique. Ici, nous avons choisis d'utiliser un modèle entraînée provenant de la librairie `spaCy`. Il s'agit du modèle le plus large disponible contenant les 300 dimensions pour chaque mot. Toutefois, il faut noter que cette nouvelle matrice de représentation des mots sera de très grande dimension (en effet, les dimensions sont : nombre de mots $\times$ 300). Pour pallier cela, nous devrons recourir à des méthodes de réduction de dimensions : l'ACP ou t-SNE.

3.2.1 Visualisation par les composantes de l'ACP

La technique de réduction de dimension la plus utilisée est naturellement l'Analyse en Composante Principale (ACP). Afin de pouvoir visualiser les mots du corpus dans un graphique, nous retiendrons les deux premières composantes (celles qui maximisent l'inertie expliquée). Pour améliorer la visualisation, nous pouvons utiliser un algorithme de clustering afin de faire ressortir les groupes de mots les plus proches (en termes de proximités). La méthode que nous décidons d'utiliser reste la plus simple : le K-Mean. Cependant, il convient pour cela de trouver

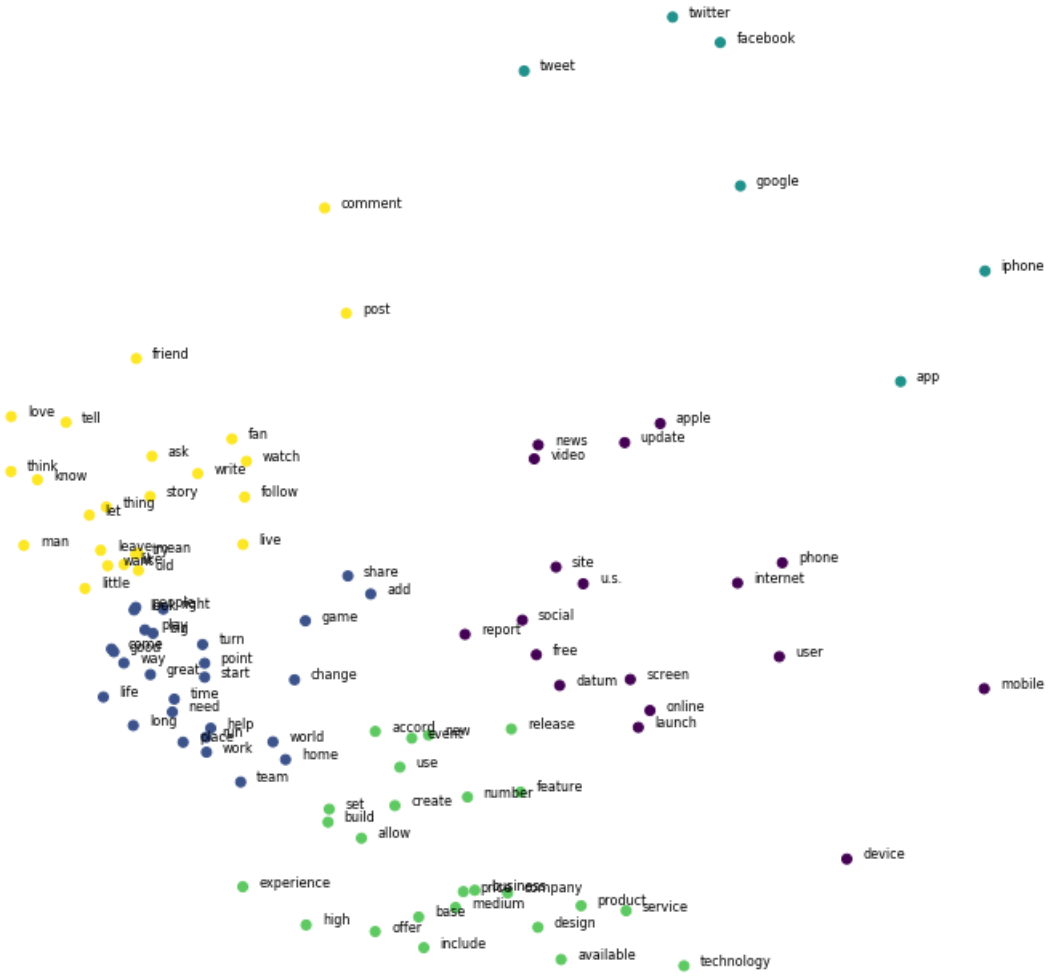
le nombre de cluster optimal. Nous utilisons le critère du coude sur la courbe des inerties total pour choisir ce nombre de clusters (qui ne correspond pas au nombre de thèmes, on s'en sert seulement seulement pour observer les groupes de mots avec un sens proche).

FIGURE 1 – Sélection des clusters : KMeans



À partir du graphique précédent (cf. Figure 1), on peut choisir entre 3 et 5 clusters. En effet, selon la règle du coude, on observe un décrochage de la courbe d'inertie entre ces deux nombres de clusters. Pour notre cas, nous choisirons d'utiliser 5 thèmes pour essayer d'en distinguer un maximum.

FIGURE 2 – – Représentation vectorielle : ACP



Cette première visualisation de l'algorithme Word2Vec sur notre jeu de données montre donc des clusters de mots proches : en **turquoise** les réseaux sociaux (Facebook, Twitter, Google, etc.), en **mauve** ce qui fait référence au web et la téléphonie (news, video, apple, phone, internet, etc.) ou encore en **vert** ce qui fait référence aux entreprises (service, technology, company, offer etc.).

3.2.2 Visualisation par les composantes t-SNE

Une autre méthode possible de visualisation est obtenue en utilisant l'algorithme t-SNE (*distributed stochastic neighbor embedding*). Il consiste à réduire en deux ou trois dimensions une matrice de grande dimensionnalité. Ainsi, si deux points sont proches (resp. éloignés) dans l'espace d'origine ils devront être proches (resp. éloignés) dans l'espace à faible dimension. Pour ce faire, cet algorithme non-linéaire (à la différence de l'ACP) se base sur une probabilité des proximités de chacun des points. L'avantage de cette méthode de réduction de dimension réside dans sa plus grande fidélité aux distances ; toutefois, les clusters ne sont pas si flagrants avec cette méthode. On gardera donc l'algorithme de clustering précédent pour faciliter la visualisation. On observe bien que les mots les plus ressemblants sont bien proches comme précédemment, mais avec une disposition différente.

FIGURE 3 – – Représentation vectorielle : t-SNE

3.3 Recherche des thèmes latents

Nous pouvons revenir à l'objectif principal de cette partie : la détection des thèmes latents. Pour l'effectuer, il nous est possible d'utiliser deux algorithmes :

— **Latent Dirichlet Allocation (LDA)** : Il s'agit d'un modèle de type génératif permettant d'expliquer des ensembles d'observations, par le moyen de groupes non observés. Ces derniers sont définis par des similarités entre les données. Par exemple, si les observations sont les mots collectés dans un ensemble de documents textuels, le modèle LDA suppose que chaque document est un mélange d'un petit nombre de sujets et que la génération de chaque occurrence d'un mot est attribuable (en termes de probabilité) à l'un des thèmes du document.

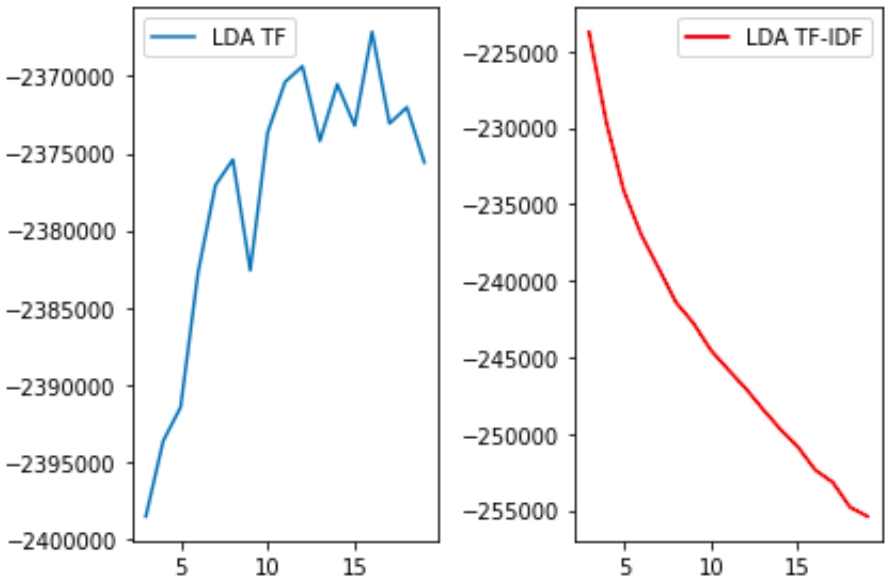
— **Non-négative Matrix Factorization (NMF)** : Il s'agit d'une technique de factorisation de matrice par la décomposition en valeur singulière (comme pour l'ACP). C'est donc une technique de réduction de dimension mais les composantes sont non-orthogonales (contrairement à l'ACP). Elle est adaptée aux matrices creuses (matrice de grandes dimensions contenant beaucoup de 0) ne contenant que des données positives, par exemple

des occurrences de mots. Les facteurs de décomposition n'étant pas orthogonaux, des superpositions peuvent apparaître : des même mots participants à plusieurs thèmes. Lee et Seung (1999) illustre cette méthode sur la classification d'un corpus de 30 991 articles de l'encyclopédie Grolier. Plutôt que de classer ces articles par thèmes choisis a priori, ils sont classés sur la base d'un vocabulaire de 15 276 mots.

3.3.1 Le nombre optimal de thèmes

Avant de commencer à utiliser un algorithme de recherche des thèmes, la première étape est de réfléchir aux hyperparamètres à utiliser. Pour trouver le nombre optimal de thème nous allons utiliser la fonction `GridSearchCV` qui permet de tester plusieurs modèles avec un système de cross-validation et de retenir celui ayant la métrique choisie (ici la log-vraisemblance) la plus élevée. À partir de cette méthode, on va pouvoir comparer le modèle LDA pour les deux types de matrice et décider du nombre de thèmes à retenir. Toutefois, il faut rester prudent car les thèmes doivent rester pertinent pour l'analyse.

FIGURE 4 – Log-vraisemblance des modèles



À partir de la Figure 4, on observe l'évolution de la log-vraisemblance selon le nombre de composantes (équivalent aux thèmes). Pour le modèle utilisant la matrice TF, on peut retenir 16 thèmes différents, tandis que pour le modèle utilisant la matrice TF-IDF, la métrique indique un nombre optimal de 2. Toutefois, d'un point de vue métier, distinguer seulement deux thèmes semble être très réducteur. Pour cela, nous décidons de suivre le nombre de thèmes de la baseline (jeu de données de départ). On retiendra donc 5 thèmes pour les matrices TF-IDF.

3.3.2 Labélisation des thèmes

Nous avons donc au total 4 propositions de sélection de thèmes pour notre jeu de données :

— un algorithme NMF avec la matrice TF (16 thèmes)
— un algorithme LDA pour la matrices TF (16 thèmes)
— un algorithme NMF avec la matrice TF-IDF (5 thèmes)
— un algorithme LDA pour la matrices TF-IDF (5 thèmes)

Après une première observation des sorties, nous avons sélectionné deux algorithmes concurrents. Le premier est celui LDA pour la matrice TF-IDF. Les mots formant chacun des cinq thèmes sont les suivants :

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9
Topic 0	police	government	people	report	country	accord	city	u.s.	state	ukraine
Topic 1	company	facebook	user	twitter	social	google	service	app	new	use
Topic 2	film	movie	trailer	episode	netflix	character	star	series	disney	premiere
Topic 3	app	apple	device	iphone	new	phone	use	android	user	samsung
Topic 4	video	game	like	time	youtube	fan	good	know	love	song

Après une recherche de la signification de l'ensemble des mots formant chacun des thèmes, nous avons labéliser les thèmes de la manière suivante :

Numéro	Thèmes labélisés
0	Politics and diplomacy
1	Internet and social media
2	Videos and movies
3	Telecomunication devices
4	Entertainment

Ici, les thèmes semblent bien correspondre mais restent tout de même très génériques. Nous pouvons alors observer la distinction en 16 thèmes pour savoir si il est possible d'être plus précis. Le second algorithme sélectionné est NMF pour la matrice TF. Les mots formant chacun des 16 thèmes sont les suivants :

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9
Topic 0	like	time	people	know	think	want	work	way	thing	good
Topic 1	child	age	mother	number	job	mom	thing	like	good	moment
Topic 2	app	ios	user	android	new	free	update	available	mobile	feature
Topic 3	use	user	site	create	file	email	web	design	allow	work
Topic 4	game	play	player	new	fan	character	world	time	team	console
Topic 5	apple	iphone	new	event	california	ipad	ios	tim cook	product	san francisco
Topic 6	u.s.	report	country	accord	president	state	people	government	official	russia
Topic 7	twitter	tweet	social	account	medium	user	follow	new	news	post
Topic 8	facebook	user	social	post	friend	people	page	like	network	share
Topic 9	phone	device	screen	new	tablet	camera	smartphone	samsung	display	design
Topic 10	video	youtube	vine	music	new	share	song	view	watch	release
Topic 11	team	final	score	look	kit	home	away	brazil	win	white
Topic 12	use	guest	light	photograph	look	instagram	shot	shoot	like	subject
Topic 13	police	protester	government	ukraine	protest	clash	officer	anti	street	ukrainian
Topic 14	company	work	job	business	product	new	startup	employee	service	help
Topic 15	google	search	glass	user	android	company	reader	service	google+	google glass

Comme précédemment, nous avons labélisé les thèmes selon les mots qui forment ce dernier. On peut remarquer que ces thèmes sont plus précis que les précédents, toutefois, on observe de la redondance (trois thèmes reviennent en double).

Numéro	Thèmes labélisés
0	other
1	family and job
2	devices telecomunication
3	web
4	sport and gaming
5	apple
6	politics and society
7	twitter
8	facebook
9	devices telecomunication
10	musics and videos
11	sport and gaming
12	photography
13	politics and society
14	business
15	google

3.3.3 Création des variables de proximités des thèmes prédits et du thème dominant

La dernière étape de cette partie consiste à l'attribution des thèmes pour chaque document et le calcul de leur proximité avec les autres thèmes (dans le cas où plusieurs thèmes interviennent). On peut afficher un extrait de ces nouvelles variables :

	Topic_nmf other	Topic_nmf family and job	Topic_nmf web	Topic_nmf apple	Topic_nmf twitter
0	0.00	0.00	0.00	0.00	0.30
1	0.00	0.00	0.00	0.10	0.00
2	0.03	0.00	0.01	0.00	0.00
3	0.06	0.02	0.00	0.04	0.04
4	0.00	0.00	0.00	0.02	0.00

	Topic_nmf musics and videos	Topic_nmf photography	Topic_nmf business	Topic_nmf google	dominant_topic
0	0.00	0.02	0.13	0.00	twitter
1	0.00	0.00	0.05	0.00	devices telecomunication
2	0.03	0.02	0.00	0.00	sport and gaming
3	0.22	0.00	0.06	0.00	devices telecomunication
4	0.03	0.00	0.10	0.03	devices telecomunication

Pour chaque article (les lignes), on retrouve la proximité en valeur numérique par rapport à chacun des thèmes choisis précédemment. La dernière colonne correspond au thème prédominant

de l'article. Par exemple, pour le premier article (le numéro 0), son titre est le suivant : « AP's Twitter to Begin Displaying Sponsored Tweets ». On remarque sur la table précédente, que le thème prédominant est Twitter. Pour l'article numéro 2, celui-ci devrait théoriquement aborder le thème du sport et du gaming, mais son titre fait référence à un astronaute : « This Astronaut Is Rooting for Notre Dame Tonight ». Toutefois, si on entre dans le contenu de l'article, ce dernier évoque le fait qu'un astronaute suive un match de football depuis une station spatiale. Donc, ce thème fait bien référence à un évènement sportif. Enfin, un dernier article qu'on peut auditer est le numéro 4 : « BeeWi's Smart Toys Put Your Smartphone in Control ». Ce dernier traite bien des smartphones au vu de son titre.

Pour le second algorithme utilisé (LDA TF-IDF), on obtient les attributions suivantes :

	Topic_lda Politics and diplomacy	Topic_lda Internet and social media
0	0.02	0.90
1	0.03	0.03
2	0.56	0.02
3	0.02	0.02
4	0.03	0.03

	Topic_lda Videos and movies	Topic_lda Telecomunication devices	Topic_lda Entertainment	dominant_topic_lda
0	0.02	0.03	0.02	Internet and social media
1	0.05	0.85	0.03	Telecomunication devices
2	0.02	0.02	0.37	Politics and diplomacy
3	0.02	0.92	0.02	Telecomunication devices
4	0.03	0.81	0.11	Telecomunication devices

Pour cette dernière attribution de thèmes, on remarque que deux des articles précédemment testés correspondent bien aux nouveaux thèmes (Internet pour Twitter et Telecommunication devices pour les smartphones). Toutefois, l'article traitant de l'astronaute qui suit un évènement sportif depuis l'espace est estimé comme appartenant au thème Diplomacy and Politics (en raison du caractère de la NASA) avec une score de 0,56. Cependant, les mots concernant l'évènement sportif augmente le score du thème du loisir (0,37), ce qui semble être exact au vu du contenu de l'article. Cette attribution de thème semble donc être également cohérente.

Pour terminer cette première partie NLP, nous conserverons les deux types d'attributions de thèmes et nous les intégrerons dans notre jeu de données. Selon les modèles de prédictions utilisés, nous réfléchirons à quelle attribution de thèmes retenir.

4 Analyse de sentiments : construction des variables polarité et subjectivité

La deuxième partie de NLP consiste à la création de variable via l'analyse de sentiment. Celle-ci est un ensemble de méthodes permettant de décrire l'attitude ou l'émotion de l'auteur d'un texte, c'est-à-dire si elles sont positives, négatives, neutres, de grande intensité ; ou encore si le texte étudié révèle un fort engagement personnel de l'auteur ou constitue simplement un énoncé

factuel.

Afin de conduire cette analyse, nous nous appuyons ici sur la librairie TextBlob. Cette dernière offre un accès simple à de nombreuses méthodes de NLP (et notamment l'analyse de sentiments), sur la base des librairies NLTK et Pattern. Ainsi, la fonction *TextBlob* nous retourne deux métriques : la polarité et la subjectivité.

— La **polarité** retournée appartient à l'intervalle $[-1\,;1]$. Une polarité de -1 signifie alors une émotion fortement négative, là où une polarité de 1 signifie une émotion fortement positive, et 0 une neutralité. La polarité nous renseigne alors à la fois sur la nature du sentiment du texte, mais aussi sur l'intensité de ce sentiment.
— La **subjectivité** se situe quant à elle dans l'intervalle $[0\,;1]$. Plus ce score est élevé, plus le texte est identifié comme étant constitué d'opinions personnelles, d'émotions ou de jugements, alors qu'une subjectivité faible sera attribuée aux textes se référant à des informations factuelles, sans engagement de l'auteur.

Il est question ici d'appliquer ces méthodes d'analyse de sentiment sur le contenu de nos articles, mais également sur leur titre. Nous pensons en effet qu'à la fois la nature des émotions, leur force et le niveau de subjectivité d'un article et de son titre peuvent avoir un effet significatif sur sa popularité. Nous étudierons d'abord la distribution des scores obtenus selon la baseline, notre texte brut et notre texte lemmatisé. Puis, nous comparerons des échantillons et des métriques d'écarts entre les valeurs obtenues.

4.1 Polarité et subjectivité de la base de données initiale (baseline)

Tout d'abord, regardons les distributions de polarité et de subjectivité dans la base de données originale qui nous a inspirés.

On voit qu'au niveau du contenu, la polarité est assez faible (Figure 5), distribuée normalement autour de 0, avec une variance très faible. Cela irait dans le sens d'articles relativement neutres du point de vue des émotions. Pour la subjectivité en revanche (Figure 6), la distribution est plutôt étalée à gauche avec une grande partie des valeurs située entre 0,3 et 0,5, signif iant un nombre d'articles relatant des faits plus élevé que le nombre d'articles plus personnels. Cependant, on est tout de même à des niveaux de subjectivité différents de 0. On est donc loin de l'idée d'articles purement factuels.

FIGURE 5 – Distribution de la polarité des contenus - Baseline

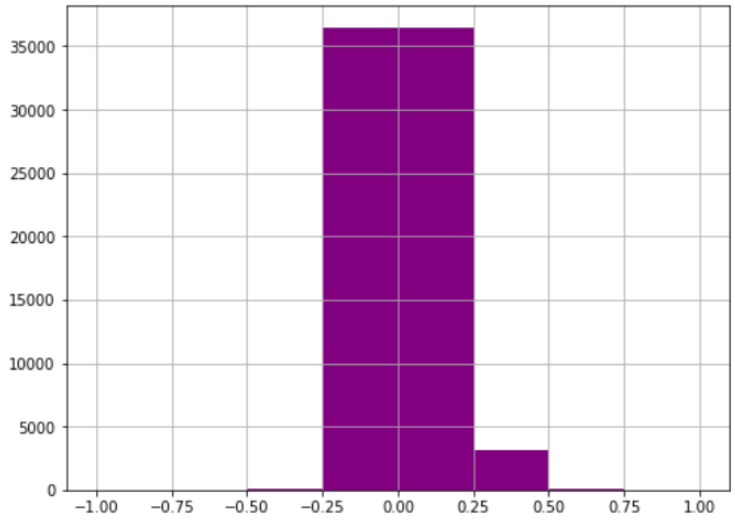
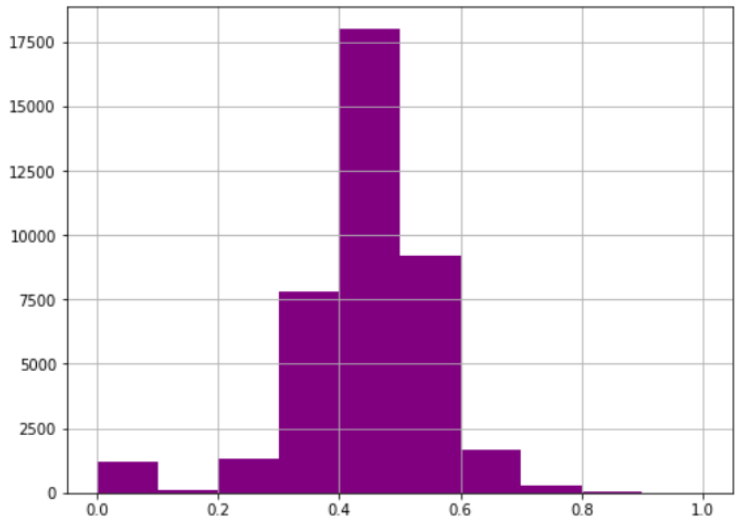


FIGURE 6 – Distribution de la subjectivité des contenus - Baseline



Au niveau des titres, on a aussi une distribution de la polarité (Figure 7) fortement concentrée autour de 0, et peu étalée. On a tout de même un peu plus de valeurs élevées (en valeur absolue) que pour le contenu. Selon nous, cela pourrait témoigner de la nécessité pour un titre d'être plus accrocheur, et de susciter plus d'émotions (positives ou négatives, tant qu'elles sont fortes) pour donner envie de cliquer. Concernant le niveau de subjectivité des titres (Figure 8), c'est beaucoup plus étalé que pour les contenus. En effet, on retrouve un très grand nombre de valeurs nulles, qui témoignent à notre sens de la nécessité pour un titre d'être direct et universel, plutôt que soumis à l'opinion de celui qui l'écrit. Notons aussi que l'étalement pour le reste des classes et ici aussi plus élevé que pour le contenu.

FIGURE 7 – Distribution de la polarité des titres - Baseline

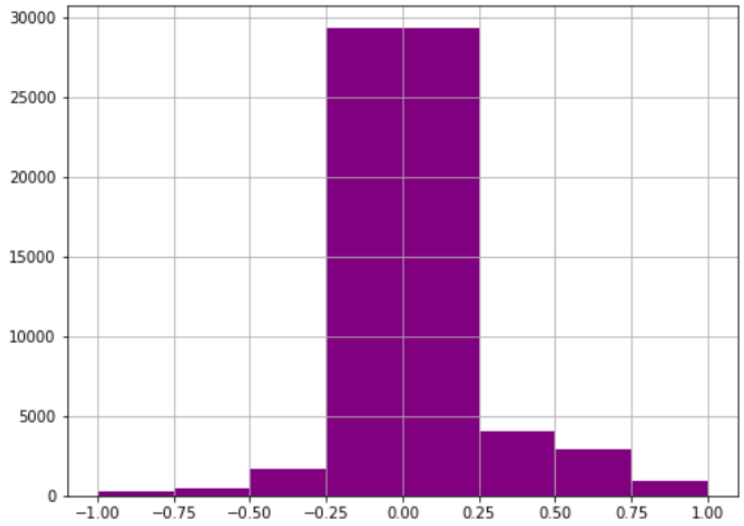
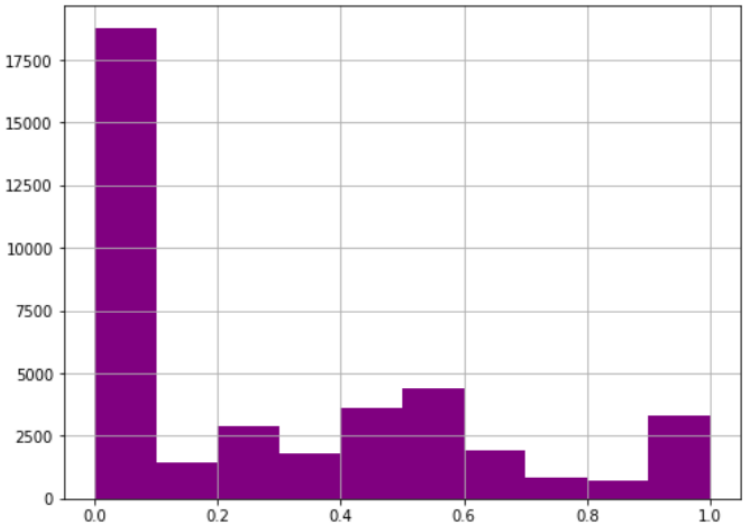


FIGURE 8 – Distribution de la subjectivité des titres - Baseline



4.2 Polarité et subjectivité de la base de données scrappée

Après avoir étudié la distribution des scores présents dans notre base de référence, construisons nos propres scores à partir de la base de données que nous avons scrappée.

Au moment de conduire une analyse de sentiments, la question se pose quant au format des données sur lesquelles nous allons travailler : sur données lemmatisées (et sans stopwords) ou non lemmatisées ? Il existe des arguments allant dans le sens de chacune de ces deux options :

— Lemmatiser les données : cela permet de simplifier l'analyse en réduisant le nombre total de mots à considérer, et donc en réduisant le bruit.
— Laisser les données telles quelles : la lemmatisation est une étape obligatoire pour de nombreuses tâches de NLP (*e.g.* topic modeling, word vectors). Cependant, dans une

analyse de sentiments, la lemmatisation peut conduire à simplifier outre-mesure des mots et des phrases et donc à rater des éléments clefs qui nous auraient permis de mieux saisir et formaliser le sentiment ressortant du texte. Ceci est d'autant plus vrai lorsque le texte est de petite taille (titre d'un article par exemple), et où la variation d'un seul élément peut avoir un fort impact sur le résultat général.

Prenons un exemple. Deux phrases ne diffèrent que d'une seule lettre : « He hates brocoli » et « He hated brocoli ». Si on les lemmatise, on obtient la même phrase : « hate brocoli ». Ainsi, en les lemmatisant, les scores seront exactement les mêmes. Cependant, le score de polarité que l'on obtient pour les deux phrases brutes est différent. En effet, nous obtenons un score de 0 pour la première forme (et donc une phrase neutre du point de vue de la polarité), et un score de -0,9 pour la seconde (une phrase très fortement négative du point de vue de la polarité). Le choix de lemmatiser ou non les textes est donc susceptible de modifier nos résultats. Nous nous pencherons ici sur ces deux options, et en retiendrons une seule au final.

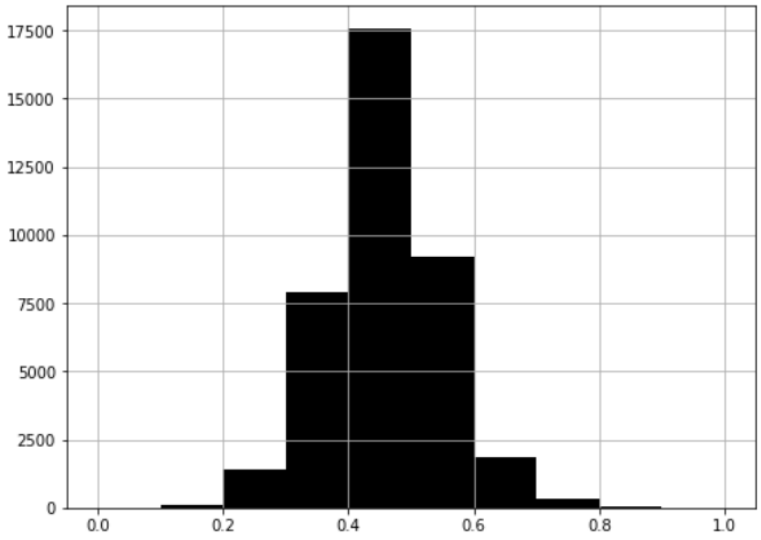
4.2.1 Textes bruts

La librairie TextBlob s'appuie sur deux modules au choix pour conduire ses analyses de sentiments. D'abord, le module PatternAnalyzer (basé sur la librairie pattern), ou alors sur le module NaivesBayesAnalyzer (basé sur la librairie NLTK). Ce dernier ayant été entraîné sur une base de données de critiques de films, nous utiliserons la première option (sélectionnée par défaut) qui est plus générale. L'analyse se fait globalement ainsi :

1. La librairie sépare les phrases en différentes parties afin d'identifier les noms, adjectifs, verbes, adverbes etc.
2. Ensuite, elle se réfère à sa base de données d'expressions/de mots/de séquences de phrases déjà notées afin d'attribuer un score aux expressions de notre texte qui matcheraient avec cette base de données. Ceci est valable à la fois pour la subjectivité et pour la polarité.
3. Enfin, l'entièreté du texte est reconstituée et les scores sont agrégés.

Appliquons cette analyse à nos textes bruts. On obtient globalement des distributions très proches de celles présentes dans la baseline. Nous pouvons en voir un exemple en comparaison la Figure 9 à la Figure 6, qui sont en effet très similaires. La différence la plus notable est la quasi absence de valeurs nulles dans l'analyse de nos textes bruts, contrairement à la subjectivité des contenus de la baseline.

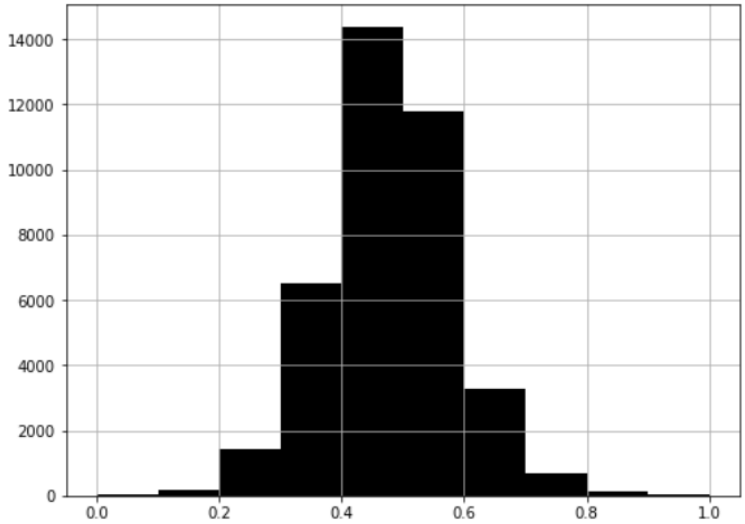
FIGURE 9 – Distribution de la subjectivité des contenus - Textes bruts



4.2.2 Textes lemmatisés

Du point de vue du contenu des articles, les données lemmatisées mènent à une distribution de la polarité très similaire à précédemment. En revanche, concernant la subjectivité (Figure 10), nous retrouvons des niveaux plus élevés et plus présents dans la tranche 0,5-0,6 que pour les deux bases précédentes. De même, du côté des titres, on obtient une distribution de polarité similaire, mais une distribution de la subjectivité un peu plus concentrée en 0.

FIGURE 10 – Distribution de la subjectivité des contenus - Textes lemmatisés



Au final, on voit que les résultats d'analyse de sentiments sont relativement proches en distributions entre les trois bases de données, bien que les différences se concentrent surtout en termes de subjectivité (plus importantes pour les textes lemmatisés dans le contenu, et plus faible dans

les titres). Aussi, la distribution baseline se rapproche légèrement plus de la distribution des scores via textes bruts que via textes lemmatisés.

4.3 Comparaisons

Après avoir étudié les distributions des scores de sentiments obtenus, adoptons une analyse plus spécifique.

4.3.1 Echantillons

D'abord, inspectons quelques échantillons des résultats. On voit tout d'abord dans l'échantillon de polarité des contenus que les trois bases de données présentent des scores relativement proches (Tableau 1), les scores des bases brut et lemmatized étant les plus proches. Ceci s'explique sûrement par les différences de méthodes d'analyse de sentiments et de préparation de données que les auteurs de la baseline ont utilisées comparativement à nos méthodes. Cependant, on voit tout de même que les scores de la base brute sont plus proches de la baseline que ne l'est la base lemmatisée. Pour l'article « What Is Internet Freedom Day ? », on a même un score exactement égal entre baseline et brut.

TABLE 1 – Polarité des contenus pour un sous-échantillon

Polarité des contenus			
title	baseline	brut	lemmatized
DOMA Decision Spurs Celebrations and Other Top...	0.255080	0.157497	0.100939
It's a War to Rule the World in 'Dawn of the P...	0.169329	0.119551	0.081358
Couple Tips Their Waitress in Crystal Meth	0.027922	0.050000	-0.035714
Gamers' Touching Tributes to Nintendo Visionar...	0.129868	0.034483	0.052202
Egyptian Stars of Netflix's 'The Square' Can't...	0.195026	0.127033	0.114094
What Is Internet Freedom Day ?	0.122887	0.122887	0.084587
Obama's Funeral Selfie : This Is Why Context Ma...	0.019906	0.086093	0.082243
Why the Ice Bucket Challenge Is a Watershed Mo...	0.137813	0.144095	-0.031361
How This Teacher Won the $1 Million TED Prize	0.079400	0.079400	0.068740
Does Jonathan Coulton Have a Copyright Case Ag...	0.142964	0.142964	0.128991

L'échantillon de subjectivité des titres nous révèle d'autres élements (Tableau 2). Un constat qui aurait pu être fait sur un échantillon de polarité des titres : la concordance entre la baseline et nos scores obtenus est bien moins évidente que pour le contenu. En effet, il y a bien moins de données, donc les différences méthodologiques ressortent plus clairement.

TABLE 2 – Subjectivité des titres pour un sous-échantillon

Subjectivité des titres			
title	baseline	brut	lemmatized
Disney Shutters LucasArts, Cancels Upcoming 'S...	0.000000	0.000000	0.400000
Security Firm Warns About App That Pays for Un...	0.000000	0.400000	0.400000
Singer Miguel Jumps on Fans' Heads in Freak Ac...	0.000000	0.000000	0.000000
Ed Sheeran Debuts 'Sing' Music Video Exclusive...	0.000000	0.000000	0.000000
What Glaciovolcanoes Can Tell Us About Past Ic...	0.800000	0.250000	0.250000
Take a Look at Baidu Eye, China's Version of G...	0.000000	0.000000	0.000000
'A Vendetta Against Suarez' : Uruguay Team Defe...	0.000000	0.000000	0.000000
Top 10 Summer Swimming Pool Dunk Videos	0.000000	0.500000	0.000000
The growing popularity of global social media ...	0.580000	0.033333	0.033333
Watch the Empire State Building's Fourth of Ju...	0.148182	0.350000	0.350000

4.3.2 Ecart absolu moyen

Voyons si nos observations formulées à la vue des échantillon ci-dessous se confirment par les chiffres, c'est-à-dire par la métrique suivante : l'écart absolu moyen entre les valeurs. Soit deux scores (de polarité ou de subjectivité), de la base de données A et de la base de données B, attribués à un texte i. On les nommera a_i et b_i. Notre métrique nous renseignera sur l'écart (en valeur absolue) moyen entre les deux scores attribués à un texte. Sa formule s'écrit donc :

$$EAM = \frac{\sum_{i=1}^{n} |a_i - b_i|}{n}$$

où n est le nombre de textes considérés.

Les résultats obtenus sont présentés dans le tableau ci-dessous :

TABLE 3 – Ecarts absolus moyens

	Contenus		Titres	
	Polarité	Subjectivité	Polarité	Subjectivité
EAM - brut/lemmatisé	0.047	0.041	0.055	0.067
EAM - brut/baseline	0.085	0.091	0.319	0.314
EAM - lemmatisé/baseline	0.100	0.107	0.324	0.315

On peut lire dans la table que nos observations faites via échantillons sont globalement exactes. En effet, les écarts les plus faibles sont systématiquement entre données brutes et lemmatisées, du fait de la similitude des méthodes employées sur nos deux BDD. Aussi, les écarts les plus élevés constatés sont toujours entre données lemmatisées et résultats de la baseline. L'analyse de sentiments via données brutes fournit donc des résultats plus proches de la baseline. Enfin, là où

les écarts observés sur le contenu sont relativement faibles, ceux observés sur les titres sont au contraire très élevés (ce qui est logique car moins de données/de mots signifie une plus grande sensibilité aux différences de méthodologie).

Afin de nous assurer les résultats les plus fiables possibles, et pour des raisons de maintien du sens originel évoquées plus haut (avec l'exemple « hate »), nous décidons de garder les résultats obtenus sur textes bruts.

5 Analyse descriptive des données

Dans cette partie, nous présenterons les données de notre nouveau jeu de données. On peut commencer tout d'abord par vérifier le nombre d'observation manquantes pour chacune des variables du jeu de données.

	Nombre d'observations manquantes
shares	0
author_name	0
nb_images	0
num_videos	0
num_hrefs	0
num_keywords	0
chanel	0
day	0
week	0
Topic_nmf other	0
Topic_nmf family and job	0
Topic_nmf web	0
Topic_nmf apple	0
Topic_nmf twitter	0
Topic_nmf facebook	0
Topic_nmf musics and videos	0
Topic_nmf photography	0
Topic_nmf business	0
Topic_nmf google	0
dominant_topic	0
Topic_nmf devices telecomunication	0
Topic_nmf politics and society	0
Topic_nmf sport and gaming	0
Topic_lda Politics and diplomacy	0
Topic_lda Internet and social media	0
Topic_lda Videos and movies	0
Topic_lda Telecomunication devices	0
Topic_lda Entertainment	0
dominant_topic_lda	0
content_polarity	0
abs_content_polarity	0
content_subjectivity	0
title_polarity	0
abs_title_polarity	0
title_subjectivity	0
month	0

On remarque qu'il n'y a pas de valeurs manquantes dans notre jeu de données. On peut donc commencer notre analyse descriptive simple selon les différents groupes de variables : les variables relatives aux articles, les variables relatives aux thèmes de LDA, les variables relatives aux thèmes de NMF. Dans une seconde partie nous nous pencherons sur les les liens entre l'ensemble de nos variables et nous regarderons les liens possibles entre chacune d'elle et notre variable à prédire ($shares$).

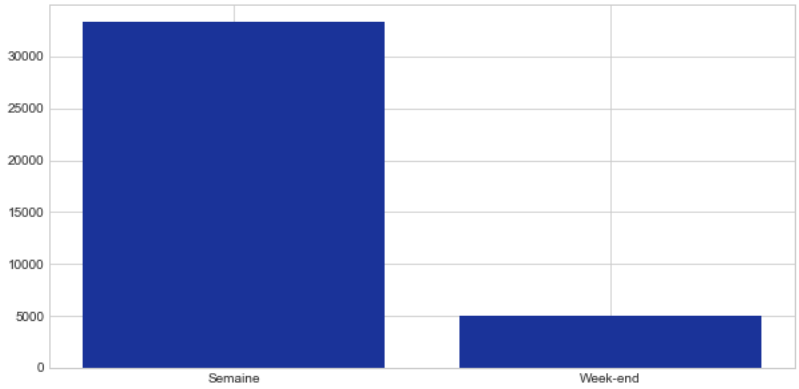
5.1 Analyse descriptive simples

La première étape lors d'une analyse descriptive est l'analyse univariée de chacune des variables du jeu de données. On peut pour les variables qualitatives observer leur fréquence, et pour les variables quantitatives observer la distribution et les box-plots.

5.1.1 Variables relatives aux articles

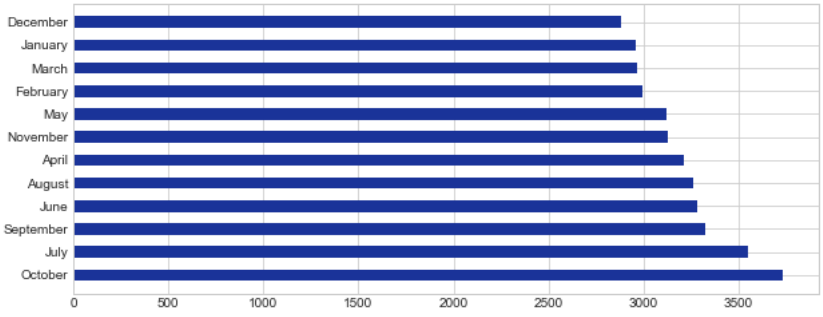
Nous avons déjà des variables pour différencier les jours de la semaine et différencier les weeks-end du reste de la semaine. Nous ajoutons une variable *month* pour les mois (inutile pour les années car le jeu de données s'étend seulement sur deux années).

FIGURE 11 – Répartition des articles : Week-end et Semaine



On remarque ici que plus de 86% des articles ont été postés en semaine, contre 14% le week-end. Outre le fait qu'il y ait seulement 2 jours de week-end, ces articles peuvent avoir été préparé à l'avance lors de la semaine en vue de leurs diffusions les samedi et dimanche.

FIGURE 12 – Répartition des articles : mois de l'année



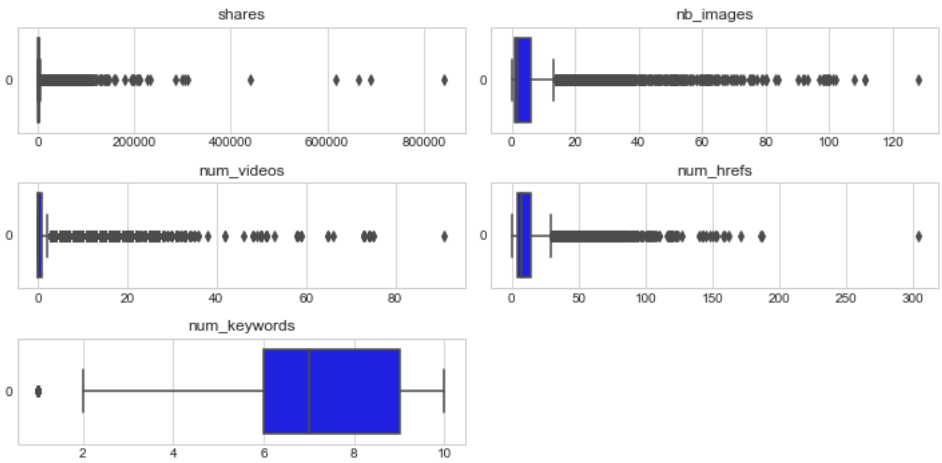
Pour la variable *month*, les articles sont équitablement répartis sur chacun des mois de l'année

(entre 9,5% et 7,5% du total des article pour chacun des mois). Cela ne semble donc pas être une variable discriminante d'un point de vue global.

	author_name
Noname	28148
Neha Prakash	1499
Sam Laird	1458
Stan Schroeder	1443
Todd Wasserman	1287
Seth Fiegerman	1267
Brian Anthony Hernandez	1233
Samantha Murphy	1093
Lorenzo Franceschi-Bicchierai	977

Pour réduire le nombre total de catégories, on remplace le nom des auteurs par *noname* quand ce dernier ne fait pas partie des 9 auteurs ayant écrit le plus d'articles. On réduit ainsi le nombre d'auteurs différents à 10 (au lieu de plus de 1100) tout en gardant l'information relative à l'auteur pour 11083 articles.

FIGURE 13 – Boxplots – Articles

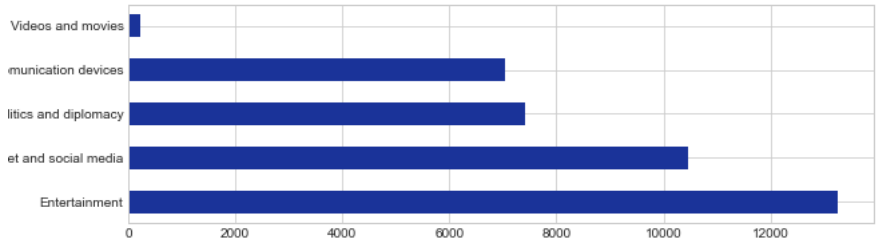


Au travers des graphiques ci-dessus, on remarque que de nombreux articles ont une popularité faible (shares proche de 0) mais que certains sont extrêmement populaires, ces articles sont de véritables buzz. De plus, la plupart des articles sont avares en éléments visuels (images et/ou vidéos) ainsi qu'en références. Cependant le nombre de keywords approches souvent le maximum possible qui est de 10, afin de favoriser le référencement des articles et donc faciliter leurs diffusions.

5.1.2 Variable relatives aux thèmes

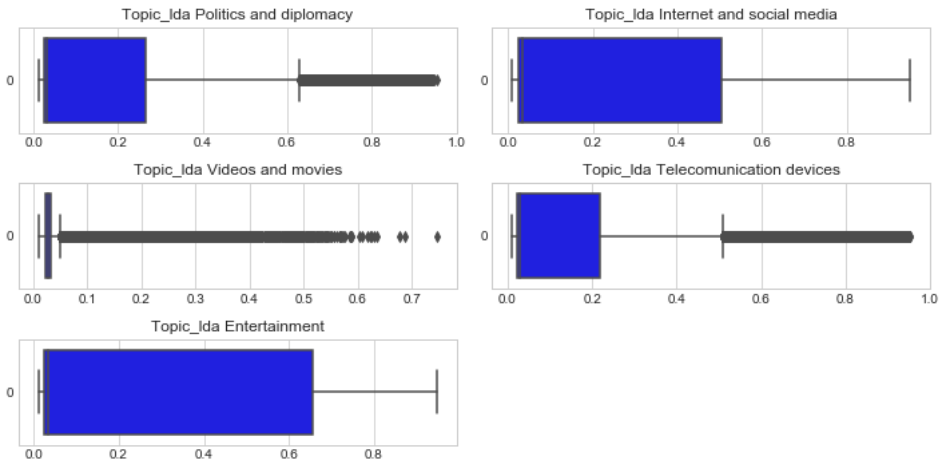
Pour cette partie relative aux thèmes, nous allons observer la répartition des thèmes pour chacun des algorithmes, puis vérifier leur cohérence par rapport aux thèmes.

FIGURE 14 – Répartition des thèmes LDA



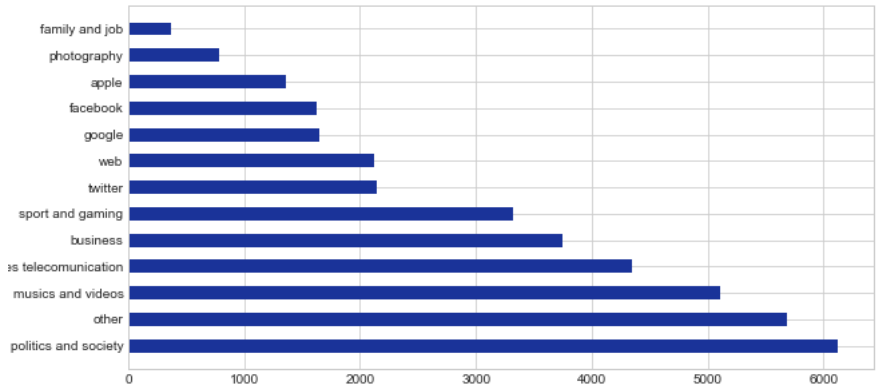
La Figure 14 montre la répartion des articles au sein des thèmes (méthode LDA). On remarque rapidement que le thème "Videos and movies" est bien moins fourni que les autres. Le thème "Entertainment" est quand à lui le plus représenté.

FIGURE 15 – Boxplots – LDA



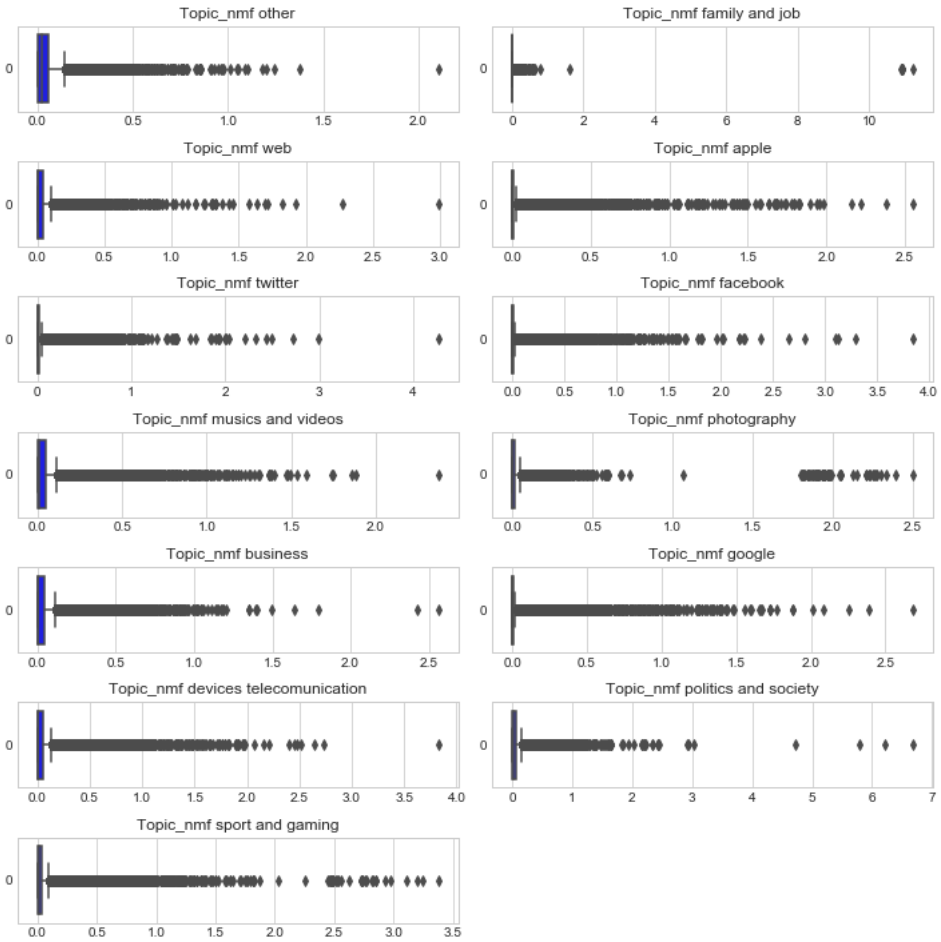
La Figure 15 représente la probabilité d'appartenance de chaque article à chacun des 5 thèmes. On remarque que le thème « Videos and movies » est très spécifique, si bien que dès qu'un article à une probabilité supérieure à 5% d'appartenir à ce thème il apparaît en valeur extrême. On peut également remarquer que les thème « Internet and social media » et « Entertainment » semblent les moins spécifiques. Cela confirme les observations liées à la Figure 14.

FIGURE 16 – Répartition des thèmes NMF



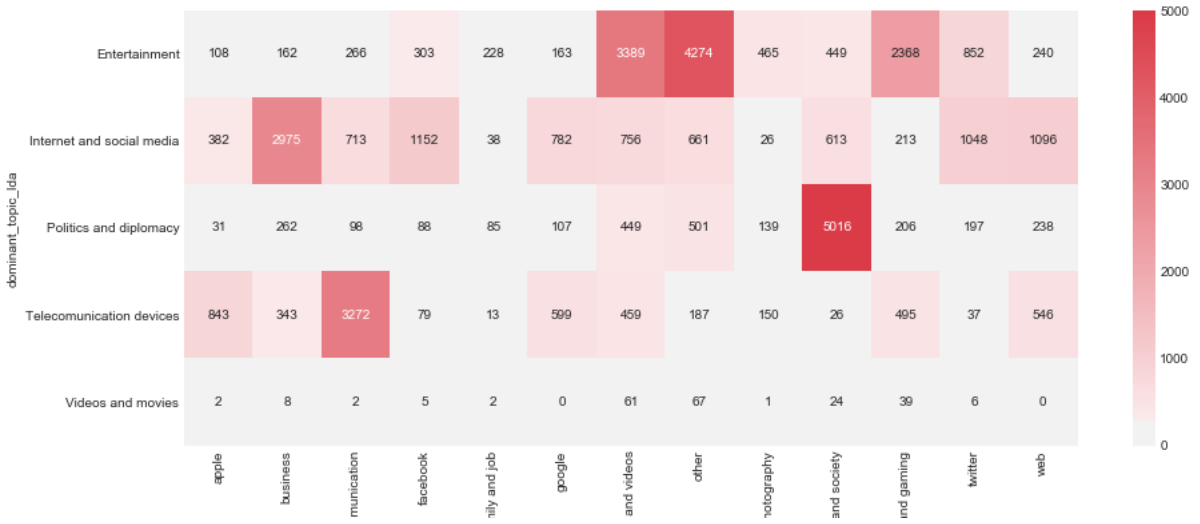
La Figure 16 montre la répartitions des articles au sein des thèmes (méthode NMF). Les thèmes « Family and job », « Photography », « Apple », « Facebook », « Google », « Web » et « Twitter » sont les moins fournis.La répartition des autres thèmes semblent plus uniforme, et le thème « Politics and society » est le plus représenté devant « Others ».

FIGURE 17 – Boxplots – NMF



La Figure 17 représente la probabilité d'appartenance de chaque article à chacun des 13 thèmes. On remarque une nette différence avec le graphique 15 : les thèmes étant plus nombreux, ces derniers semblent être plus spécifiques. On remarque également que le thème qui semble couvrir le plus de sujets est le thème « Other » ce qui est cohérent.

FIGURE 18 – Crosstab des deux algorithmes



La Figure 18 indique que les classements par thèmes sont globalement cohérents entre les deux méthodes. Par exemple : sur les 4332 articles classés « devices telecom » par l'algorithme NMF, 3265 le sont aussi par l'algorithme LDA (soit plus de 75%). En revanche, lorsque le thème LDA est trop large (par exemple Entertainment), le thème NMF permet d'obtenir une information plus précise.

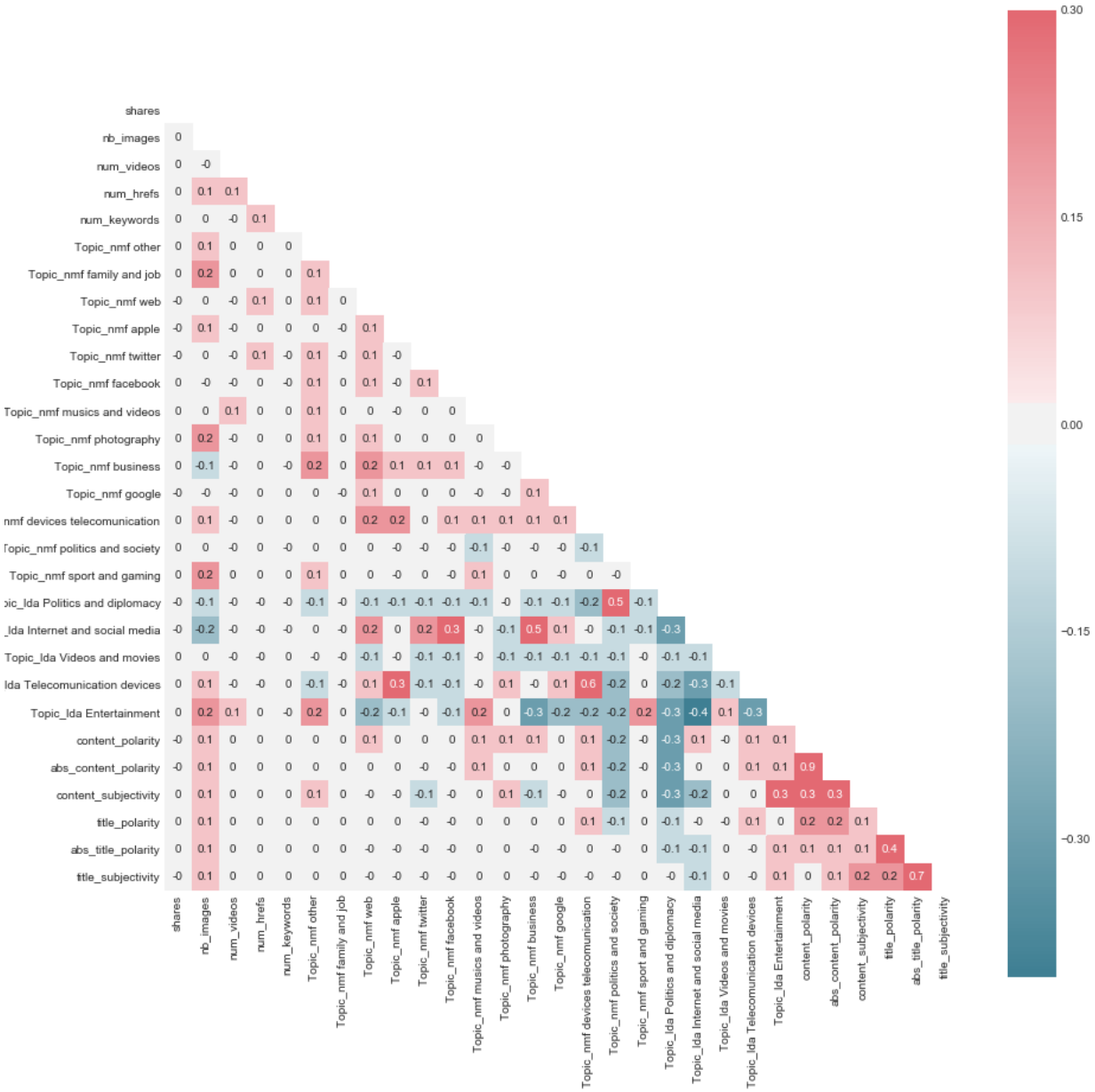
5.2 Analyse des liens entre variables

La seconde étape dans l'analyse descriptive de variable est celle de la prise en compte des liens entre chacune des variables. Pour cela, nous commencerons par identifier les liens possibles entre chacune des variables puis nous nous pencherons sur la variable *shares* : notre objectif de prédiction.

5.2.1 L'ensemble des variables

La méthode la plus utilisée pour déterminer le sens de la relation entre deux variables quantitatives est l'analyse de la corrélation.

FIGURE 19 – Corrélation entre les variables quantitatives



D'après la figure précédente (cf. Figure 19), on constate qu'il existe de nombreuse corrélation nulles. C'est à dire qu'il n'existe pas de liens statistique entre les deux variables.

Toutefois, on observe qu'il existe des liens positifs et négatifs plus ou moins forts entre certaines d'entre-elles. Certaines variables représentant des thèmes provenant des deux méthodes se rap-

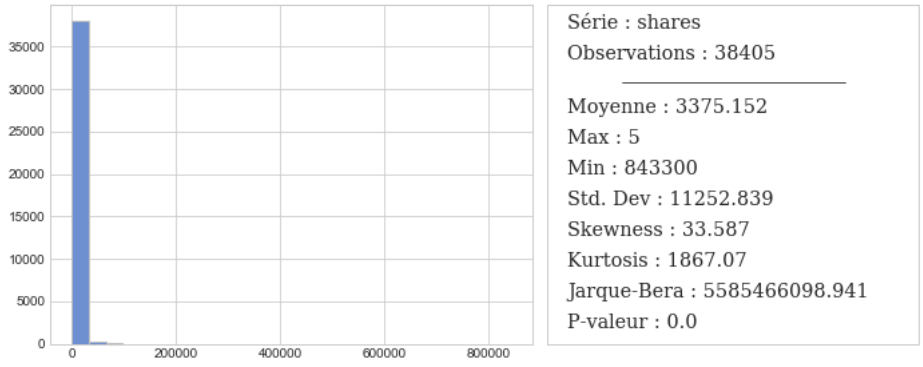
prochent comme « Topic lda politic and diplomacy » et « Topic nmf politics and society » avec un coefficient de corrélation de 0,5. C'est également le cas de la variable « Topic lda Internet and social media » qui est notamment corrélée avec « Topic nmf business » (0.5), « Topic nmf facebook » (0.3), « Topic nmf twitter » (0.2) et « Topic nmf web » (0.2). On peut supposer que ces corrélations sont liés au fait que le nombre de thème dans la méthode NMF est plus important et donc que les thèmes de la méthode LDA sont fondus.

Outre cela, on retrouve le plus fort coefficient de corrélation entre les variables jumelles « abs content polarity » et « content polarity » (coef. de 0,9, l'une étant la valeur absolue de l'autre). On remarque également que les variables de l'analyse de sentiment sont globalement corrélées.

5.2.2 Liens avec la variable à prédire : *shares*

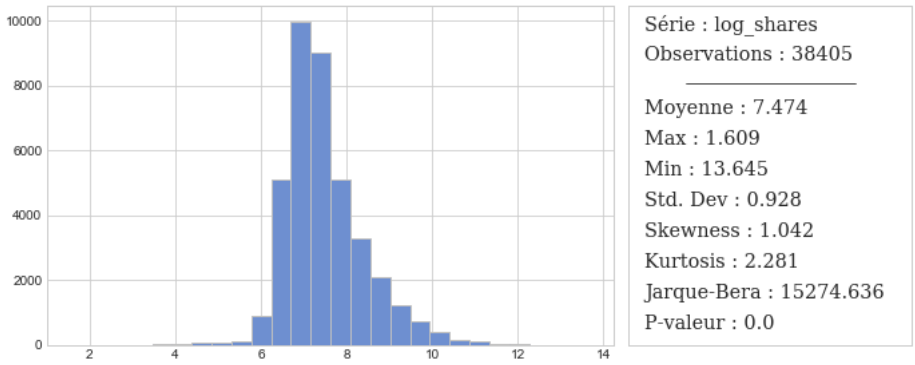
La coeur de notre sujet est de prédire la variable *shares*. Toutefois, nous devons dans un premier temps observer quelques indicateurs statistiques pour se donner une idée de sa forme et de ses caractéristiques.

FIGURE 20 – Statistiques descriptive de *Shares*



La Figure 20 présente quelques informations descriptives de la variable *shares*. On constate que cette série varie entre 5 et 843 000 et que la moyenne est située à 3300. La médiane est elle située à 1400. Ceci indique une forte dissymétrie de la distributions. La valeur du kurtosis confirme ce ressentie : 1800. Visuellement, on remarque que les valeurs sont très étalées vers la droite. Enfin, on peut terminer par un test de normalité (celui de Jarque-Bera) qui conclut à la non normalité des données avec une statistique de test incroyablement élevée. Cette distribution semble problématique pour utiliser des modèles paramétrique, on peut donc essayer de transformer cette variable en logarithme pour observer comment elle se comporte sous cette forme.

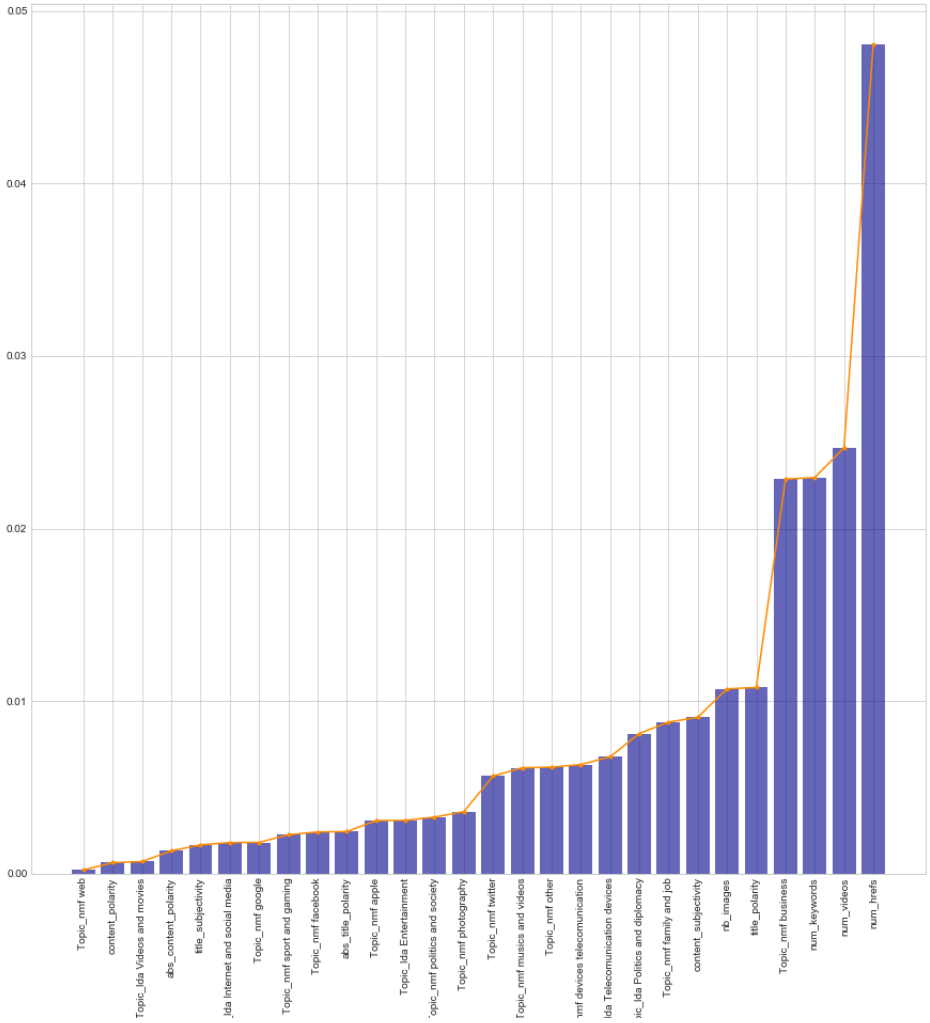
FIGURE 21 – Statistiques descriptive de *logshares*



La Figure 21 montre cette fois-ci la variable *shares* transformée en log. Tout d'abord, les valeurs varient entre 1,6 et 13, ce qui permet d'obtenir un écart-type beaucoup plus faible que précédemment. Le kurtosis est également plus faible que prédécemment mais reste tout de même positif (donc beaucoup de valeurs restent à droite de la distribution). Enfin, visuellement la variable semble davantage de forme normale que prédécemment. Toutefois, la statistique de Jarque-Bera ne conclut toujours pas à la normalité des données. La p-valeur reste toujours très faible (inférieur à 0,001). En conclusion, on doit rejeter l'hypothèse nulle de normalité.

On peut maintenant se concentrer sur les liens des autres variables du jeu de données avec notre variable cible.

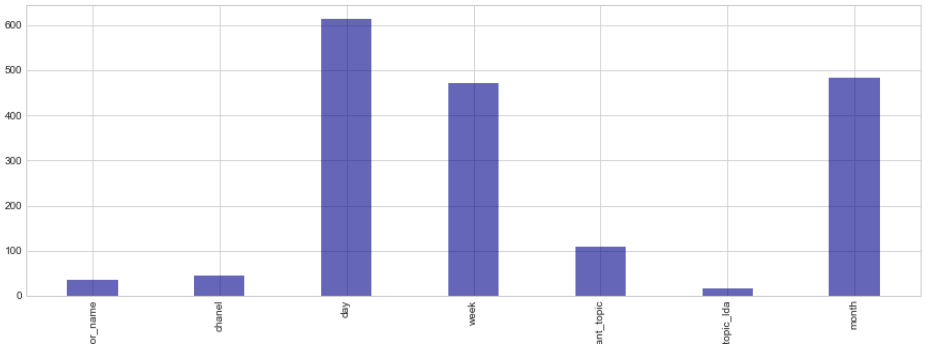
FIGURE 22 – Corrélation de pearson avec *shares*



Une première mesure est d'observer l'histogramme des corrélations de chacune des variables avec notre variable cible (cf. Figure 22).

Pour les variables qualitatives, nous testons leur liens à l'aide du test de Kruskall-Wallis et de la statistique de test associée. Pour rappel, une valeur élevée de la statistique indique qu'il existe une forte liaison entre la variable qualitative et la variable cible. Les variables présente pour les tests sont les suivantes : Nom, jour de la semaine, week-end, dominant topic LDA et NMF, mois.

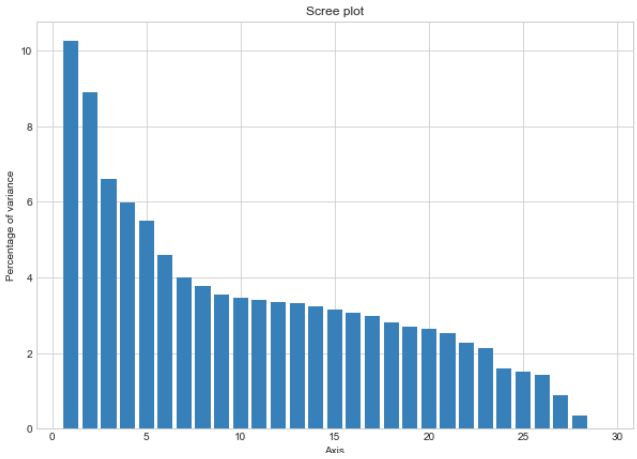
FIGURE 23 – Statistiques du test de Kruskall-Wallis



5.3 Analyse exploratoire : l'Analyse en Composante Principale

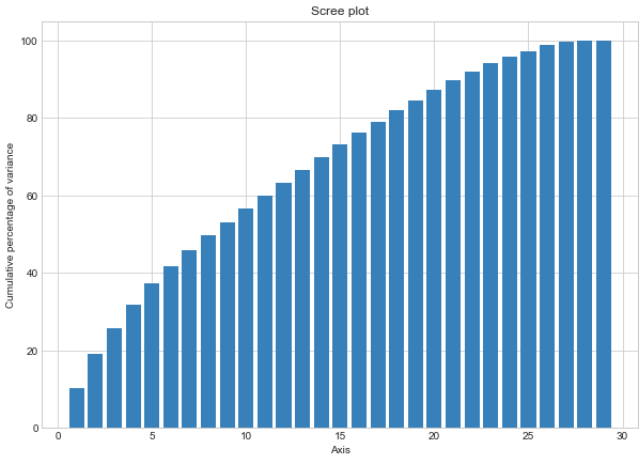
Pour terminer cette partie d'analyse descriptive, nous allons utiliser une Analyse en Composante Principale pour observer notre jeu de données sur un nombre plus faible de dimensions.

FIGURE 24 – Inertie en pourcentage de chaque axe



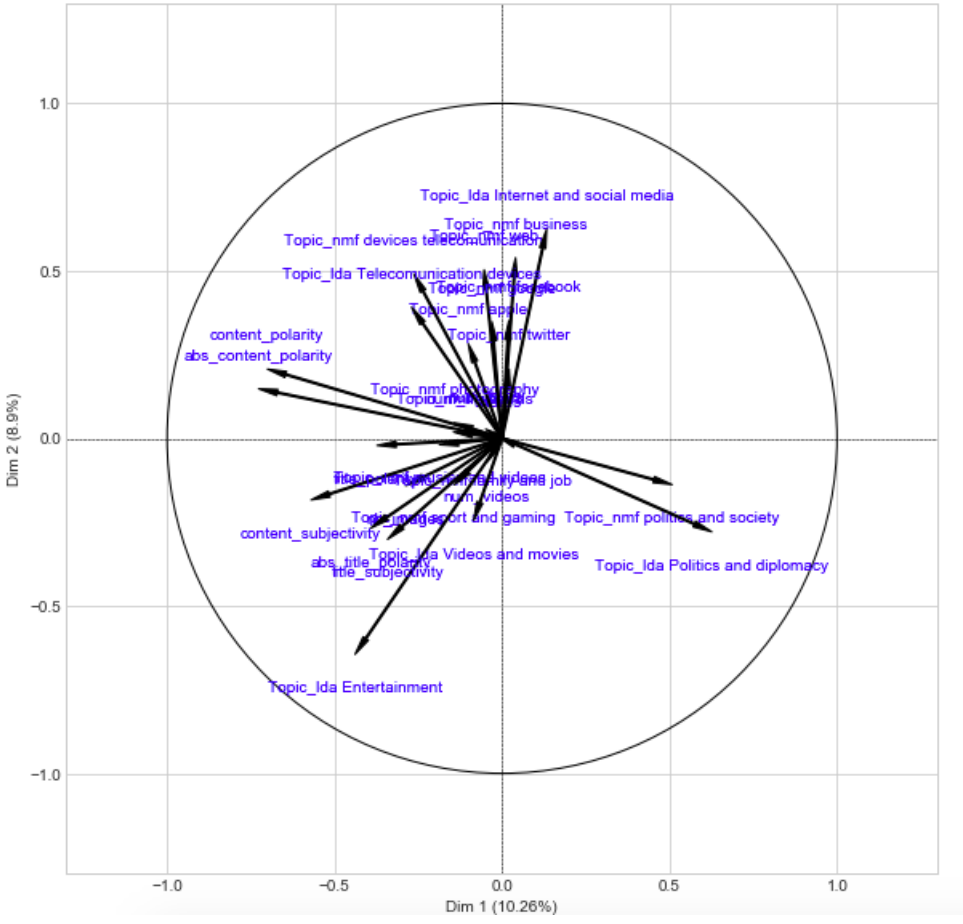
Le graphique 24 indique l'inertie en pourcentage de chaque axes. On remarque rapidement que cette inertie est très faible, les premier et second axes ne comportant seulement que 10% et 8% de l'inertie totale. Cela posera problème lors de la représentation graphique sur un cercle des corrélations.

FIGURE 25 – Inertie en pourcentage cumulé



On retrouve le constat précédent dans le graphique 25 : afin d'obtenir une inertie satisfaisante d'environ 80%, il faudrait prendre près de 20 dimensions.

FIGURE 26 – Cercle des corrélations



Cela se ressent dans le cercle des corrélations : il est difficile de discerner des groupes corrélés. Cependant, on remarque certains rapprochements : la variable « content_polarity » avec sa jumelle « abs_content_polarity » ou encore la variable « Topic nmf politics and society » avec son homologue « Topic lda Politics and diplomacy ».

6 Modèle de régression : prédiction de la variable *shares*

Rappelons que l'objectif est ici de prédire au mieux le nombre de partages d'un article. Comme dit précédemment, une fonction logarithmique est utilisée pour linéariser la variable cible, ce qui permet une meilleure prédiction même avec un modèle linéaire similaire. Cette variable étant une variable quantitative, les modèles adaptés seront donc des solutions à des problèmes de régression.

Pour ce faire nous allons donc commencer par effectuer une régression linéaire multiple comme premier modèle. Puis, nous utiliserons des modèles non linéaires (*e.g.* arbres de décision), afin de comparer les différents résultats obtenus. L'enjeu de l'utilisation d'un arbre de régression réside dans le fait de trouver les variables les plus importantes à la prédiction du nombre de partages d'un article.

6.1 Régression linéaire

Tout d'abord, nous commencerons par un modèle simple : prédire le nombre de partages d'un article à partir des différentes variables quantitatives disponible dans notre jeu de données : le nombre d'images, le nombre de vidéos, le nombre de référence d'un article, le nombre de mots clés, et utiliser un encodage si l'article est publié le week-end ou en semaine, et un second encodage concernant le mois de publication de l'article.

Pour ce premier modèle, la pratique veut que nous découpions notre jeu de données en deux échantillons :

— un échantillon de train : ici nous utiliserons 70% du jeu de données.
— un échantillon de test : ici nous utiliserons 30% du jeu de données.

Pour chacun des échantillons, il nous faudra ajouter une constante (intercept) pour que le modèle soit valide. Cette constante est la valeur moyenne attendue de Y lorsque tous les X sont nulles. Pour une régression simple avec un prédicteur X, si X est égal à 0, la constante est simplement la valeur moyenne attendue de Y à cette valeur. Si X n'est pas égal à 0, alors la constante n'a pas de signification intrinsèque.

FIGURE 27 – OLS : shares

OLS Regression Results

Dep. Variable:	shares	**R-squared:**	0.021
Model:	OLS	**Adj. R-squared:**	0.021
Method:	Least Squares	**F-statistic:**	115.3
Date:	Tue, 21 Apr 2020	**Prob (F-statistic):**	5.14e-121
Time:	19:44:47	**Log-Likelihood:**	-36019.
No. Observations:	26883	**AIC:**	7.205e+04
Df Residuals:	26877	**BIC:**	7.210e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	7.1859	0.022	320.109	0.000	7.142	7.230
nb_images	0.0010	0.001	1.510	0.131	-0.000	0.002
num_videos	0.0068	0.001	5.015	0.000	0.004	0.009
num_hrefs	0.0073	0.001	14.277	0.000	0.006	0.008
num_keywords	0.0238	0.003	7.967	0.000	0.018	0.030
week	0.2282	0.017	13.645	0.000	0.195	0.261

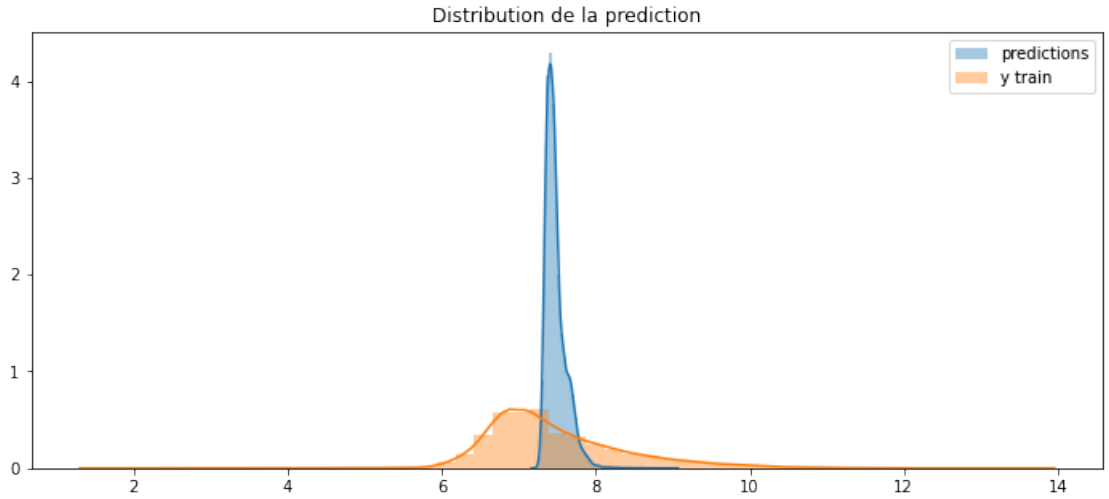
Omnibus:	5066.731	**Durbin-Watson:**	1.996
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	11514.243
Skew:	1.081	**Prob(JB):**	0.00
Kurtosis:	5.367	**Cond. No.**	69.9

On remarque par exemple que *week* a un effet positif sur le nombre de *shares*. Au même titre que le nombre d'image, de mots clés, de vidéos et de href ont une influence positive. On retient les informations suivante de notre modèle :

— Le MSE est de : 0.853
— Le R^2 est de : 0,021

Ce R^2 très faible nous montre que notre modèle a un pouvoir explicatif très limité : il n'est capable d'expliquer que 2% de la variabilité de notre cible. Aussi, les distributions comparées de la prédictions et des vraies valeurs ci-dessous nous montrent clairement que la constante est simplement la variable la plus importante du modèle. En effet, on a une distribution prédite très concentrée autour de sa moyenne, témoignant ainsi des coefficients très proches de 0, même lorsqu'ils sont significativement positifs.

FIGURE 28 – Distribution des *shares* et de nos prédictions



6.2 Régression linéaire avec des variables encodées

Nous allons ici agrémenter nos données avec des variabes catégorielles qui nous semblent être pertinentes. De nombreux algorithmes de machine learning ne peuvent pas fonctionner directement sur des données catégorielles. Ils exigent que toutes les variables d'entrée et de sortie soient numériques.

En général, c'est surtout une contrainte d'efficacité des algorithmes plutôt que des limitations strictes sur les algorithmes eux-mêmes. Cela signifie que les données catégorielles doivent être converties sous une forme numérique.

Il existe deux méthode pour convertir les données :

- **Integer Encoding :** Dans un premier temps, chaque valeur de catégorie unique se voit attribuer une valeur entière. Par exemple, "rouge" est 1, "vert" est 2 et "bleu" est 3. Cette méthode est appelée label encoding ou integer encoding et est facilement réversible. Les valeurs entières ont une relation ordonnée naturelle entre elles et les algorithmes peuvent être capables de comprendre et d'exploiter cette relation.
- **One-Hot Encoding :** Pour les variables catégorielles où il n'existe pas de relation ordinale de ce type, le codage des nombres entiers n'est pas suffisant. En fait, l'utilisation de ce codage et le fait de laisser le modèle supposer un ordre naturel entre les catégories peuvent entraîner de mauvaises performances ou des résultats inattendus.

Dans ce cas, un one-hot encoding peut être appliqué. C'est là que la variable codée est entierement supprimée et qu'une nouvelle variable binaire est ajoutée pour chaque valeur unique. Dans l'exemple de la variable "couleur", il y a 3 catégories et donc 3 variables binaires sont nécessaires. Une valeur "1" est placée dans la variable binaire pour la couleur et des valeurs "0" pour les autres couleurs.

Pour notre prochain modèle, nous avons à nouveau divisé notre dataset en train/test. Nous avons en suite effectué un one-hot encoding pour les variables *day*, *month*, *chanel* et *dominant_topic*.

On remarque ainsi une légère amélioration du R^2, même si les résultats restent mauvais. La plupart des coefficients sont positifs et significativement différents de 0. Au niveau des coefficients, on peut voir que dans la catégorie jours de la semaine, ceux sont les jours samedi et dimanche qui ont les coefficients les plus élevés, ce qui signifierait qu'il serait préférable de publier son article le week-end, afin de maximiser son audience.

Dep. Variable :	shares	**R-squared :**	0.033
Model :	OLS	**Adj. R-squared :**	0.032
Method :	Least Squares	**F-statistic :**	23.01
Date :	Tue, 21 Apr 2020	**Prob (F-statistic) :**	5.39e-164
Time :	20 :17 :30	**Log-Likelihood :**	-35851.
No. Observations :	26883	**AIC :**	7.178e+04
Df Residuals :	26842	**BIC :**	7.212e+04
Df Model :	40		

	coef	std err	t	P> |t|	[0.025	0.975]
const	5.1232	0.082	62.137	0.000	4.962	5.285
nb_images	0.0004	0.001	0.532	0.595	-0.001	0.002
num_videos	0.0066	0.001	4.889	0.000	0.004	0.009
num_hrefs	0.0073	0.001	14.286	0.000	0.006	0.008
num_keywords	0.0240	0.003	8.053	0.000	0.018	0.030
is_Friday	0.7347	0.018	40.370	0.000	0.699	0.770
is_Monday	0.6629	0.018	37.661	0.000	0.628	0.697
is_Saturday	0.9153	0.023	38.976	0.000	0.869	0.961
is_Sunday	0.8687	0.023	38.428	0.000	0.824	0.913
is_Thursday	0.6621	0.017	38.786	0.000	0.629	0.696
is_Tuesday	0.6340	0.017	36.959	0.000	0.600	0.668
is_Wednesday	0.6454	0.017	37.655	0.000	0.612	0.679
is_Apr	0.5818	0.020	29.349	0.000	0.543	0.621
is_Aug	0.3608	0.019	18.511	0.000	0.323	0.399
is_Dec	0.4083	0.021	19.445	0.000	0.367	0.449
is_Feb	0.4482	0.020	21.888	0.000	0.408	0.488
is_Jan	0.4354	0.020	21.365	0.000	0.395	0.475
is_Jul	0.3323	0.019	17.400	0.000	0.295	0.370
is_Jun	0.3119	0.020	15.778	0.000	0.273	0.351
is_Mar	0.6169	0.020	30.315	0.000	0.577	0.657
is_May	0.4386	0.020	21.796	0.000	0.399	0.478
is_Nov	0.4169	0.020	20.656	0.000	0.377	0.457
is_Oct	0.3748	0.019	20.024	0.000	0.338	0.412
is_Sep	0.3973	0.020	20.334	0.000	0.359	0.436
is_Business	0.5957	0.106	5.603	0.000	0.387	0.804
is_Culture	0.6238	0.106	5.874	0.000	0.416	0.832
is_Entertainment	0.5912	0.106	5.588	0.000	0.384	0.799
is_Shopping	0.8924	0.815	1.095	0.273	-0.705	2.489
is_Social Good	0.6455	0.119	5.443	0.000	0.413	0.878
is_Tech	0.6190	0.106	5.833	0.000	0.411	0.827
is_U.S.	0.5902	0.110	5.352	0.000	0.374	0.806
is_World	0.5656	0.106	5.321	0.000	0.357	0.774
is_apple	0.3108	0.030	10.336	0.000	0.252	0.370
is_business	0.3412	0.021	16.226	0.000	0.300	0.382
is_devices telecomunication	0.3778	0.020	19.020	0.000	0.339	0.417
is_facebook	0.4952	0.027	18.264	0.000	0.442	0.548
is_family and job	0.4840	0.055	8.866	0.000	0.377	0.591
is_google	0.4273	0.027	15.668	0.000	0.374	0.481
is_musics and videos	0.3641	0.018	20.720	0.000	0.330	0.399
is_other	0.3880	0.017	22.371	0.000	0.354	0.422
is_photography	0.4399	0.037	11.783	0.000	0.367	0.513
is_politics and society	0.3688	0.020	18.778	0.000	0.330	0.407
is_sport and gaming	0.3690	0.021	17.441	0.000	0.328	0.411
is_twitter	0.3519	0.024	14.590	0.000	0.305	0.399
is_web	0.4052	0.024	16.624	0.000	0.357	0.453

Omnibus :	5274.992	**Durbin-Watson :**	1.999
Prob(Omnibus) :	0.000	**Jarque-Bera (JB) :**	12228.712
Skew :	1.114	**Prob(JB) :**	0.00
Kurtosis :	5.439	**Cond. No.**	1.66e+16

6.3 Sklearn LinearRegression et validation croisée

Utilisons maintenant un autre modèle de la bibliothèque Sklearn afin de pouvoir appliquer une validation croisée : **Linear Regression**

On retrouve les métriques suivantes :

— R2 score : 0.0309
— MSE train : 0.8431
— MSE test : 0.8086

6.3.1 Cross validation

La cross validation, est une technique de validation de modèle qui permet d'évaluer les résultats d'un modèle généralisé à un ensemble de données indépendantes. Elle est principalement utilisée dans des contextes où l'objectif est la prédiction et où l'on veut estimer la précision d'un modèle prédictif dans la pratique. Après avoir utilisé notre cross validation, nous obtenons :

— MSE : 7.072515282403405e+16
— RMSE : 84098248.6274 +/- 252294743.1118

Ces résultats confirment non que le modèle n'est pas très performant et donc pas très adapté pour effectuer une prédiction. On peut donc essayer d'utiliser d'autre modèle, pouvant tenter de repérer des effets non linéaires.

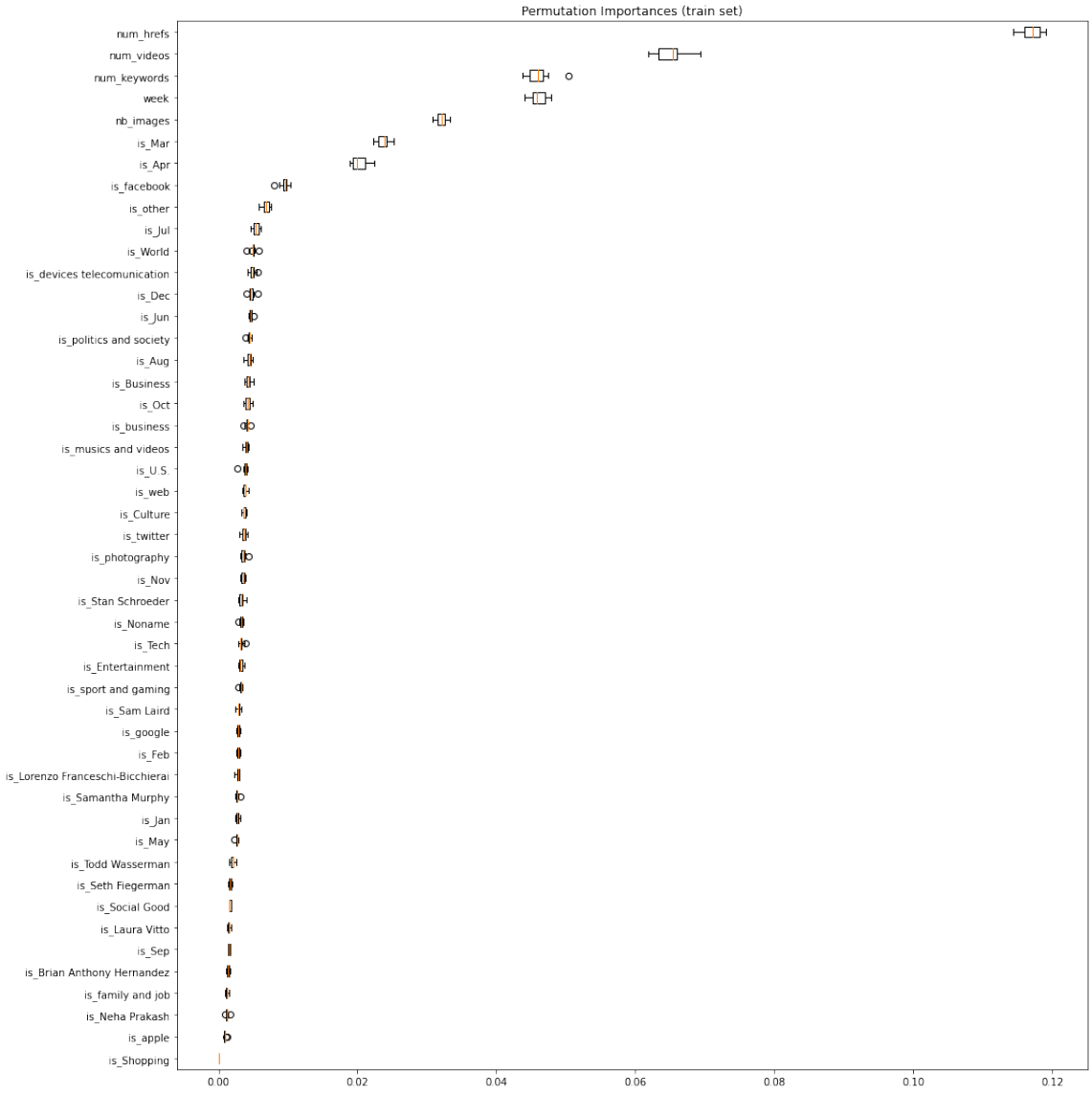
6.4 Régression non linéaire

La régression non linéaire a pour but d'ajuster un modèle non linéaire pour un ensemble de valeurs afin de déterminer la courbe qui se rapproche le plus de celle des données de Y en fonction de X. Pour ce modèle, nous avons utilisé les variables :
nb_images, *num_videos*, *num_hrefs*,*num_keywords*,*week*, *month*,*chanel*,*dominant_topic*, *author_name*.
Dont *month*,*chanel*,*author_name*,*dominant_topic* qui ont été encodées grace à *get_dummies*.

6.5 Permutation feature importance

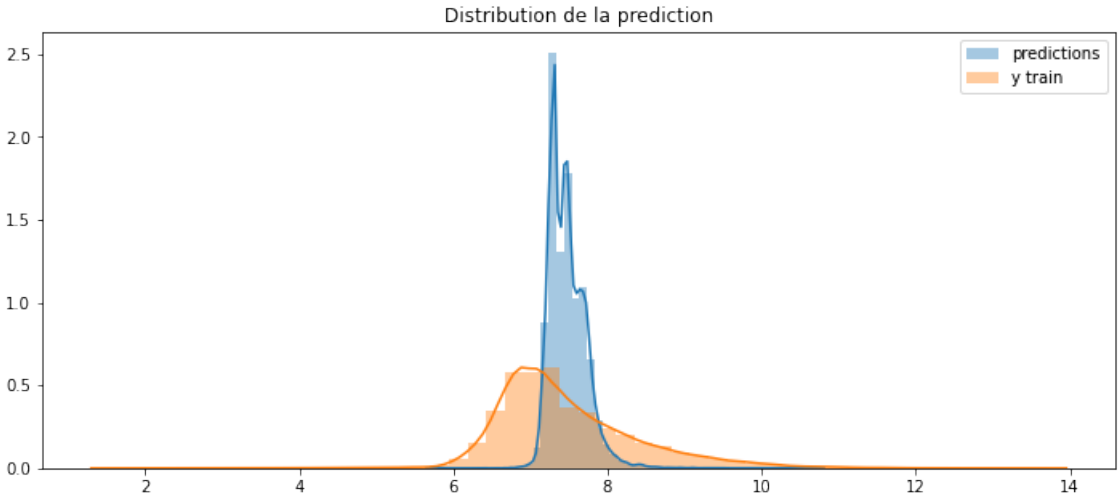
Permutation feature importance est une technique d'inspection du modèle. Cette technique est particulièrement utile pour les estimateurs non linéaires. La permutation feature importance est définie comme étant la diminution du score d'un modèle lorsqu'une feature est mélangée de manière aléatoire. Cette procédure rompt la relation entre les features et la cible, la diminution du score du modèle est donc un indicateur de la mesure dans laquelle le modèle dépend de la feature. On peut donc afficher l'importance des caractéristiques des estimateurs pour un ensemble de données donné.

FIGURE 29 – Permutation importance



On remarque bien ici que les données corrélées à *Shares* sont les plus discriminante dans cette régression. Au niveau de la comparaison des distributions des vraies valeurs et des valeurs prédites, on remarque que la forme de la courbe est moins univoque et plus étalée que ce que nous avions obtenu avec la régression linéaire.

FIGURE 30 – Distribution des *shares* et de nos prédictions



On retrouve les métriques suivantes concernant notre cross validation sur le Random Forest Regressor :

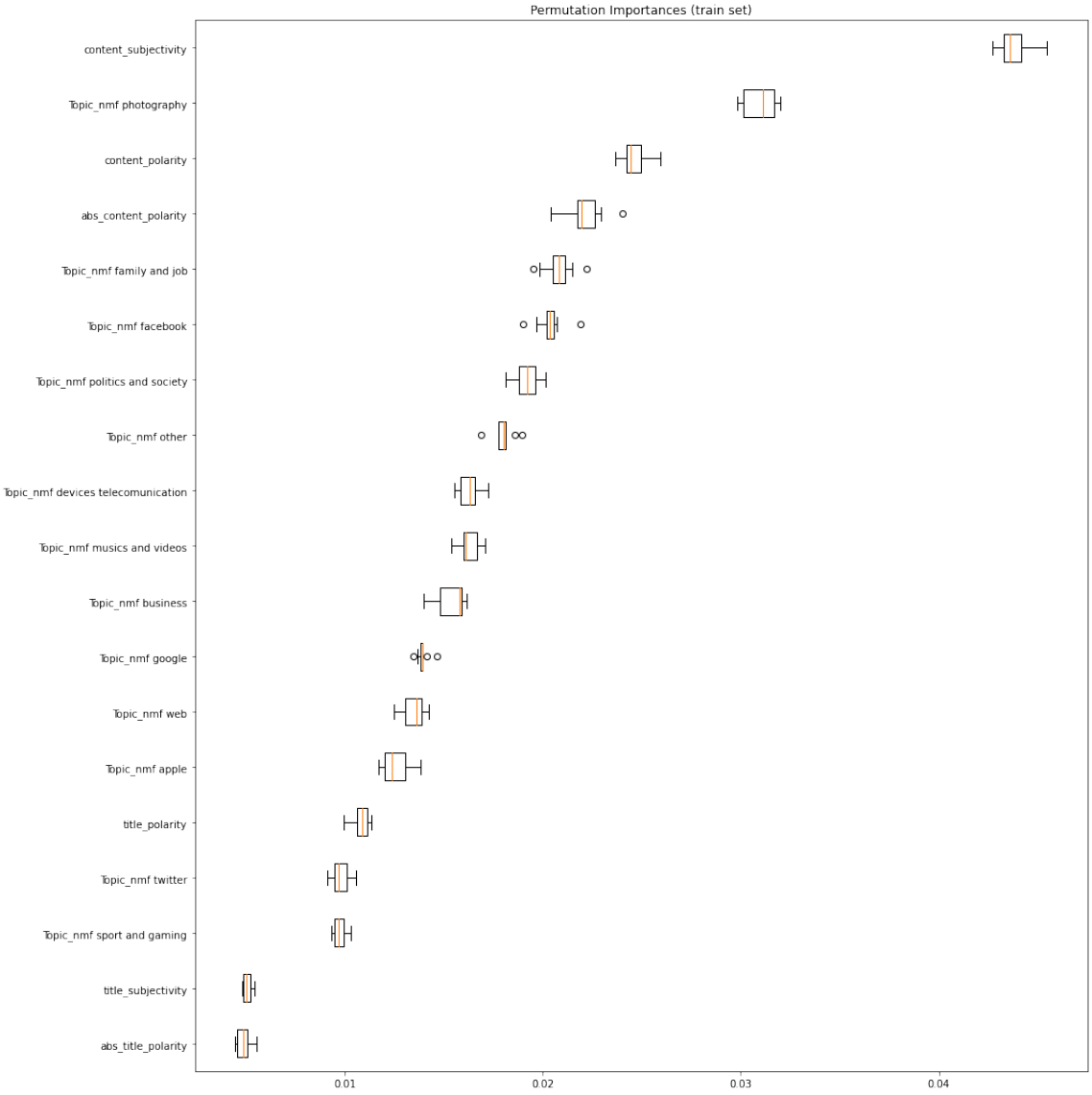
— MSE : 0.8467873963268125
— RMSE : 0.9184 +/- 0.0581

6.5.1 Random Forest : topics NMF et sentiments

Intéressons nous maintenant aux topics NMF. Nous allons les coupler à l'analyse de sentiment pour essayer de prédire au mieux le nombre de partages. Les scores indiquent :

— R2 score : -0.0092
— MSE train : 0.7905
— MSE test : 0.8421

FIGURE 31 – Distribution des *shares* et de nos prédictions

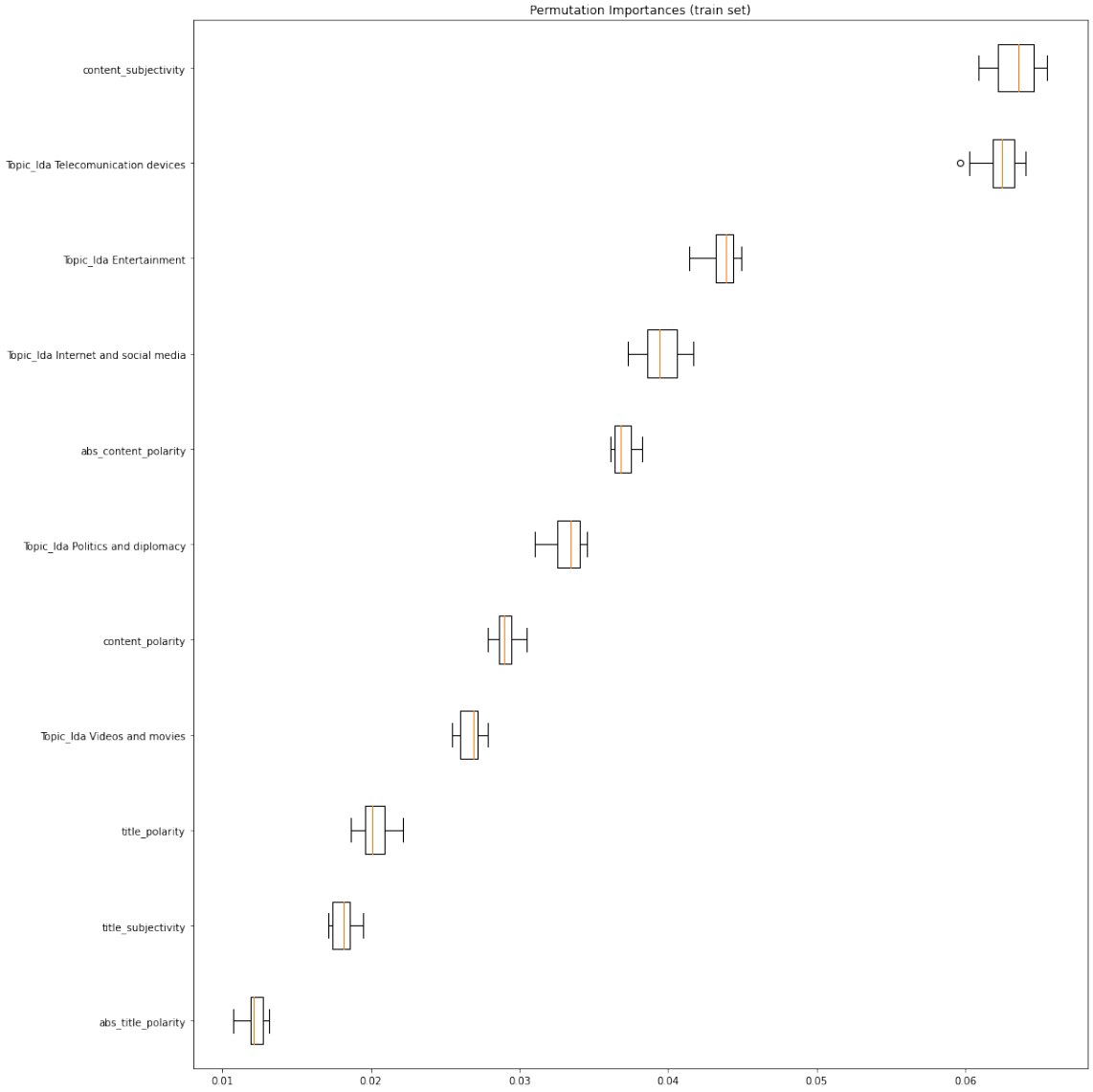


6.5.2 Random Forest : topics LDA et sentiments

Le modèle pour les variables des thèmes sur le LDA :

- R2 score : -0.0148
- MSE train : 0.7896
- MSE test : 0.8468

FIGURE 32 – Distribution des *shares* et de nos prédictions



6.6 Modèle final

L'idée ici va être de mettre toutes les variables dans le modèle puis de sélectionner les plus importantes.

Voici les 30 variables les plus importantes sur lesquelles nous allons appliquer un Random Forest Regressor :

- *is_sportandgaming*
- *is_business*
- *is_twitter*
- *is_StanSchroeder*,
- *is_politicsandsociety*

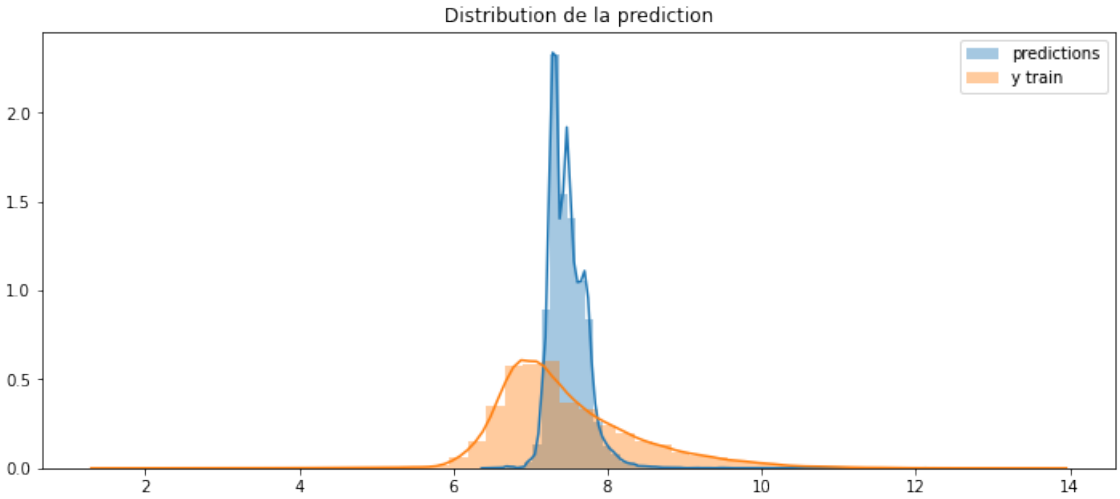
— is_Aug
— is_Noname
— is_World
— is_Feb
— $is\_Entertainment$
— $is\_U.S.$
— is_Dec
— is_Oct
— $is\_Culture$
— is_Nov
— $is\_devicestelecomunication$
— $is\_Business$
— is_Jun
— $is\_musicsandvideos$
— is_Jul
— is_Tech
— $is\_facebook$
— is_other
— is_Apr
— is_Mar
— nb_images
— $week$
— $num\_keywords$
— num_videos
— num_hrefs

On retrouve les métriques suivantes pour ce modèle :

— R2 score : 0.0293
— MSE train : 0.7428
— MSE test : 0.81
— RMSE : 0.9184 +/- 0.0581

On revient ainsi à des niveaux de performance similaires à ceux de notre modèle linéaire de base, avec l'ajout des variables one-hot encoded, que ce soit en termes de R^2 ou de MSE.

FIGURE 33 – Distribution des *shares* et de nos prédictions



6.7 Comment améliorer nos résultats ?

En prenant la décision de ne pas utiliser le jeu de données original, nous devions par nous même reconstituer ce jeu de données. Or, il y a certaines variables que nous n'avons pas reproduites, comme le nombre de mots par article par exemple. C'est à l'issue de nos prédiction que nous nous en sommes rendus compte et qu'il pourrait être intéressant de les rajouter dans le but d'obtenir de meilleurs résultats.

Aussi, une approche régressive ici semble se confronter au problème des valeurs extrêmes. La prochaine partie consistera à traiter notre problème comme un problème de classification, et non plus de régression.

7 Modèle de classification : prédiction de la variable *popular*

7.1 Discrétisation de la variable à prédire

La discrétisation est une méthode de transformation d'une variable quantitative en une variable qualitative. Autrement dit, c'est le fait de réaliser un découpage en classe d'une variable quantitative. Cette méthode possède plusieurs avantages :

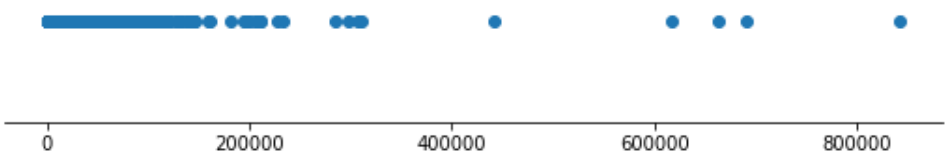
- Certaines techniques statistiques ne fonctionnent que sur des variables qualitatives (notamment les classifieurs).
- Minimiser l'impact des valeurs extrêmes sur le modèle.
- Faciliter l'interprétation des résultats.

Toutefois, si nous voulons utiliser une méthode de discrétisation, la difficulté réside dans le fait de maintenir un bon équilibre concernant la perte d'information : doit-on garder un nombre faible de classes au risque de perdre trop d'informations, ou un nombre élevé de classes et ainsi

réduire l'utilité de la discrétisation ? Il faut noter que la meilleure discrétisation possible reste celle du métier. Toutefois, lorsque l'expertise du métier n'est disponible, nous devons recourir à des méthodes statistiques guidées par les caractéristique des données.

Nous avons déjà vu sur la Figure XX que la distribution de notre variable cible est très asymétrique (son coefficient d'asymétrie est bien au dessus de 0). On affiche également la variable que nous souhaitons discrétiser (*shares*) sur un axe horizontal afin de se donner une meilleure idée de la répartition des données (Figure 34).

FIGURE 34 – Répartition des valeurs pour la variable *shares*



7.1.1 Les méthodes usuelles de discrétisation

Plusieurs méthodes de discrétisation existent. Ces dernières ont toutes leurs avantages et leurs inconvénients mais leurs choix dépendent de la nature des données.

Il existe tout d'abord les méthodes usuelles :

— **Quantiles** : il s'agit de réaliser une partition des données en des classes de taille égale (même nombre d'individus par classe). L'avantage est de créer des classes équilibrées. Cependant, cette méthode ne prend pas en compte la distribution des valeurs. Ainsi, des valeurs très différentes peuvent se retrouver dans la même classe ou encore des valeurs similaires peuvent se retrouver dans des classes différentes.
— **Étendue** : il s'agit de diviser les données avec des bornes de mêmes amplitudes. Cette méthode facile à interpréter est souvent utilisée pour réaliser des classes d'âges. En dehors de ce cas, elle est peu utilisée car elle ne convient pas aux distributions trop asymétriques (les classes pourraient être très inégales).

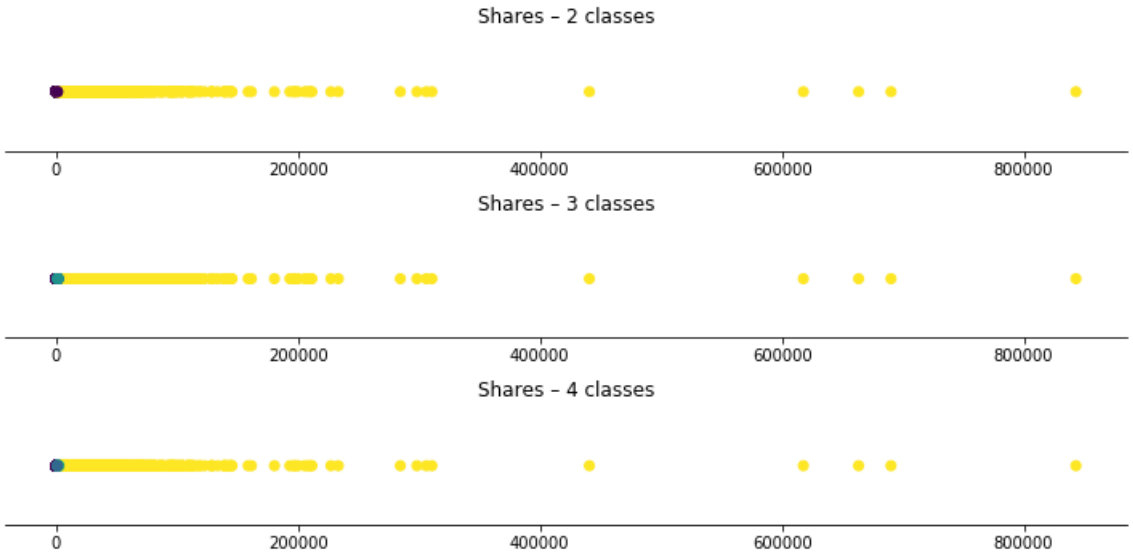
Pour ces deux méthodes, les seuils retenus ne tiennent pas compte des caractéristiques des données. Elles posent également la question du choix du nombre de classe. Il existe différents indicateurs permettant de choisir le nombres, k, de classes (Tableau 4). Après avoir calculés les indicateurs provenant de la littérature, nous remarquons que le nombre de classes proposées est trop important pour répondre à notre problématique. Nous décidons donc de limiter notre choix entre deux et quatre classes.

TABLE 4 – Indicateurs du nombre de classes idéal

	Nombre de classes
Brooks-Carruthers	22
Huntsberger	16
Sturges	15
Scott	722
Freedman-Diaconis	7673

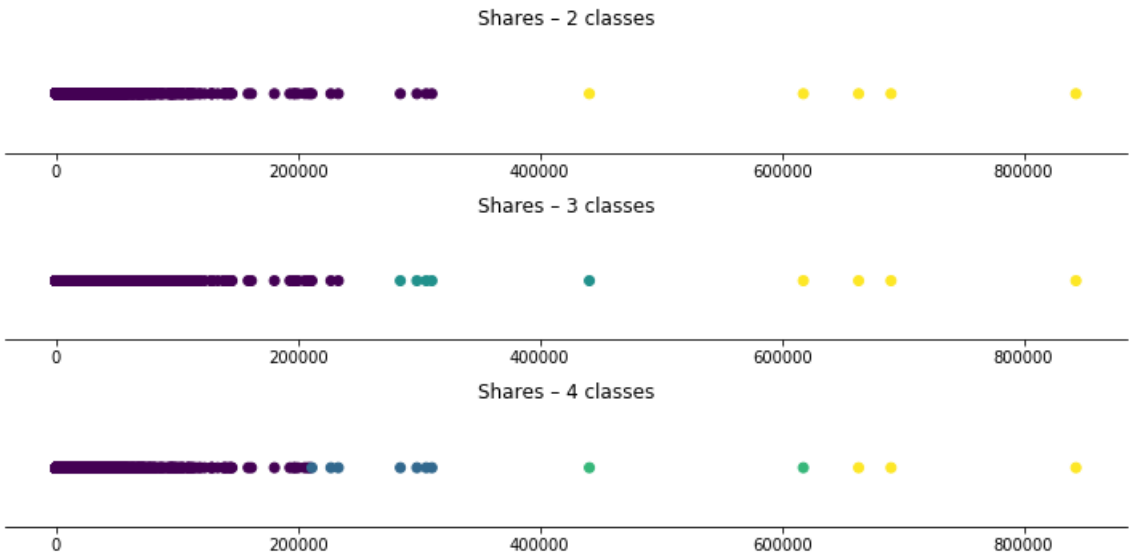
Discrétisation par les quantiles : Pour la première méthode, la discrétisation se fait un niveau de la médiane lorsqu'il y a deux classes. Celle-ci étant située à 1400. Avec cette méthode, les deux classes auraient la même taille. Cette méthode reste valable et c'est celle qui a été choisie par les auteurs de l'article de recherche du jeu de données initial. Lorsque plus de classes sont construites, les bornes de séparations sont sont 1100 et 2100 pour trois classes ; 946, 1400 et 2800 pour quatre classes (Figure 35).

FIGURE 35 – Discrétisation par les quantiles



Discrétisation par l'étendue : Avec cette méthode, on prend uniquement en compte les valeurs de la variable. Toutefois, nous estimons que cette méthode pas adaptée car elle amène à considérer un article partagé 200000 fois comme non-populaire. Pour que la discrétisation soit de bonne qualité, il faudrait prendre en compte la dispersion des données. Les bornes issues de cette méthode sont les suivantes (voir Figure 36) : pour deux classes (421 652), pour trois classes (281 103 et 562 201), pour quatre classes (210828, 421652 et 632476).

FIGURE 36 – Discrétisation par l'étendue



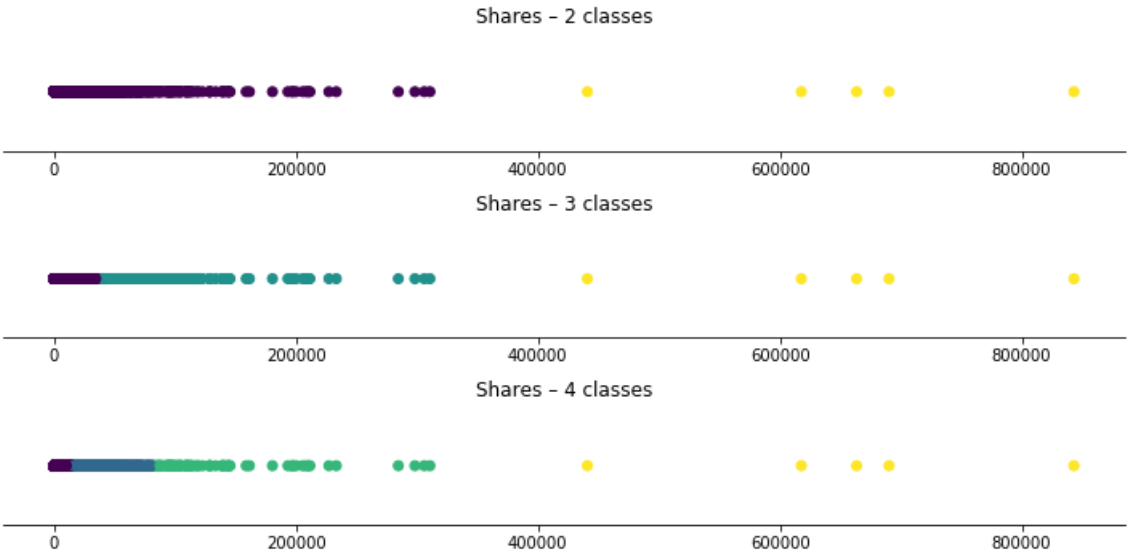
7.1.2 Méthodes par algorithmes : KMeans, Arbre de régression ou CAH

Il existe d'autre méthode qui tiennent compte de la dispersions des données. Les données peuvent être organisées en « paquets » plus ou moins homogènes. Le nombre de classes étant difficile à estimer, on s'intéresse donc aux caractéristiques de dispersion des données. On se limitera donc à une séparation binaire.

- **K-Means** : il s'agit d'une méthode utilisant l'algorithme du K-Moyennes pour le partitionnement des données. On initialise un nombre de classes k et ces dernières sont attribuées selon la classe la plus proche en termes de distance. Cette méthode est très dépendante de l'initialisation des clusters et aux valeurs extrêmes.
- **Classification Hiérarchique** : La séparation des données se fait à partir d'un algorithme de classification hiérarchique (ascendant ou descendant). On commence par N classes, puis on regroupe les points dont le regroupement minimise un critère d'agrégation (méthode de Ward par exemple), et on recommence jusqu'à avoir une même classe pour toutes les données. La méthode de sélection se fait par la visualisation d'un dendrogramme.
- **Arbre de décision** : il s'agit de séparer les données par un algorithme d'arbre de décision. À partir des branches de l'arbre on pourra choisir un nombre de classes optimales qui séparera au mieux les données selon la métrique choisie.

Discrétisation par les K-Means : La discrétisation par le KMeans semble être meilleure comparé à l'utilisation des méthodes usuelles. Toutefois, celle en deux classes reste assez problématique pour définir un article populaire ou non populaire. La discrétisation en trois classes, elle, permet de bien remarquer trois types d'article distincts. Les bornes des classes sont les suivantes : deux classe (327265), trois classes (39316 et 363555), et quatre classes (15533, 81971 et 393201). La méthode du KMean ne semble pas répondre à notre problématique.

FIGURE 37 – Discrétisation par les K-means



Discrétisation par CAH : On trouve les bornes suivantes : de 0 à 43 700 partages, de 43 700 à 441 000 partages et plus de 441 000 partages (Figure 38 et Figure 39). Cette méthode affiche un nombre élevé de partage pour la première classe, ce qui ne semble pas cohérent avec notre problématique. De plus, elle créé des classes très déséquilibrées, nous décidons donc de ne pas la retenir.

FIGURE 38 – Dendrogramme de la CAH

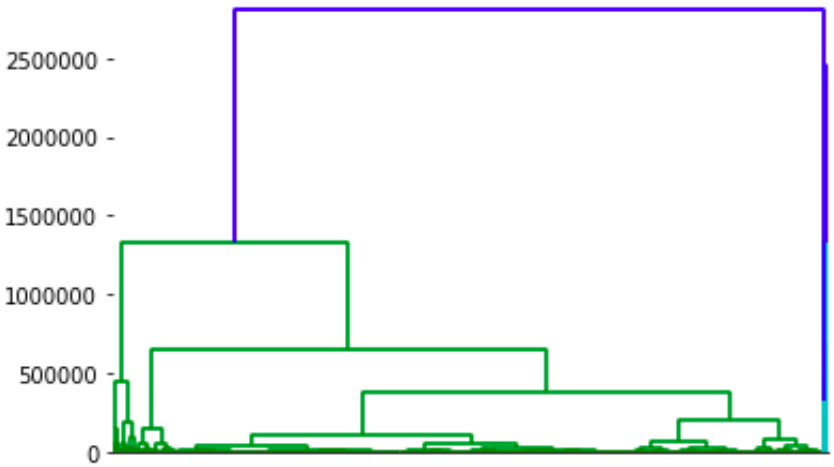
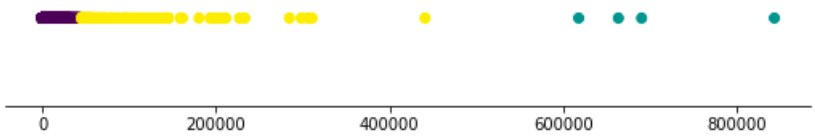
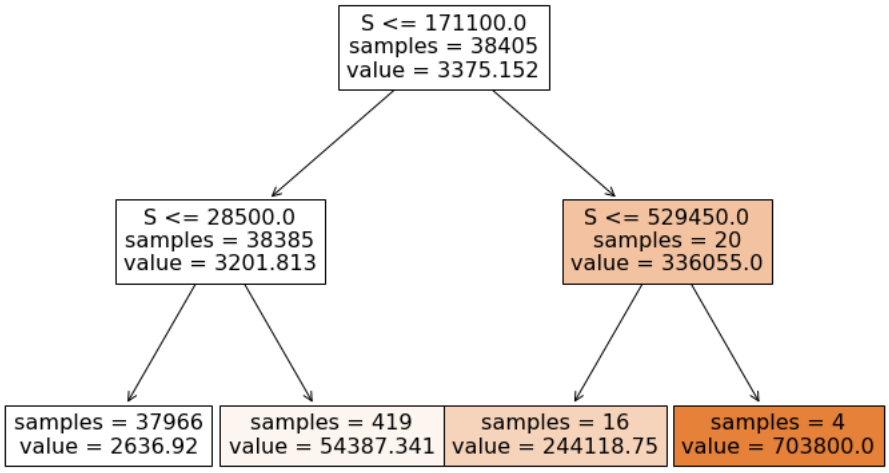


FIGURE 39 – Discrétisation par CAH



Discrétisation par les arbres de décision : La dernière méthode consiste à utiliser un arbre de régression. Sur la base de l'arbre de régression construit, nous pouvons opter soit pour une partition en deux classes, soit pour une partition en quatres classes (voir Figure 40). Dans le premier cas, la partition serait entre les article avec moins de 171 000 partages (soit 38 385 articles) et ceux ayant plus de 171 000 partages (soit 20 articles). Ainsi, nous constatons encore une fois que les classes générées seraient trop déséquilibrées et que la première classe ne répondrait pas à nos attentes.

FIGURE 40 – Discrétisation par les arbres de décision



7.1.3 Choix de discrétisation retenu

Au final, aucune des méthodes par algorithme utilisées ne fournit de résultats satisfaisant. En effet, soit la variable cible est discrétisée en un nombre trop important de classes, soit le seuil choisi est trop important de sorte que les classes créées sont déséquilibrées. Par conséquent, nous décidons d'opérer une discrétisation par quantile en utilisant la médiane. Ce choix pallie les défauts des autres méthodes. Notre nouvelle variable cible est donc la variable *popular* telle que : $popular = \begin{cases} 0 & \text{si } shares \leq 1400 \\ 1 & \text{sinon} \end{cases}$

7.2 Modélisation

Dans ce qui suit, nous désignons la variable cible par Y et les variables explicatives par X. Notons également que aurons besoin de métriques pour évaluer les performances de nos algorithmes. En

plus des métrique communes (telles que l'accuracy ou l'AUC), nous fournissons des métriques « plus fines » (sensitivité, spécificité, précision) permettant d'évaluer avec plus de précision si nos algorithmes prédisent mieux une des deux classes. On peut en effet penser que les rédacteurs d'articles sont plus intéressés par le fait de savoir si un article sera populaire plutôt que non-populaire.

7.2.1 Modélisation par régressions logistiques

Régression logistique usuelle : Nous commençons par construire un modèle de régression logistique usuelle. Le modèle de régression logistique est un des modèles paramétriques permettant de modéliser l'espérance conditionnelle à $X = x$ de Y : $\mathbb{E}(Y|X = x)$. Le modèle de régression logistique est linéaire et facilement interprétable ; il nous servira de *benchmark* pour la suite. On remarque (Tableau 5) que les performances de la régression logistique sont légèrement supérieures sur l'échantillon d'apprentissage, ce qui révèle la présence de sur-apprentissage. Au vu du nombre de variables (et donc de paramètres estimés) inclues dans le modèle, cela était attendu. Pour pallier cet écueil nous mettons en place une régression logistique pénalisée par la norme L_1.

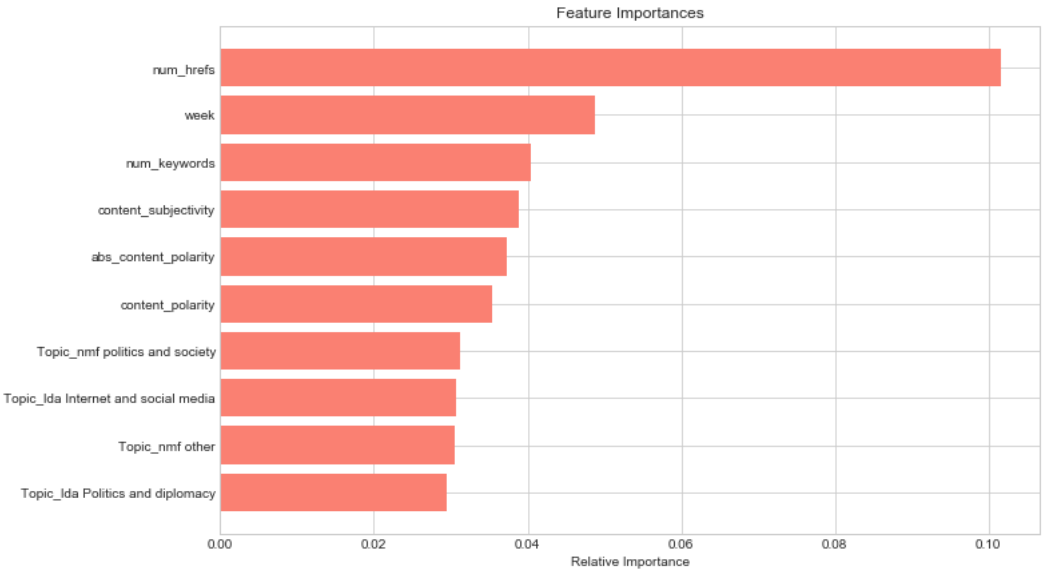
Régression logistique pénalisée : Les performances de la régression logistique pénalisée sont, dans l'ensemble, très légèrement supérieures à celles du modèle logistique classique. En revanche, si l'on s'intéresse plus précisément à la classe des articles populaires, nous remarquons que les métriques sont légèrement inférieures à celles du modèle logistique usuel (Tableau 5). Nous considérons toutefois cette différence comme non significative au vu de la faiblesse des écarts. Une des raisons pouvant expliquer l'absence d'amélioration tient au fait que la régression logistique, même pénalisée, ne capte que les relations linéaires présentent dans les données. Une solution consiste à explicitement modéliser des interactions entre variables. Se pose alors la question de savoir quelles interactions choisir et pourquoi. Une autre alternative consiste à utiliser des algorithmes capables de détecter et modéliser les relations non linéaires par eux-mêmes. Nous optons pour cette seconde option car elle présente le double avantage de ne pas augmenter le nombre de variables et de ne pas introduire de biais lié aux choix des interactions par le modélisateur.

7.2.2 Modélisation par Random Forest

L'algorithme Random Forest peut être décrit comme une combinaison d'arbre de décision de type CART, de bagging et de sélection aléatoire des X à chaque noeud des arbres composant la forêt. Ainsi, nous notons B le nombres d'arbres de décision composant la forêt et $mtry$ le nombre de variables candidates sélectionnées à chaque noeud. L'idée sous jacente à cet algorithme est de décorréler les B arbres indivuels de sorte à réduire la variance finale des prédictions : plus $mtry$ est faible, moins la variance est élevée ; plus $mtry$ est important, plus la variance est élevée. Enfin, cet algorithme présente également l'avantage de fournir un classement des variables selon leur importance relative pour la prédiction de la variable cible.

Pour construire notre modèle, nous mettons en place une procédure de validation croisée avec 5 folds. Différentes valeurs sont testées pour les hyperparamètres. Plus précisément, nous avons $B \in \{500, 1000\}$, $mtry \in \{10, 15, 20, 30, 50\}$ et $d \in \{1, 5, 50, 75, 100, 150\}$ avec d le nombre d'observations minimum requis pour qu'un noeud puisse être considéré comme un noeud terminal. Cela représente un total de $2 \times 5 \times 6 = 60$ modèles concurrents. Les hyperparamètres retenus sont $B = 500$, $mtry = 10$ et $d = 50$. Les résultats sont fournis dans le Tableau 5. Les performances de l'algorithme Random Forest sont légèrement supérieures à celles des modèles logistiques, notamment sur les articles considérés comme populaires. Toutefois, on note bien que cette amélioration est très légère et que nous somme dans un cas de sur-apprentissage au vu des différences de performances entre l'échantillon d'apprentissage et l'échantillon de test. Le classement des 10 premiers prédicteurs est fourni par la Figure 41. Nous observons que le classement contient des variables diverses : relatives au temps, au nombre de référence, au thème et à l'analyse de sentiment.

FIGURE 41 – Classement des 10 premières variables selon leur importance relative pour la prédiction (Random Forests)



7.2.3 Modélisation par Gradient Boosting

Comme les Random Forests, le boosting est une méthode ensembliste. Le boosting consiste à combiner plusieurs modèles dits « faibles » (*weak learners* en anglais) en un modèle dit « fort » (*strong learner*). Ci-dessous, lorsque nous utilisons le terme de boosting, nous faisons référence au Gradient Boosting et ses dérivés ; et nous considérons le cas où les *weak learners* sont des arbres de décision simples. Contrairement aux arbres d'un algorithme Random Forest qui sont indépendants les uns des autres, ceux utilisés dans le cadre d'un algorithme de boosting sont construits de manière séquentielle : chaque arbre dépend des résultats de l'arbre le précédent. Par conséquent, la mise en point d'un algorithme de boosting recquiert la fixation du nombre D d'arbres construits et de leur profondeur δ ; fixer une valeur trop importante pour un de ces

deux paramètres augmente le risque de sur-apprentissage. La solution à ce problème réside dans l'introduction d'un taux d'apprentissage $\lambda \in [0, 1]$ permettant de pénaliser l'ajout d'un arbre supplémentaire au modèle final.

Pour construire notre modèle, nous mettons en place une procédure de validation croisée avec 3 folds. Différentes valeurs sont testées pour les hyperparamètres. Plus précisément, nous avons $D \in \{400, 500, 800\}$, $\delta \in \{4, 7, 9\}$ et $\lambda \in \{0.1, 0.05, 0.01\}$. Cela représente un total de $3 \times 3 \times 3 = 27$ modèles concurrents. Les algorithmes de boosting recquiérant une puissance de calcul plus importante que les algorithmes de type bagging, nous allégons volontairement la procédure de validation. Les hyperparamètres retenus sont $D = 400$, $\delta = 4$ et $\lambda = 0.01$. Au niveau des performances (Tableau 5), on n'observe pas d'amélioration par rapport aux autres modèles sur l'échantillon de test ; le sur-apprentissage est plus léger que dans le cas de l'algorithme Random Forest. Le classement des 10 premiers prédicteurs est fourni par la Figure 42. Nous observons que les deux classements diffèrent de manière assez sensible. Alors que l'algorithme Random Forest accorde une grande importance à des variables apportant une information diverse, le Gradient boosting se focalise presque exclusivement sur les informations apportant une information temporelle.

FIGURE 42 – Classement des variables selon leur importance relative pour la prédiction (Gradient Boosting)

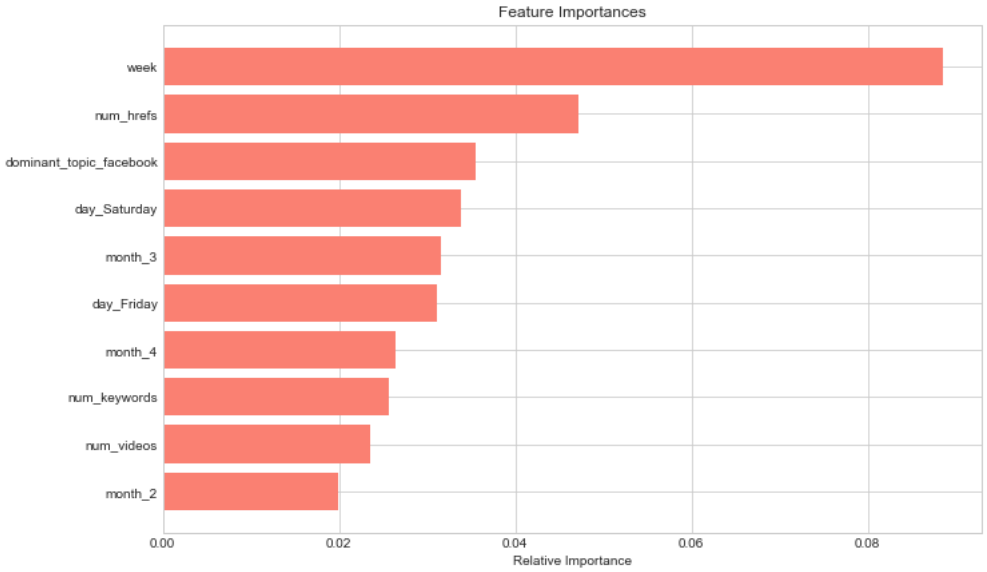


TABLE 5 – Performances des modèles

Modèle	Accuracy	Precision	Recall	Specificity	Sensitivity	Balanced-accuray	AUC
Échantillon d'apprentissage							
Logistique	0.58	0.58	0.51	0.64	0.51	0.58	0.57
Logistique-LASSO	0.58	0.59	0.50	0.65	0.50	0.58	0.58
Random Forest	0.68	0.68	0.61	0.73	0.61	0.68	0.67
Gradient Boosting	0.61	0.62	0.53	0.69	0.53	0.61	0.63
Échantillon de test							
Logistique	0.57	0.56	0.50	0.63	0.50	0.56	0.56
Logistique-LASSO	0.57	0.56	0.48	0.65	0.48	0.56	0.57
Random Forest	0.57	0.57	0.51	0.64	0.51	0.57	0.57
Gradient Boosting	0.58	0.57	0.50	0.65	0.50	0.57	0.57