Rapport Features Engineering

I) Analyse des jeux de données AGnews et Brown

Le corpus AGnews dispose de 4 classes : World, Sports, Business et Sci/Tech. j'ai donc créé une dataframe avec le fichier train.csv du de AGnews(Fig.1).

	classe	title	content
0	Business	Wall St. Bears Claw Back Into the Black (Reuters)	Reuters - Short-sellers, Wall Street's dwindli...
1	Business	Carlyle Looks Toward Commercial Aerospace (Reu...	Reuters - Private investment firm Carlyle Grou...
2	Business	Oil and Economy Cloud Stocks' Outlook (Reuters)	Reuters - Soaring crude prices plus worries\ab...
3	Business	Iraq Halts Oil Exports from Main Southern Pipe...	Reuters - Authorities have halted oil export\f...
4	Business	Oil prices soar to all-time record, posing new...	AFP - Tearaway world oil prices, toppling reco...

Fig.1

Cette dataframe dispose de 3 colonnes : classe, title ainsi que content.
Ensuite j'ai concaténé ces 3 colonnes sous forme de liste et stocké cela dans "classes_documents"(Fig.2).

```
["Business:Wall St. Bears Claw Back Into the Black (Reuters) Reuters - Short-sellers, Wall Street's dwindling band of ultra-cyn
ics, are seeing green again.",
 'Business:Carlyle Looks Toward Commercial Aerospace (Reuters) Reuters - Private investment firm Carlyle Group, which has a rep
utation for making well-timed and occasionally controversial plays in the defense industry, has quietly placed its bets on anot
her part of the market.',
 "Business:Oil and Economy Cloud Stocks' Outlook (Reuters) Reuters - Soaring crude prices plus worries about the economy and th
e outlook for earnings are expected to hang over the stock market next week during the depth of the summer doldrums.",
 'Business:Iraq Halts Oil Exports from Main Southern Pipeline (Reuters) Reuters - Authorities have halted oil export flows from
the main pipeline in southern Iraq after intelligence showed a rebel militia could strike infrastructure, an oil official said
on Saturday.']
```

Fig.2

De plus j'ai également créé une liste appelé "corpus"(Fig.3) de tuples (catégorie,texte) pour pouvoir analyser AGnews au niveau statistiques.

```
[('Business',
  "Wall St. Bears Claw Back Into the Black (Reuters) Reuters - Short-sellers, Wall Street's dwindling band of ultra-cynics, a
re seeing green again."),
 ('Business',
  'Carlyle Looks Toward Commercial Aerospace (Reuters) Reuters - Private investment firm Carlyle Group, which has a reputatio
n for making well-timed and occasionally controversial plays in the defense industry, has quietly placed its bets on another
part of the market.'),
 ('Business',
  "Oil and Economy Cloud Stocks' Outlook (Reuters) Reuters - Soaring crude prices plus worries about the economy and the outl
ook for earnings are expected to hang over the stock market next week during the depth of the summer doldrums."),
 ('Business',
  'Iraq Halts Oil Exports from Main Southern Pipeline (Reuters) Reuters - Authorities have halted oil export flows from the m
ain pipeline in southern Iraq after intelligence showed a rebel militia could strike infrastructure, an oil official said on
Saturday.'),
```

Fig.3

J’ai effectué une répartition des classes pour chaque catégorie en utilisant les listes compréhensions pour ainsi me permettre de savoir le nombre de mots moyens par classes dans chaque article (Fig.4).

```
Mean number of words : 38.699958333333335
Mean number of words in Business : 38.85856666666667
Mean number of words in World: 39.66466666666667
Mean number of words in Sports: 38.27146666666667
Mean number of words in Sci_Tech: 38.00513333333333
```

Fig.4

On observe que le nombre de mots est assez faible par article et que le nombre de mots est équitable pour chaque classe.
Agnews dispose de 30000 articles par catégories soit donc 120000 articles. Cela peut donc compenser le nombre assez faible présent dans chaque articles du dataset.

Pour le corpus Brown j’ai donc lu le fichier json fourni sous forme de dataframe (Fig.5).

	category	id	text
0	adventure	cn01	\n\n\tDan Morgan told himself he would forget ...
1	adventure	cn02	\n\n\tGavin paused wearily. ``You can't stay h...
2	adventure	cn03	\n\n\tThe sentry was not dead. He was, in fact...
3	adventure	cn04	\n\n\t``So it wasn't the earthquake that made ...
4	adventure	cn05	She was carrying a quirt, and she started to r...

Fig.5

La dataframe “corpus_brown” dispose de 3 colonnes category id ainsi que le texte. Pour les étapes de l’analyse de texte la colonne “id” ne va pas être utile.
Le corpus Brown dispose de plus de catégories que le corpus AGnews.
De plus on peut compter dans ce corpus 15 catégories réparties inéquitablement contrairement à AGnews (Fig.6).

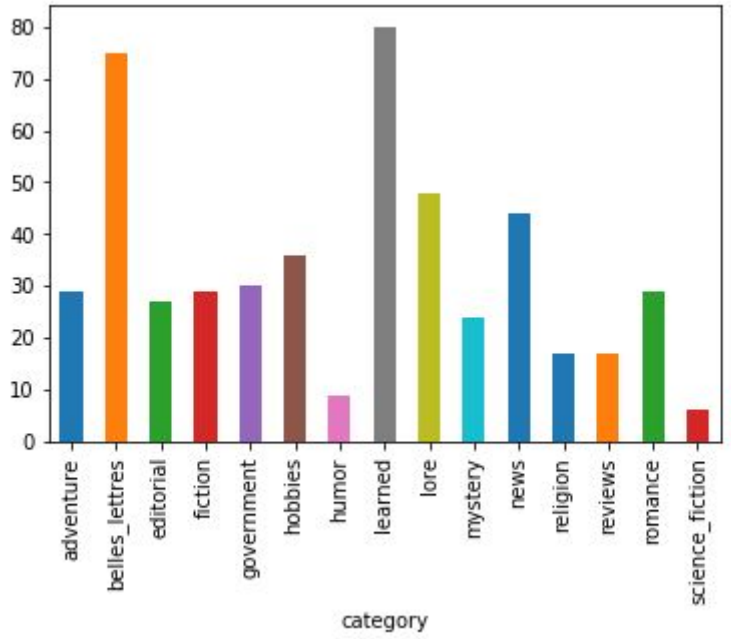


Fig.6

Comme pour AGnews j'ai effectué une analyse statistiques sur le nombre de mots moyens par article pour chaque catégorie (Fig.7).

```
mean word learned 12588.325
mean word belles_lettres 12130.8
mean word lore 12091.770833333334
mean word news 12278.727272727272
mean word hobbies 12161.75
mean word government 13170.166666666666
mean word fiction 11320.103448275862
mean word romance 11354.344827586207
mean word adventure 11348.448275862069
mean word adventure science_fiction 11832.0
mean word adventure editorial 12169.666666666666
mean word adventure mystery 11243.958333333334
mean word adventure religion 12096.823529411764
mean word adventure reviews 12714.117647058823
mean word adventure humor 11904.888888888889
```

Fig.7

On observe qu'il y a un nombre intéressant de mot par articles mais comme il n'y a pas beaucoup d'articles pour chaque catégorie peut-être que cela ne sera pas assez. De plus on voit que le nombre moyen de mots varie assez d'une catégorie à une autre. En effet par exemple pour la catégorie "government" on a en moyenne 13170 mots par article alors que pour la catégorie "mystery" on a en moyenne 11243 mots par article.

II) Les différentes étapes du Pipeline.

La première étape pour l'analyse de corpus a été le parsing. En effet il a fallu créer une fonction "parse" qui transforme le texte en minuscule, enlève la ponctuation dans le texte, effectué une étape de tokenization à l'aide des espaces, supprime les mots qui sont contenus dans la liste de stopwords et qui ne sont pas des chiffres. Elle permet également de réaliser la stemmatisation et créer des bigrams.
L'utilisation de la méthode download() ouvre une nouvelle fenêtre montrant le NLTK downloader qui m'a permis de télécharger la liste des stopwords de langue anglaise.
Pour rappel les "stopwords " sont des mots qui n'apportent pas de sens lors de l'analyse lexicale d'un texte, c'est pour cela qu'il faut les exclure.
Pour pouvoir garder la racine des mots il faut utiliser un stemmer, en effet la stemmatisation est une sorte de normalisation car de nombreuses variantes de mots ont le même sens. La raison pour laquelle il est important d'effectuer la stemmatisation est que cela permet de raccourcir la recherche et ainsi de normaliser les phrases. Pour cela il a fallu créer un stemmer pour la langue anglaise.
Ensuite la fonction "parse" va retourner une liste de dictionnaires avec pour clé le mot et pour valeur le nombre d'occurrences dans l'article tout cela grâce à Counter.

La deuxième étape a été la vectorisation. En effet pour cela il a fallu créer une fonction "vectorize" qui va transformer le dictionnaire de mots-occurrences en vecteurs et retourner la matrice transformé en vecteur ainsi que les noms des features.

La troisième étape a été la classification. En effet pour cela une fonction “classify_and_evaluate” grâce au classifieur multinomial Naive Bayes qui convient à la classification avec des caractéristiques discrètes comme dans notre cas le nombre de mots pour la classification de texte. De plus elle va permettre de générer des estimations croisées pour chaque point de données en entrée.

Voici les résultats pour le corpus AGnews (Fig.8).

Nombre d'erreurs : 13193

	precision	recall	f1-score	support
Business	0.85	0.85	0.85	30000
Sci/Tech	0.85	0.86	0.86	30000
Sports	0.94	0.97	0.96	30000
World	0.91	0.88	0.89	30000
avg / total	0.89	0.89	0.89	120000

Fig.8

Voici les résultats pour le corpus Brown (Fig.9).

Nombre d'erreurs : 279

	precision	recall	f1-score	support
adventure	0.50	0.59	0.54	29
belles_lettres	0.30	0.76	0.43	75
editorial	0.67	0.07	0.13	27
fiction	0.44	0.28	0.34	29
government	0.65	0.50	0.57	30
hobbies	0.71	0.33	0.45	36
humor	0.00	0.00	0.00	9
learned	0.55	0.57	0.56	80
lore	0.33	0.17	0.22	48
mystery	0.60	0.25	0.35	24
news	0.64	0.80	0.71	44
religion	1.00	0.06	0.11	17
reviews	0.00	0.00	0.00	17
romance	0.33	0.48	0.39	29
science_fiction	0.00	0.00	0.00	6
avg / total	0.48	0.44	0.41	500

Fig.9

On peut expliquer les différences par le fait que dans le corpus AGnews la base d’apprentissage est beaucoup plus grande et il y a seulement 4 classes du coup le modèle apprend mieux plus facilement. Au contraire dans le corpus Brown le dataset est petit seulement 500 articles et il y a plus de catégories donc pour de meilleurs résultats il faudrait augmenter considérablement augmenter le dataset.

Une autre étape a été ensuite de garder les mots avec le plus d'occurrences dans les articles pour pouvoir avoir des variables pertinentes et effectuer une meilleure classification. On a utilisé le vecteur de booléens “keeping” pour construire la liste des mots gardés.

C'est très utile puisque “keeping” et “feature_names” sont alignés, c'est-à-dire que ce sont des listes de même taille, stockant des informations sur les mêmes choses dans le même ordre, par construction.

Ensuite on utilise une transformation TF-IDF car cela est plus intéressant car cette transformation renvoie un score au lieu de renvoyer le nombre d'occurrences comme countvectorizer. C’est mieux d’avoir des valeurs comprises entre 0 et 1 au lieu de valeurs qui peuvent être très différentes.

III) Temps de calcul AGnews

Pour tester le temps de calcul des différentes étapes j’ai donc créé une deuxième fonction “parse_test” sans stemmatisation. Tableau représentant les différentes étapes de calcul avec et sans stemmatisation:

temps	sans stemmatisation	avec stemmatisation
parse	11.838203900664427	61.02196373758937
vectorize	3.4904322878508367	2.921866710456527
classification	1.456380816891084	1.2669371163125334
tfidf	0.145136090619701	0.16597442399512374

On remarque grâce à ce tableau que l’ajout d’une étape de stemmatisation dans la fonction parse augmente considérablement le temps de calcul. Par contre grâce à la suite des étapes c’est à dire vectorisation et classification sera moins coûteux en temps de calcul.

IV) Topic modelling: LSA et LDA

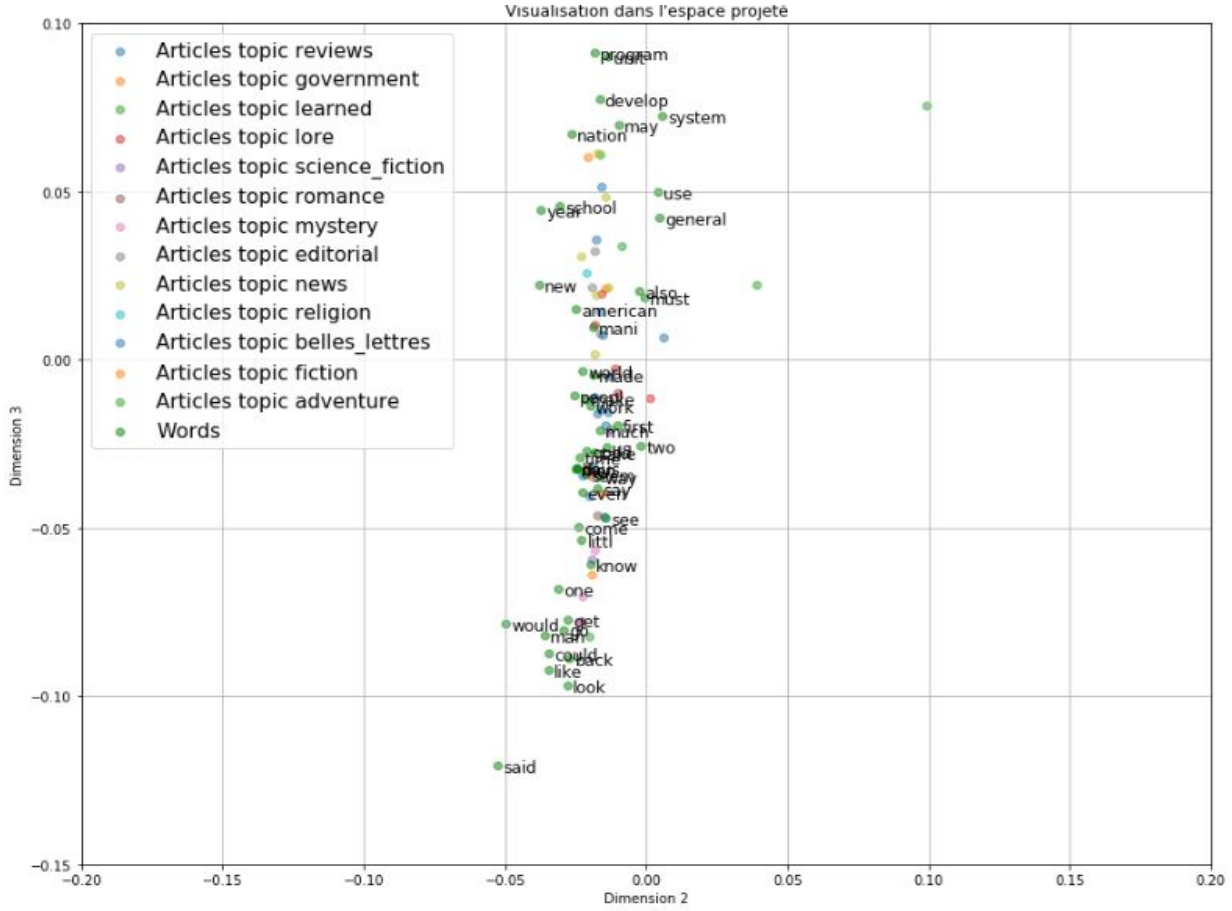
LSA

Nous allons chercher à résumer les documents texte en utilisant des phrases de document, les termes de chaque phrase du document, et leur appliquer SVD à l'aide d'une sorte des poids caractéristiques tels que les poids TF-IDF. Le principe de base de LSA est que dans tout document, il existe une structure latente entre termes qui sont liés dans le contexte et doivent donc également être corrélés dans le même espace singulier.

Pour AGnews j’ai donc sélectionné les classes Sports et World, puis sélectionné les 50 mots les plus représentés c’est à dire avec le plus d’occurrences dans ses 2 classes. Voici donc la représentation LSA:

On remarque que la classe World parle de différentes choses c'est pour cela qu'il y a des clusters différents en haut, à droite et au centre. Au contraire pour la catégorie sport on remarque que le cluster de mots se dirige surtout vers le bas à gauche.

Pour Brown j'ai sélectionné les 50 mots les plus représentés c'est à dire avec le plus d'occurrences dans ses 2 classes. Voici donc la représentation LSA:



On observe bien que les mots sont classé en fonction de leur similarité.

LDA

LDA est un modèle génératif probabiliste permettant d'expliquer des ensembles d'observations, par le moyen de groupes non observés, eux-mêmes définis par des similarités de données.
Il faut prendre l'argmax de chaque entrée de la matrice de topic pour obtenir la catégorie de topic prédite pour chaque mot. Ces catégories de sujets sont ensuite caractérisées par leurs mots les plus fréquents.
le modèle LDA est appliqué à la fonction de terme de document TF-IDF.
matrice, qui est décomposée en deux matrices, à savoir une matrice de topic-document et une matrice topic-terme. On utilise la matrice de termes de sujets stockée dans lda.components_ pour récupérer les poids pour chaque terme par sujet.

Résultats pour AGnews:

```
Topic: 0
  coach, footbal, england, japan, madrid, arsenal, manag, club
Topic: 1
  game, red, peac, sox, season, ap, boston, darfur
Topic: 2
  kill, palestinian, iraqi, iraq, bomb, attack, polic, baghdad
Topic: 3
  nuclear, china, north, talk, un, south, korea, trade
Topic: 4
  presid, elect, bush, iran, leader, minist, iraq, vote
Topic: 5
  win, victori, lead, leagu, first, score, cup, point
```

On remarque qu'avec LDA les mots sont relativement bien classé. En effet chaque mot est associé à un poids et les mots ayant des poids assez proche sont mis dans un même topic. Ici j'ai choisi de prendre les 8 mots avec le plus d'occurrences pour chaque topic.

Résultats pour Brown:

```
Topic: 0
  food, cattl, feed, seed, meat, rector, hudson, use
Topic: 1
  af, cost, oper, state, rate, tax, electron, industri
Topic: 2
  one, southern, snake, south, negro, cool, river, feet
Topic: 3
  social, may, one, must, system, nation, mean, number
Topic: 4
  church, god, music, christian, jazz, christ, cathol, song
Topic: 5
  student, colleg, cell, foam, af, dictionari, faculti, chlorin
Topic: 6
  said, look, would, like, could, back, go, get
Topic: 7
  school, junior, citi, said, board, would, shelter, educ
Topic: 8
  state, govern, presid, unit, feder, nation, democrat, new
Topic: 9
  wine, would, matsuo, one, marin, fogg, coolidg, alex
Topic: 10
  stain, wright, deegan, miriam, andi, conjug, mike, nonspecif
Topic: 11
  one, poem, art, year, book, new, say, artist
Topic: 12
  morgan, hearst, theresa, clay, cousin, shu, chandler, command
Topic: 13
  mantl, palmer, linda, player, henrietta, john, bobbi, leagu
Topic: 14
  arteri, mercer, stress, helva, platform, emot, bronchial, gyro
```