DS P6 – Catégorisez automatiquement des questions

Sommaire

Contexte - Objectif

Les données

Dataset: stackoverflow-qa.csv (50000 questions exportées du site stackoverflow, 23 variables, 50000 lignes)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 23 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Id                     50000 non-null  int64
 1   PostTypeId             50000 non-null  int64
 2   AcceptedAnswerId       29765 non-null  float64
 3   ParentId               0 non-null      float64
 4   CreationDate           50000 non-null  object
 5   DeletionDate           0 non-null      float64
 6   Score                  50000 non-null  int64
 7   ViewCount              50000 non-null  int64
 8   Body                   50000 non-null  object
 9   OwnerUserId            49047 non-null  float64
 10  OwnerDisplayName       2786 non-null   object
 11  LastEditorUserId       27684 non-null  float64
 12  LastEditorDisplayName  1531 non-null   object
 13  LastEditDate           28133 non-null  object
 14  LastActivityDate       50000 non-null  object
 15  Title                  50000 non-null  object
 16  Tags                   50000 non-null  object
 17  AnswerCount            50000 non-null  int64
 18  CommentCount           50000 non-null  int64
 19  FavoriteCount          15172 non-null  float64
 20  ClosedDate             2267 non-null   object
 21  CommunityOwnedDate     177 non-null    object
 22  ContentLicense         50000 non-null  object
dtypes: float64(6), int64(6), object(11)
memory usage: 8.8+ MB
```

Système de suggestion de tags

Réaliser un modèle d'assignation de tags pertinents pour une question donnée.

Contexte - Problématique

- Quelles variables utiliser pour déterminer les tags pertinents?
- Comment interpréter les questions et trouver leurs thèmes?
- Quel modèle choisir pour catégoriser les questions?
- Comment améliorer les performances du modèle?

Exploration - Préparation

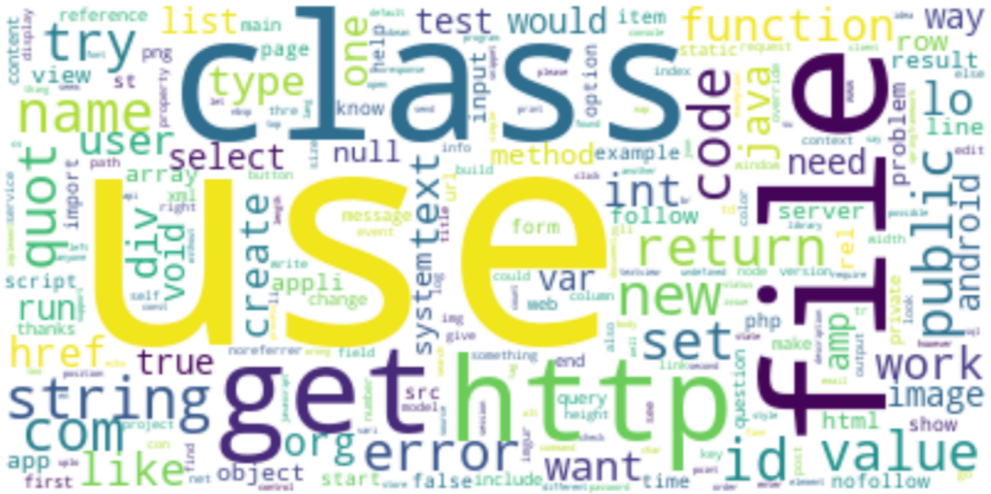
- Suppression des caractères spéciaux (tag html, ponctuations, ...)
- Suppression des stopwords
- Tokenisation
- Lemmatisation

Title	Body	Tags	FilteredBody	FilteredTitle	BodyTokens	TitleTokens	BodyLemmas	TitleLemmas
wordpress > create category as child of "uncategorized" (catid=1)	i've got a script in my functions.php file that checks for the existence of several categories that are required by my theme, and if they are not present, it creates them... `if(!get_cat_id('my-category')){wp_cr eate_category('my-category');}` i need to modify this script in two ways... first, i need to create the category so that its parent category is the "uncategorized" category (or id 1). second, if the category already exists, but its parent is not id 1, i need to make it so.	wordpress categories	got script functions php file checks existence several required theme present creates need modify script two ways first need create parent un id second alre exists parent id need make	wordpress create category child uncategorized catid 1	['got', 'script', 'functions', 'php', 'file', 'checks', 'existence', 'several', 'required', 'theme', 'present', 'creates', 'need', 'modify', 'script', 'two', 'ways', 'first', 'need', 'create', 'parent', 'un', 'id', 'second', 'alre', 'exists', 'parent', 'id', 'need', 'make']	['wordpress', 'create', 'category', 'child', 'uncategorized', 'catid']	['get', 'script', 'function', 'php', 'file', 'check', 'existence', 'several', 'require', 'theme', 'present', 'creates', 'need', 'modify', 'script', 'two', 'way', 'first', 'need', 'create', 'parent', 'un', 'id', 'second', 'alre', 'exists', 'parent', 'id', 'need', 'make']	['wordpress', 'create', 'category', 'child', 'uncategorized', 'catid']
how to reshape matlab matrices for this example?	i have a 40x16 matrix or 8 5x16 one below the other i.e. aligned vertically. i want to get a 5x128 matrix from that such that i align the 8 5x16 matrices horizontally. is there an efficient/quicker (rather than the hardcoded for loops) way to do this? i want the individual 5x16 matrices intact.	matlab matrix reshape	matrix x one e aligned verti want get matrix align x matrices horizontally efficient quicker rather har loops way want individual matrices int	reshape matlab matrices example	['matrix', 'one', 'aligned', 'verti', 'want', 'get', 'matrix', 'align', 'matrices', 'horizontally', 'efficient', 'quicker', 'rather', 'har', 'loops', 'way', 'want', 'individual', 'matrices', 'int']	['reshape', 'matlab', 'matrices', 'example']	['matrix', 'one', 'align', 'verti', 'want', 'get', 'matrix', 'align', 'matrix', 'horizontally', 'efficient', 'quicker', 'rather', 'har', 'loop', 'way', 'want', 'individual', 'matrix', 'int']	['reshape', 'matlab', 'matrix', 'example']
regex syntax validator	does anybody know any tool for validating the syntax of an regular expression? i dont want to validate if it matches or not with some text but i want to see if there are syntax errors in the regex (missing parenthisis etc). also, what about syntax highlighting? it would be a great help when writing complex regex.	regex syntax highlighting validation	anybody know tool vali syntax regular expression dont want vali matches text want see syntax errors regex missing parenthisis etc also syntax highlighting would great help writing complex regex	regex syntax validator	['anybody', 'know', 'tool', 'vali', 'syntax', 'regular', 'expression', 'dont', 'want', 'vali', 'matches', 'text', 'want', 'see', 'syntax', 'errors', 'regex', 'missing', 'parenthisis', 'etc', 'also', 'syntax', 'highlighting', 'would', 'great', 'help', 'writing', 'complex', 'regex']	['regex', 'syntax', 'validator']	['anybody', 'know', 'tool', 'vali', 'syntax', 'regular', 'expression', 'dont', 'want', 'vali', 'match', 'text', 'want', 'see', 'syntax', 'error', 'regex', 'miss', 'parenthisis', 'etc', 'also', 'syntax', 'highlight', 'would', 'great', 'help', 'write', 'complex', 'regex']	['regex', 'syntax', 'validator']

Exploration – Sac de mots (Bag of words)

dfSumBowTitlesHead

	count
use	202
file	122
get	99
error	85
value	73
work	68
array	65
string	64
android	61
data	59
code	58
php	58
java	54
server	54
add	54

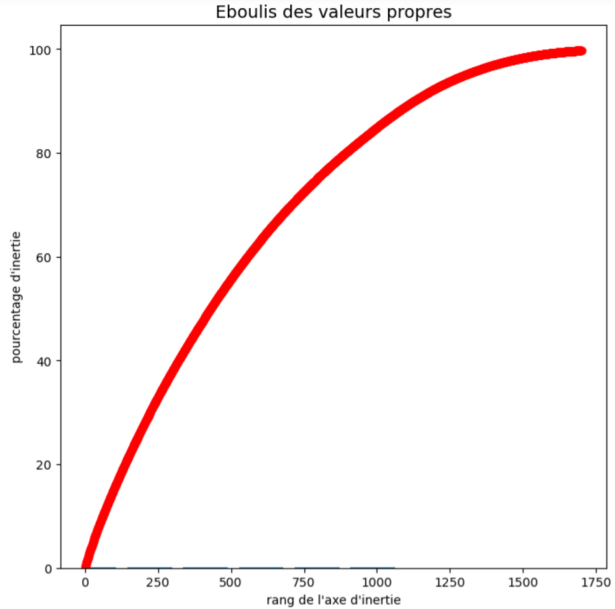


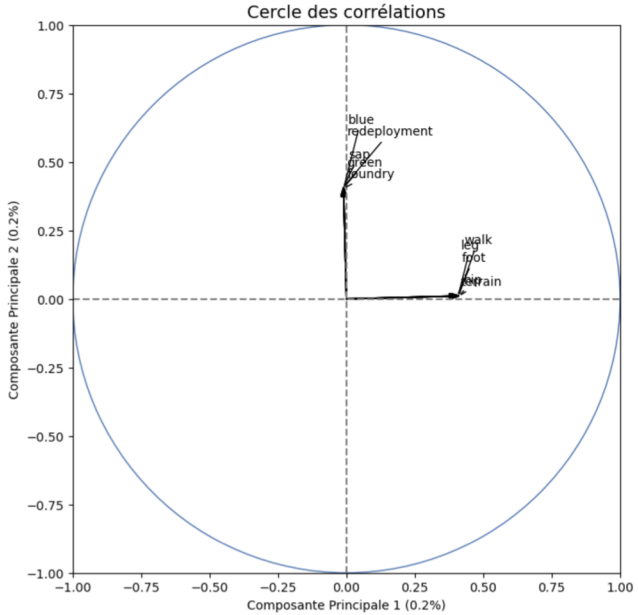
- Comptage individuel de chaque mot
- Ignore la sémantique des mots

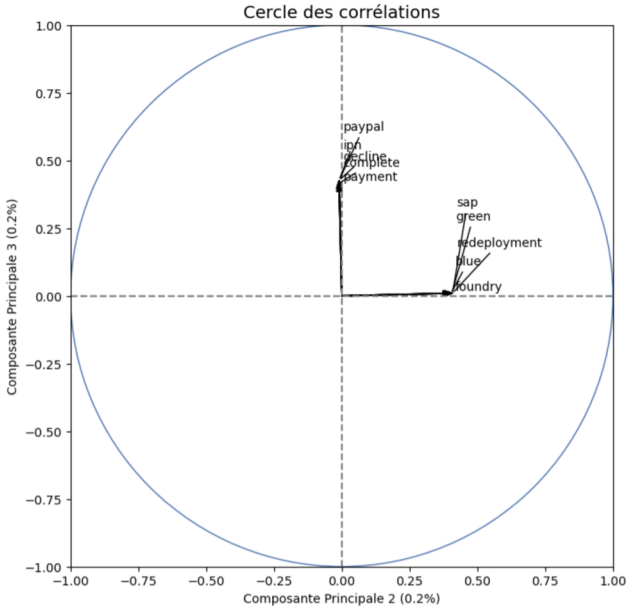
dfSumBowBodiesHead

	count
use	1940
class	1472
file	1391
get	1385
http	1270
string	1229
new	1197
code	1145
id	1127
name	1120
com	1098
value	1026
public	1002
error	998

Exploration – Analyse en Composantes Principales







- Chaque Composante principale n'explique que 0,2% de la variance
- Il faut au moins 1750 composantes pour expliquer 100% de la variance

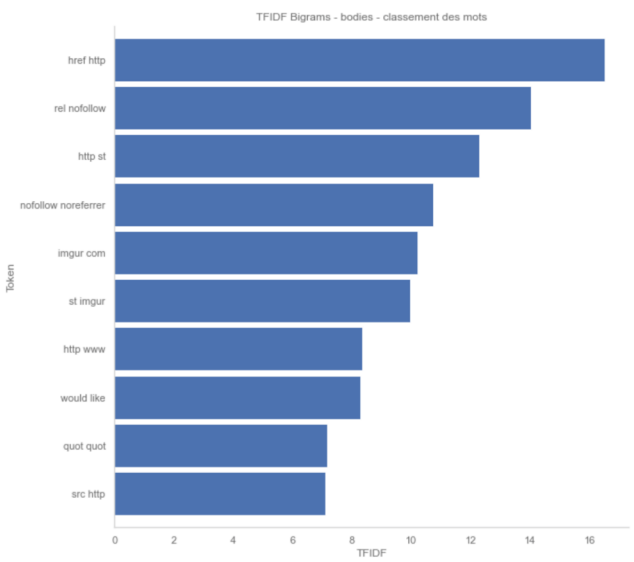
Exploration – vectorisation TF-IDF (principe)

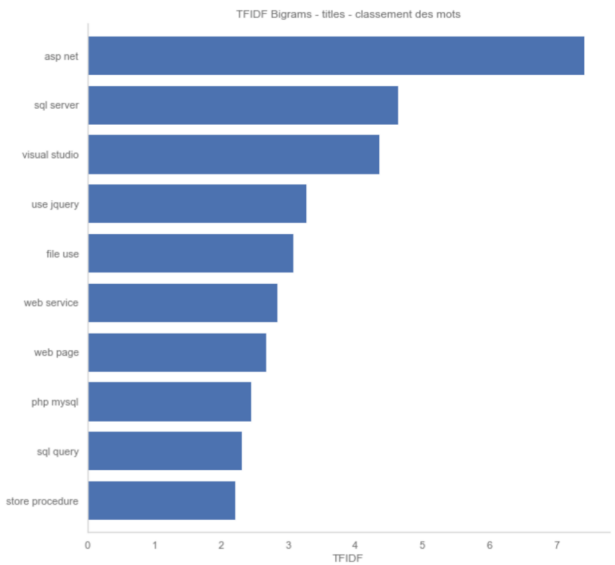
	text	tf	idf
0	Eddard Stark is a king in the north.	1	3
1	A king but one king : kings are everywhere.	2	3
2	Hodor was different : he was not a king .	1	3
3	But the North could not change without him.	0	3

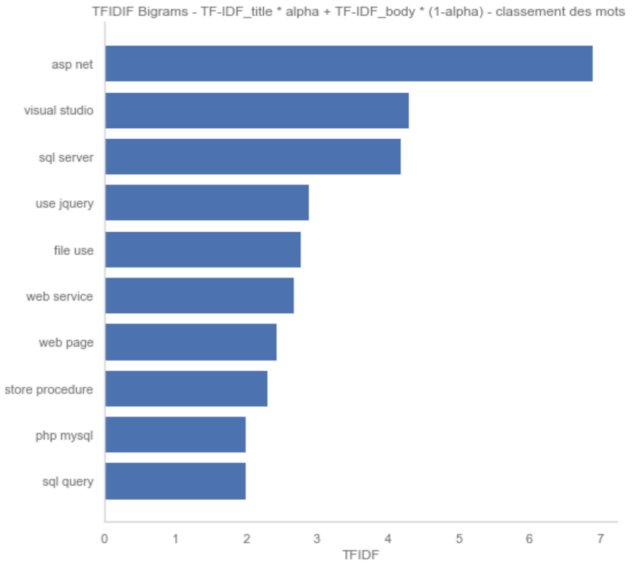
	king	was	the	not	a	he	one	north	kings	is	in	him	everywhere	A	different	could	change	but	are	Stark	North	Hodor	Eddard
0	0.333333	0.0	0.5	0.0	0.5	0.0	0.0	0.0	1.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0
1	0.666667	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0
2	0.333333	2.0	0.0	0.5	0.5	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
3	0.000000	0.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0

- Term Frequency: nombre d'occurrence du mot par document
- Inverse Document Frequency: nombre de document contenant le mot

Exploration – vectorisation TF-IDF

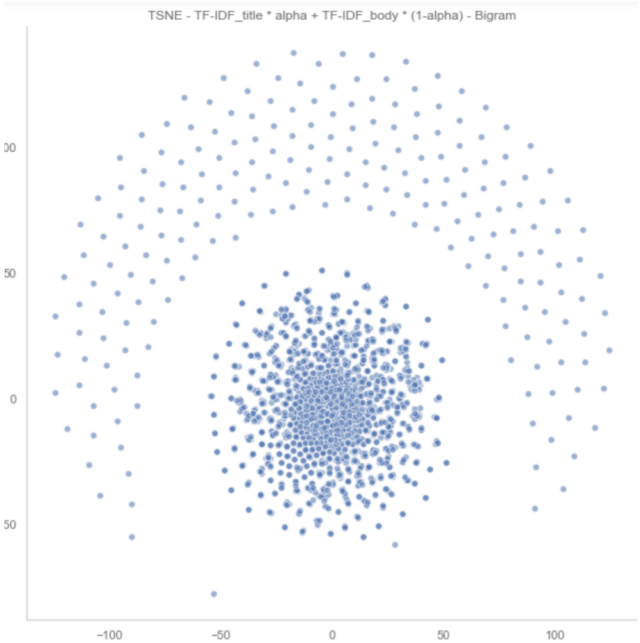


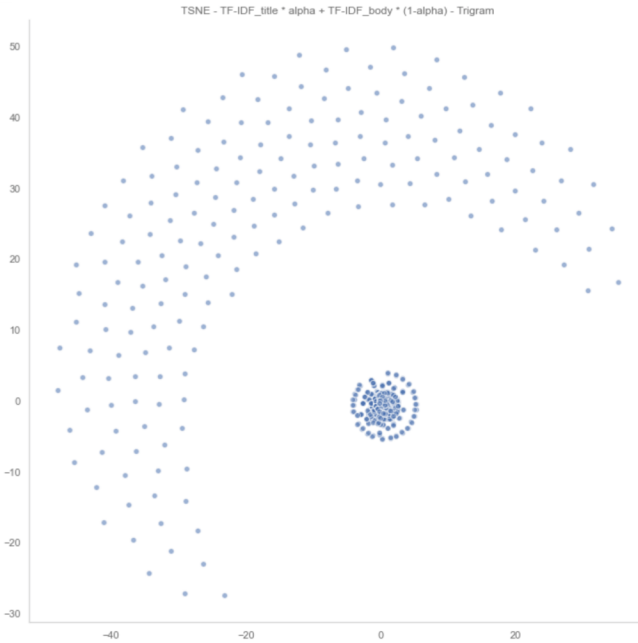




- Choix du bigramme car plus intelligible
- Pour tenir compte du titre, nous pondérons titre et corps

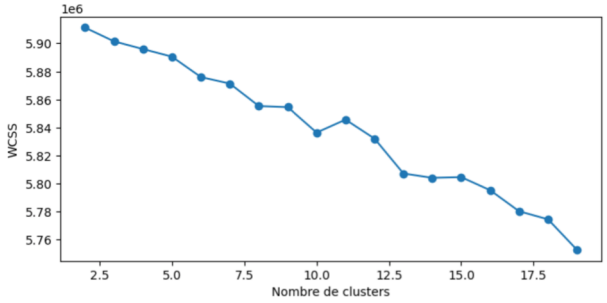
Exploration – Visualisation TSNE

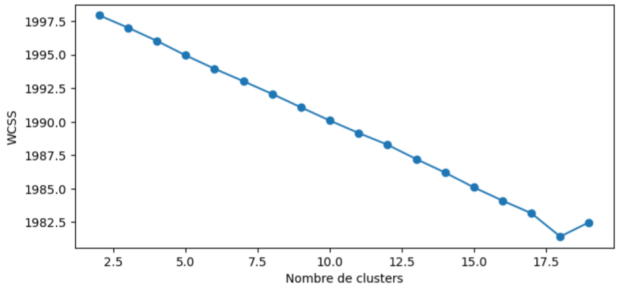


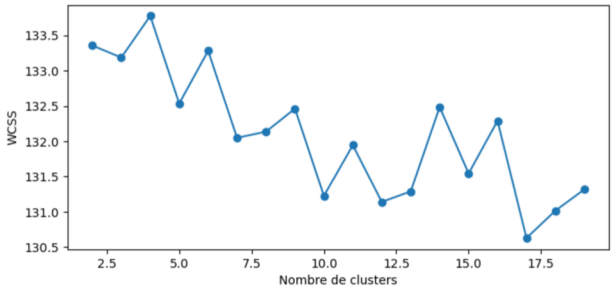


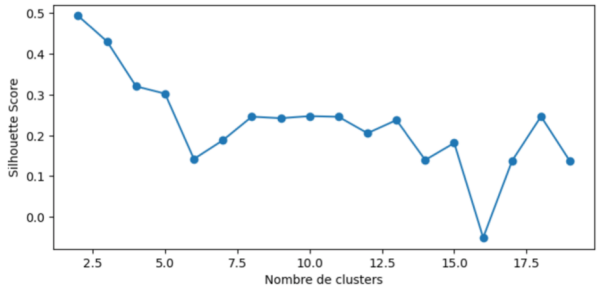
- Ensemble plus nets avec les bigrammes

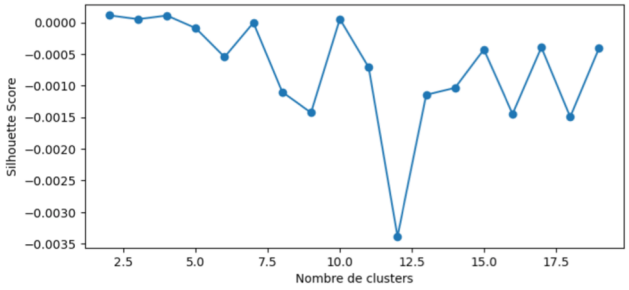
Modélisation – K-Means

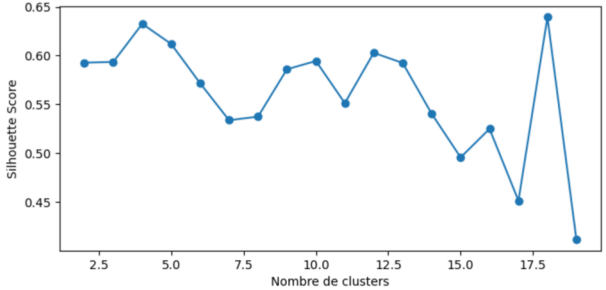








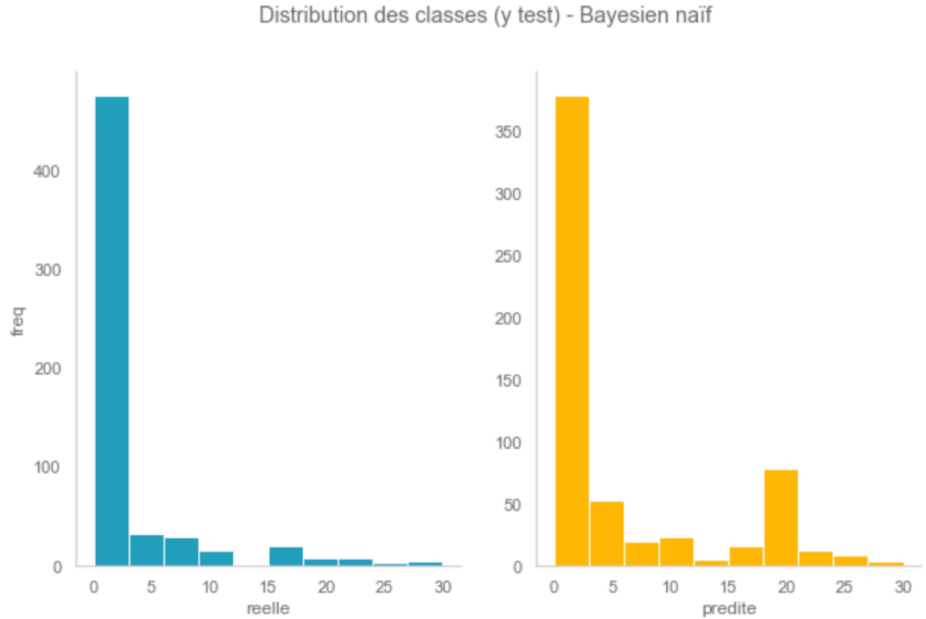




- Monogramme, bigramme: score de silhouette << 0.5
- Trigramme: score de silhouette / nombre de clusters proche de 0.5 et instable

Modélisation – Bayesien naif

Metrique	Accuracy	ARI
Score	0.546667	0.083325

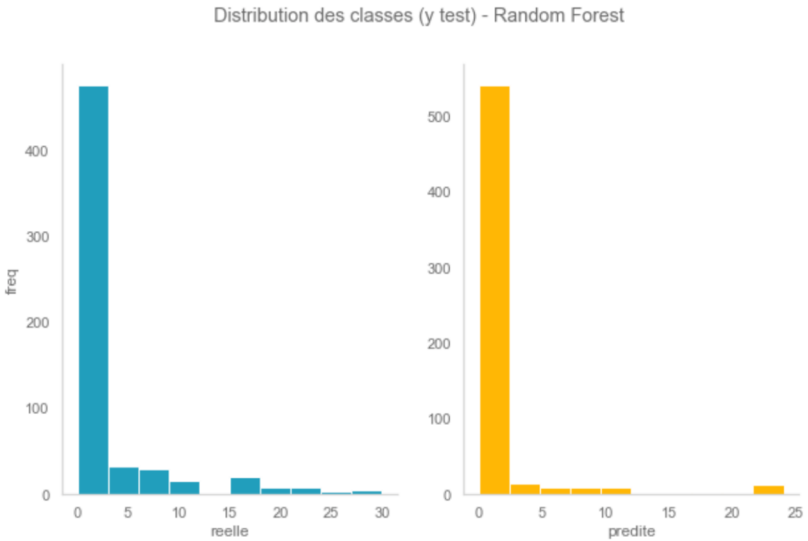


- Exactitude faible et stabilité quasi nulle

Class	Tag	TP	FP	TN	TPR	FPR	Accuracy
28	mvc	1	2	322	0.500000	0.006173	0.990798
12	iphone	0	2	323	0.000000	0.006154	0.990798
30	wordpress	0	1	323	0.000000	0.003086	0.987768
14	wpf	0	3	323	0.000000	0.009202	0.987768
10	mysql	0	3	323	0.000000	0.009202	0.981763
6	jquery	1	2	322	0.200000	0.006173	0.981763
23	spring	0	6	323	0.000000	0.018237	0.978788
24	xml	2	7	321	0.500000	0.021341	0.972892
22	visual	1	4	322	0.142857	0.012270	0.969970
1	javascript	0	4	323	0.000000	0.012232	0.967066
11	asp	1	6	322	0.166667	0.018293	0.967066
19	json	1	9	322	0.142857	0.027190	0.955621
5	net	5	10	318	0.500000	0.030488	0.955621
4	java	9	9	314	0.600000	0.027864	0.955621
9	html	2	12	321	0.333333	0.036036	0.952802
7	python	0	6	323	0.000000	0.018237	0.952802
2	android	4	11	319	0.444444	0.033333	0.952802
3	php	6	14	317	0.666667	0.042296	0.950000
8	sql	2	8	321	0.133333	0.024316	0.938953
17	server	3	13	320	0.142857	0.039039	0.912429
18	ruby	1	67	322	1.000000	0.172237	0.828205
0	NaN	286	75	37	0.625821	0.669643	0.567663
21	studio	0	0	0	0.000000	0.000000	0.000000

Modélisation – Random forest

Metrique	Accuracy	ARI
Score	0.818333	0.388118

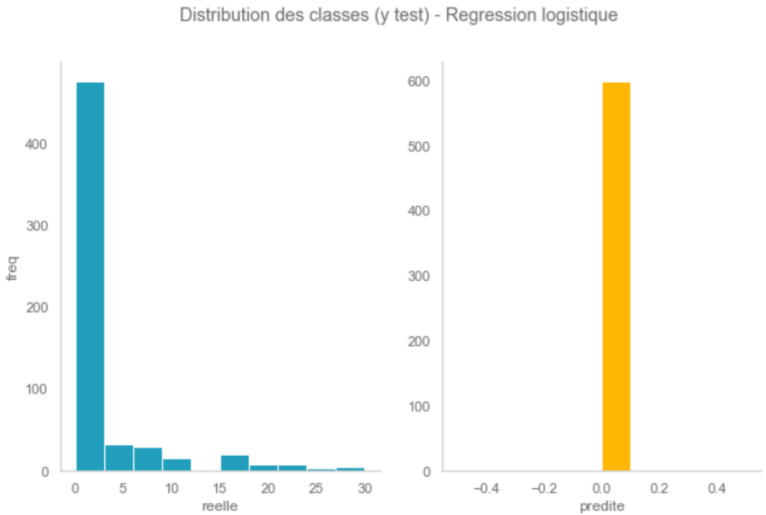


Class	Tag	TP	FP	TN	TPR	FPR	Accuracy
12	iphone	1	0	442	1.000000	0.000000	1.000000
18	ruby	1	0	442	1.000000	0.000000	1.000000
24	xml	3	1	440	0.750000	0.002268	0.995506
22	visual	6	2	437	0.857143	0.004556	0.993274
6	jquery	2	1	441	0.400000	0.002262	0.991051
11	asp	5	4	438	0.833333	0.009050	0.988839
2	android	6	3	437	0.666667	0.006818	0.986637
7	python	5	1	438	0.500000	0.002278	0.986637
8	sql	9	1	434	0.600000	0.002299	0.984444
3	php	4	4	439	0.444444	0.009029	0.980088
5	net	1	1	442	0.100000	0.002257	0.977925
4	java	3	4	440	0.200000	0.009009	0.965142
0	NaN	443	89	0	0.969365	1.000000	0.811355

- Exactitude élevée mais stabilité faible

Modélisation – Régression logistique

Metrique	Accuracy	ARI
Score	0.765	0.0



Class	Tag	TP	FP	TN	TPR	FPR	Accuracy
0	NaN	457	143	0	1.0	1.0	0.761667

- Exactitude élevée pour la classe ‘autres’

Modélisation –XGBoost

Metrique	Accuracy	ARI
Score	0.801667	0.313737

Class	Tag	TP	FP	TN	TPR	FPR	Accuracy
11	asp	4	1	442	0.666667	0.002257	0.993318
22	visual	6	2	440	0.857143	0.004525	0.993318
24	xml	3	2	443	0.750000	0.004494	0.993318
6	jquery	1	1	445	0.200000	0.002242	0.988914
2	android	5	3	441	0.555556	0.006757	0.984547
3	php	3	3	443	0.333333	0.006726	0.980220
5	net	2	2	444	0.200000	0.004484	0.978070
8	sql	7	2	439	0.466667	0.004535	0.978070
4	java	2	4	444	0.133333	0.008929	0.963283
17	server	1	2	445	0.047619	0.004474	0.952991
0	NaN	446	98	0	0.975930	1.000000	0.803604

- Exactitude élevée, classe ‘autres’ sur-représentée

Modélisation – Réseau neuronal

- Entrées: 2964 (bi grammes)
- Sortie: 30 classes
- 70 itérations
- 2 couches + drop out
- Écart faible entre entrainement et validation => surapprentissage faible

Metrique	Accuracy	ARI
Score	0.948333	0.87553

	Class	Tag	TP	FP	TN	TPR	FPR	Accuracy
18	30	wordpress	3	0	531	1.000000	0.000000	1.000000
10	10	mysql	3	0	531	1.000000	0.000000	1.000000
4	4	java	15	1	519	1.000000	0.001923	0.998131
5	5	net	8	0	526	0.888889	0.000000	0.998131
6	6	jquery	4	0	530	0.800000	0.000000	0.998131
8	8	sql	13	0	521	0.928571	0.000000	0.998131
17	24	xml	4	1	530	1.000000	0.001883	0.998131
15	22	visual	6	1	528	0.857143	0.001890	0.996269
2	2	android	8	0	526	0.800000	0.000000	0.996269
3	3	php	9	2	525	1.000000	0.003795	0.996269
7	7	python	8	0	526	0.800000	0.000000	0.996269
11	11	asp	5	1	529	0.833333	0.001887	0.996269
12	12	iphone	1	2	533	1.000000	0.003738	0.996269
16	23	spring	1	2	533	1.000000	0.003738	0.996269
14	19	json	4	0	530	0.571429	0.000000	0.994413
9	9	html	5	3	529	0.833333	0.005639	0.992565
1	1	javascript	4	1	530	0.571429	0.001883	0.992565
13	17	server	17	2	517	0.850000	0.003854	0.990724
0	0	NaN	451	15	83	0.982571	0.153061	0.958707

API – Déploiement

- Isolation de version d'os et de logiciels par package
- Facilement portable
- Plus léger, plus rapide à créer/déployer que les machines virtuelles

FastAPI

- Taches Asynchrones
- Documentation automatique d'api

HEROKU

https://questiontag.herokuapp.com

- Gratuit
- 1 app web + 1 app backend
- 512 Mb Ram

https://github.com/8huit/OC_DS_P6

Conclusion

- Les +
  - Première analyse de texte
  - Efficacité du modèle de réseau neuronal

- Les –
  - Pas de validation croisées
  - Pas de prise en compte de la relation entre les mots

- Pour la suite
  - Approfondir l'analyse contextuelle
  - Tester le modèle sur + de volumétrie
  - Modèle LSA pour résumer
  - Exploration sémantique avec le word embedding
  - Graphe de connaissance entité-relation