
OPENCLASSROOMS & CentraleSupélec

# Rapport de projet N 5

# Catégorisez automatiquement des questions

Irina Maslowski

19 janvier 2022

## Table des matières

# 1 Introduction

Stack Overflow (SO) est un site web d'échanges de questions et réponses ayant pour sujet l'informatique. L'objectif du projet N°5 est d'aider les utilisateurs novices de SO à trouver des mots clés correspondant à la question qu'ils aimeraient poser. Le cahier des charges du projet demande de développer une application web basique qui fait un appel à un modèle d'apprentissage automatique [1] de prédiction de tags.

Ce type de tâche est équivalent à la détection du sujet du texte. En traitement automatique des langues (TAL) [2], cette tâche peut être formulée et abordée différemment en fonction du sous-domaine du TAL. Nous pouvons distinguer cinq principales approches :

— purement statistique en analyse de discours [2] ;
— Machine Learning (ML) supervisé pour l'extraction de mots clés en extraction d'information [1] ;
— ML non supervisé pour la modélisation de thème de document [6] ;
— à base de graphes pour l'extraction des relations sémantiques [6].
— réseaux de neurones pour l'extraction de mots ou phrases clés [3]

Ces approches peuvent également être combinées pour former des solutions alternatives ([5]).

Selon [3], l'utilisation de réseaux de neurones pour l'extraction de mots clés est encore rare, car demandant beaucoup de données pour l'entrainement et son efficacité par rapport aux autres approches n'est pas démontrée. De plus, les réseaux de neurones sont plus complexes à mettre en place que les approches classiques de ML. Dans le travail présent, nous nous concentrons donc sur des approches de Machine Learning, car le sujet du projet est utilitaire et ne demande pas d'analyse linguistique profonde. Dans les sections suivantes de ce rapport, nous décrirons d'abord le corpus, i.e. la manière dont il a été extrait, ses particularités et les traitements appliqués pour le rendre davantage exploitable. Ensuite nous parlerons des expérimentations que nous avons effectuées pour prédire des mots clés. Nous présenterons l'application web développée en tant qu'interface du modèle de Machine Learning choisi. Enfin, nous conclurons sur les résultats du projet.

1. Pour désigner ce dernier, dorénavant, nous utiliserons le terme anglophone Machine Learning (ML).
2. Le terme populaire anglais est NLP (Natural Language Processing).

# 2 Corpus "Questions StackOverflow 2020 - 2021"

La construction du corpus "Questions StackOverflow 2020 - 2021" comprend trois étapes :

1. extraction de données ;
2. exploration du corpus ;
3. nettoyage du corpus.

Ci-dessous nous décrivons chaque étape.

## 2.1 Extraction de données

Le site-web StackExchange propose une interface (voir Figure 1) pour interroger la base de données des questions et des réponses des utilisateurs et télécharger les données obtenues selon une requête. Nous avons suivi la

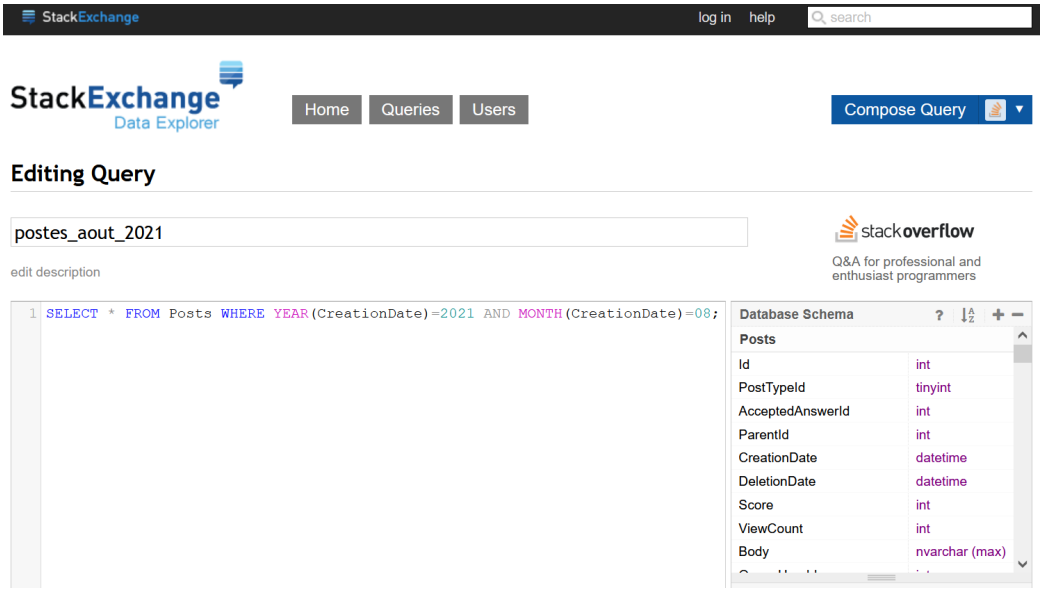


FIGURE 1 – Capture d'écran de l'interface de *Data Explorer* de *StackExchange*

méthodologie suivante pour extraire les données :

— prévoir deux jeux de données distincts :
  1. corpus de travail ;
  2. corpus de test ;

— extraire les publications selon les critères établis :

1. une question doit avoir une réponse acceptée ;
2. score d'"utilité perçue" pour une question $> 0$ ;
3. la période temporelle pour le corpus de travail est comprise entre 01/10/2020 et 01/10/2021 ;
4. le corpus de test correspond aux questions du mois d'octobre 2021 ;

— ne pas explorer le corpus de test.

La Figure 2 montre le nombre de questions par mois que nous avons récupéré en suivant les critères établis. Il est intéressant de noter que cette figure

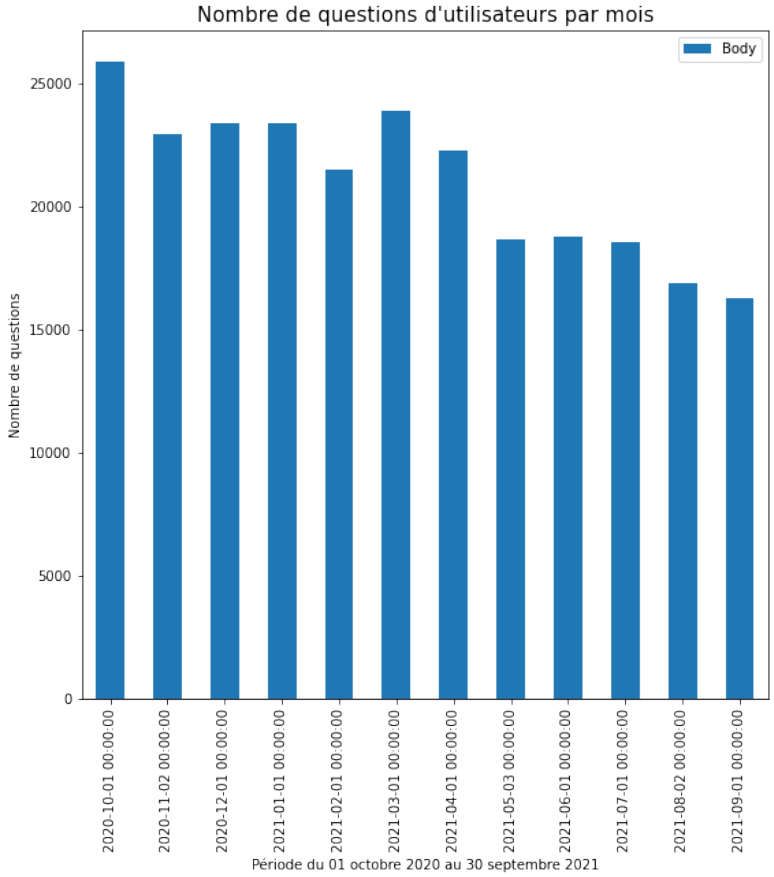


FIGURE 2 – Nombre de questions par mois entre 01/10/2020 et 01/10/2021

nous permet d'observer la diminution progressive de nombre de questions par mois. Toutefois, nous ne pouvons pas avancer d'hypothèse pour expliquer ce phénomène, car ce type d'analyse sort de la portée du sujet de notre travail.

## 2.2 Exploration du Corpus

Pour se familiariser avec les données en notre possession, nous avons exploré le texte des questions des utilisateurs et les tags associés. Nous avons obtenu un Corpus [3] de 252 420 publications [4], contenant du code html, dont nous avons extrait le titre de chaque question, son contenu et les tags associés. La langue du corpus est l'anglais. Le corpus contient 27 535 825 mots avec, en moyenne, 93 mots par question. Comme nous le voyons sur le diagramme en boîte (voir Figure 3), la majorité des questions n'excède pas 95 - 100 mots.

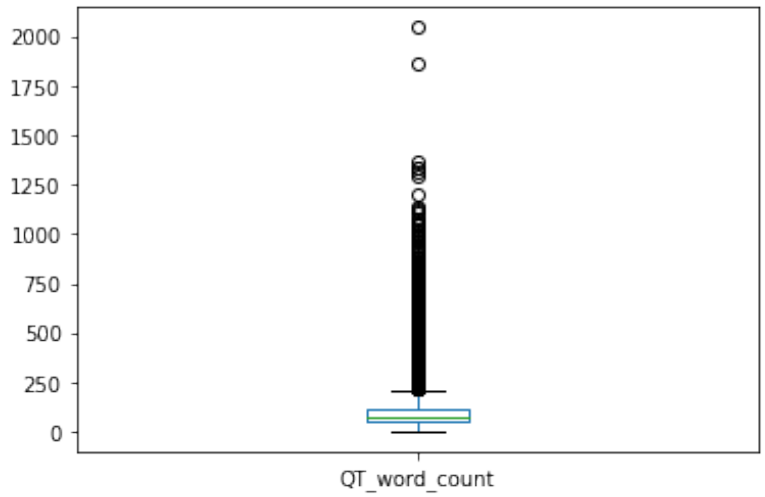


FIGURE 3 – Nombre de mots dans les questions sur StackOverflow

Puisque les questions concernent l'informatique, la majorité contient du code informatique. Le nombre de mots différents représente 1,7% du nombre total des mots. Cela signifie que seul quelques mots spécifiques peuvent permettre de distinguer la sous-thématique à laquelle appartient chaque question dans la thématique commune de programmation [5]. Les hapaxes sont des commandes spécifiques, telles que "initializer-list" ou "conflict-merge-target".
En ce qui concerne les catégories syntaxiques des mots composant les messages, ce sont en majorité les noms (16%) et les déterminants (10%). Cela est probablement lié à la présence des noms de commandes informatiques en grande quantité. Toutefois, les phrases restent bien structurées selon la

3. Ici le corpus de travail
4. Une publication représente ici un ensemble comprenant une question, une réponse et les mots clés.
5. Nous utilisons ce terme ici comme un synonyme du terme "l'informatique".

grammaire anglaise et compréhensibles pour les initiés. Voici un exemple d'une question utilisateur :

```
'I am performing an aggregated array collection using
    the following code in pyspark:[code] I know
    functions like collect forces data into a single
    node. Is it possible to achieve the same result
    while at the same time leveraging the power of
    distributed cloud computing?'
```

Bien que testé, nous n'avons pas réduit le corpus aux lemmes, car les lemmatiseurs ne sont pas adaptés au lexique métier. Cette approche pénaliserait, par exemple, les termes, tels que 'css' -> 'cs' ou encore 'https' -> 'http'. Nous avons supprimé des mots vides[6].

Une analyse rapide des collocations en forme de bigrammes et trigrammes les plus fréquentes permet d'avoir une première idée du contenu des messages :

```
[('Archer', 'Achiever'),
 ('AssetsStarting', 'Biomecs'),
 ('HKCU', 'SOFTWAREMicrosoftOffice160WordOptions'),
 ('Hawke', 'Uma'),
 ('IDbConnection', 'cnn')]

[('ca', 'nt', 'figure'),
 ('ca', 'nt', 'find'),
 ('ca', 'nt', 'get'),
 ('ca', 'nt', 'seem'),
 ('could', 'nt', 'find')]
```

Ces collocations nous permettent de supposer qu'il s'agit de descriptions techniques de difficultés rencontrées par les utilisateurs de SO, ce qui correspond bien à la vocation de ce site-web. Nous remarquons également que les bigrammes, obtenus ici avec la mesure d'association PMI (Pointwise Mutual Information), à l'encontre de trigrammes, permettent de faire ressortir des termes métier. Le nuage de mots sur la Figure 4 représente le contenu des questions composant le corpus de façon plus visuel.

Notre corpus contient 776 026 tags, où 24 410 tags sont des tags distincts (3%). Pour une question, l'utilisateur peut associer un ou plusieurs tags. Dans le corpus que nous avons recueilli, les utilisateurs ont attribué, en moyenne, trois tags par question (voir la Figure 5).

Les tags les plus répandus sont :

6. Voir la liste des mots supprimés en Annexe 5.


FIGURE 4 – Nuage de mots du corpus "Questions StackOverflow 2020 - 2021"

'python', 'javascript', 'r', 'pandas', 'reactjs', 'java', 'html', 'c++', 'c#', 'css'

(voir Figure 6).

Les algorithmes de ML ne sont pas capables de traiter des données brutes, car il n'est pas possible de prévoir tous les formats qu'ils puissent avoir. Nous décrivons les transformations que nous avons appliquées au corpus dans la Sous-section suivante 2.3.

## 2.3 Transformation des données

Nous avons sauvegardé le corpus SO dans un *dataframe Pandas*. Avant de transformer le format textuel de notre corpus en format utilisable par les algorithmes de ML, nous l'avons nettoyé, en supprimant les lignes comportant des cellules vides. Ensuite, nous avons assemblé le contenu des corps des questions avec leurs titres. À cette étape, la première brique de notre futur pipeline est constitué. Le module des prétraitements consiste en la tokenisation par la bibliothèque NLTK et en la suppression de nombres et de mots vides, dont la liste se trouve en Annexe 5.

Les algorithmes de ML nécessitent la transformation de données en forme de caractéristiques (ou "features" en anglais). Dans le cas de l'analyse textuelle, les mots doivent être transformés en vecteurs, une approche dite "sac-de-mots" [7]. Les vecteurs peuvent représenter soit les comptes d'occurrences de mots, soit le nombre de mots pondéré en fonction de leur apparition dans un document par rapport à la totalité du corpus (voir la définition de la méthode TF-IDF [8]). Nous avons choisi cette dernière méthode permettant

7. "bag-of-words" en anglais
8. TF-IDF

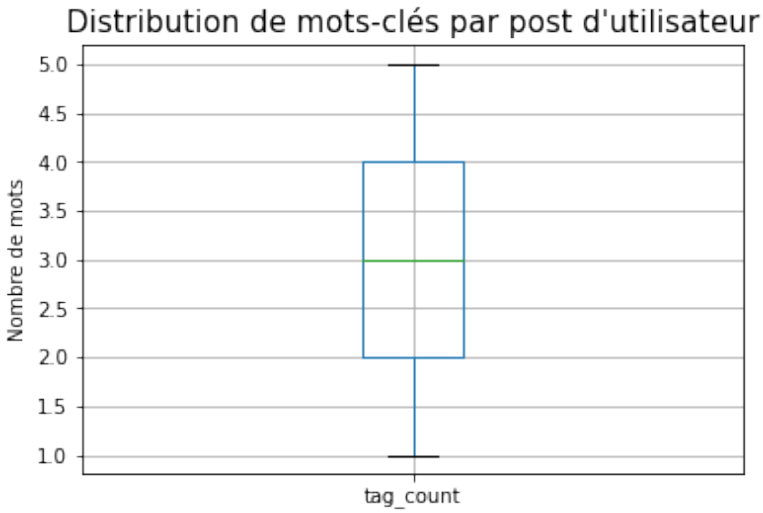


FIGURE 5 – Nombre de mots dans les questions sur StackOverflow

de capturer l'importance d'un mot dans une question utilisateur de SO. En appliquant *TfidfVectorizer* de *sklearn* avec les paramètres $\boldsymbol{max\_df = 0.6}$, $\boldsymbol{min\_df = 9}$[9], nous avons obtenu une matrice creuse de taille 168 996 × 23 119. Le nombre important de paramètres à gérer doit être pris en compte lors du choix de l'algorithme de ML et des prétraitements, dont ce dernier peut avoir besoin.
Cette considération concerne également les tags ou mots-clés. Ainsi, nous avons réduit le nombre de mots clés à cent pour réduire le nombre de paramètres qu'un algorithme de ML doit gérer pour apprendre le modèle de données. Nous les avons triés manuellement et ont obtenu une liste de 99 mots clés.

Dans la Section 3 nous discutons des méthodes de ML applicables aux données textuelles pour la prédiction des mots-clés et avec des caractéristiques comparables à celles de notre corpus.

# 3 Prédiction de mots clés

Comme nous l'avons vu dans l'introduction (Section 1), deux solutions du domaine de ML s'offrent à nous : ML non-supervisé et supervisé. [6] passent en revue les méthodes plus ou moins populaires pour la modélisation du thème d'un document. Les auteurs proposent un arbre de décision qui nous permet de choisir facilement un algorithme non-supervisé. La Figure 7

9. La méthode de choix de valeurs pour ces paramètres est décrite dans la Section 3.

représente un extrait de cet arbre de décision, correspondant au chemin que nous avons pris pour choisir l'allocation de Dirichlet latente (de l'anglais Latent Dirichlet Allocation) ou LDA [10].

[3] regroupent dans leur analyse les travaux à l'état de l'art de l'extraction de mots clés et de résumé. Les algorithmes supervisés les plus utilisés pour ces tâches, selon les auteurs, sont SVM, Naïve-Bayes et les forêts aléatoires. Les algorithmes tels que CRF ou HMM sont utilisés pour des tâches demandant davantage d'analyse sémantique. Par conséquent, nous avons testé les algorithmes supervisés suivants : Multinominal Naïve Bayes, régression logistique, SVM-SVC, forêts aléatoires, Gradient Boosting Classifier et Passive Agressive Classifier. Nous décrivons les pipelines construits et les résultats obtenus de nos expérimentations dans les sous-sections suivantes.

## 3.1 Modèles non-supervisés

Notre pipeline de recherche de paramètres pour l'apprentissage de modèle non-supervisé consiste de boucles permettant de trouver une meilleure combinaison entre le poids d'importance accordé au titre de la question, les paramètres de TF-IDF (***max_ df*** et ***min_ df***) et le nombre de thèmes à prédire. Malheureusement, lors de l'exécution du pipeline, nous avons observé les temps d'exécution supérieurs à neuf heures et la croissance du score de perplexité proportionnel à l'augmentation du nombre de thèmes à prédire. Toutefois, lors de la visualisation des vingt thèmes obtenus avec la librairie Python *pyLDAvis* [4], les résultats semblent être cohérents (voir les Figures 8 et 9). Nous avons donc changé de métrique d'évaluation des résultats du modèle pour *cohérence*, considérée comme une métrique plus proche des évaluations humaines (voir l'article de Shashank Kapadia sur le site-web "Medium" Evaluate Topic Models: Latent Dirichlet Allocation (LDA)).

Nous avons rapporté les meilleurs résultats obtenus dans la Table 1 .

Pour le modèle final, nous avons choisi les paramètres correspondants à 20 thèmes. Moins de thèmes nous paraît trop peu pour capter une variété suffisante de sujets. Malgré les résultats obtenus relativement cohérents sur un corpus de test comportant 15 questions, le modèle ne parvient pas à faire une prédiction pour un corpus consistant d'une seule question et attribue le même poids à tous les thèmes.

L'efficacité limitée du LDA sur notre corpus peut être expliquée par plusieurs éléments. Premièrement, l'examen plus détaillé des mots-clés proposés

10. "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." (Définition de LDA proposée sur Wikipedia. Consulté le 28/01/2022).

| **cohérence** | **poids de titre, n (titre x n)** | **max_df** | **min_df** | **nombre de thèmes** |
|---|---|---|---|---|
| 0.83 | 2 | 0.40 | 8 | 5 |
| 0.81 | 1 | 0.60 | 10 | 10 |
| 0.78 | 3 | 0.60 | 9 | 10 |
| **0.77** | 1 | 0.60 | 9 | **20** |

TABLE 1 – Scores de cohérence obtenus

indique la nécessité de tri supplémentaire des mots de corpus. Ainsi, il serait intéressant de ne laisser que des noms ou des bigrammes, représentantes des paires : [nom + nom] ou [adjectif + nom]. Deuxièmement, une des hypothèses faite par LDA est que tous les sujets sont indépendants les uns des autres [6]. Corpus "Questions StackOverflow 2020 - 2021" est constitué de questions qui appartiennent à un sujet global : l'informatique. Il ne correspond donc pas au corpus "idéal" attendu par le modèle. Compte tenu des résultats obtenus par le model de ML non-supervisé, nous avons procédé aux tests des modèles supervisés que nous décrivons dans la sous-section suivante.

## 3.2 Modèles supervisés

Nous avons choisi sur *sklearn* un éventail des modèles supervisés qui couvre les algorithmes les plus utilisés :

1. régression logistique ;
2. Naïve Bayes multinomial ;
3. SVM SVC ;
4. Gradient Boosting Classifier (GBC) ;
5. les forets aléatoires ;
6. Passive Aggressive Classifier.

Les modèles à base d'arbre se sont avérés trop gourmand en mémoire. Pour pouvoir obtenir un résultat quelconque avec ce type de modèles, nous avons testé une réduction dimensionnelle jusqu'à 100 composants par les algorithmes suivants : LSA, Sparse PCA, MiniBatch SPCA, NMF et LDA. De plus, nous avons dû réduire radicalement le nombre d'estimateurs : jusqu'à dix pour les forêts aléatoires et jusqu'à cinq, pour GBC. L'application de ces mesures a entrainé l'obtention de scores très bas pour ces algorithmes. La Table 2 présente les scores que nous avons pu obtenir pour des différents modèles testés. Lorsqu'un modèle ou une configuration n'est pas présente dans

ce tableau, cela signifie que nous avons eu des problèmes techniques que nous n'avons pas pu résoudre et nous n'avons obtenu aucun résultat.

| **Modèle supervisé** | **Temps moyen d'entrainement (sec)** | TF-IDF | | **Score de Jaccard** |
|---|---|---|---|---|
| | | **max_df** | **min_df** | |
| Dummy Classifier | 159.7 | 0.7 | 10 | 0.008 |
| Random Forest + LDA | 2143.6 | 0.6 | 8 | 0.02 |
| Random Forest + LSA | 557.0 | 0.6 | 8 | 0.03 |
| Multinominal Naïve Bayes | 415.0 | 0.5 | 11 | 0.05 |
| Random Forest + NMF | 1179.4 | 0.6 | 8 | 0.07 |
| GBC + LDA | 4297.0 | 0.6 | 10 | 0.13 |
| Gradient Boosting Classifier (GBC) | 2441.2 | 0.6 | 10 | 0.14 |
| GBC + NMF | 3270.6 | 0.6 | 10 | 0.23 |
| Logistic Regression | 746.7 | 0.6 | 10 | 0.33 |
| Passive Aggressive Classifier | 434.3 | 0.6 | 9 | **0.45** |

TABLE 2 – Résultats obtenus par les modèles de ML supervisés

Comme le montre le tableau des résultats, nous avons obtenu le meilleur score avec le modèle *Passive Aggressive Classifier*. Nous avons retenu ce modèle pour la suite de notre projet. La Table 3 montre les résultats obtenus à l'évaluation sur le corpus de test.

Le score de Jaccard de l'évaluation montre que le modèle n'est pas surentrainé et permet, a priori, d'obtenir une prédiction satisfaisante. Nous avons intégré ce modèle dans une API pour avoir une interface utilisateur.

| Modèle supervisé | Score de Jaccard d'entrainement | Score de Jaccard de test | Précision | Rappel | F-score |
|---|---|---|---|---|---|
| Passive Aggressive Classifier | 0.45 | 0.37 | 0.48 | 0.75 | 0.58 |

TABLE 3 – Scores d'évaluation pour le modèle Passive Aggressive Classifier

# 4 API

Nous avons développé une API Flask avec une interface simple (voir Figure 10). Elle est disponible à l'adresse imas.pythonanywhere.com. Le code de l'application est disponible sur GitHub [11]. L'API permet à un utilisateur renseigner le titre de la question, la question et obtenir une proposition de mots-clés qui y correspondent, en appuyant sur le bouton "Predict".

L'application a été testée par trois experts en informatique volontaires. Les tests ont montré que le modèle donne des prédictions dès que le texte est suffisamment long (approximativement 30 mots) et comporte un ou plusieurs mots-clés plusieurs fois. Cela reste variable en fonction des mots-clés mentionnés. Par exemple, il suffit une dizaine de mots et un mot-clé "MYSQL" pour obtenir une prédiction "MYSQL". Pour les mots-clés plus complexes et moins répandus dans le corpus, comme "google-cloud-platform", il faut un texte de plus de 35 mots et plusieurs mentions de Google Cloud.

# 5 Conclusion

Ce rapport présente en grandes lignes les résultats de travail sur le projet de catégorisation automatique des questions. L'analyse de corpus "Questions StackOverflow 2020 - 2021" et de la littérature sur le sujet, nous a permis de faire les choix des approches et des algorithmes. Nous avons appris un modèle non-supervisé et plusieurs modèles supervisés pour prédire les mots-clés reflétant le thème de la question d'utilisateur du site web StackOverflow. Le modèle choisi donne des résultats satisfaisants. Toutefois, il serait intéressant de nettoyer davantage le corpus et de faire des tests d'autres algorithmes pour améliorer le score de Jaccard.

11. GitHub du projet.

# Références


[1] Slobodan Beliga, Ana Meštrović, and Sanda Martinčić-Ipšić. An overview of graph-based keyword extraction methods and approaches. *Journal of information and organizational sciences*, 39(1) :1–20, 2015.

[2] Costas Gabrielatos. Keyness analysis : Nature, metrics and techniques. In *Corpus Approaches to Discourse*, pages 225–258. Routledge, 2018.

[3] Zara Nasar, Syed Waqar Jaffry, and Muhammad Kamran Malik. Textual keyword extraction and summarization : State-of-the-art. *Information Processing & Management*, 56(6) :102088, 2019.

[4] Carson Sievert and Kenneth Shirley. Ldavis : A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, pages 63–70, 2014.

[5] Miroslav Tushev, Fahimeh Ebrahimi, and Anas Mahmoud. Domain-specific analysis of mobile app reviews using keyword-assisted topic models. 2022.

[6] Ike Vayansky and Sathish AP Kumar. A review of topic modeling methods. *Information Systems*, 94 :101582, 2020.


# A Liste de mots vides

"'m", "'ve", "'s", "'d", "'re", "'ll", "'t", 'she', 'were', 'before', 'an', 'just', 'd', 'from', 'needn', 'these', "won't", 'y', 'up', "mightn't", 'yours', 'can', 'few', 'he', 'didn', 'our', 'theirs', "aren't", "'m", "haven't", 'them', 'out', 'no', 'very', 'doing', 'most', "wasn't", "you're", 'other', 'we', "you'd", 'couldn', 'ours', 've', 'than', 'yourself', 're', 'mustn', 'her', 'mightn', 'too', 'its', 'when', 'hers', 'whom', 'where', 'now', "'ve", 'about', "should've", 'of', 'was', "don't", 'which', 'hadn', 'what', 'but', 'above', 't', 'such', 'm', 'himself', 'down', 'have', 'isn', "'d", 'after', 'him', "mustn't", 'again', "doesn't", 'so', 'that', 's', 'haven', "she's", 'those', 'are', 'if', 'his', 'does', 'how', 'am', 'your', 'and', 'is', 'under', 'my', 'been', 'their', 'why', "you've", 'then', 'or', 'wasn', 'who', "weren't", 'it', "'s", 'this', 'should', 'themselves', 'once', "hadn't", 'i', 'shan', 'yourselves', "'re", 'between', "didn't", 'to', 'ourselves', 'do', "couldn't", 'any', 'me', 'off', 'because', 'each', 'more', 'not', 'all', 'myself', 'you', 'into', 'own', 'the', 'having', 'itself', 'being', 'they', "isn't", "shouldn't", "'t", 'aren', "'ll", 'o', 'until', "it's", 'some', 'will', "hasn't", 'by', 'shouldn', "wouldn't", "you'll", 'hasn', 'there', 'as', 'wouldn', 'with', 'below', 'same', 'nor', 'over', 'be', 'both', 'against', "needn't", 'ma', 'here', 'during', 'on', 'in', 'at', 'has', 'doesn', 'a',

’while’, ’ain’, ’ll’, ’weren’, ’won’, "that’ll", ’for’, ’further’, "shan’t", ’through’, ’only’, ’herself’, ’don’, ’had’, ’did’

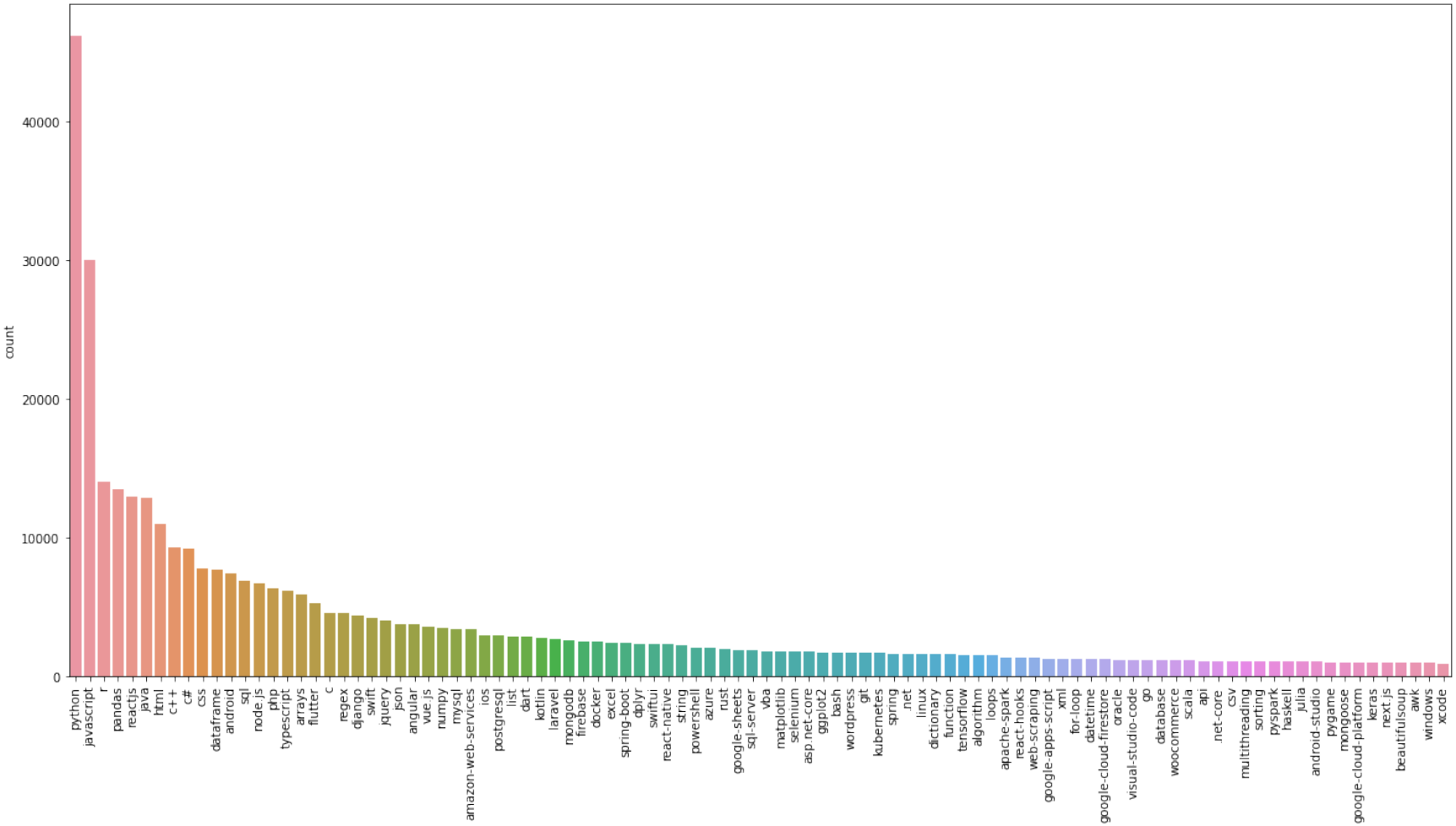


FIGURE 6 – Nombre de tags dans le Corpus

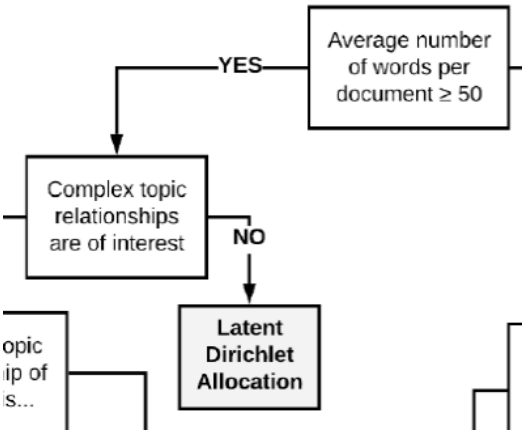


FIGURE 7 – Chemin de décision extrait de [6]

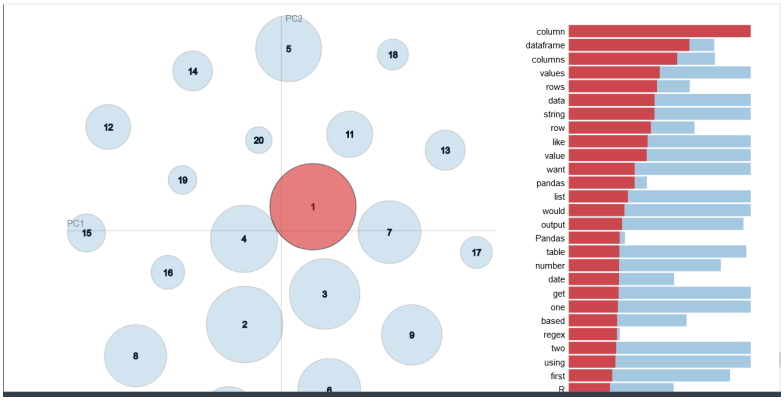


FIGURE 8 – Carte des distances entre les thèmes. Thème 1.

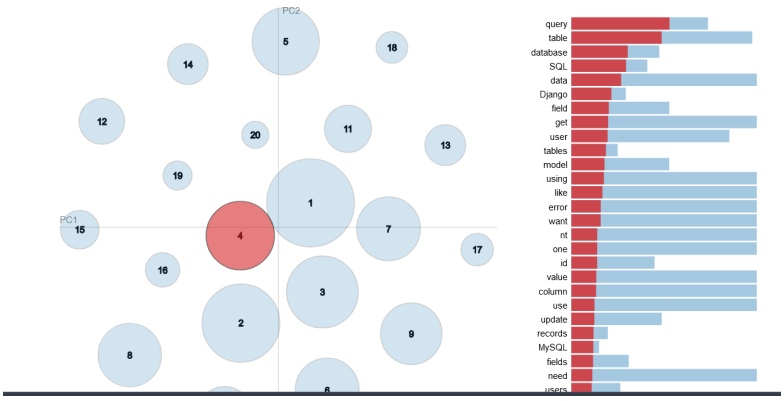


FIGURE 9 – Carte des distances entre les thèmes. Thème 4.


FIGURE 10 – Capture d'écran de l'API