Rapport du projet 5

Catégorisez automatiquement des questions

Openclassroom - Parcours Ingénieur en Machine Learning

Jonas Zeff

30/04/2022

Sommaire

1 - Contexte

Stack Overflow est un site web proposant des questions et réponses sur un large choix de thèmes concernant la programmation informatique.

Pour poster une questions les utilisateurs doivent compléter un formulaire avec un titre, la question et 1 à 5 tags en relation avec leur question.

2 - Enjeux

Proposer un système de suggestion de tags à partir de la question posée par des nouveaux utilisateurs sur Stackoverflow

3 - Méthode

1. Comparer l'approche supervisée à la non supervisée
2. Evaluer les prédictions et sélectionner l'approche la plus efficace
3. Déployer API pour proposer les tags en lien avec la question posée par l'utilisateur

4 - Extraction et filtrage des données

L'extraction sélective du dataset s'est faite avec des requêtes SQL sur la base de données de Stackoverflow appelée Stackexchange explorer.

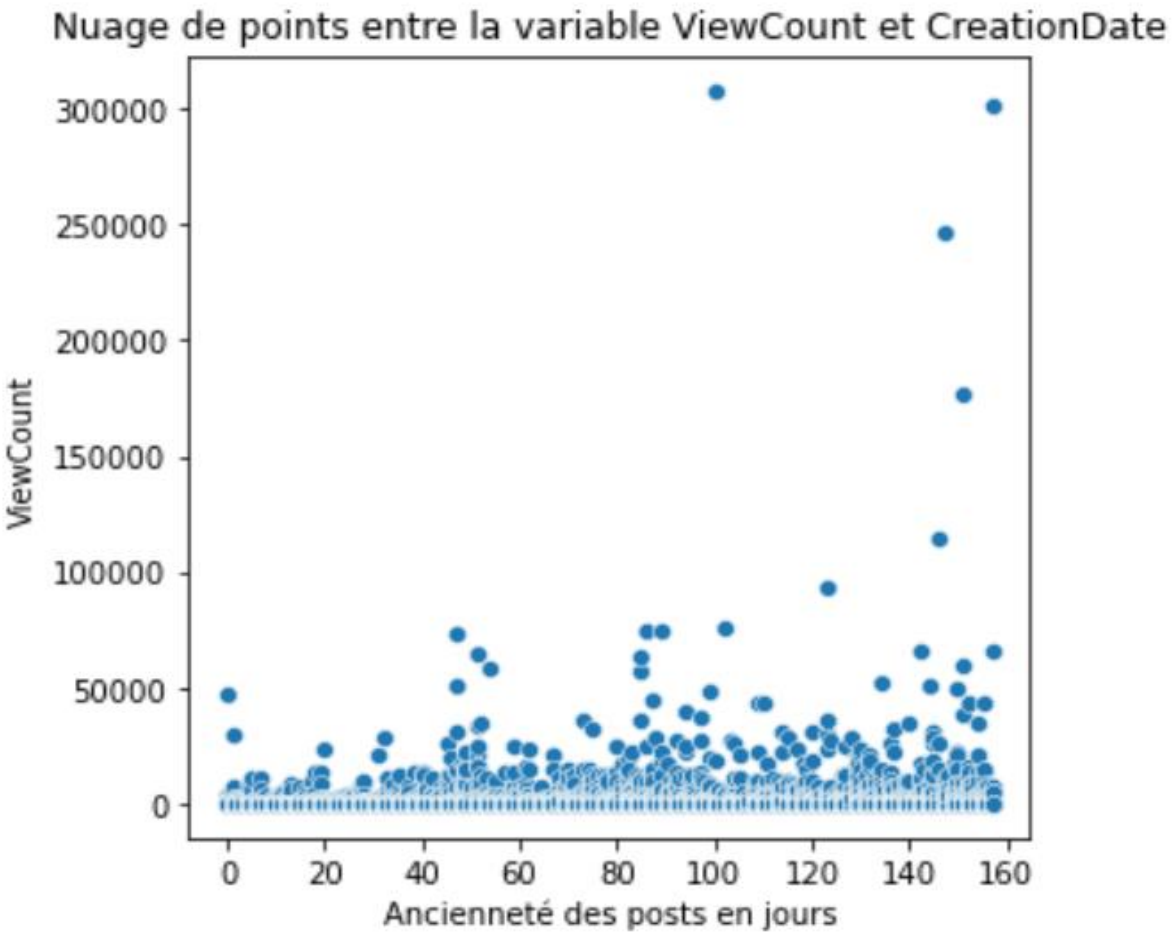
L’extraction s’est accompagnée d’une filtration des données afin de limiter le nombre de données, avoir les données les plus récentes et conserver les valeurs non nulles en vue des analyses exploratoires et de l’entraînement des algorithmes.

Voici la démarche suivie :

- Extraction des 50 000 lignes les plus récentes
- Filtrage par date (du 2020-01-01 au 2022-02-01)
- Score non nul
- Nombre de vues non nul
- Nombre de réponses non nul
- Nombre de commentaires non nul
- Nombre de mise en favoris non nul
- Nombre de vues par jour supérieur à 5

Le dernier point ayant été filtré directement dans le notebook.

5 - Exploration des données

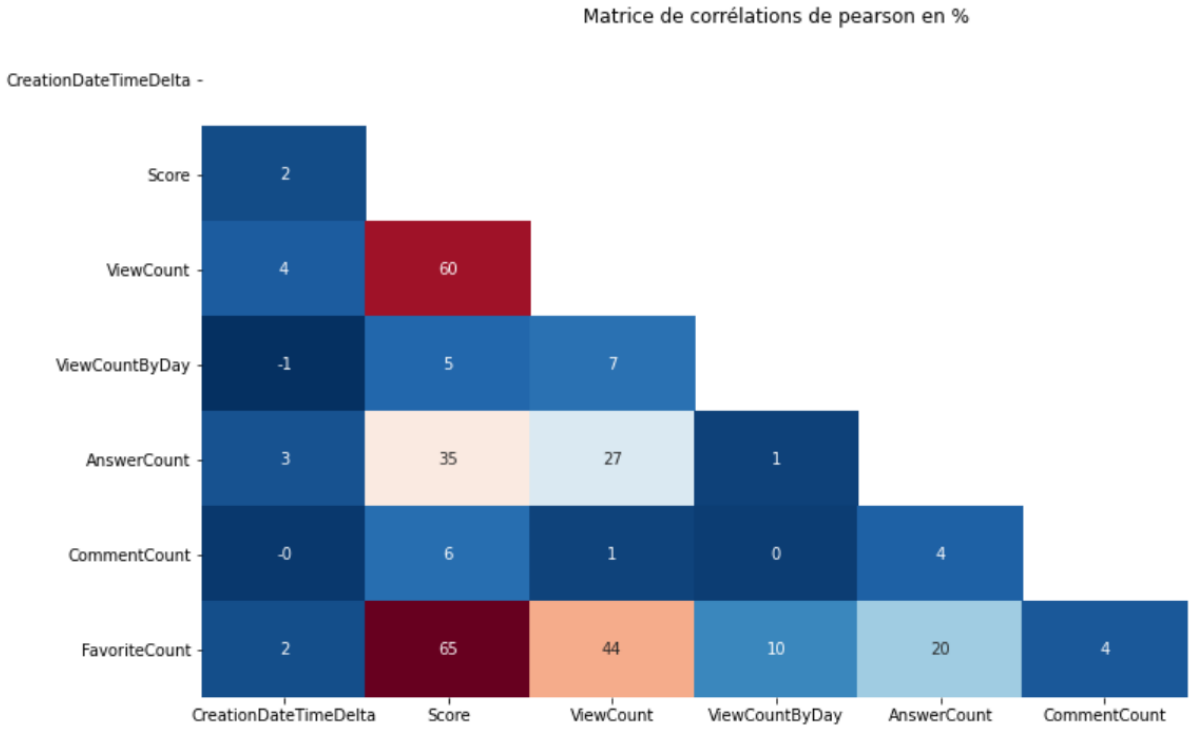


Suite à ce graphique nous pouvons émettre l'hypothèse que plus un post est ancien, plus il a eu le temps d'être vu.

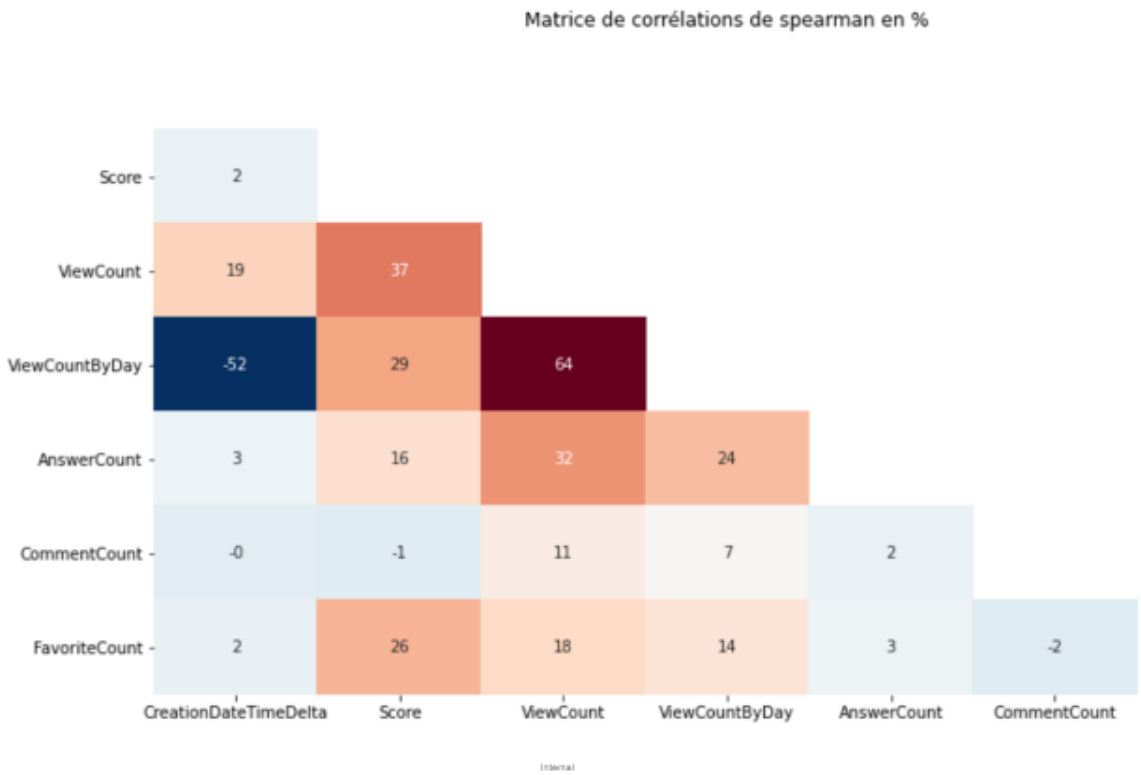
On va venir vérifier cette hypothèse avec des analyses statistiques concrètes en y incluant une nouvelle variable, le nombre de vues par jour (ViewCountByDay).

Tests statistiques de corrélation

```
--------------------------------------------------
Test statistique de corrélation linéaire de Pearson entre CreationDateTimeDelta et ViewCount

Coeficient de corrélation linéaire de Pearson: 0.04010561371755175
 p valeur: 2.93107420231872e-19

L'hypothèse H0 d'indépendance peut être rejetée avec un risque de 5%

--------------------------------------------------
--------------------------------------------------
Test statistique de corrélation de rang de Spearman entre CreationDateTimeDelta et ViewCount

Coeficient de corrélation de rang de Spearman: 0.19394881235579342
 p valeur: 0.0

L'hypothèse H0 d'indépendance peut être rejetée avec un risque de 5%

--------------------------------------------------
```



Cette matrice de corrélations de Pearson présente les corrélations monotones, c’est-à-dire proportionnelles entre les variables.



Cette matrice de corrélations de Spearman présente les corrélations non monotones.

6 - Pré-traitement

Pour entraîner nos algorithmes traité les données à ce fait en suiavnt les points suivant :

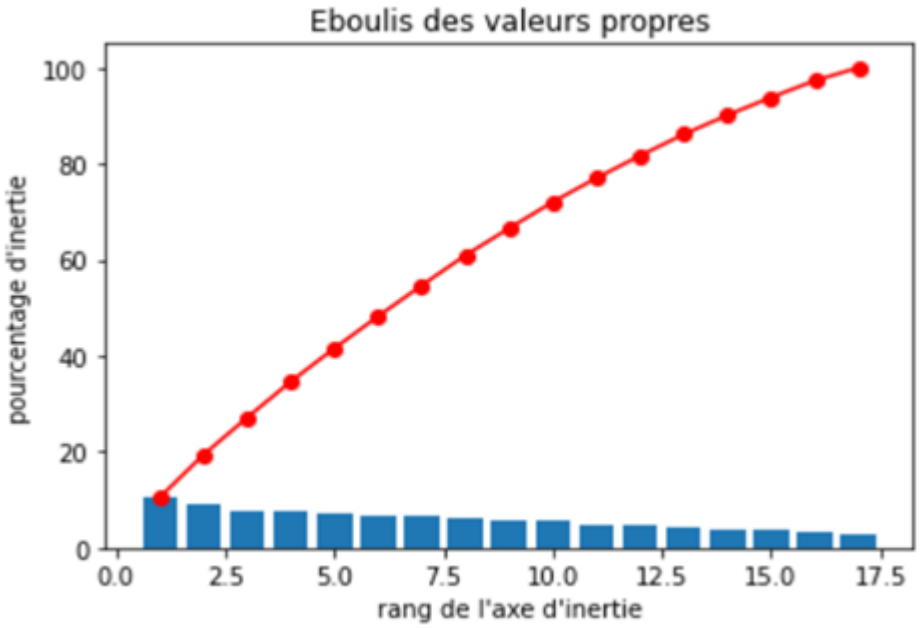
- Nettoyage des balises HTML
- Nettoyage du texte
    - Conserver les caractères alphabétiques
    - Conserver les termes avec plus de 3 lettres

    - Passage du texte en minucules
- Tokenisation pour prendre en compte les termes des phrases individuellement et suppression des Stopwords tels que « however», « then », « else », etc.
- Conservation des noms avec POS qui permet de filtrer les termes en fonction de leur fonction grammaticale par rapport à un dictionnaire et par rapport à la fonction grammaticale du terme précédent.
- Lemmatisation afin de considérer le radical lexique des termes et ne pas prendre en compte les versions conjuguais d'un même terme.
- Suppression des documents vides après pré-traitement.
- Fréquence des tokens en ne gardant que les 200 termes les plus fréquents.
- Vectorisation du corpus : TF – IDF qui est un modèle qui prend en compte la fréquence du terme dans le document et le nombre de documents dans lesquels le terme apparaît afin d'une part minorer l'impact des termes qui apparaissent dans de nombreux documents et afin de normaliser la taille des documents.

7 - Approche supervisée

L'approche supervisée demande les pré-traitements spécifiques des données suivant :

- Dédoublonnage des labels
- Partition des données
- Réduction des données par PCA pour réduire la dimensionnalité des variables nécessaires pour entrainer nos modèles en réduisant le nombre variables au minimum tout en conservant 85% de l'inertie, soit 13 composantes principales.



- Vectorisaton des labels

Pour évaluer les scores de prédiction des différents modèles supervisés testés (KNN, SVM, RandomForest, GradientBoosting) et sélectionner celui qu'on va utiliser nous utilisons le score relatif à la capacité du modèle à prédire les tags en correspondance aux tokens du corpus présenté, ici la colonne micro_ precision.

Les colonnes micro_recall et micro_f1 évaluent respectivement la capacité à retrouver les documents dans lesquels les tags sont présents et une mesure harmonique des deux précédentes. Ces deux mesure ne sont pas directement incluent dans la sélection du modèle mais nous permet d'avoir une analyse plus fine des résultats.

	micro_precision	micro_recall	micro_f1
knn	0.478571	0.054339	0.097597
svm	0.721311	0.035685	0.068006
Random Forest	0.420561	0.036496	0.067164
Gradient Boosting	0.161593	0.055961	0.083133

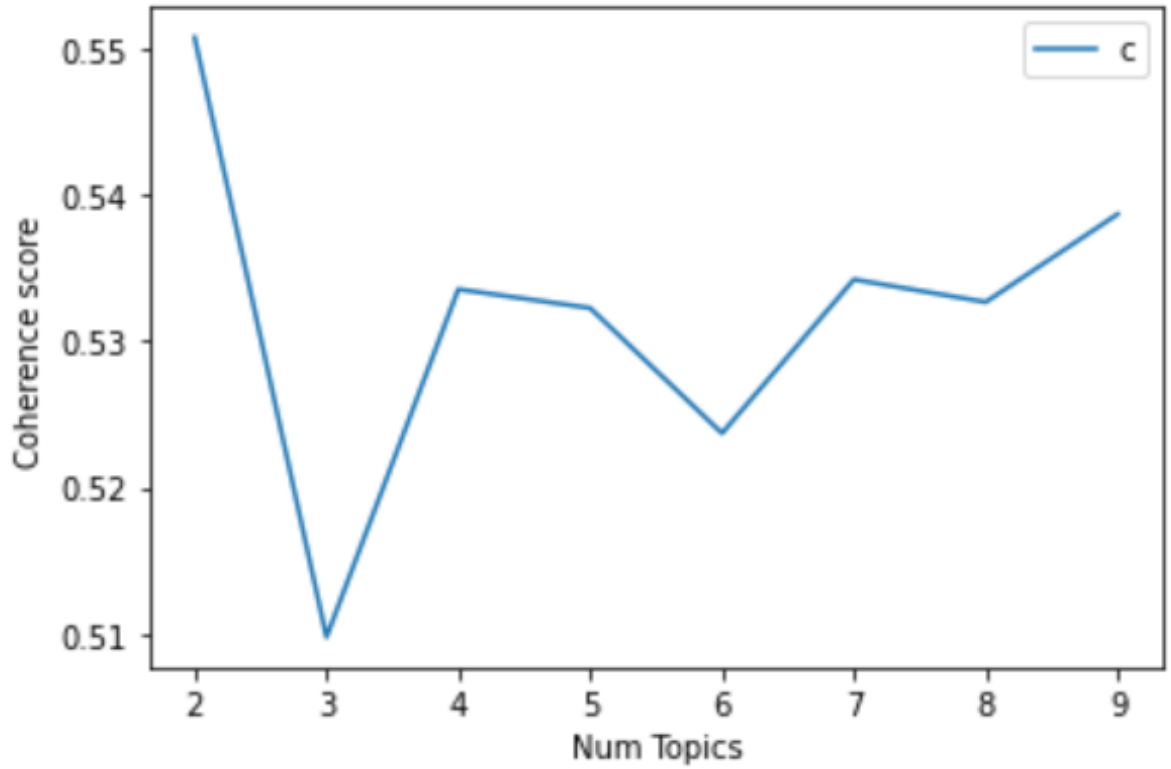
8 - Approche non supervisée

L'approche non supervisée prend en compte des algorithmes de prédiction/classification qui ne nécessitent pas d'être entrainés avec des valeurs prédites en avance.

J'ai utilisé à ce fait le modèle LDA (Latent Dirichlet Allocation) qui permet classer les données par sujet en prenant en considération les points suivants :

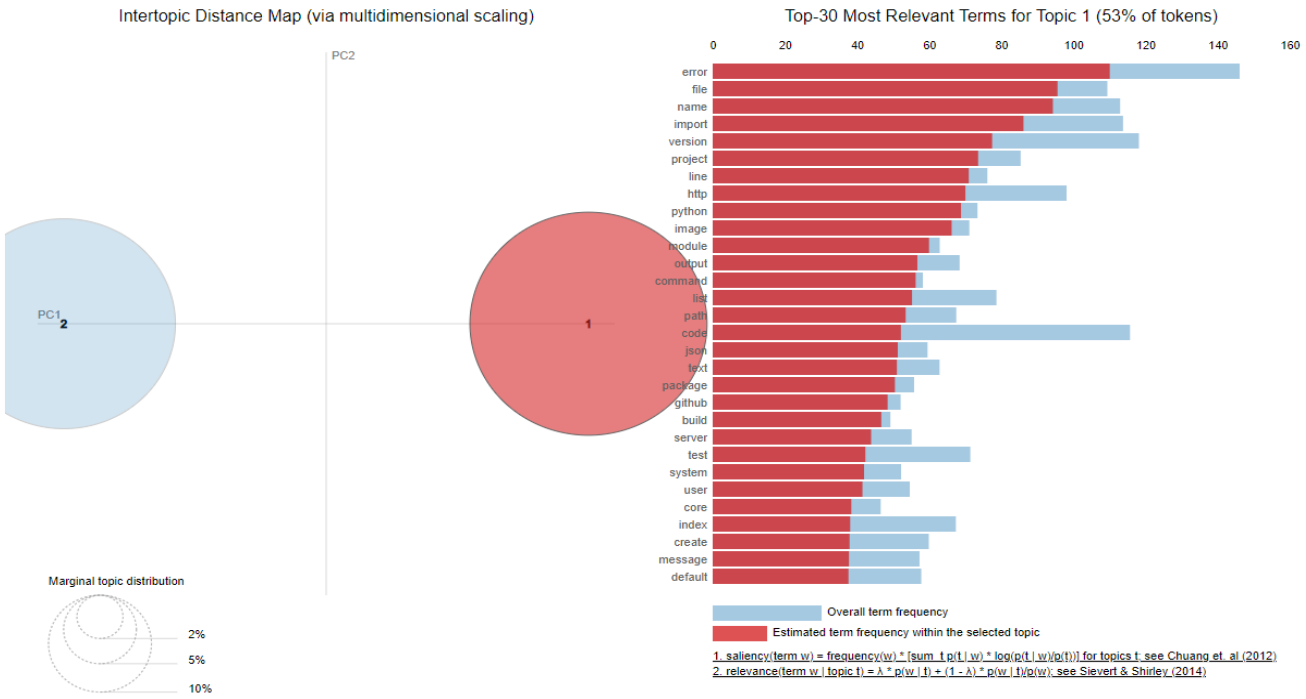
- Un sujet (*Topic*) regroupe les termes en relation avec une thématique proche.
- Les sujets sont répartis en proportions différentes au sein des documents.
- Un sujet est composé d'une distribution des termes associés à ce terme.

Pour entrainer ce modèle il est nécessaire de déterminer en amont le nombre de sujets en évaluant le degré de similarité sémantiques entre les termes d'un même sujet avec le score dit de cohérence.




```
Num Topics = 2  has Coherence Value of 0.5507
Num Topics = 3  has Coherence Value of 0.5098
Num Topics = 4  has Coherence Value of 0.5335
Num Topics = 5  has Coherence Value of 0.5322
Num Topics = 6  has Coherence Value of 0.5289
Num Topics = 7  has Coherence Value of 0.5342
Num Topics = 8  has Coherence Value of 0.5327
Num Topics = 9  has Coherence Value of 0.5387
```

Dans ce cas nous pouvons voir que le score de cohérence est le plus élevé quand le nombre de sujets est de 2. Nous pouvons observer la distribution des 30 termes les plus fréquents par sujet (ici le sujet 1) dans le graphique ci-dessous.



9 - Comparaison des deux approches

Les résultats des deux approches sont comparés aux tags pré-traités du document.

	pré-traités	aprroche supervisée	approche non supervisée
0	[node, reactjs, webpack, webstorm]	[]	[error, project, issue]
1	[java, android, kotlin, gradle]	[java]	[module, error, core, file, build, message]
2	[javascript, reactjs, install, yarnpkg]	[]	[application, error, create]
3	[architecture, compiler, optimization]	[]	[time, size, code, return]
4	[webpack]	[]	[module, http, github, error, command, path, n...
5	[swift, uinavigationcontroller, xcode]	[]	[application, code, default, problem, http]
6	[android, react, native]	[]	[error, project, build, test]
7	[exception, segmentation, fault]	[]	[time, question, solution]
8	[python, algorithm, sorting]	[]	[question, work]
9	[azure, azure, devops, azure, repos]	[]	[list]
10	[python, cpython, python, internals]	[python]	[value]
11	[openssl, certificate, let, encrypt]	[]	[return, code, type, error, message, http, pro...
12	[statement, syntax]	[]	[]
13	[reactjs, node, module, package, json, postcss]	[]	[import, module, package, http, json, error, v...
14	[array, initialization, undefined, behavior, z...	[]	[time, size, number, problem, class, error, va...
15	[performance, time, precision, trigonometry]	[]	[time, project]
16	[python, python, rounding, integer, division]	[python]	[python]
17	[xcode, swift, package, manager]	[]	[package, error, file, project, version, build...
18	[performance, hashtable]	[]	[size, problem, example, work, value, code, re...
19	[python, list, caching]	[]	[time, code]
20	[node, reactjs, sas]	[]	[version, error]
21	[powershell, visual, studio, code, bash]	[]	[code, something, default]
22	[multidimensional, array, language, lawyer, un...	[]	[const, function, something, call]
23	[android, flutter, android, studio]	[]	[version, project]
24	[auto, concept, integral]	[]	[return, code, type, something]

Nous pouvons voir que sur 25 itérations sur différents documents l'approche non supervisée prédit un nombre de tags corrects plus élevé que l'approche supervisée.

Au-delà des résultats chaque approche à ses avantages et désavantages.

	Avantages	**Désavantages**
Approche supervisée	• Modèles et indicateurs connus	• Pré-traitements nombreux
Approche non supervisée	• Pré-traitements peu nombreux • Un modèle unique • Réduction dimensionnelle	• Evaluation difficile des performances du modèle

C'est en vue de ces critères et des résultats de prédiction que je choisie l'approche non supervisée pour déployer l'API

10 - Déploiement de l'API

Voici le lien de l'API déployée en utilisant streamlit :

http://share.streamlit.io/zartrock/tagsproposition/main/code.py