Projet : Catégoriser automatiquement des questions

Antoine Maby - 15/09/2021

Problématique

Stack Overflow est un site célèbre de questions-réponses

Il faut entrer plusieurs tags de manière à retrouver facilement la question par la suite

Suggérer des tags relatifs à la question posée.

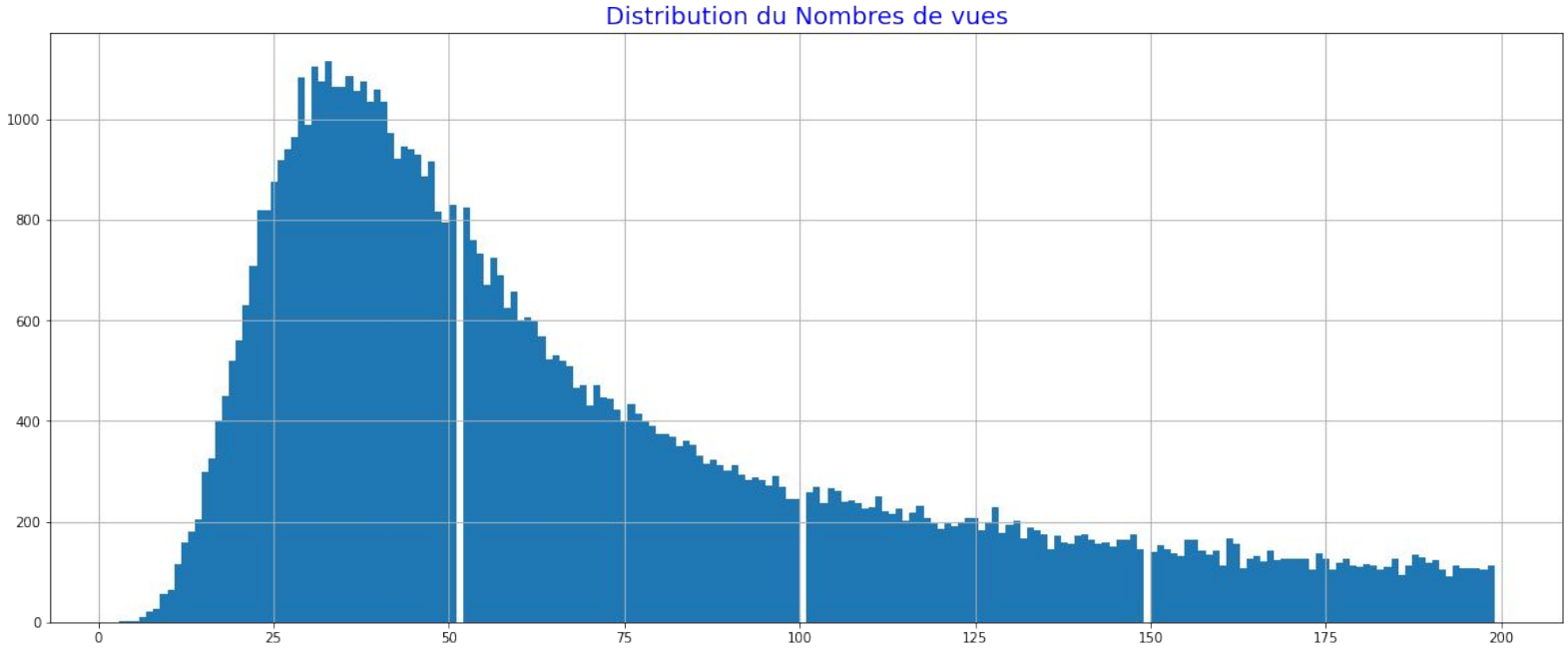
Récupération des données

- Stack Overflow propose un outil d'export de données Stack Exchange

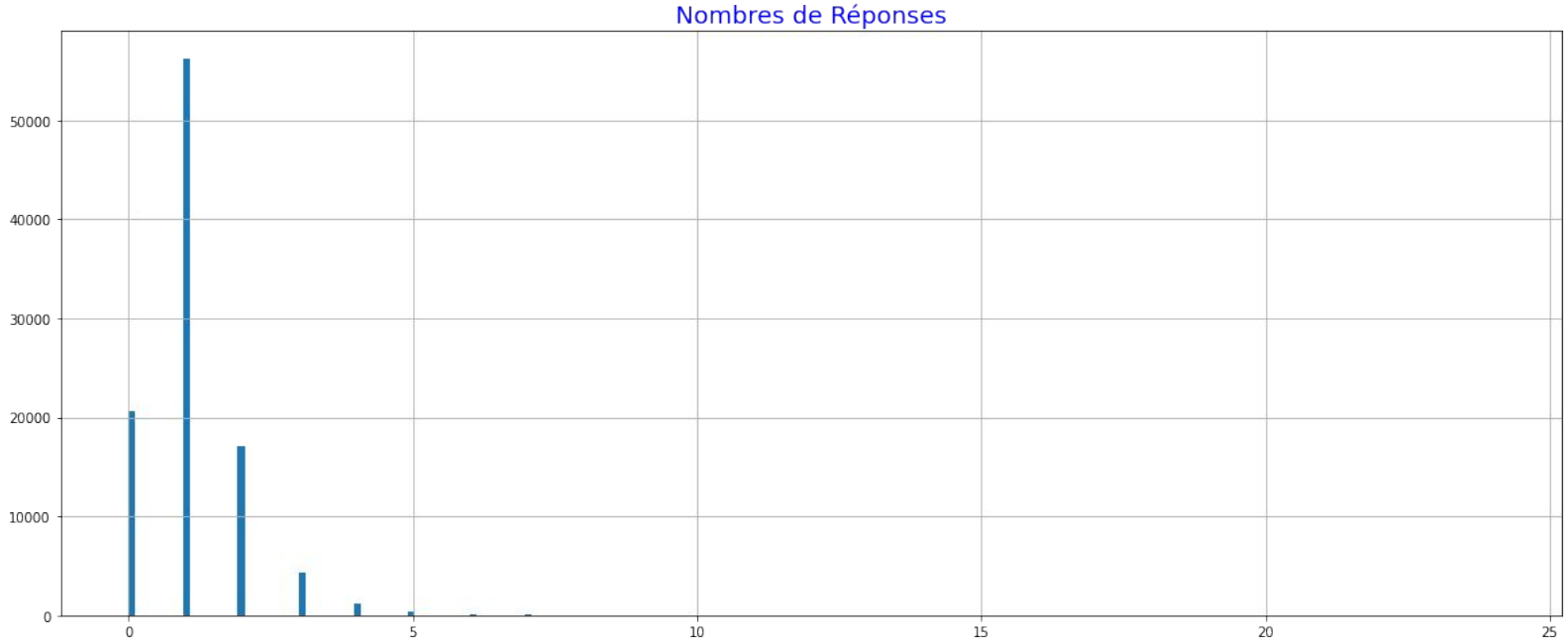
- Nous allons déterminer des critères pour réduire les posts par leur qualité

- Regardons les variables que nous allons utiliser

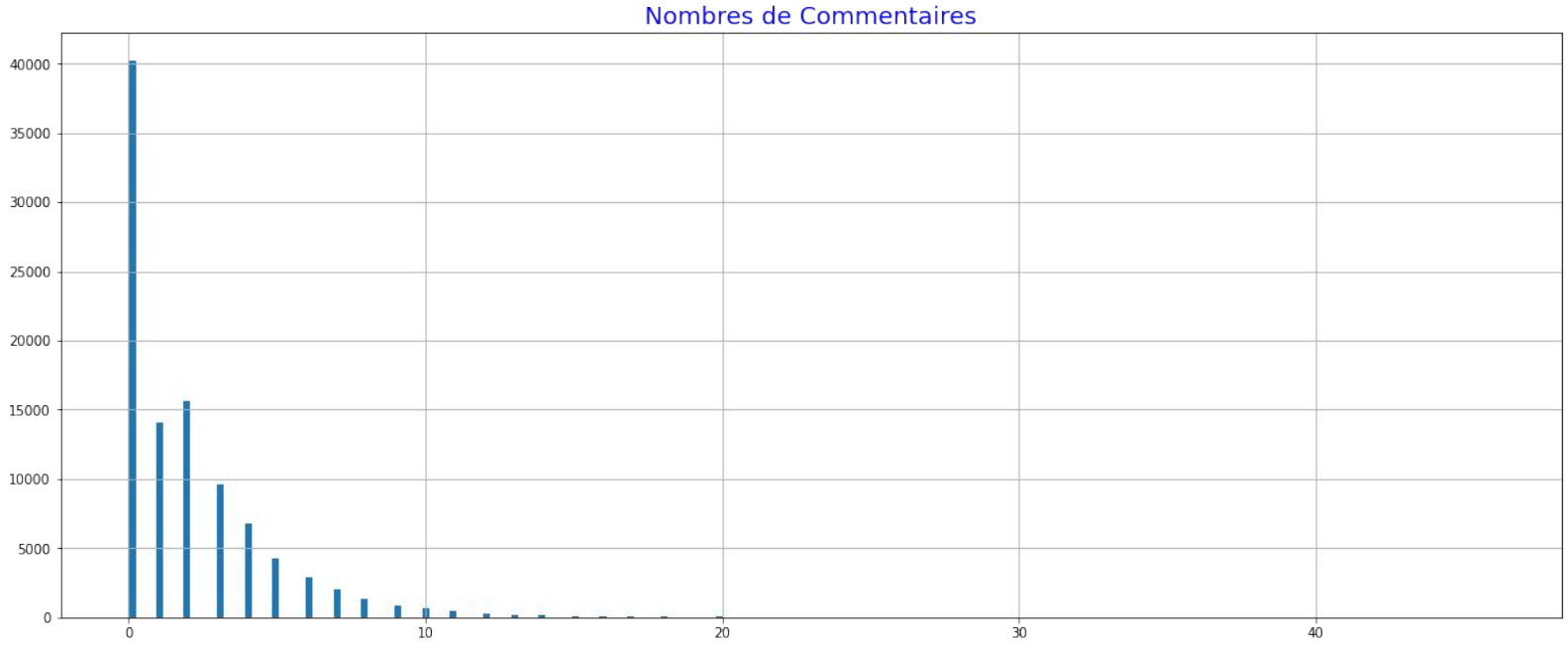
Récupération des données



Récupération des données



Récupération des données



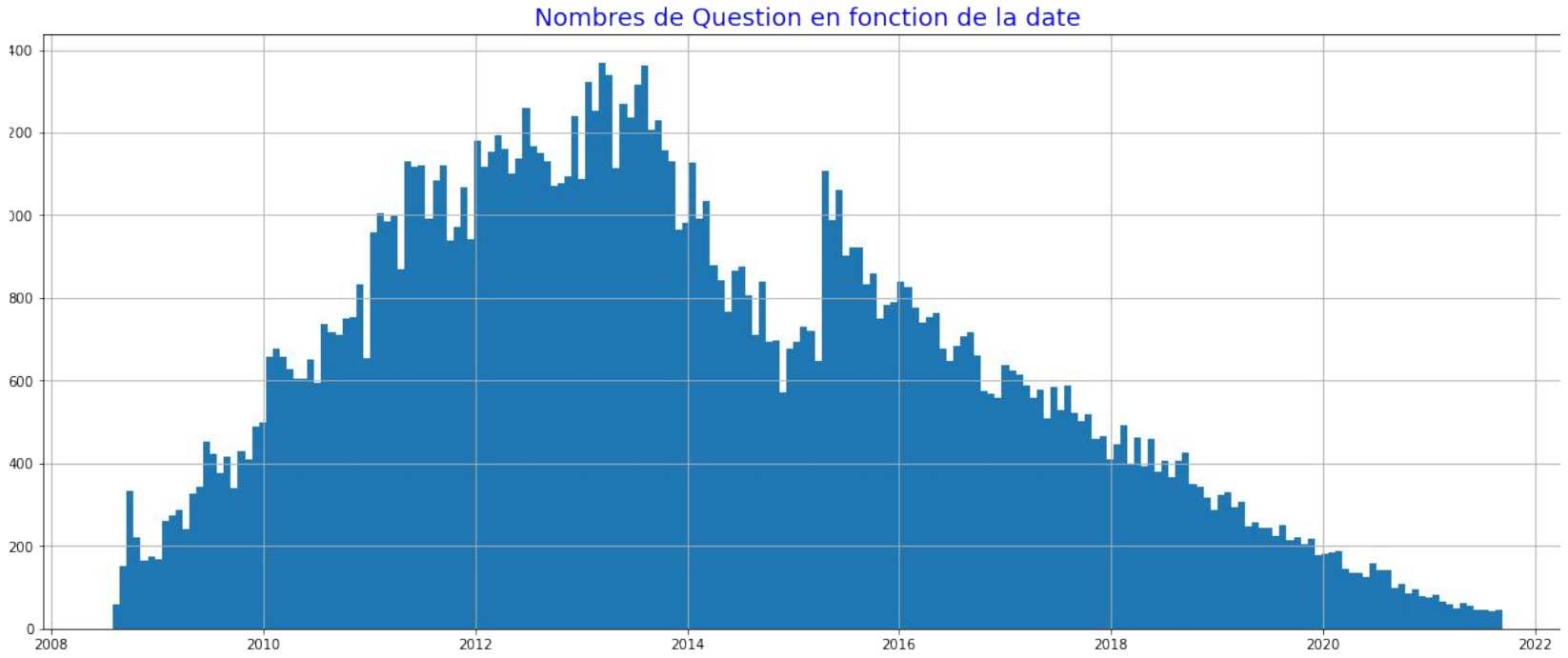
Récupération des données

Maintenant que nous avons nos critères pour déterminer la qualité des posts

Nous avons récupéré tous les posts avec cette qualité depuis la création de Stack Overflow

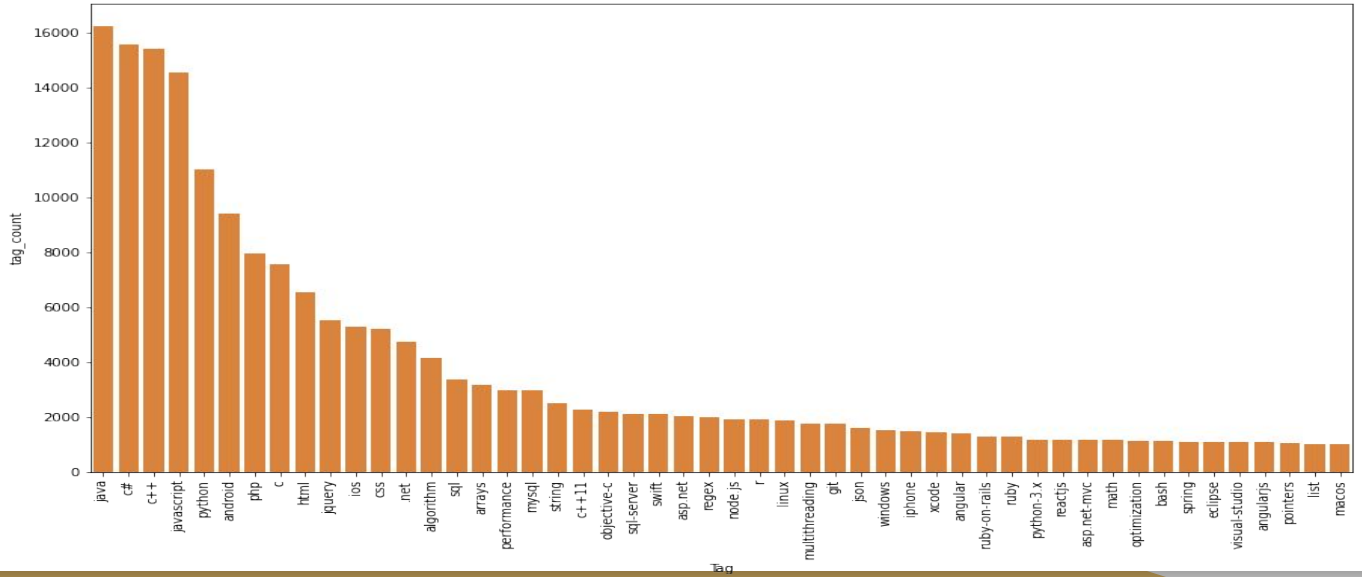
Passons à la partie exploration et au nettoyage de notre corpus

Nettoyage et Exploration du corpus



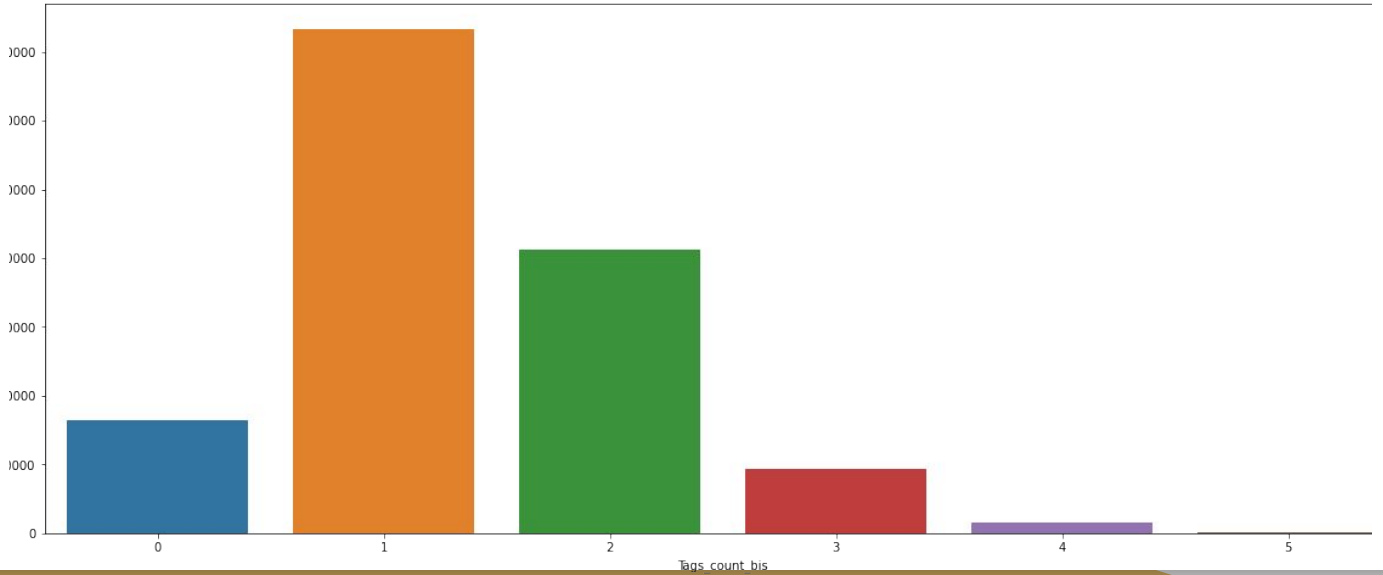
Nettoyage et Exploration du corpus

Il y a plus de 18 000 tags différents dans notre corpus, regardons les plus utilisés



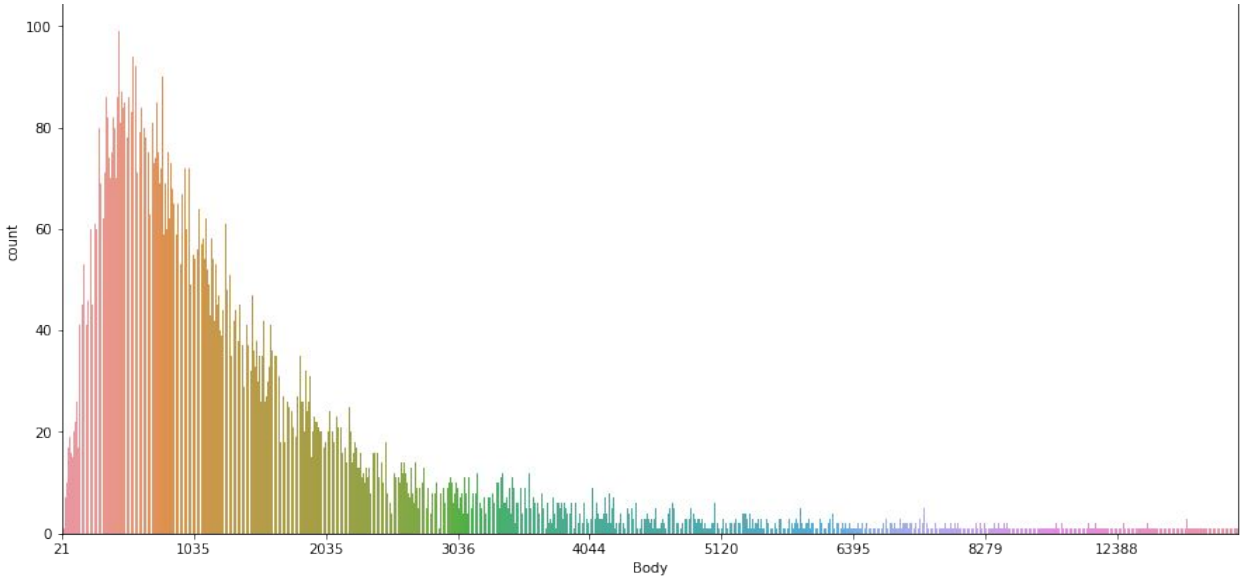
Nettoyage et Exploration du corpus

Après restrictions aux 50 tags les plus utilisés, regardons le nombre de tag par post



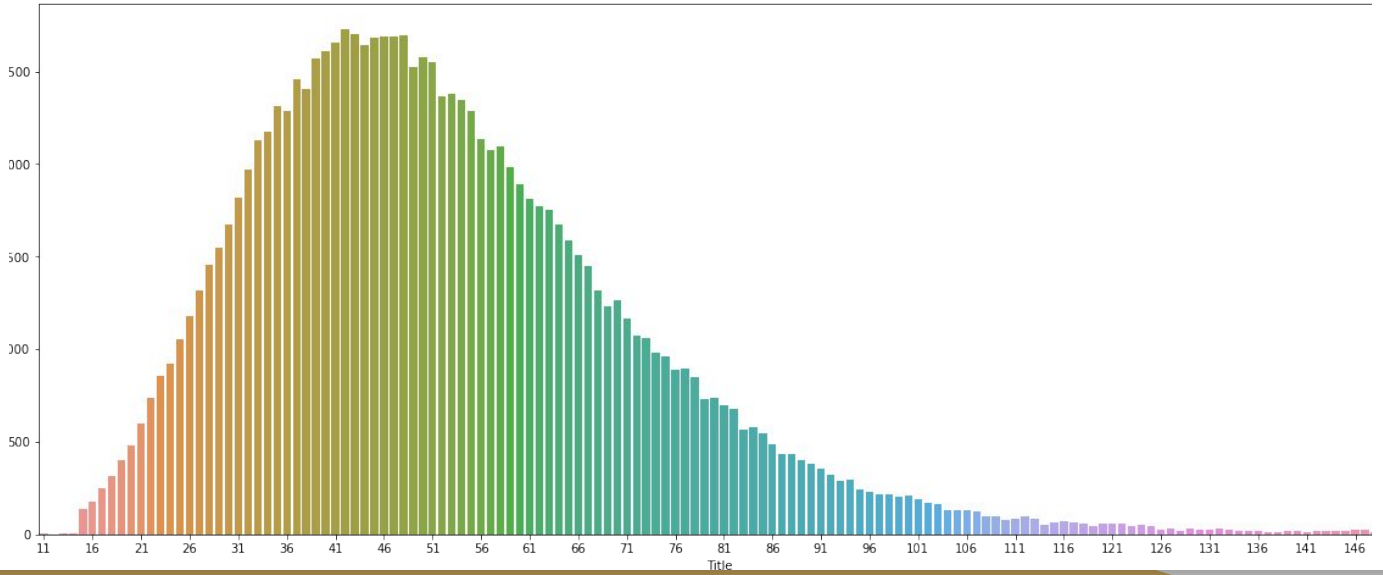
Nettoyage et Exploration du corpus

Nous allons passer au nettoyage de notre corpus. Regardons la longueur du corps



Nettoyage et Exploration du corpus

Regardons aussi la longueur des titres



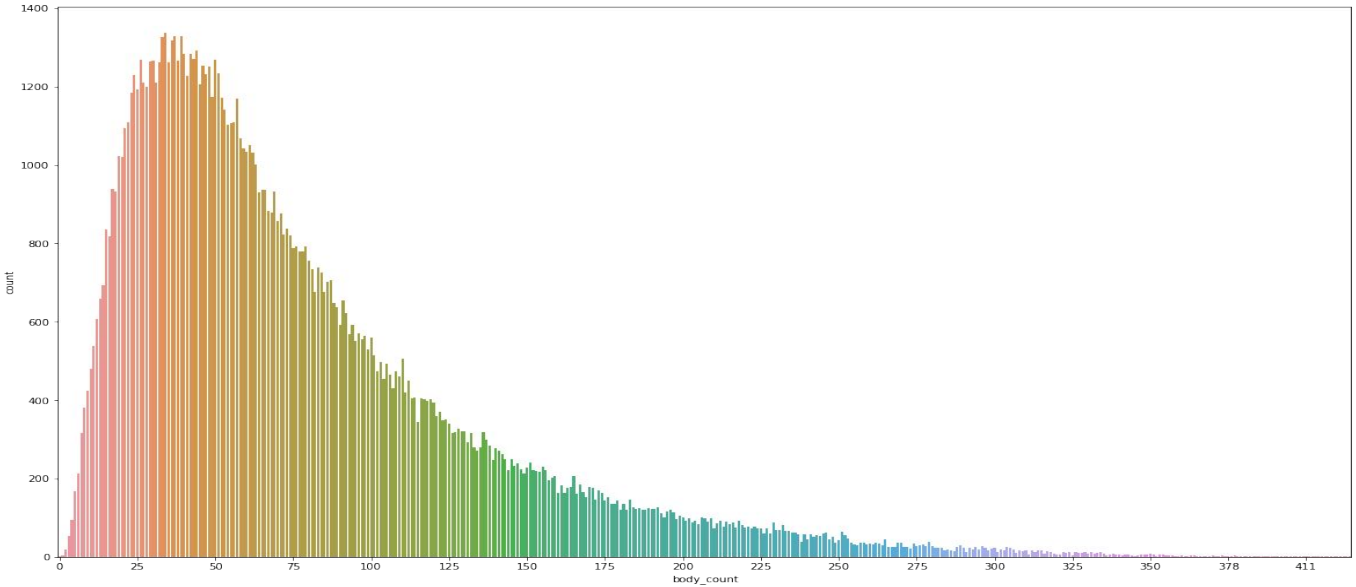
Nettoyage et Exploration du corpus

Pour nettoyer le corpus, nous allons supprimer

- Les bannières HTML
- Mettre tout le texte en minuscules
- Supprimer les caractères Unicode
- Suppression des espaces supplémentaires
- Suppression de la ponctuation
- Suppression des liens
- Supprimer les nombres
- Supprimer les Stopwords
- Tockeniser
- Lemmatiser

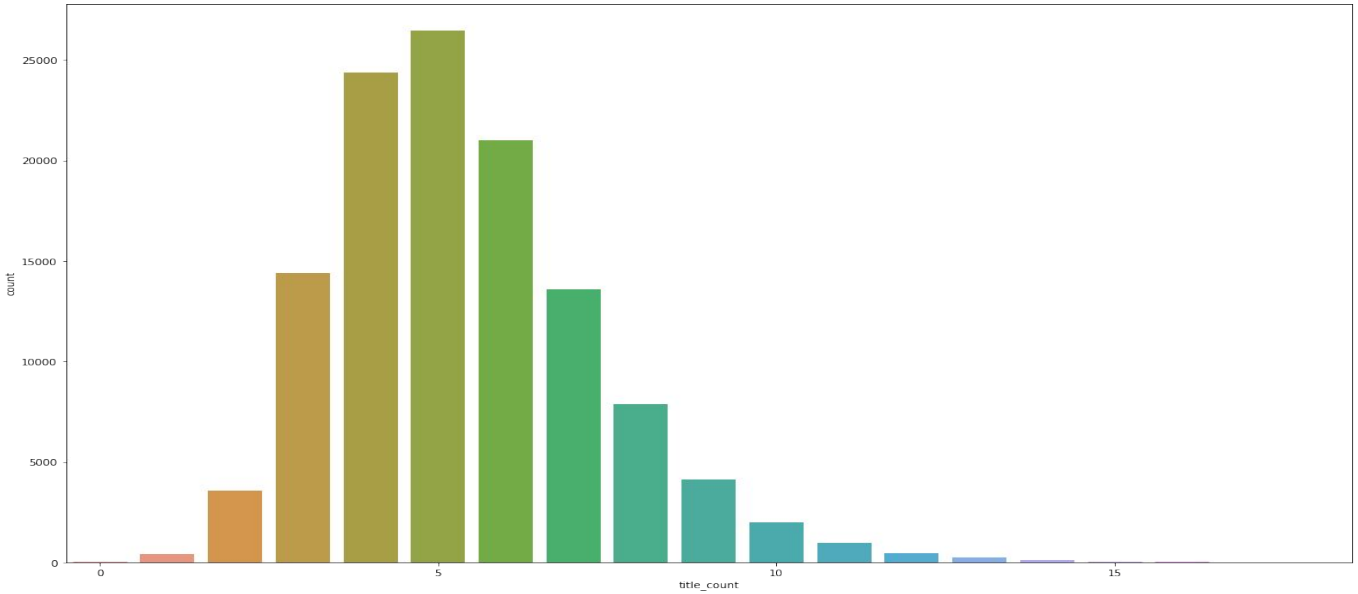
Nettoyage et Exploration du corpus

Grâce à cela, nous avons ainsi pour le corps du post, le nombre de Token



Nettoyage et Exploration du corpus

Et le Nombre de Token des titres



Nettoyage et Exploration du corpus

Maintenant que nous avons nettoyé notre corpus, passons à l'assignation des tags aux posts

Traitons l'automatisation des données textuelles avec des modèles supervisés et non supervisés

Modèle non supervisé

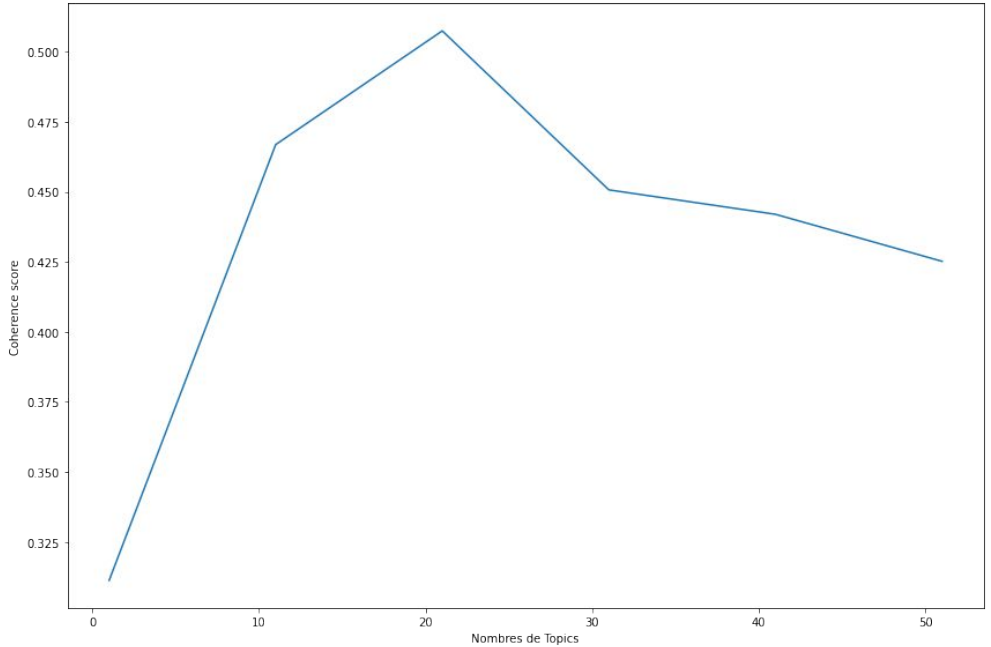
Latent Dirichlet Allocation est un modèle probabiliste qui permet d'obtenir des Clusters de Posts

Pour cela, nous devons d'abord déterminer le bon nombre de Clusters en utilisant le cohérence score

Nous ferons ensuite une assignation de tags à partir de ces clusters obtenus

Modèle non supervisé

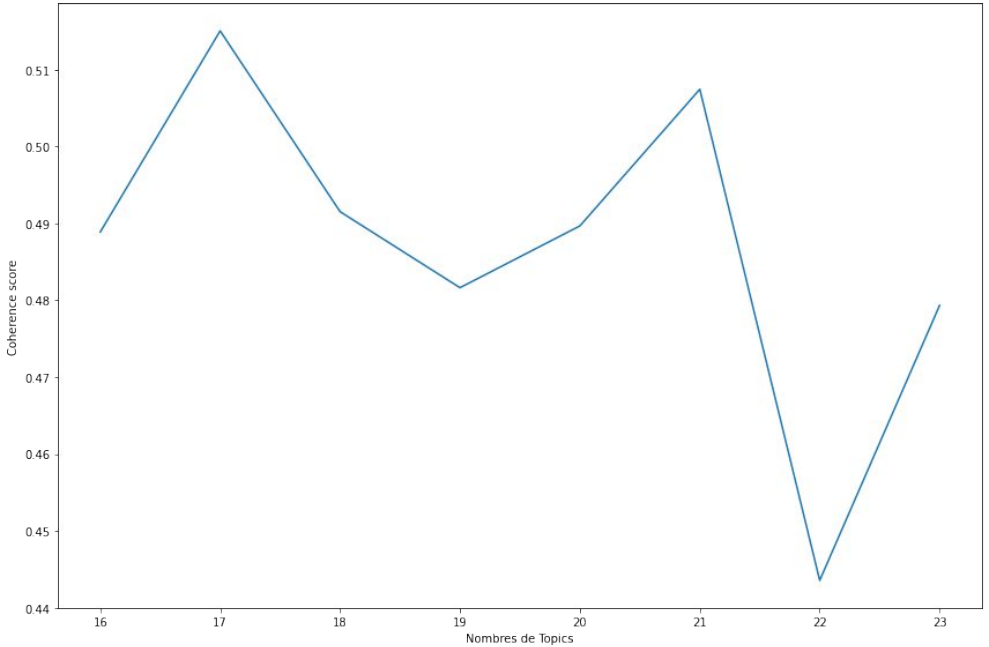
Tracé du score de cohérence en fonction du nombre de Topic



Modèle non supervisé

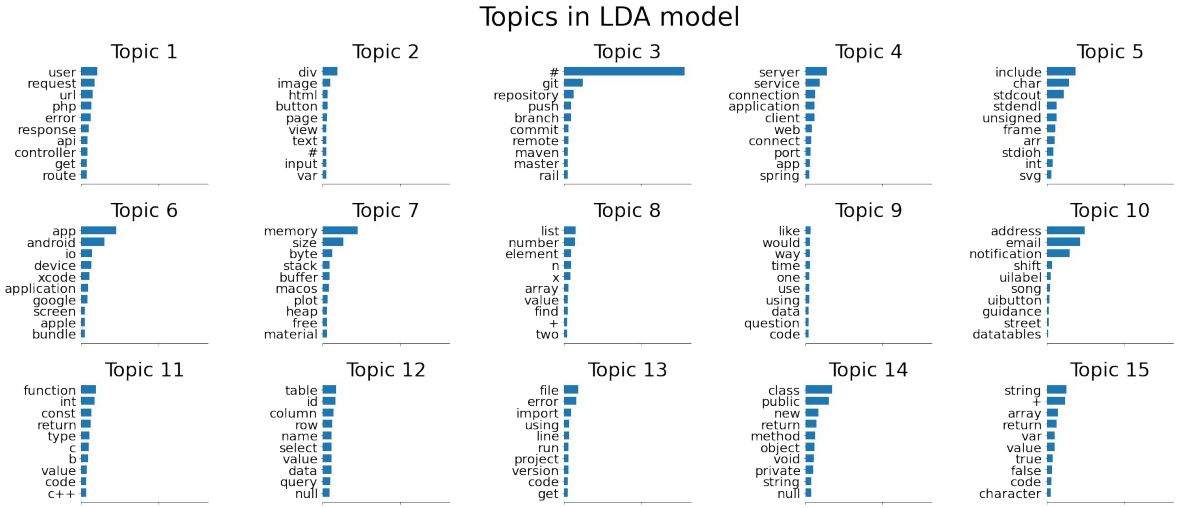
Tracé du score de cohérence en fonction du nombre de Topic

Le Score de cohérence le plus élevé est obtenu pour 17 Topics



Modèle non supervisé

Maintenant que nous avons obtenu le nombre de Topics optimaux, regardons les mots qui les composent



Modèle non supervisé

Pour déterminer les tags associé à chaque post, nous allons :

- Prendre l'intersection de nos 50 tags avec les mots des Topics pour taguer les Topics
- Chaque post est associé à un Topic
- Prendre l'intersection entre ces tags et les mots du posts pour déterminer le tag

Modèle non supervisé

Maintenant que nous avons assigné les tags à chaque post. Nous pouvons regarder les performances en les comparant avec les tags réels

	LDA
F1	0.24
Recall	0.18
Précision	0.51
Accuracy Moyenne	0.19
Accuracy Faible	0.28
Accuracy Forte	0.13

Modèle supervisé

Passons maintenant aux modèles supervisés

Pour chaque modèle, nous avons effectué une GridSearchCV pour déterminer les hyper paramètres sur une partie de nos données

Nous avons ensuite regardé les performances de ces paramètres sur l'ensemble des données en refaisant une GridSearchCV

Ensuite, nous pourrons comparer les performances sur le jeu de données Test

Modèle supervisé

Pipeline : LogisticRegression et OnevsRestClassifier

Hyperparamètres :

- C : paramètre de régularisation

- Penalty : Norme de la pénalité

Modèle supervisé

Regardons maintenant les performances de ce modèle sur les sets de train

	LDA	Logistic
F1	0.24	0.65
Recall	0.18	0.57
Précision	0.51	0.76
Accuracy Moyenne	0.19	0.66
Accuracy Faible	0.28	0.77
Accuracy Forte	0.13	0.55

Modèle supervisé

Pipeline : RandomForestClassifier et OnevsRestClassifier

Hyperparamètres :

- n estimator : Nombre d'arbres
- max depth : Profondeur de l'arbre
- min sample split : Nombre minimum pour séparer un noeud
- min sample leaf : Nombre minimum pour être un noeud

Modèle supervisé

Regardons maintenant les performances de ce modèle sur les sets de train

	LDA	Logistic	RandomForrest
F1	0.24	0.65	0.65
Recall	0.18	0.57	0.79
Précision	0.51	0.76	0.56
Accuracy Moyenne	0.19	0.66	0.90
Accuracy Faible	0.28	0.77	0.92
Accuracy Forte	0.13	0.55	0.86

Modèle supervisé

Pipeline : MLPCLassifier

Hyperparamètres :

- Hidden Layer Size : Nombres de neurones à la ième couches
- Learning rate init : Taux d'apprentissage initiale
- Max iter : Nombre maximal d’itération
- Alpha : Paramètre de pénalité

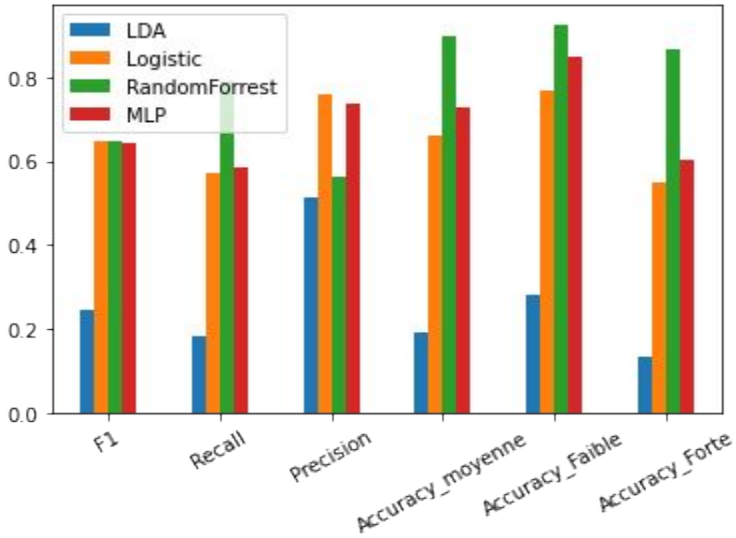
Modèle supervisé

Regardons maintenant les performances de ce modèle sur les sets de train

	LDA	Logistic	RandomForrest	MLP
F1	0.24	0.65	0.65	0.64
Recall	0.18	0.57	0.79	0.58
Précision	0.51	0.76	0.56	0.74
Accuracy Moyenne	0.19	0.66	0.90	0.73
Accuracy Faible	0.28	0.77	0.92	0.85
Accuracy Forte	0.13	0.55	0.86	0.60

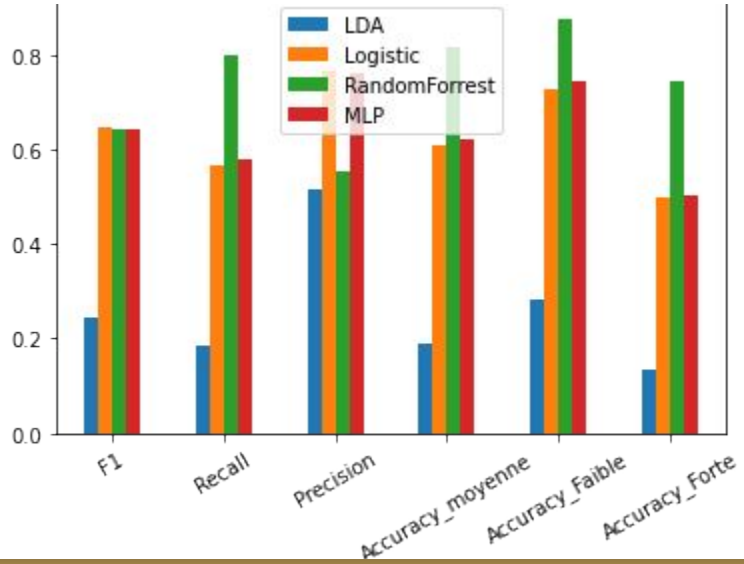
Choix du modèle

Maintenant que nous avons tous les modèles optimisés, regardons les performances pour choisir le modèle final



Choix du modèle

Il semblerait que le RandomForest soit le plus performant, regardons les performances sur un set de Test



Complément

- Création d'une classe pour utiliser les différents modèles

- Création de répertoires pour les différents Notebook et une API
- https://github.com/Ulytonio/API-Stackoverflox
- https://github.com/Ulytonio/Stack-Overflow-Maby-Antoine

- Création d'une API avec Fast API et Heroku
- https://api-stackoverflow.herokuapp.com/docs#/default/Question_Pr_dictor__get

- Pickles pour stocker tous les modèles finaux

Merci pour votre attention