

OPENCLASSROOMS

Projet N° 5

Catégorisez automatiquement des questions

Guillaume Lemêle - *Machine Learning Engineer* - 10/2022 - 06/2023

SOMMAIRE

CATÉGORISEZ AUTOMATIQUEMENT DES QUESTIONS

-
- Étape n° 1
- Présentation de la problématique et du prétraitement effectué, cleaning, feature engineering et EDA
- Étape n° 2
- Présentation de l'approche non supervisée et features extraction
- Étape n° 3
- Présentation de l'approche supervisée et du modèle final sélectionné
- Étape n° 4
- Démonstration de l'API et conclusion

Présentation de la problématique et du
prétraitement effectué, cleaning, feature
engineering et exploration

ÉTAPE N° 1



Présentation de la problématique



► Problématique :

- Stack Overflow est un site célèbre de questions-réponses liées au développement informatique.

► Objectif :

- Pour les nouveaux utilisateurs , développer un système de suggestion de tags pour le site

► Mission :

- Appliquer des méthodes d'extraction de features spécifiques des données textuelles.
- Mettre en œuvre une approche non supervisée afin de proposer des mots clés.
- Mettre en œuvre une approche purement supervisée

Récupération de la base de données

1. Requête SQL : 50000 individus sur la base de données (SELECT TOP 50000).
2. Nous filtrons la récupération du jeu de données avec différents critères :
 - Ce sont des questions (PostTypeId = 1)
 - Les plus vues (ViewCount > 10) ;
 - Pouvant être considérées comme favorites par des internautes (FavoriteCount >= 0, car si > 0, nous donne 137 individus seulement) ;
 - Considérées comme pertinentes par des internautes (Score > 0) ;
 - Ayant eu au moins une réponse (AnswerCount > 0)
 - Comprenant chacune 5 tags

Prétraitement effectué

- ❑ Web scraping : BeautifulSoup

Récupération des informations d'un site web en fonction des balises HTML

Retrait des balises liées au code (pour le contexte des questions sur StackOverflow)

- Retrait des balises de code
- Suppression des adresses URL
- Permet de conserver uniquement la question, sans le code.

Cleaning

❑ Construction du mapping des langages :

- *Avant d'enlever certaines imperfections, nous créons un mapping afin de sauvegarder des items important, par exemple C++ devient « language C++ »*

❑ Nettoyage du texte avec Regex:

- Enlever la ponctuation et les caractères spéciaux
- Nous compilons tous les signes à enlever (`r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[*\(\)\,\]|(?:%[0-9a-fA-F][0-9a-fA-F]))+'`) et nous utilisons Regex.

❑ Suppression des stop words :

- Mots répétitifs et qui ne véhiculent pas de sens particulier.
- Exemple : « i », « me », « the ». Etc.

❑ Mise en minuscule

Feature engineering

❑ Tokenisation

- Conversion d'un texte en une séquence de "tokens" ou de "jetons". Un token est une unité élémentaire du texte; un mot ou une ponctuation.

❑ Fusion des variables « Titre » et « question ».

- Pour ainsi avoir plus d'éléments de texte à notre disposition

❑ Création d'une fonction de offrant un choix entre la racinisation (stemmatisation) et la lemmatisation

- Cette fonction permettra d'optimiser l'analyse du texte en réduisant les mots à leur racine ou leur forme canonique

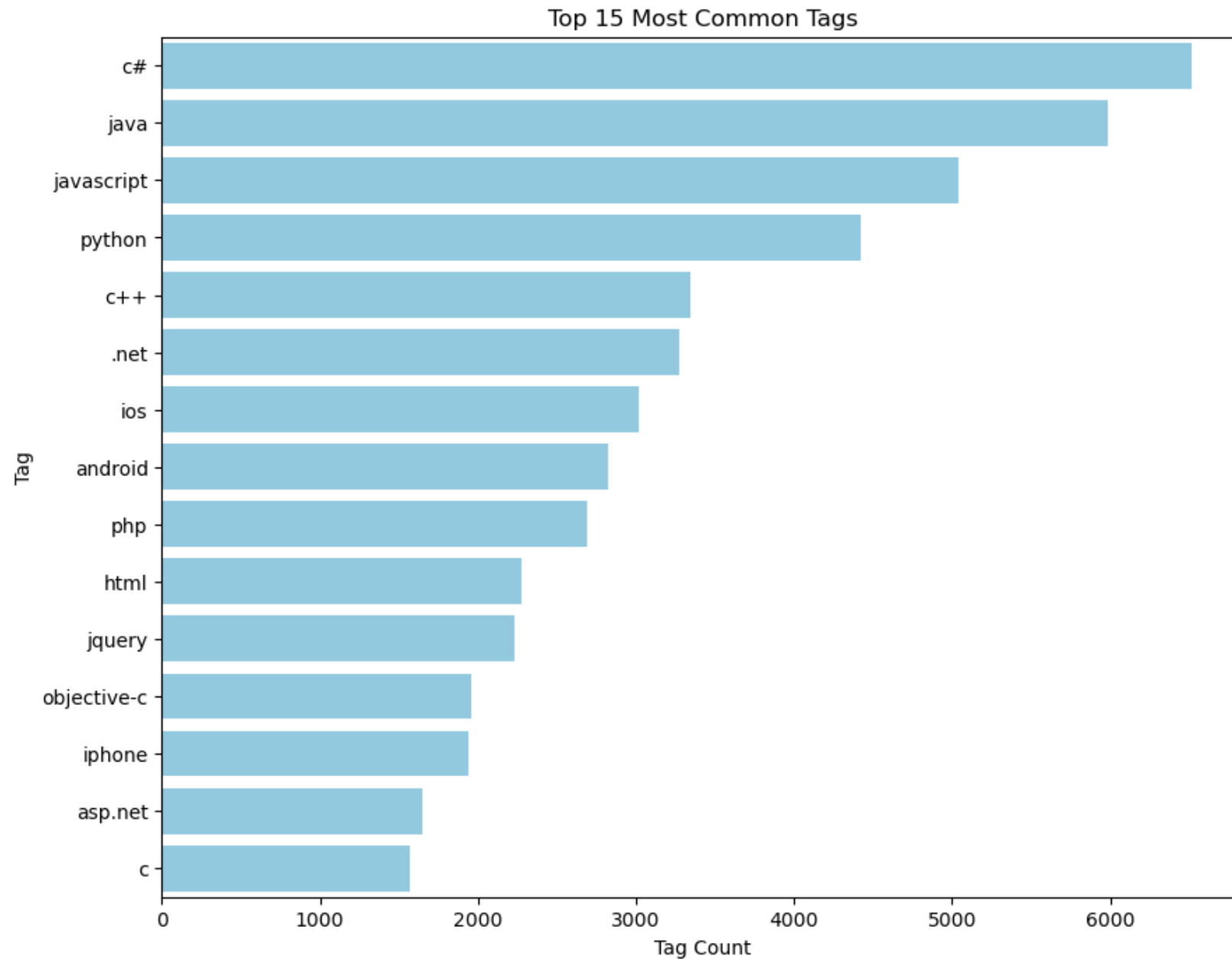
Feature engineering

- ❑ Sélection des tags :
 - Sélection des tags pertinents pour l'étude ; les 15 tags les plus communs
 - Conservation des individus contenant au moins 1 tag du top 15 des tags.
 - Supprimer des autres tags au sein de la nouvelle colonne.

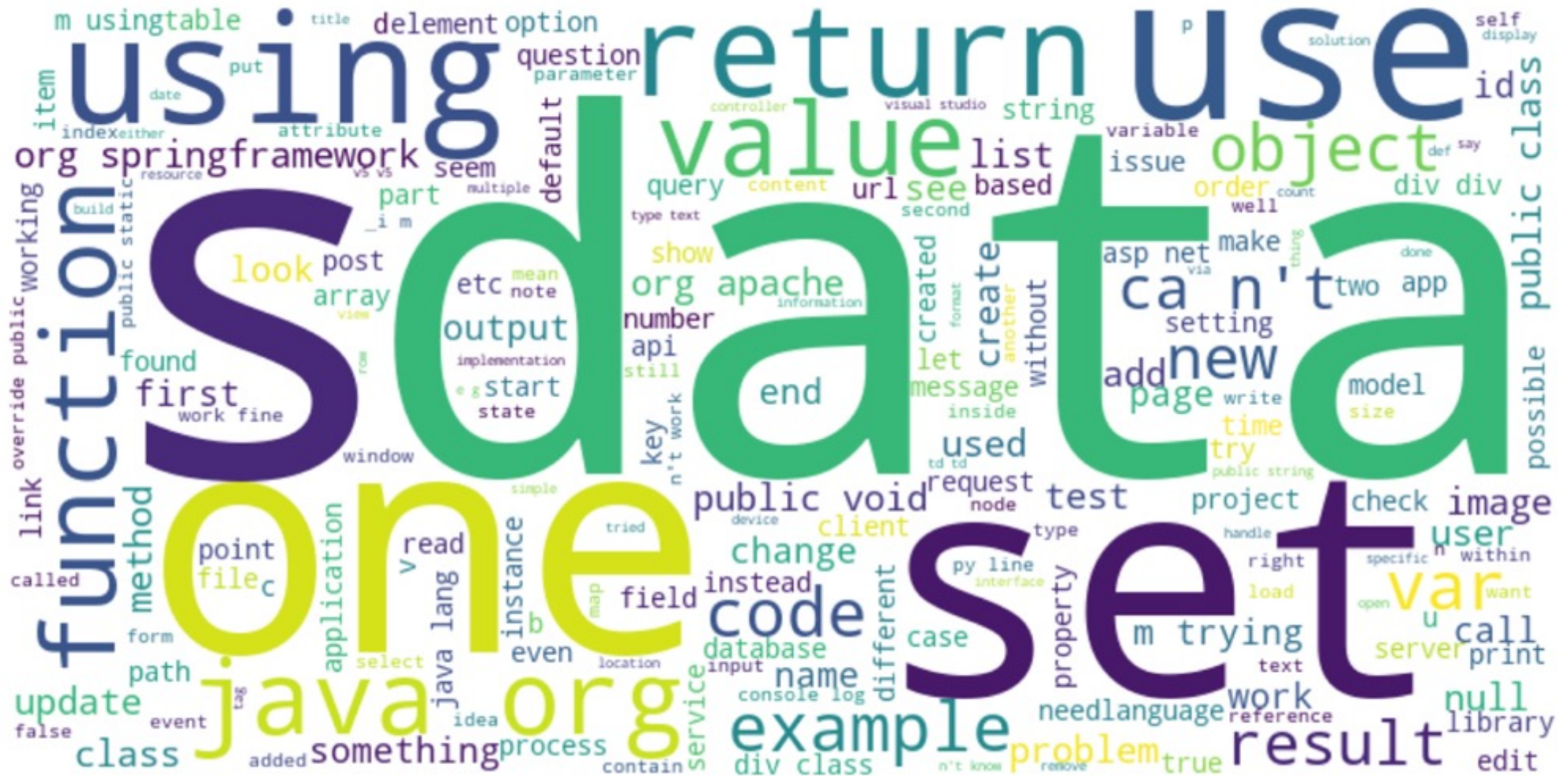
- ❑ Features encoding :
 - Utilisation du MultilabelBinarizer pour encoder les features

- ❑ Création d'une nouvelle variable avec une marque propre : « _ »
 - Faire une distinction claire entre les variables d'origines et le feature engineering.

EDA



EDA : nuage de mots (Lemmatisation: _TITLE+BODY)



**Présentation de l'approche non
supervisée et du feature extraction**

ÉTAPE N° 2



Features extraction

Nous appliquons 5 types de features extraction au sein de notre projet :

1. Méthodes basées sur la fréquence des mots :
 - Bag of Words (BoW)
 - Term Frequency-Inverse Document Frequency (TF-IDF)
2. Méthodes basées sur les embeddings :
 - Word Embedding (comme Word2Vec, GloVe)
 - Universal Sentence Encoder (USE)
 - BERT (Bidirectional Encoder Representations from Transformers)

Features extraction

1. Bag of Words (BoW) :

1. Une représentation simple
2. Qui compte la fréquence des mots dans un document
3. Mais ne prend pas en compte l'ordre des mots
4. => Facile à comprendre et à mettre en œuvre mais à ses défauts

2. Term Frequency - Inverse Document Frequency (TF-IDF) :

1. Une représentation qui prend en compte l'importance relative des mots dans un document
2. TF : La fréquence du mot dans le document et l'ensemble du corpus
3. IDF : Attribution d'un poids plus faible aux mots fréquents et un poids plus élevé aux mots rares
4. Permet de mieux discriminer les documents en fonction de leur contenu ; par exemple, dans un corpus qui lié à la guerre, les documents liés à la résistance (par l'utilisation de termes uniques).

Features extraction : Word2Vec

- ❑ Représentation vectorielle des mots
 - Permet de capturer les relations sémantiques et syntaxiques entre les mots

- ❑ Plongement de mots (Word Embedding)
 - Transformation des mots en vecteurs numériques
 - Facilite la manipulation et l'analyse des données textuelles

- ❑ Modèle utilisé
 - glove_vectors_word2vec.txt
 - Basé sur l'algorithme GloVe (Global Vectors for Word Representation))
 - Entraîné sur un large corpus de textes : capture les cooccurrences de mots à l'échelle globale
 - Permet de gagner du temps et des ressources & facilite l'analyse de texte

Features extraction : BERT

❑ Compréhension du contexte

- Tient compte de l'ordre des mots et du contexte grâce à l'apprentissage bidirectionnel et contextuel
- Utilise des mécanismes d'attention et d'auto-attention pour une meilleure représentation du contexte
- Plongement de mots mais contrairement à Word2Vec qui se concentre sur les relations sémantiques et syntaxiques, ici, nous exploitons les réseaux de neurones pour comprendre les phrases dans leur contexte

❑ Modèle utilisé

- « all-MiniLM-L6-v2 »
- Entraîné sur un large corpus de textes (Voir sources)
- Peu gourmand en ressources

Features extraction : USE

- ❑ Compréhension du contexte
 - Analyse la signification globale de la phrase plutôt que des mots individuels
 - Encodage sur des phrases entières pour mieux représenter le contexte
 - Utilise aussi des mécanismes d'attention pour identifier les relations entre les mots
- ❑ Indépendance par rapport à la longueur de la phrase
 - Gère efficacement des phrases de différentes longueurs
- ❑ Modèle utilisé
 - "Universal Sentence Encoder" (USE)
 - Disponible sur TensorFlow Hub : <https://tfhub.dev/google/universal-sentence-encoder/4>
 - Offre des performances élevées par rapport aux ressources demandées.

Présentation de l'approche non supervisée

❑ LDA (Latent Dirichlet Allocation)

- Méthode non supervisée d'extraction de thèmes :
- Détermine les thèmes, les distributions de mots sur les thèmes, et la fréquence d'apparition de chaque thème dans le corpus.

❑ Inférence

- Utilisation de la librairie LDA de Gensim pour générer des mots-clés

❑ Visualisation

- PyLDAVis pour la visualisation des résultats LDA
- Intègre Plotly pour des graphiques interactifs (Voir diapo suivant).

❑ Évaluation du modèle LDA

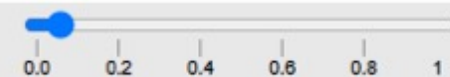
- Paramètre λ pour l'évaluation des performances

Out[20]:

Selected Topic:

Slide to adjust relevance metric:(2)

$\lambda = 0.06$



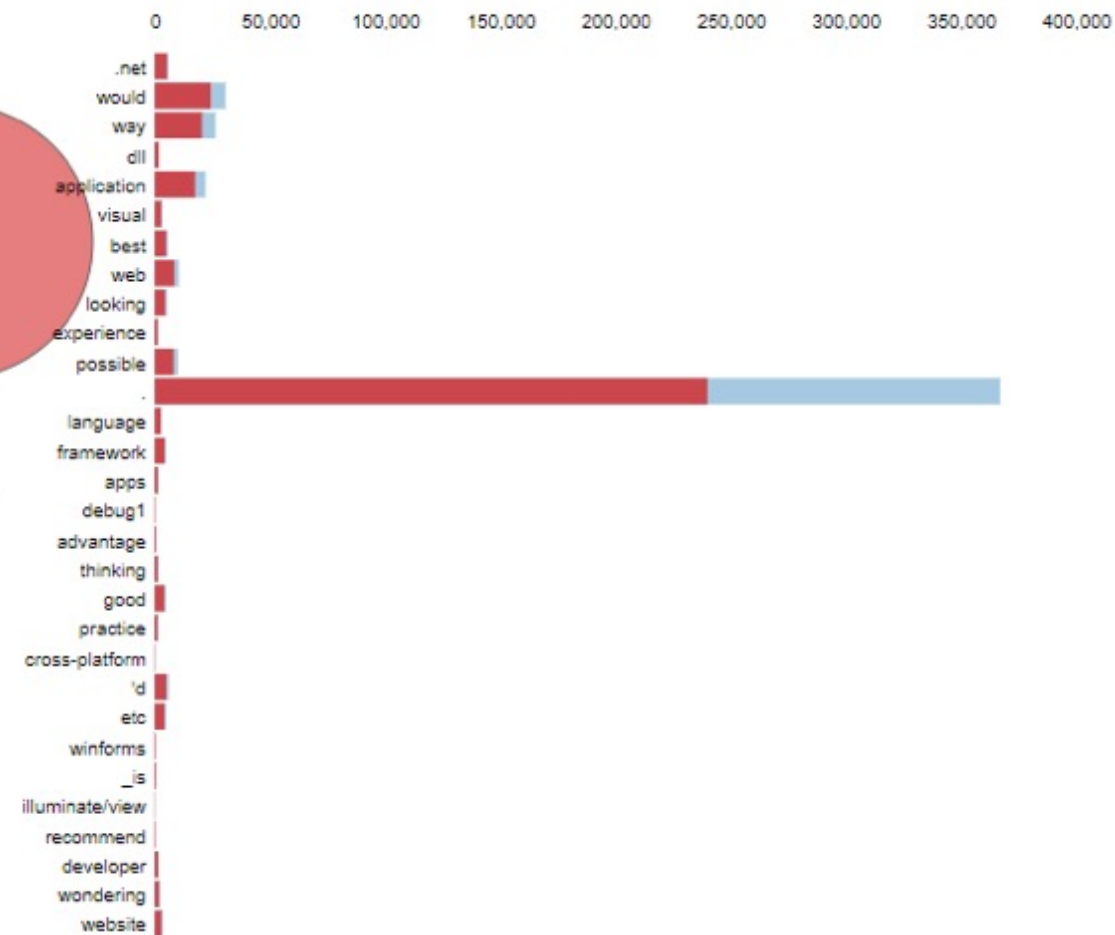
Intertopic Distance Map (via multidimensional scaling)



Marginal topic distribution



Top-30 Most Relevant Terms for Topic 1 (27.5% of tokens)



Overall term frequency

Estimated term frequency within the selected topic

1. $\text{saliency}(\text{term } w) = \text{frequency}(w) * (\sum_t p(t|w) * \log(p(t|w)/p(t)))$ for topics t ; see [Chuang et al \(2012\)](#)

2. $\text{relevance}(\text{term } w | \text{topic } t) = \lambda * p(w|t) + (1 - \lambda) * p(w|t)/p(w)$; see [Sievart & Shirley \(2014\)](#)

Présentation de l'approche supervisée et
du modèle final sélectionné

ÉTAPE N° 3



Process

1. Séparation du jeu de données
 - Création d'un ensemble d'entraînement (train set) et d'un ensemble de test (test set)
2. Ajustement de la validation croisée
 - Choix du nombre de plis (folds = 3) pour la cross-validation
3. Sélection des paramètres et validation croisée
 - Utilisation de GridSearchCV et RandomSearchCV pour l'optimisation des hyperparamètres
4. Ajustement des hyperparamètres
5. Entraînement et évaluation des modèles
 - a) Modèle de base (Baseline Extra Trees)
 - b) Modèles plus complexes (Random Forest, régression logistique)

Process

5. Métriques de performance pour la classification

❑ Distance de Jaccard

Mesure la similarité entre deux groupes d'éléments en comparant le nombre d'éléments qu'ils ont en commun par rapport au nombre total d'éléments dans les deux groupes.

❑ Analyse de la précision et du rappel pour évaluer la qualité des prédictions.

Le rappel mesure la proportion de vrais positifs qui sont correctement identifiées par le modèle

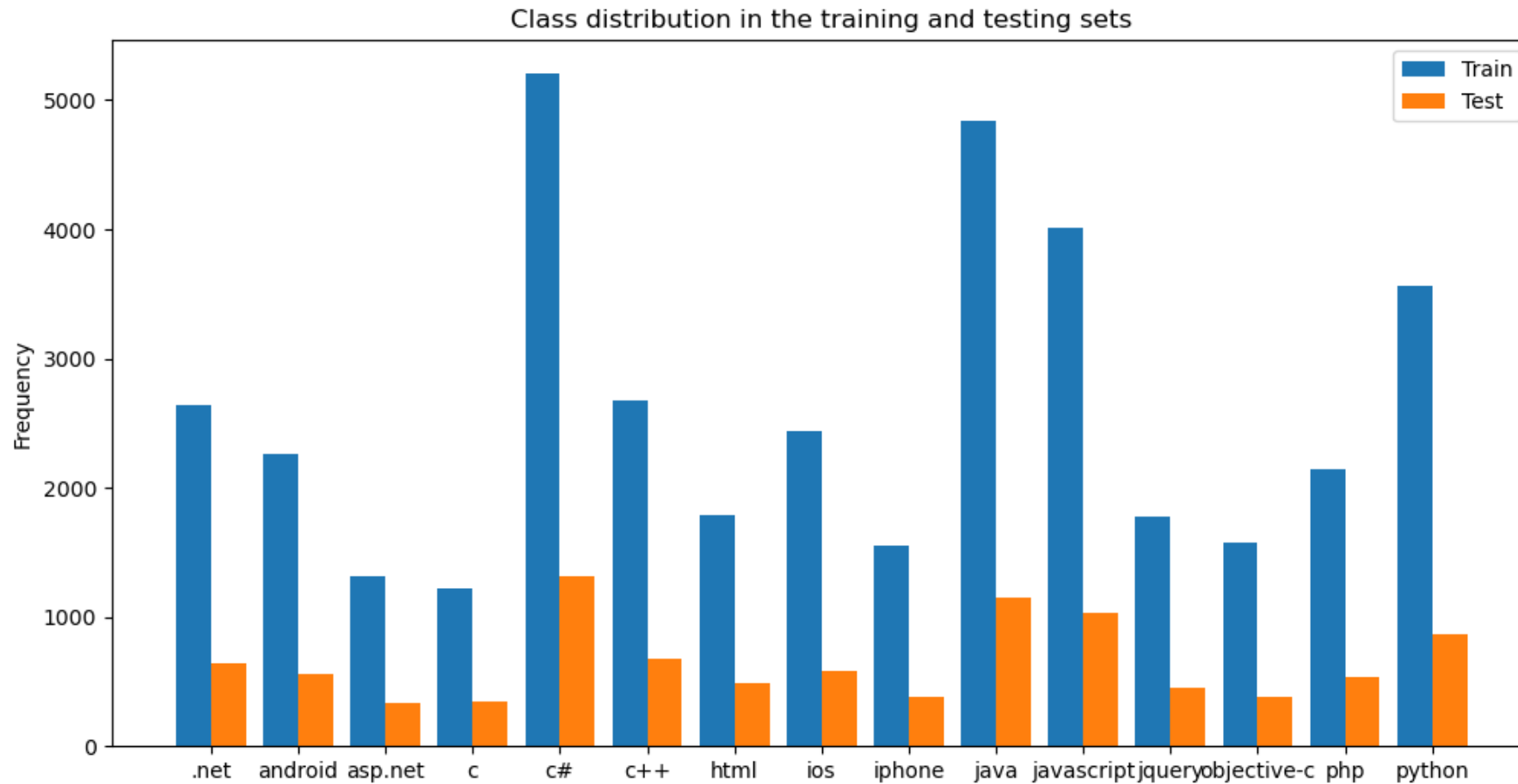
La précision mesure la proportion de vrais positifs qui sont réellement positifs

❑ F1 score : moyenne harmonique de la précision et du rappel

7. Sélection du modèle final

Process

- Représentation des tags au sein de Xtrain et Xtest
- Pas de déséquilibre de classes



Machine learning : Extra Trees MODELE BASELINE

► Meilleurs hyperparamètres : {
 'n_estimators': [500],
 'max_depth': [None, 30],
 'min_samples_split': [2],
 'min_samples_leaf': [1]
}

► Temps d'exécution : 245s

► Features extraction : TF-IDF

Classifier report:

	precision	recall	f1-score	support
.net	0.88	0.01	0.02	637
android	0.99	0.30	0.46	558
asp.net	0.81	0.06	0.12	331
c	1.00	0.01	0.02	348
c#	0.87	0.05	0.09	1317
c++	0.93	0.19	0.31	678
html	0.71	0.09	0.16	484
ios	0.84	0.08	0.15	586
iphone	1.00	0.01	0.01	384
java	0.93	0.24	0.38	1154
javascript	0.83	0.16	0.27	1035
jquery	0.83	0.07	0.14	456
objective-c	0.33	0.00	0.01	380
php	1.00	0.02	0.05	540
python	0.95	0.25	0.40	866

Jaccard score: 0.1376

Machine learning : Random Forest

► Meilleurs hyperparamètres : {
'n_estimators': [100],
'max_depth': [None],
'min_samples_split': [5],
'min_samples_leaf': [2],
'bootstrap': [False]
}

► Temps d'exécution : 1102s

► Features extraction : TF-IDF

Classifier report:

	precision	recall	f1-score	support
.net	0.83	0.04	0.07	637
android	0.98	0.44	0.61	558
asp.net	0.77	0.10	0.18	331
c	0.76	0.05	0.09	348
c#	0.77	0.17	0.28	1317
c++	0.85	0.28	0.42	678
html	0.67	0.14	0.23	484
ios	0.79	0.18	0.30	586
iphone	0.75	0.02	0.05	384
java	0.93	0.38	0.54	1154
javascript	0.79	0.29	0.42	1035
jquery	0.83	0.15	0.25	456
objective-c	0.67	0.03	0.06	380
php	0.97	0.06	0.11	540
python	0.96	0.37	0.54	866

Jaccard score: 0.2336

Machine learning : Régression logistique

► Meilleurs hyperparamètres : {

```
'estimator__C': 0.001,  
'estimator__max_iter': 100,  
'estimator__penalty': 'none',  
'estimator__solver': 'lbfgs'  
}
```

► Temps d'exécution : 245s

► Features extraction : TF-IDF

Classifier report:

	precision	recall	f1-score	support
.net	0.65	0.22	0.33	637
android	0.95	0.74	0.83	558
asp.net	0.72	0.41	0.53	331
c	0.69	0.32	0.44	348
c#	0.71	0.44	0.54	1317
c++	0.76	0.47	0.58	678
html	0.66	0.35	0.46	484
ios	0.80	0.51	0.63	586
iphone	0.58	0.21	0.31	384
java	0.86	0.62	0.72	1154
javascript	0.81	0.60	0.69	1035
jquery	0.82	0.60	0.69	456
objective-c	0.61	0.23	0.34	380
php	0.82	0.49	0.61	540
python	0.93	0.74	0.83	866

Jaccard score: 0.4895

Machine learning : Régression logistique

Classifier report:

	precision	recall	f1-score	support
.net	0.54	0.38	0.44	637
android	0.93	0.87	0.90	558
asp.net	0.71	0.51	0.60	331
c	0.72	0.53	0.61	348
c#	0.82	0.72	0.77	1317
c++	0.82	0.71	0.76	678
html	0.69	0.51	0.58	484
ios	0.75	0.68	0.71	586
iphone	0.66	0.52	0.58	384
java	0.87	0.81	0.84	1154
javascript	0.81	0.80	0.80	1035
jquery	0.80	0.74	0.77	456
objective-c	0.66	0.55	0.60	380
php	0.90	0.83	0.86	540
python	0.94	0.91	0.93	866

► Meilleurs hyperparamètres : {

'estimator__C': 0.001,
'estimator__max_iter': 100,
'estimator__penalty': 'none',
'estimator__solver': 'lbfgs'
}

► Temps d'exécution : ~s

► Features extraction : SBERT

Jaccard score: 0.6802

Machine learning : Régression logistique

► Meilleurs hyperparamètres : {

```
'estimator__C': 0.001,  
'estimator__max_iter': 100,  
'estimator__penalty': 'none',  
'estimator__solver': 'lbfgs'  
}
```

► Temps d'exécution : ~s

► Features extraction : USE

Classifier report:

	precision	recall	f1-score	support
.net	0.57	0.36	0.44	637
android	0.89	0.82	0.85	558
asp.net	0.69	0.47	0.56	331
c	0.69	0.54	0.61	348
c#	0.78	0.72	0.75	1317
c++	0.78	0.63	0.70	678
html	0.66	0.48	0.55	484
ios	0.70	0.67	0.68	586
iphone	0.59	0.51	0.55	384
java	0.84	0.81	0.82	1154
javascript	0.79	0.75	0.77	1035
jquery	0.76	0.65	0.70	456
objective-c	0.60	0.52	0.56	380
php	0.84	0.73	0.78	540
python	0.91	0.86	0.88	866

Jaccard score: 0.6449

Modèle final sélectionné

- ▶ La meilleure features extraction est celle de BERT.
- ▶ Le meilleur modèle de machine learning est celui de la régression logistique.
- ▶ Le meilleur Jaccard Score est de 0,68
- ▶ Avec les meilleurs hyperparamètres suivant : {'estimator__C': 0.001, 'estimator__max_iter': 100, 'estimator__penalty': 'none', 'estimator__solver': 'lbfgs'}
- ▶ Temps d'exécution : <1minute

Démonstration de l'API et conclusion






ÉTAPE N° 4







Démonstration de l'API : présentation

- ❑ Déploiement en local de l'API avec Gradio (avec accès à une API temporaire de 3 jours sur Gradio si besoin est) pour tester et démontrer l'interaction avec le modèle.
- ❑ Utilisation de Gradio pour créer une interface utilisateur gratuite, simple et intuitive.
- ❑ Hébergement du modèle sur Hugging Face, création d'un « espace » pour faciliter la démonstration du modèle final sélectionner, et ensuite le partage et la réutilisation.
- ❑ Lien de l'API disponible :
https://huggingface.co/spaces/GuillaumeLemele/tags_stackoverflow
- ❑ Gestionnaire de versions :
 - GitHub : https://github.com/GuillaumeLemele/P5_OPC_IML
 - Hugging Face :
https://huggingface.co/spaces/GuillaumeLemele/tags_stackoverflow/tree/main

Démonstration de l'API : exemple de prédiction

Spaces:  GuillaumeLemele / **tags_stackoverflow**   like 0  Running  Logs

App  Files  Community  Settings 

text

"This is a new input code example for using py in panda with jupyter notebook please how i do it."

Method

stem

Nettoyer

Soumettre

Processed Text

` ` new input code exampl use py panda jupyt notebook pleas . "

Predicted Tags

python

ALLER
PLUS
LOIN

- ❑
- ❑ Point n° 1
- ❑ Améliorer le nettoyage de la donnée pour améliorer les résultats
- ❑ Point n° 2
- ❑ Tester d'autres algorithmes de machine learning tels que XGBOOST



DISCUSSION

MERCI POUR VOTRE ÉCOUTE !

Guillaume Lemêlé

Parcours : Machine Learning Engineer

10/2022 - 06/2023