Catégorisez automatiquement des questions

Catégorisez automatiquement des questions

Compétences évaluées:

- Représenter graphiquement des données à grandes dimensions

- Prétraiter des données non structurées pour obtenir un jeu de données exploitable

- Mettre en œuvre des techniques d’extraction de features pour des données non structurées

- Mettre en œuvre des techniques de réduction de dimension

Problématique - Interprétation

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

Pour les développeurs, les recherches se font géralement ainsi :

- ce que je veux faire
- type d'object
- **nom de la librairie**
- nom du language

Cependant pour les novices, il serait intéressant de suggérer **des tags** relatifs à la question posée.

> système de suggestion de plusieurs tags

Problématique - Pistes de recherche

- **une approche non supervisée : LDA**

- **une approche combinée : OnevsRest SVC et LDA**
    - OneVsRestClassifier(LogisticRegression)
    - OneVsRestClassifier(SVC)
    - OneVsRestClassifier(MultinomialNB)

- **une approches supervisée : OnevsRest SVC**
    - OneVsRestClassifier(LogisticRegression)
    - OneVsRestClassifier(SVC)
    - OneVsRestClassifier(MultinomialNB)

- **une approche supervisée en multi-output** : **KNN**
    - MultiOutputClassifier(GaussianNB)
    - MultiOutputClassifier(RandomForest)
    - MultiOutputClassifier(KNeighborsClassifier)
    - MultiOutputClassifier(MultinomialNB)

Cleaning

Récupération des données sur 5 csv

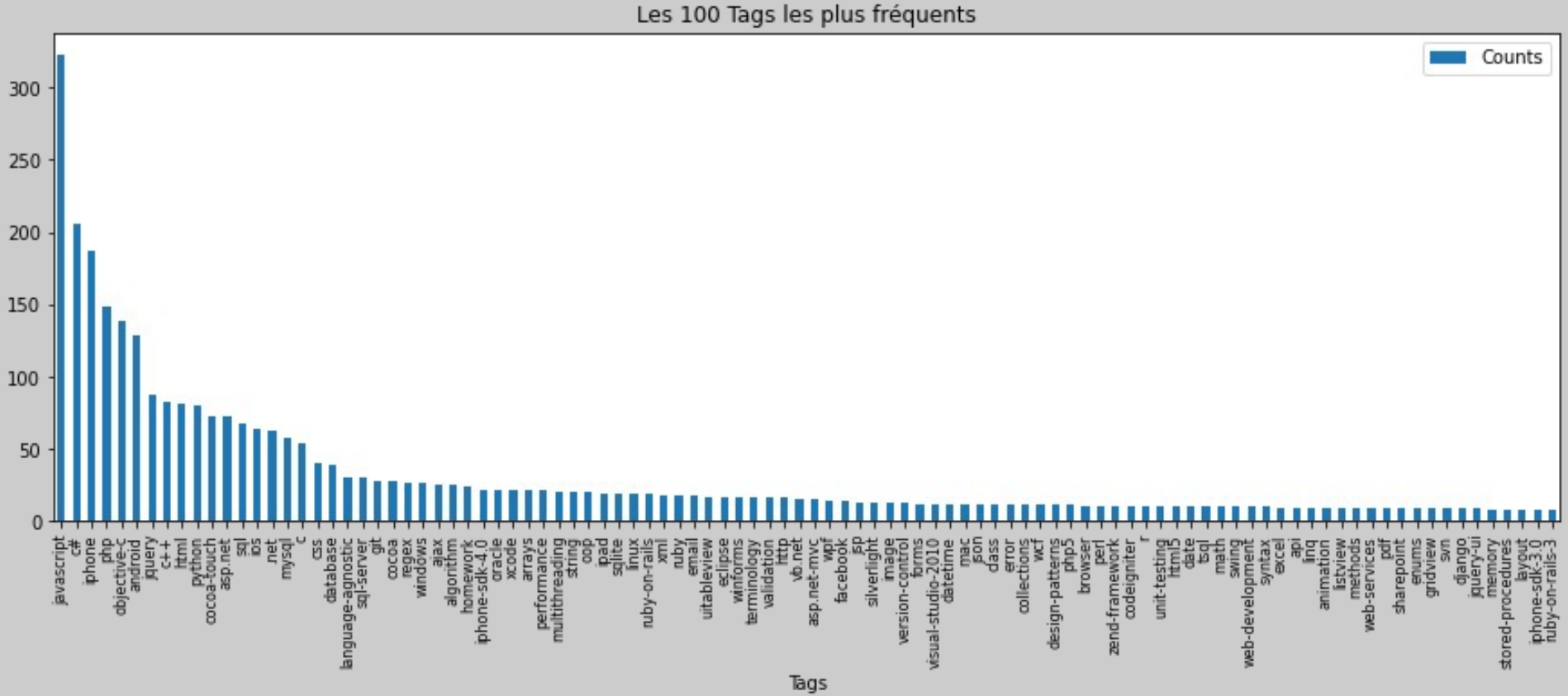
- Suppression des NaN sur les colonnes `Titles` et `Tags`
- Filtration des lignes sur les colonnes > `(2015, 2)`

```
AnswerCount > 10
FavoriteCount > 200
ViewCount > 5000
Score > 15
```

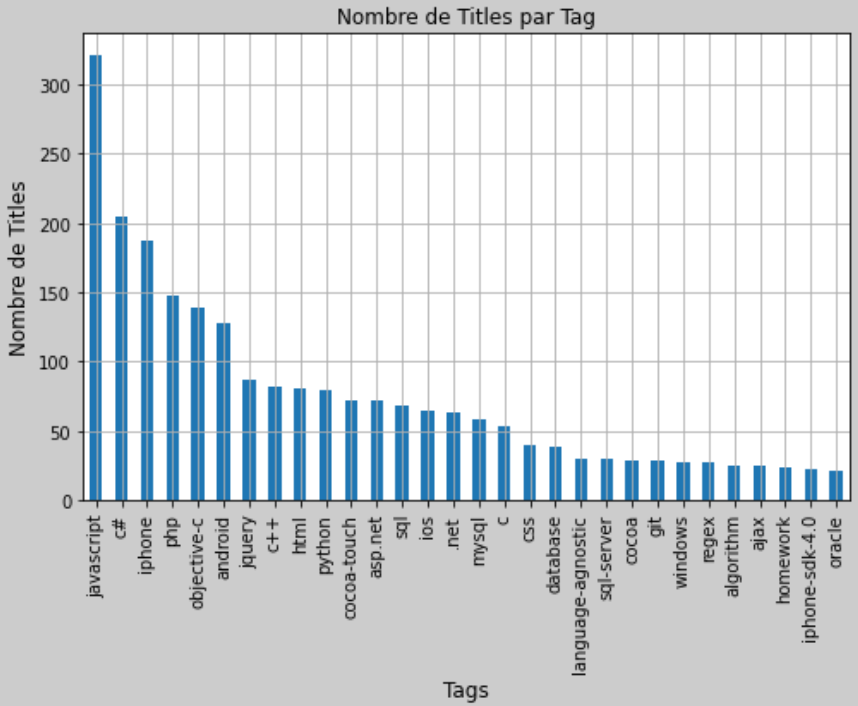
Nettoyage des Tags

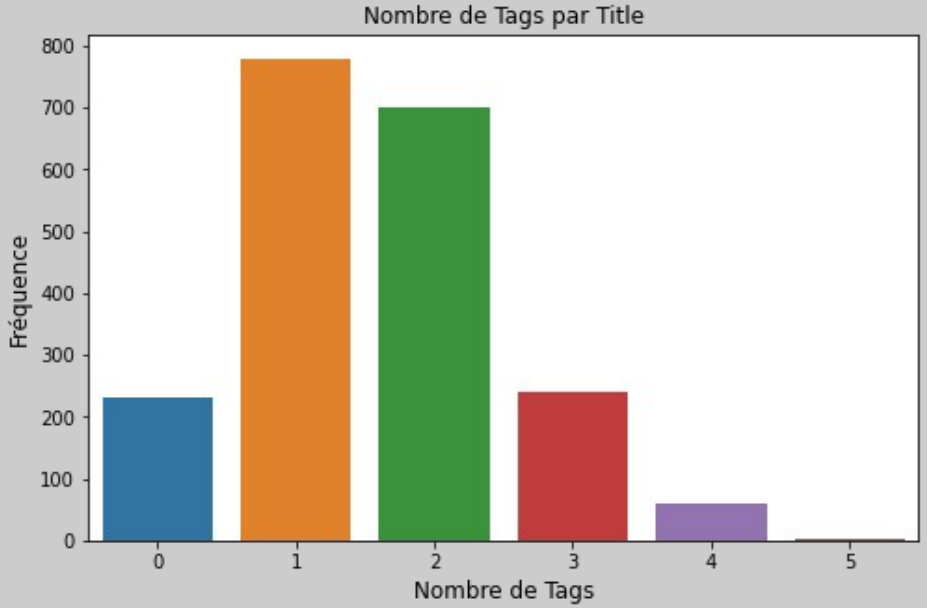
- Suppression des balises avec regex
- Typage: string to list of Tags
- Selection des Tags parmi les 100 tags les plus fréquents
- Transformation : list of Tags to Dummies

Cleaning


Les 100 Tags les plus fréquents
Counts
300
250
200
150
100
50
0
javascript
c#
iphone
php
objective-c
android
jquery
c++
html
python
cocoa-touch
asp.net
sql
ios
net
mysql
c
css
database
language-agnostic
sql-server
git
cocoa
regex
windows
ajax
algorithm
homework
iphone-sdk-4.0
oracle
xcode
arrays
performance
multithreading
string
oop
ipad
sqlite
linux
ruby-on-rails
xml
ruby
email
uitableview
eclipse
winforms
terminology
validation
http
vb.net
asp.net-mvc
wpf
facebook
jsp
silverlight
image
version-control
forms
visual-studio-2010
datetime
mac
json
class
error
collections
wcf
design-patterns
php5
browser
perl
zend-framework
codeigniter
r
unit-testing
html5
date
tsql
math
swing
web-development
syntax
excel
api
linq
animation
listview
methods
web-services
pdf
sharepoint
enums
gridview
svn
django
jquery-ui
memory
stored-procedures
layout
iphone-sdk-3.0
ruby-on-rails-3
Tags

Exploration





Cleaning

Nettoyages des Titles :

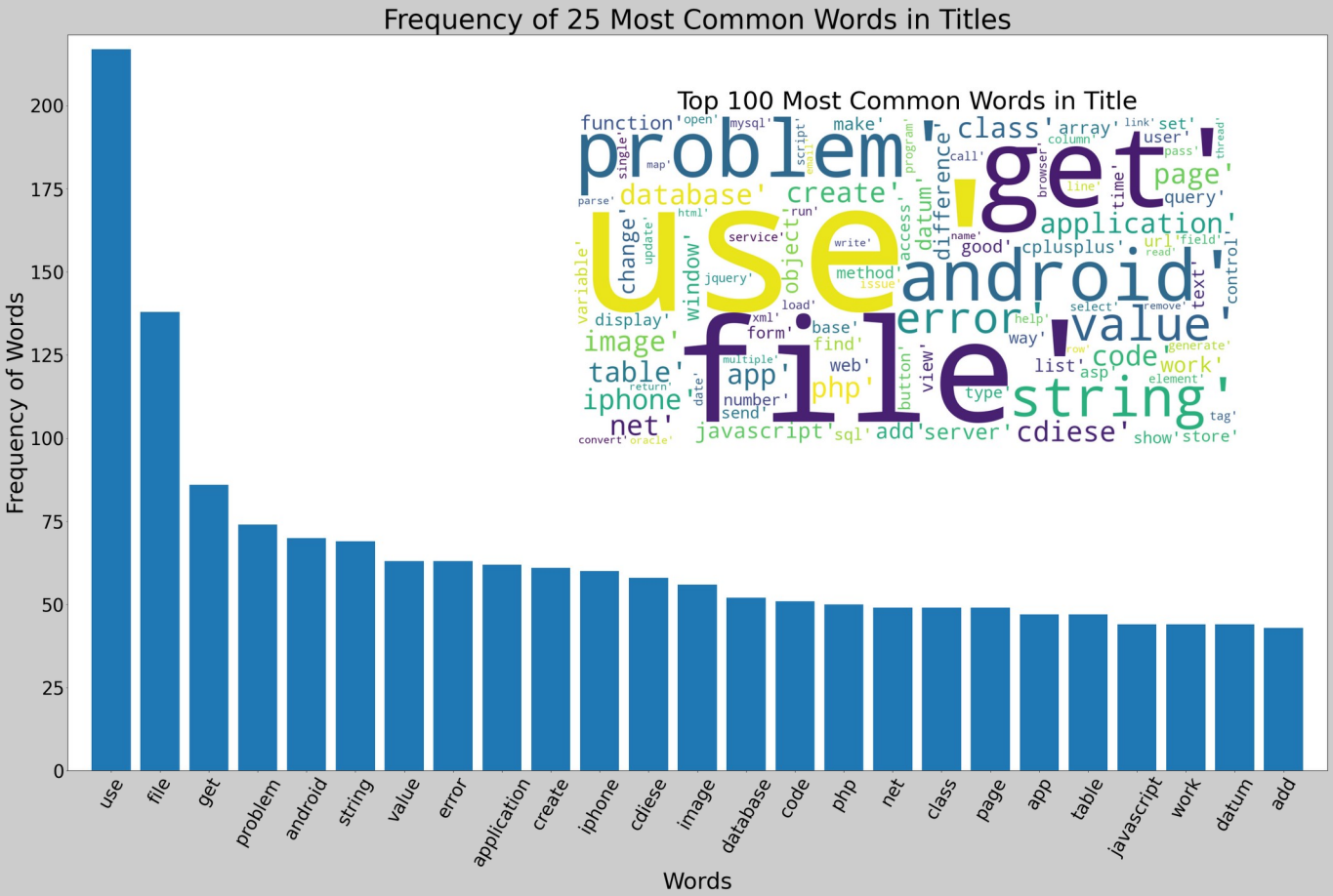
Tokenisation : list of sentences to list of words
- suppression des caractères / \ < > - = ..
- sauf c++ > cplusplus
- sauf c# > cdiese

Stopwords

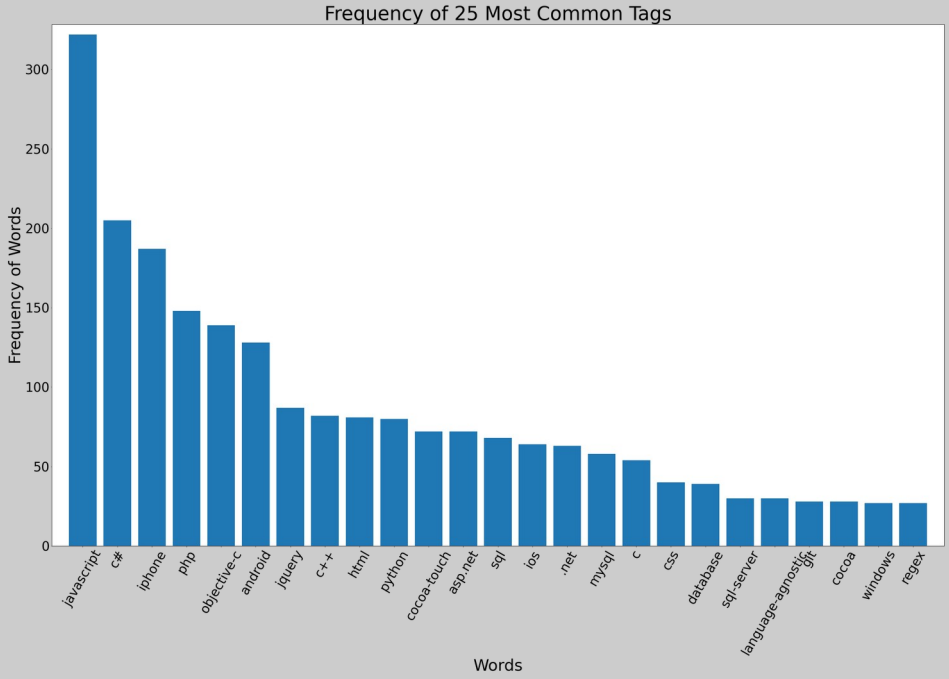
Lemmatization : noms en masculin singulier, verbes à l'infinitif > sens

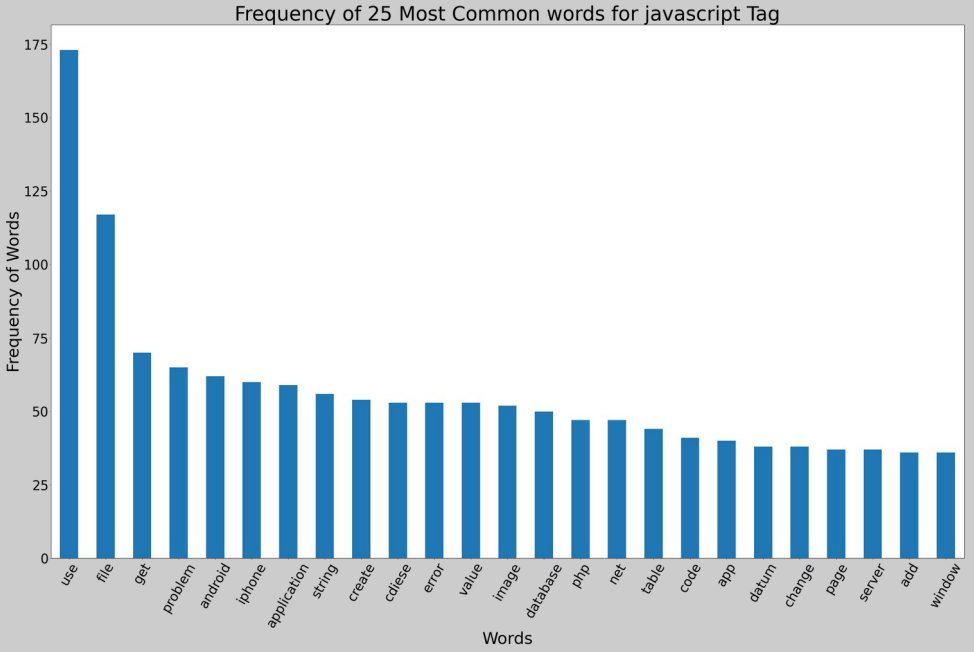
```
DF (2015, 101)
```

Exploration



Exploration





Feature engineering

Méthodes d'extraction de features

- CountVectorizer : `max_features`
- TfidfVectorizer : `max_features`
- CountVectorizer TfidfTransformer : `max_features`
- CountVectorizer TfidfTransformer : `ngram_range max_features`

- Reduction de dimension

  - CountVectorizer TruncatedSVD : `max_features  n_components`
  - CountVectorizer TfidfTransformer TruncatedSVD : `max_features n_components`
  - word2vec : `vector_size`

- **une approche non supervisée : LDA**

- **une approche combinée : OnevsRest SVC et LDA**
    - OneVsRestClassifier(LogisticRegression)
    - OneVsRestClassifier(SVC)
    - OneVsRestClassifier(MultinomialNB)

- **une approches supervisée : OnevsRest SVC**
    - OneVsRestClassifier(LogisticRegression)
    - OneVsRestClassifier(SVC)
    - OneVsRestClassifier(MultinomialNB)

- **une approche supervisée en multi-output** : KNN
    - MultiOutputClassifier(GaussianNB)
    - MultiOutputClassifier(RandomForest)
    - MultiOutputClassifier(KNeighborsClassifier)
    - MultiOutputClassifier(MultinomialNB)

Modélisation

Méthologie pour les 4 approches:

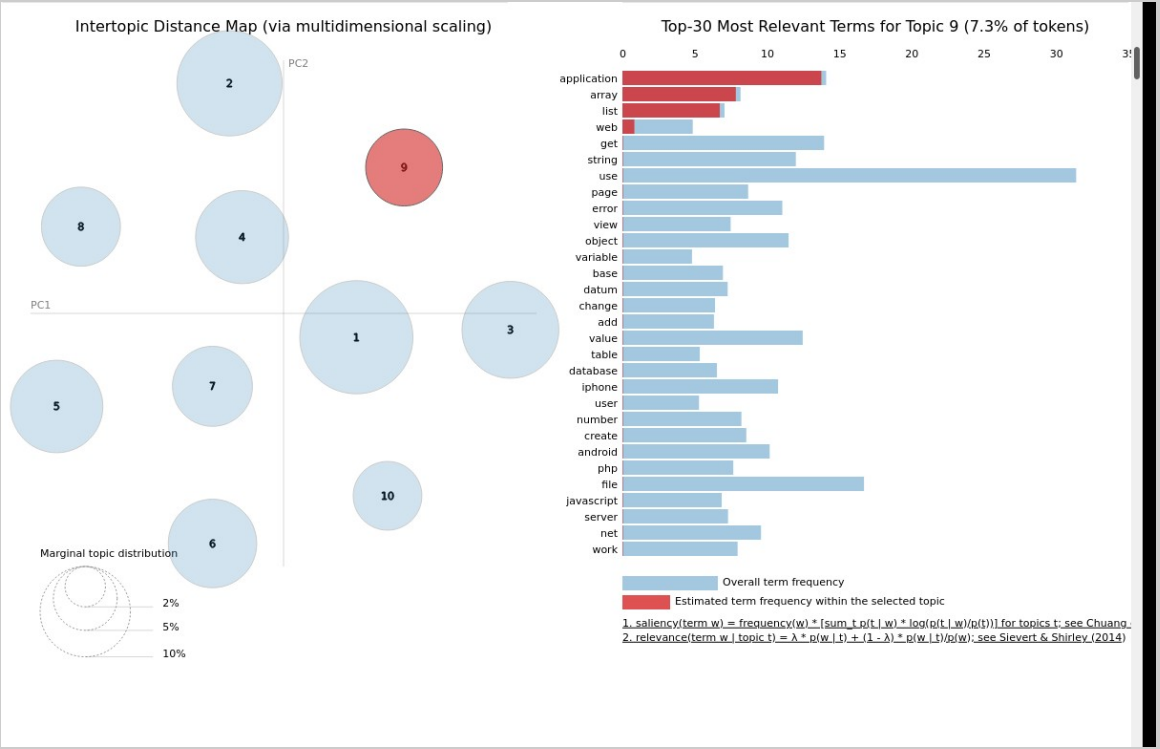
- Split test & train data

- GridSerachCV sur les vectorizers :
  - `max_features`
  - `ngram_range`
  - `use_idf`
- Récupération des scores pour chaque modèle selon leurs types de vectorizers.
- Choix du **meilleur modèle avec son vectorizer**

- GridSeachCV sur le modèle
- Comparaison des performances avant et après le GridSearchCV
- Création ou prédiction des topics

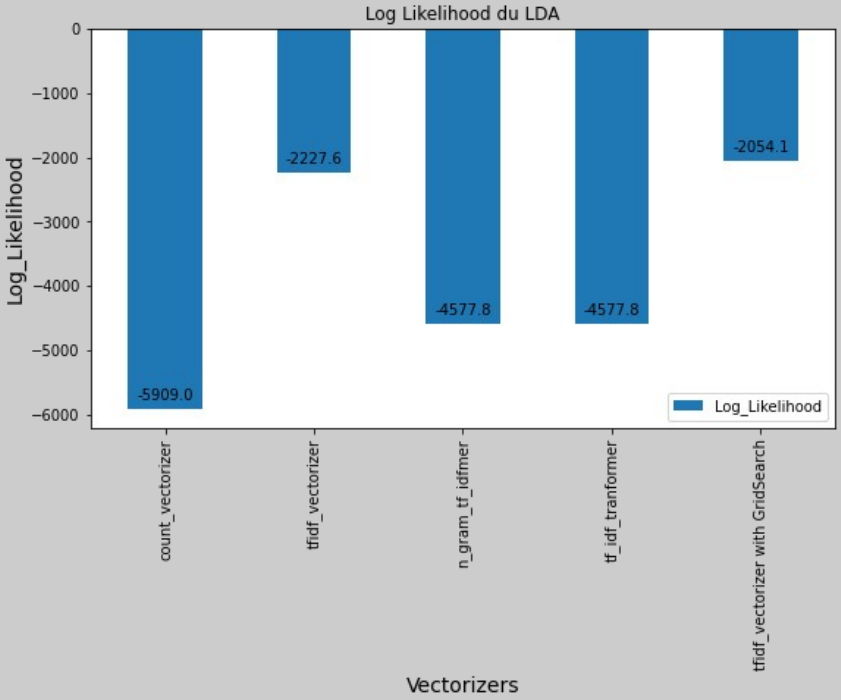
LDA - Topics

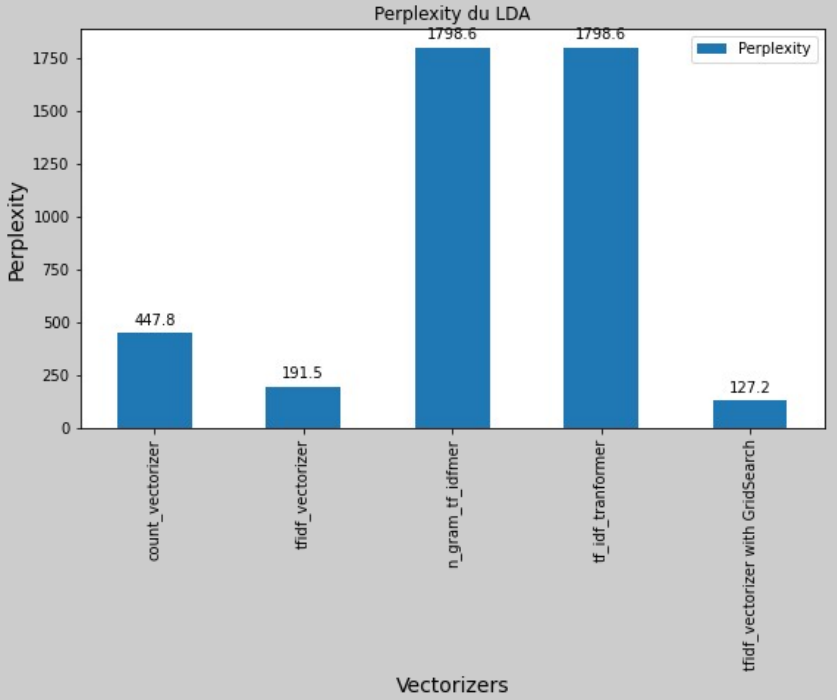
LDA: 16 tags



Fréquence de chaque mot dans le topic et dans le corpus global.

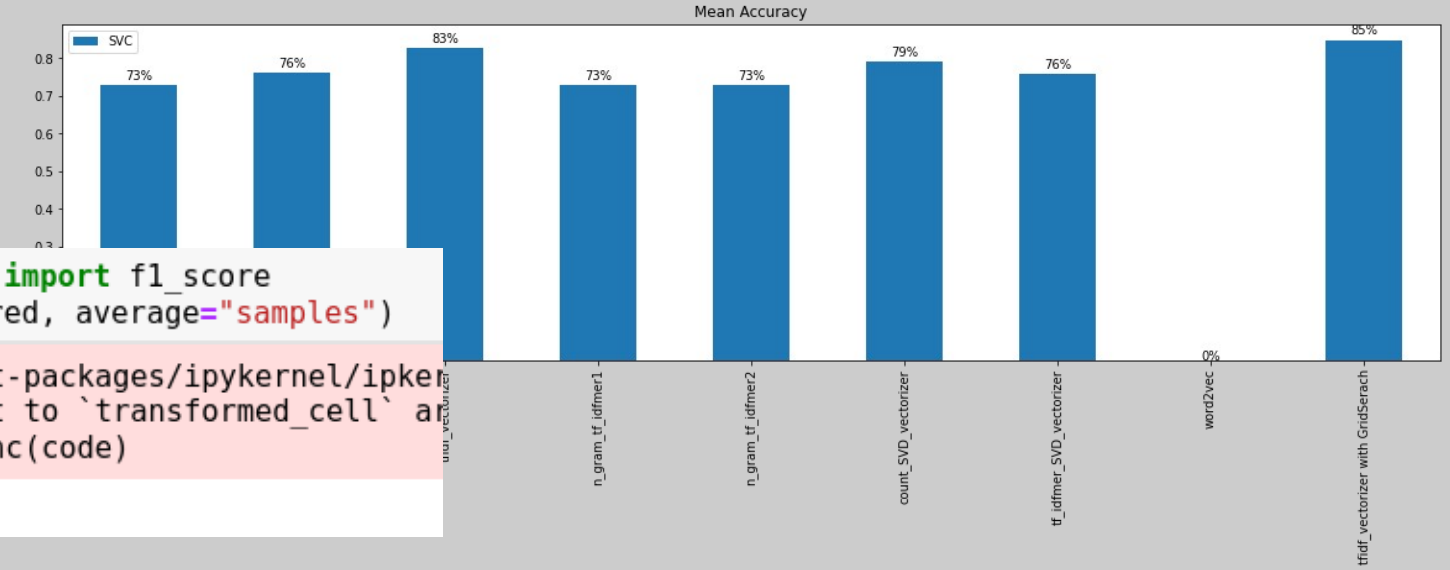
Performances - LDA





LDA avec tfidf_vectorizer

Performances – OneVsRest SVC sur la LDA




```
from sklearn.metrics import f1_score
f1_score(y_test, y_pred, average="samples")
```

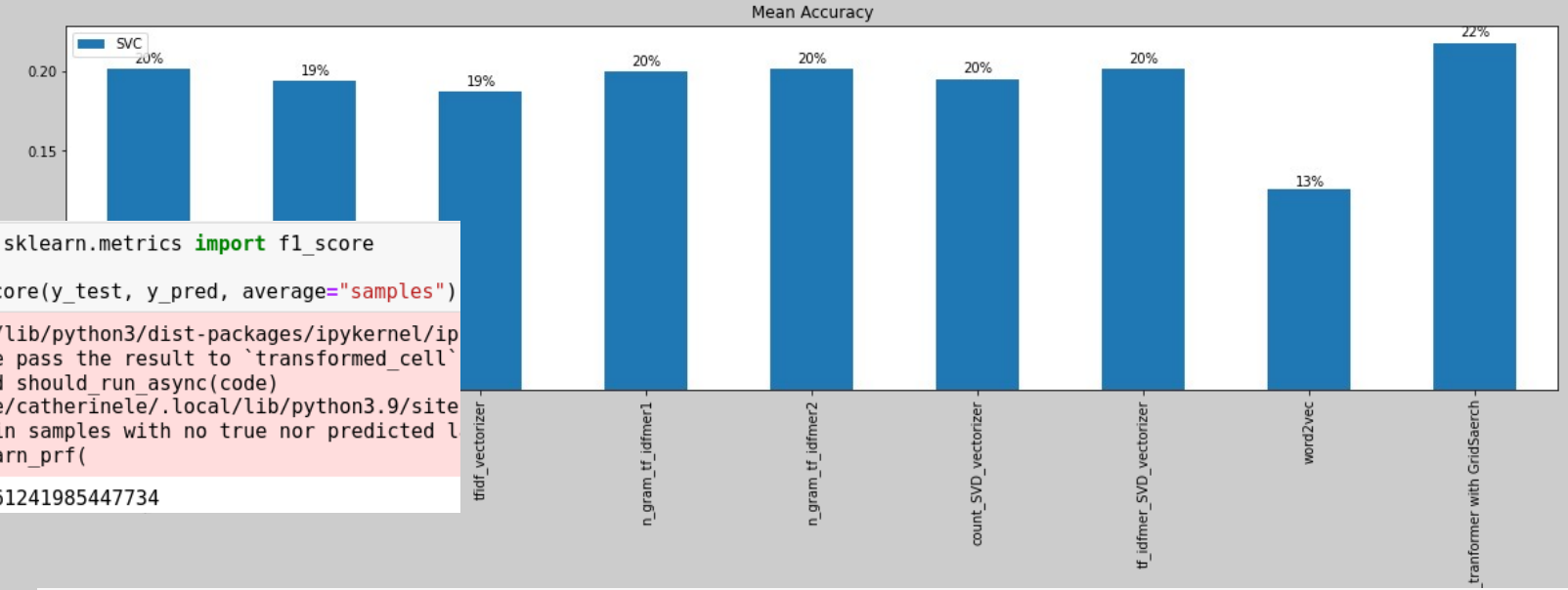
```
/usr/lib/python3/dist-packages/ipykernel/ipker
lease pass the result to `transformed_cell` ar
  and should_run_async(code)
```

```
0.8981060375160225
```

```
def j_score(y_true,y_pred):

    jaccard = np.minimum(y_true,y_pred).sum(axis=1)/np.maximum(y_true,y_pred).sum(axis=1)

    return  print(jaccard.mean()*100)

j_score(y_test, df_tags)
```

```
88.40621472996659
```

Performances – OneVsRest SVC




```
from sklearn.metrics import f1_score

f1_score(y_test, y_pred, average="samples")
```

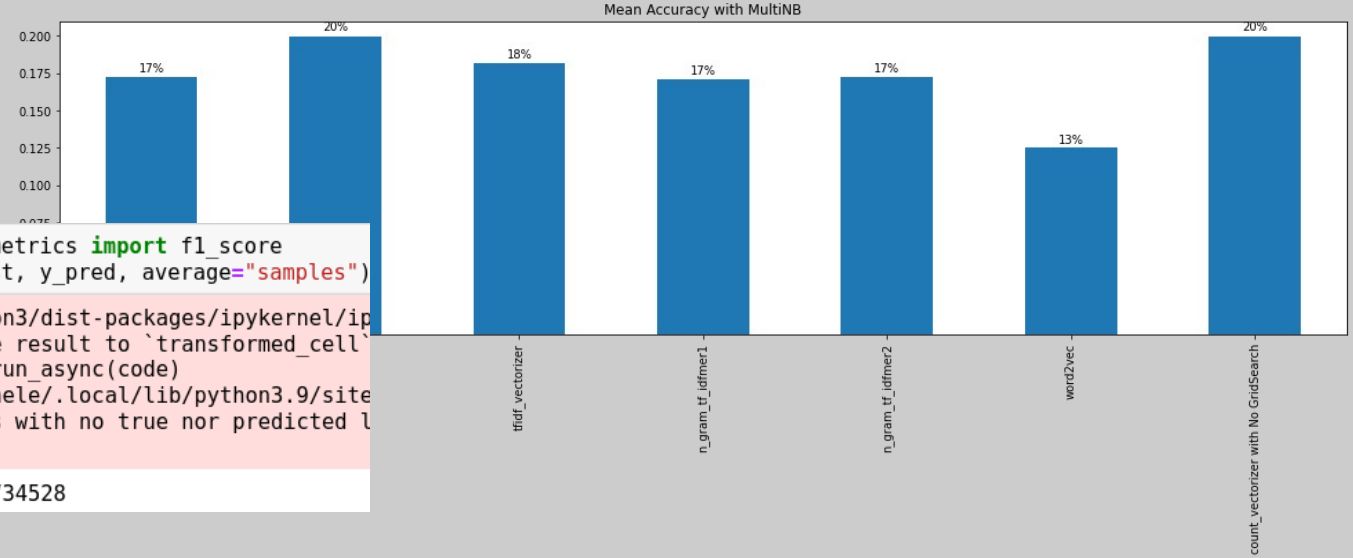
```
/usr/lib/python3/dist-packages/ipykernel/ip
lease pass the result to `transformed_cell`
  and should_run_async(code)
/home/catherinele/.local/lib/python3.9/site
0.0 in samples with no true nor predicted l
  _warn_prf(
```

```
0.1961241985447734
```

```
def j_score(y_true,y_pred):

    jaccard = np.minimum(y_true,y_pred).sum(axis=1)/np.maximum(y_true,y_pred).sum(axis=1)

    return  print(jaccard.mean()*100)

j_score(y_test, df_tags)
```

```
19.431372549019603
```

Performances – Multi-outputs MultiNB




```
from sklearn.metrics import f1_score
f1_score(y_test, y_pred, average="samples")
```

```
/usr/lib/python3/dist-packages/ipykernel/ip
lease pass the result to `transformed_cell`
  and should_run_async(code)
/home/catherinele/.local/lib/python3.9/site
0.0 in samples with no true nor predicted l
  _warn_prf(
```

```
0.15633599884734528
```

```
def j_score(y_true,y_pred):

    jaccard = np.minimum(y_true,y_pred).sum(axis=1)/np.maximum(y_true,y_pred).sum(axis=1)

    return  print(jaccard.mean()*100)

j_score(y_test, df_tags)
```

```
15.554607508532426
```

Conclusion

Les modèles par ordre croissant dans leurs performances:

Multi-outputs MultinomialNB
OneVsRest SVC
OneVsRest SVC sur la LDA

Ainsi, il est plus intéressant d'adopter l'approche combinée OneVsRest SVC sur la LDA.

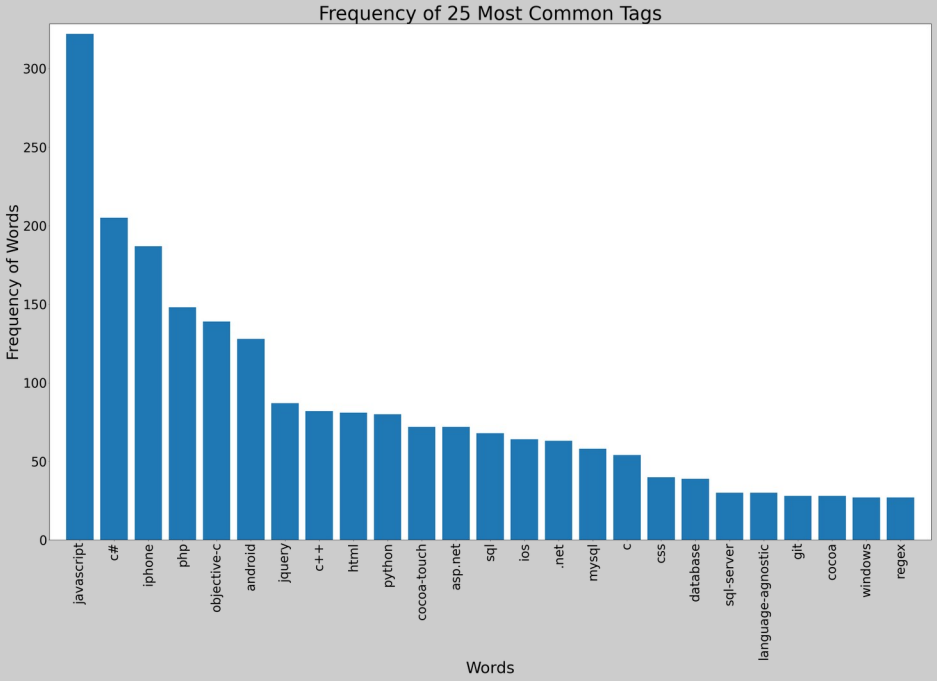
Modèle final – SVC on LDA

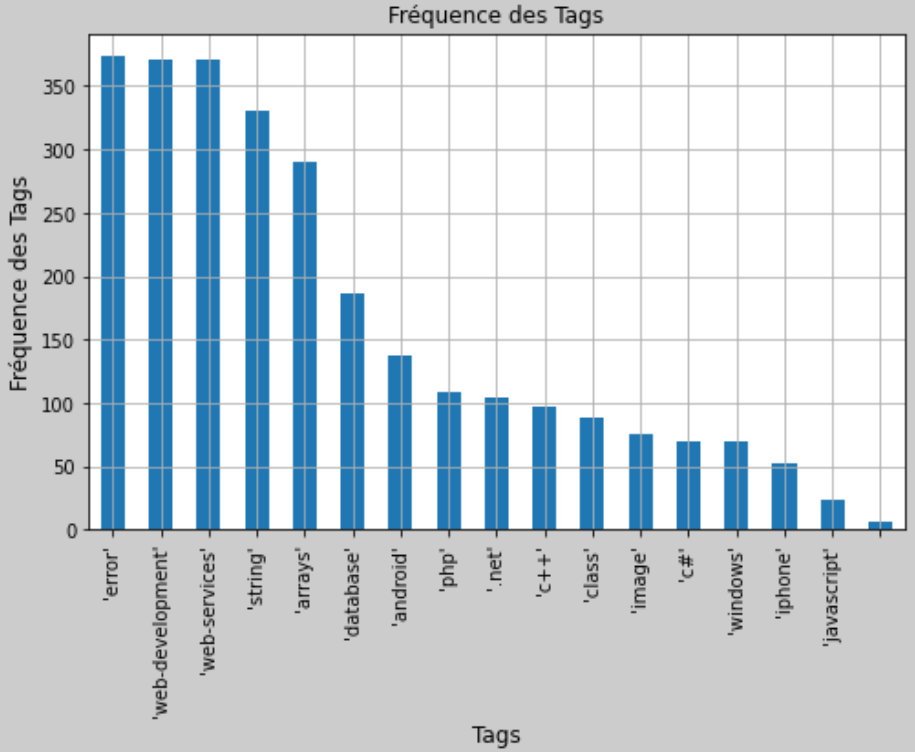
Topics de la LDA

```
["arrays", "string", "error", "web-development", "web-services"],
["javascript", "c++"] ,
["image", "string", "database", "php","c++"],
[".net", "database"],
["android", "c#", "database","web-development", "web-services"],
["iphone"],
["arrays", "string","database"],
["error", "php", "android" ,"web-development", "web-services" ],
["arrays",  "android", "class", "image","c++"],
["class", "windows" ,".net", "error"]]
```

	Title	LDA	SVC_LDA
0	break format nsstre multiple line	['arrays', 'string', 'error', 'web-development...	('arrays', 'error', 'string', 'web-development...
1	intellisense text script template	['arrays', 'android', 'class', 'image', 'c++']	('android', 'arrays', 'c++', 'class', 'image')
2	problem post large amount datum web server iphone	['.net', 'database']	('iphone',)
3	delete post day old	['arrays', 'string', 'error', 'web-development...	('arrays', 'error', 'string', 'web-development...
4	write mysql query look element key array	['arrays', 'string', 'error', 'web-development...	('arrays', 'error', 'string', 'web-development...

Modèle final – SVC on LDA

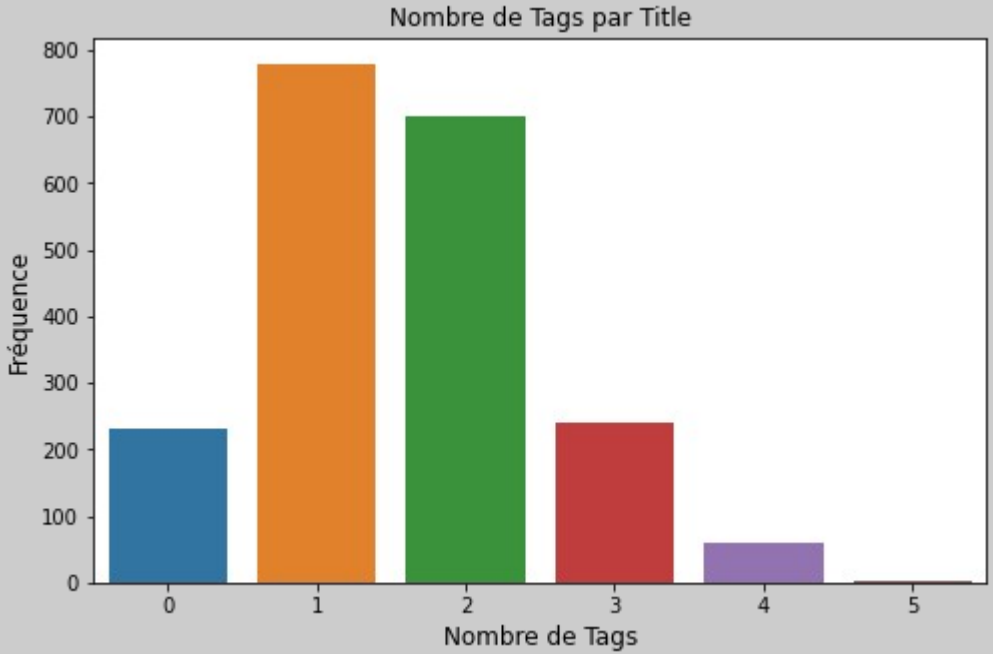




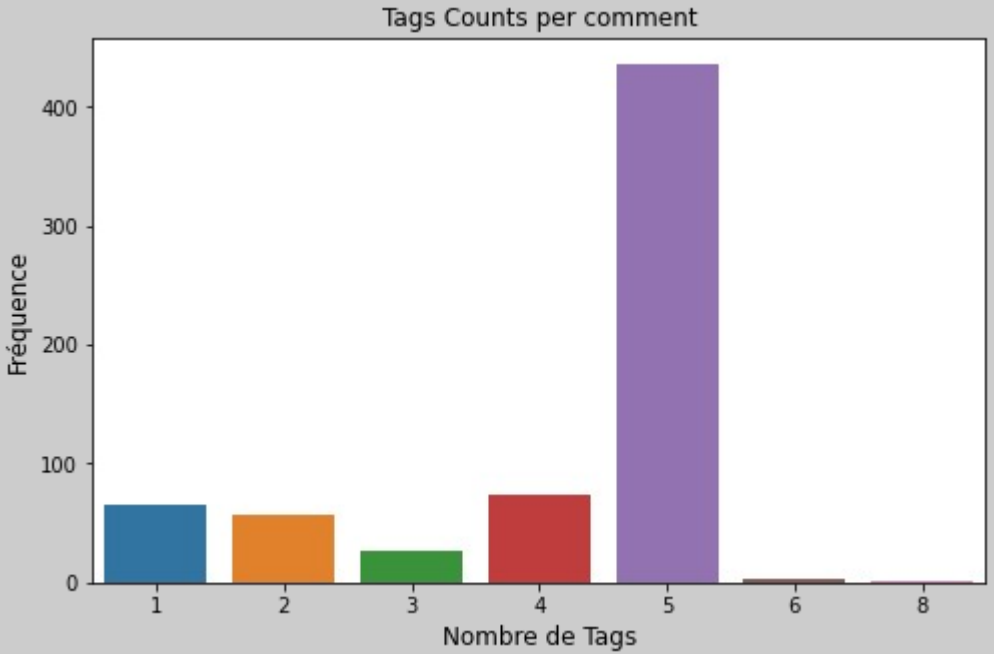
AVANT

APRES

Modèle final – SVC on LDA



AVANT



APRES

Conclusion

Application sur l'API avec Streamlit :

- Entrée manuelle de la question
- Nettoyage du texte en entrée
- Prédiction du Topic avec **OnevsRest SVC sur LDA**

Lien Github pour les commits et pour l'application :
https://github.com/CatherineLE/Stackoverflow

Lien pour l'application:
https://share.streamlit.io/catherinele/stackoverflow/app_streamlit.py

Merci pour votre écoute !
Avez-vous des questions ?