Catégorisation automatique de questions

Thomas Weber

Introduction

- Stack Overflow est un site célèbre de question-réponses liées au développement informatique.
- Plusieurs tags sont associés à chaque question afin de pouvoir retrouver facilement la question par la suite.
- Objectif: développer un système de suggestion de tag en utilisant un algorithme de machine learning.
- 2 approches testées: supervisée et non-supervisée.

Récupération des données - SQL

```
SELECT Id, CreationDate, Body, Title, Tags FROM Posts
WHERE PostTypeId = 1 ORDER BY Rand()
ASC OFFSET 0 ROWS FETCH NEXT 50000 ROWS ONLY
```

Pré-traitement des données textuelles

How to change button's size in checkbox

0

I tried to change Button in Checkbox by using png file, but the image was too big and it made the Button too big.

I want to change the Button's size to 20dp. Then I set layout_width and layout_height in CheckBox, but it didn't solve the problem.

Could you please teach me how to change the size?

```
<CheckBox
    android:layout_width="20dp"
    android:layout_height="20dp"
    android:text="New CheckBox"
    android:layout_margin="5dp"
    android:button="@drawable/my_check_box"
    android:layout_weight="1" />
```

my_check_box.xml

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_checked="false"
        android:drawable="@drawable/checkbox_normal" />
    <item android:state_checked="true"
        android:drawable="@drawable/checkbox_selected" />
</selector>
```

android button checkbox size

```
In [13]: custom_tokenizer_bis(df.iloc[1].Title) + ' ' + custom_tokenizer_bis(df.iloc[1].Body)

Out[13]: 'button size checkbox button checkbox png file image button button size 20dp set layout_width layout_height checkbox solve teac
         h size my_check_box.xml'
```

Pré-traitement des données textuelles

- Suppression des bouts de code
- Suppression des balises HTML, chiffres, urls et ponctuation
- Passage en minuscule
- Tokenisation: transformation d’un texte en liste de mots
- Lemmatisation: transformation des différentes formes d’un mot sous une même racine

Pré-traitement des données textuelles

NLTK stopwords

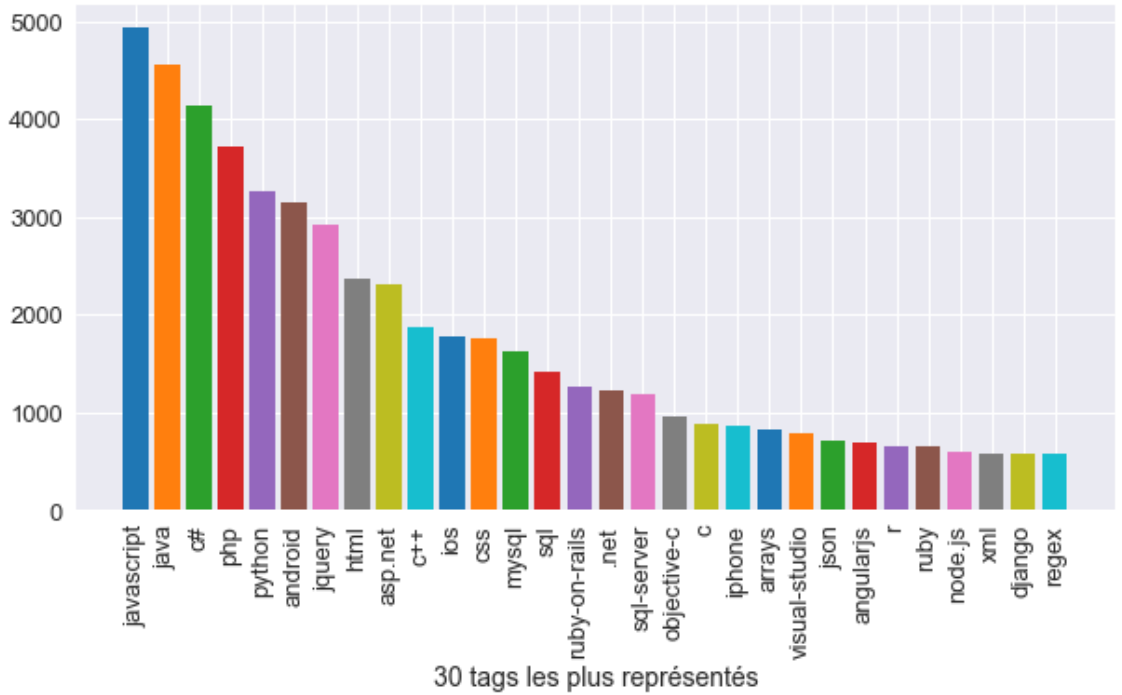
i me my myself we our ours ourselves you you're you've you'll you'd your yours yourself yourselves he him his himself she she's her hers herself it it's its itself they them their theirs themselves what which who whom this that that'll these those am is are was were be been being have has had having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

Additionnal stopwords

n't 's 'm using like way would use '' `` get one need work want 've know problem best make anyone something example question could also good solution find used new 'd ... change create looking possible able trying thing seems however working number different see without ca really two better first look think name go found another even simple add etc well help much still tried thanks say currently issue point open right since back many sure -- take lot done getting might every people setting seem anything actually current based come writing write within instead try available fine custom going easy give may around show let given difference large via mean got far keep 're always similar already us box suggestion thought rather either created kind creating reason wondering someone ’ written put several place though else called second must little x long please never great 'll seen main correct pretty v everything wrong o small quite enough certain maybe existing useful send 1 ie probably cause made last contains across thinking old added least avoid e.g ‘ simply next making automatically reading basically nice m side whether yet various ii 2 i.e recently changed anybody easily whole needed wo worked often nothing big exactly per directly perhaps manually unfortunately correctly u bad ok others wanted feel obviously later actual regular sometimes ideally top matter 3.5 b yes le left three allows fairly 2005

Pré-traitement des données textuelles

- Transformation des tags sous forme de liste.
- Filtrage des tags pour ne garder que les principaux (nombre d'occurrences > 150)
- MultiLabelBinarizer: transformation de scikit-learn adaptée à des problèmes multi-label.



Choix d’une métrique

Score de similarité de Jaccard:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

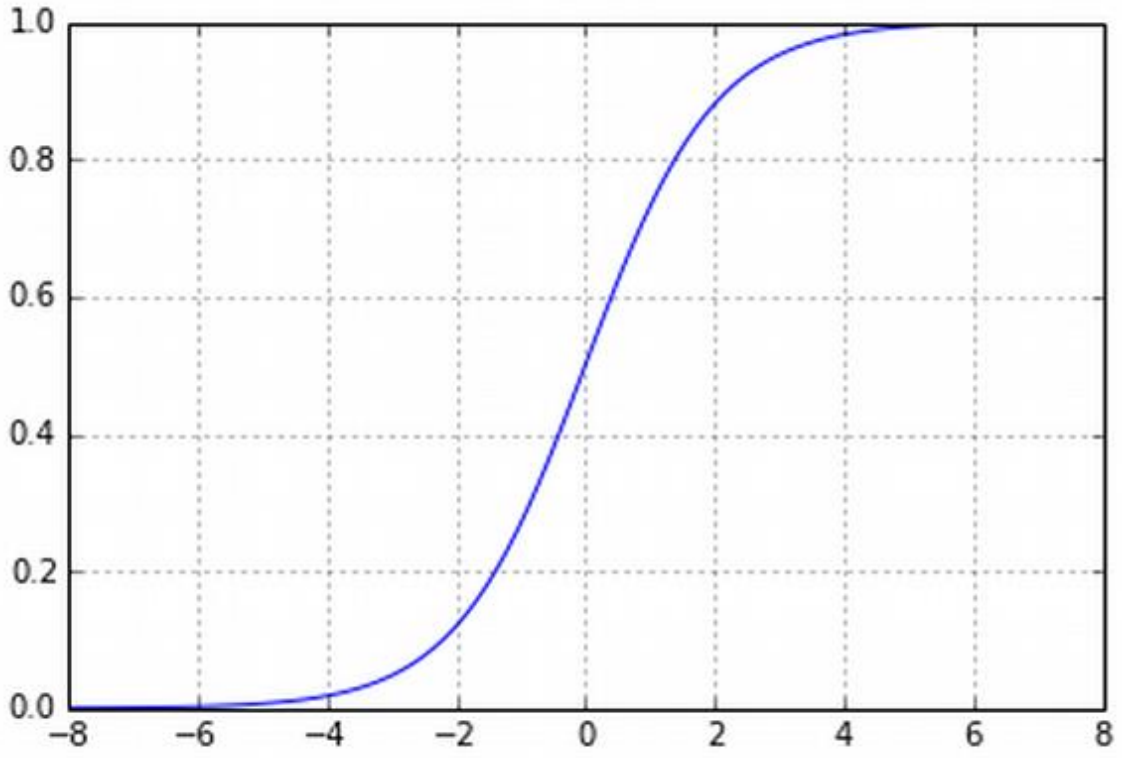
A = l’ensemble des tags réels

B = l’ensemble des tags prédits

Score entre 0 (deux ensembles disjoints) et 1 (deux ensemble égaux).

Approche supervisée

- Modèle choisi: Régression logistique
- Seuil de probabilité variable (modification de la classe LogisticRegression de Scikit-learn)
- Préparation des entrées: Transformation TF-IDF (Term Frequency – Inverse Document Frequency)



Approche supervisée – Validation croisée

- 2 paramètres pour la TF-IDF:
  - max_features: nombre de mots du vocabulaire
  - ngram_range: prise en compte des co-occurrences

- 3 paramètres pour la régression logistique:
  - estimator_penalty: norme à utiliser pour la régularisation
  - C: coefficient de régularisation
  - threshold: seuil de probabilité

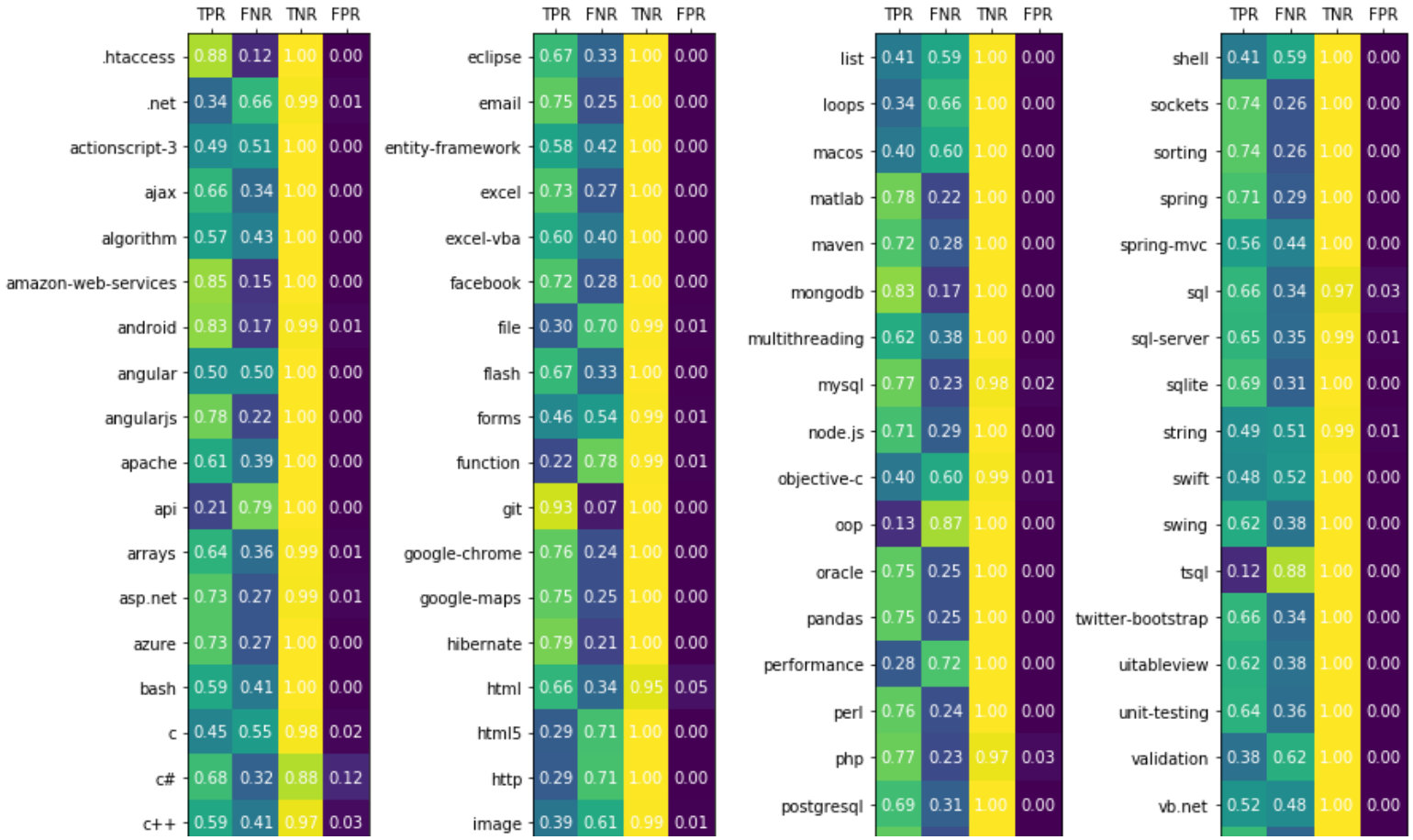
Approche supervisée – Résultats

- Paramètres obtenus:
  - max_features = 10000
  - ngram_range = (1,3)
  - estimator_penalty = L1
  - C = 1
  - threshold = 0.15

- Score sur le jeu de test: 0.504

	y_pred	y_true
0	(unit-testing,)	(maven, unit-testing)
1	(c#,)	(c#,)
2	(java,)	(java,)
3	(ios,)	(iphone, objective-c)
4	()	(validation,)
5	(angularjs, html5, java, spring)	(angularjs, spring)
6	(ios, iphone)	(ios,)
7	(c#, c++, visual-studio)	(c++,)
8	(delphi,)	(delphi,)
9	(c++, r)	(file, r)
10	(ruby, ruby-on-rails)	(javascript, ruby, ruby-on-rails)
11	(laravel, php)	(laravel, php)
12	(node.js,)	(javascript, node.js)
13	(javascript, python, regex)	(python, regex)
14	(javascript,)	(css, html, javascript, jquery)
15	(asp.net, c#)	(asp.net, ruby-on-rails)
16	(node.js,)	(node.js,)
17	(git,)	(git,)

Approche supervisée – Résultats



Approche non-supervisée

- Modélisation de sujets sous-jacents avec la Latent Dirichlet Allocation (LDA)

- Hypothèses:
  - Chaque document du corpus est un ensemble de mots sans ordre.
  - Chaque document aborde un certain nombre de thèmes dans différentes proportions qui lui sont propres.
  - Chaque mot possède une distribution associée à chaque thème. On peut ainsi représenter chaque thème par une probabilité sur chaque mot.

- Préparation des entrées: Transformation TF (Term Frequency)

Approche non-supervisée

- Pour comparer les 2 approches, il faut pouvoir obtenir une prédiction de tags à partir des thèmes de la LDA.
- On considère qu'un thème est associé à une question si sa probabilité est supérieure à un seuil.
- Puis parmi les thèmes associés on regarde les mots les plus représentatifs et on ne garde que ceux qui sont aussi des tags.
- Au final, c'est plus une approche semi-supervisée.

Approche non-supervisée – Validation croisée

- 4 paramètres obtenus par validation croisée:
  - Nombre de thèmes: 10
  - Learning decay (paramètre qui contrôle la vitesse d'apprentissage de la LDA): 0.9
  - Seuil de probabilité pour associer un thème à une question: 0.2
  - Nombre de mots à considérer dans chaque thème: 30

Approche non-supervisée – Résultats

Score sur le jeu de test: 0.06

	y_pred	y_true
0	(eclipse, file, python)	(maven, unit-testing)
1	(ajax, class, function, javascript, json, php,...	(c#,)
2	(ajax, class, function, javascript, json, php,...	(java,)
3	(ajax, class, function, javascript, json, list...	(iphone, objective-c)
4	(file, function, javascript, list, php, string)	(validation,)
5	(android, api, http)	(angularjs, spring)
6	(api, django, facebook, function, html, image,...	(ios,)
7	(android, api, eclipse, file, http, python)	(c++,)
8	(android, api, database, date, facebook, http,...	(delphi,)
9	(eclipse, file, function, javascript, list, ph...	(file, r)
10	(eclipse, file, python)	(javascript, ruby, ruby-on-rails)
11	(android, api, django, html, http, image, php)	(laravel, php)
12	(android, api, django, eclipse, file, html, ht...	(javascript, node.js)
13	(function, html, javascript, jquery, json, lis...	(python, regex)
14	(eclipse, file, html, javascript, jquery, json...	(css, html, javascript, jquery)
15	(android, api, facebook, http, sql)	(asp.net, ruby-on-rails)
16	(ajax, class, eclipse, file, function, javascr...	(node.js,)
17	(android, class, eclipse, file, git, java, mat...	(git,)

Approche non-supervisée - Analyse

Score faible car:

- Trop de tags retournés: à chaque thème associé à une question ça va tout de suite être 3 à 4 tags associés d'un coup.
- Ne peut retourner que des tags qui sont aussi présents dans le texte alors que parfois ils n'y sont pas.

Approche non-supervisée - Thèmes

```
Topics in LDA model:
Topic #0:
['app', 'server', 'project', 'web', 'application', 'net', 'url', 'view', 'request', 'client']
Topic #1:
['run', 'error', 'test', 'command', 'version', 'running', 'line', 'thread', 'python', 'program']
Topic #2:
['value', 'array', 'list', 'item', 'data', 'element', 'loop', 'object', 'result', 'map']
Topic #3:
['image', 'code', 'php', 'html', 'page', 'form', 'node', 'js', 'post', 'display']
Topic #4:
['code', 'error', 'function', 'method', 'object', 'call', 'variable', 'class', 'following', 'type']
Topic #5:
['class', 'java', 'module', 'static', 'video', 'android', 'error', 'git', 'interface', 'cell']
Topic #6:
['text', 'jquery', 'string', 'div', 'code', 'json', 'content', 'html', 'size', 'button']
Topic #7:
['table', 'query', 'column', 'row', 'data', 'database', 'sql', 'date', 'mysql', 'xml']
Topic #8:
['user', 'data', 'control', 'application', 'model', 'server', 'time', 'component', 'service', 'set']
Topic #9:
['file', 'page', 'button', 'window', 'code', 'click', 'link', 'folder', 'script', 'read']
```

Comparaison des deux approches

Approche supervisée

- Bonnes performances
- Rapide une fois que le modèle est entraîné
- Très peu de faux positifs

Approche non-supervisée

- Bonne vision des sujets abordés dans un corpus de documents
- Peut faire ressortir des nouveaux sujets pas encore couverts par les tags actuels

Déploiement des solutions

- API: http://weber-thomas.fr/ocr/project6
- Github: https://github.com/serphone/stackoverflow_tags