Projet MTI815 - Été 2022

Système de communication vocale

Mini-projet : Comparaison des méthodes de réduction de dimensionnalité LSA & NMF

Thomas Carteret
CART13129905

25 may 2022

Corpus et prétraitement

D'après le Larousse 2022, un 'document' est une pièce écrite servant d'information, de preuve. Dans notre cas, nous disposons d'écrits regroupant des titres de journaux selon un thème commun. Notre corpus est un recueil de 12 fichiers textes (.txt) sur des thèmes variés dans des langues différentes.

Afin de nettoyer le corpus, j'ai procédé à diverses étapes de nettoyage de ce dernier (ces traitements sont effectués lors du parsage ligne par ligne des documents .txt):

1. **Minuscule :**
```
lines[index] = lines[index].lower()
```
2. **Ponctuation :**
```
lines[index] = re.sub(r'[^\w\s]','',lines[index])
```
3. **Stop words :**
```
lines[index] = remove_stopwords(lines[index])
```

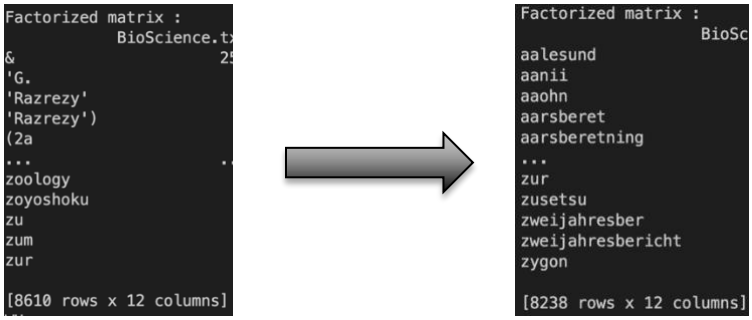
On retrouve ci-dessous la liste et le nombre de stop words de Gensim qui ont été enlevés de mon corpus (on notera qu'il ne s'agit que des stop words anglais).

```
Nb Stop words :  337
frozenset({'then', 'towards', 'nothing', 'beforehand', 'hereafter', 'back', 'forty', 'we', 'even', 'and', 'there', 'none', 'about', 'somewhere', 'hundred', '
regarding', 'herein', 'used', 'why', 'beside', 'five', 'me', 'as', 'into', 'serious', 'it', 'hereby', 'etc', 'against', 'my', 'among', 'would', 'anything', '
in', 'that', 'across', 'always', 'were', 'eleven', 'all', 'ten', 'up', 'than', 'on', 'though', 'to', 'ie', 'seemed', 'only', 'less', 'former', 'with', 'ours'
, 'they', 'amoungst', 'should', 'his', 'describe', 'of', 'because', 'yours', 'part', 'fifteen', 'such', 'anyone', 'this', 'thereby', 'could', 'never', 'almos
t', 'several', 'say', 'either', 'done', 'will', 'alone', 'full', 'around', 'however', 'thin', 'somehow', 'show', 'your', 'no', 'yourselves', 'four', 'being',
 'move', 'whatever', 'was', 'hereupon', 'who', 'ourselves', 'moreover', 'other', 'without', 'until', 'throughout', 'a', 'most', 'did', 'made', 'fill', 'altho
ugh', 'cant', 'through', 'become', 'yourself', 'whence', 'cannot', 'our', 'doing', 'empty', 'thick', 'quite', 'before', 'latterly', 'toward', 'their', 'mysel
f', 'whenever', 'over', 'becomes', 'together', 'very', 'i', 'often', 'became', 'sixty', 'nowhere', 'may', 'be', 'whether', 'have', 'please', 're', 'under', '
one', 'make', 'upon', 'still', 'is', 'if', 'itself', 'sometimes', 'same', 'via', 'due', 'own', 'off', 'else', 'the', 'thereafter', 'an', 'per', 'various', 'h
erself', 'its', 'been', 'are', 'something', 'how', 'put', 'find', 'from', 'whoever', 'hasnt', 'when', 'go', 'or', 'least', 'doesn', 'them', 'whole', 'twelve'
, 'us', 'nor', 'some', 'you', 'others', 'along', 'where', 'amount', 'during', 'once', 'mostly', 'nobody', 'except', 'has', 'km', 'by', 'both', 'de', 'didn',
'after', 'had', 'few', 'anywhere', 'down', 'eg', 'just', 'someone', 'computer', 'everyone', 'but', 'rather', 'him', 'much', 'ever', 'six', 'keep', 'whereupon
', 'besides', 'might', 'therein', 'yet', 'amongst', 'above', 'seems', 'out', 'afterwards', 'these', 'within', 'seem', 'next', 'hence', 'therefore', 'sometime
', 'must', 'anyhow', 'so', 'does', 'latter', 'sincere', 'front', 'cry', 'each', 'three', 'whereas', 'what', 'first', 'otherwise', 'everywhere', 'noone', 'mor
e', 'whom', 'fire', 'found', 'formerly', 'can', 'himself', 'really', 'nevertheless', 'too', 'which', 'wherein', 'thence', 'hers', 'bill', 'namely', 'now', 'u
sing', 'everything', 'onto', 'any', 'co', 'third', 'meanwhile', 'give', 'twenty', 'enough', 'again', 'un', 'detail', 'another', 'whereafter', 'name', 'furthe
r', 'becoming', 'thereupon', 'whither', 'interest', 'well', 'bottom', 'couldnt', 'while', 'thru', 'whereby', 'con', 'nine', 'here', 'those', 'mine', 'for', '
he', 'also', 'anyway', 'take', 'at', 'already', 'themselves', 'whose', 'perhaps', 'see', 'system', 'below', 'don', 'every', 'she', 'beyond', 'thus', 'inc', '
elsewhere', 'last', 'between', 'eight', 'wherever', 'kg', 'not', 'am', 'two', 'many', 'neither', 'indeed', 'get', 'behind', 'unless', 'do', 'seeming', 'since
', 'mill', 'side', 'her', 'call', 'top', 'fifty', 'ltd'})
```

BONUS :

4. **Lettres seules :** Les lettres seules ont été supprimées puisqu'en effet elles n'apportent aucunes informations pertinentes.
```
lines[index] = re.sub(r'[^a-z][a-z]{1}[^a-z]','',lines[index]) # signle letter in a line
lines[index] = re.sub(r'^[a-z]{1}[^a-z]','',lines[index]) # signle letter start of line
lines[index] = re.sub(r'[^a-z][a-z]{1}$','',lines[index]) # signle letter end of line
```
5. **Chiffres :** De la même façon, ils n'apportent pas d'informations pertinentes
```
lines[index] = re.sub(r'[0-9]+', '', lines[index]) # suppression des chiffres
```
6. **'Journal' :** Il s'agit d'un mot extrêmement récurent qui n'est pas discriminant
```
lines[index] = re.sub(r'journal$','',lines[index]) # mot journal non discriminant
```
(Ajouté après l'écriture du rapport améliore tous les résultats)
```
MSE NMF :  27.617152844182392
MSE LSA :  27.601442481295045
```

Après ce nettoyage en plus de donner du sens aux mots, on passe d'un corpus de 8610 mots à 8238 mots.


Factorized matrix :
BioScience.tx
&
2
'G.
'Razrezy'
'Razrezy')
(2a
...
zoology
zoyoshoku
zu
zum
zur
[8610 rows x 12 columns]
Factorized matrix :
BioSc
aalesund
aanii
aaohn
aarsberet
aarsberetning
...
zur
zusetsu
zweijahresber
zweijahresbericht
zygon
[8238 rows x 12 columns]

Utilisation de l'environnement

On génère donc ensuite la matrice factorisée TF comme suit :

```
# Matrix factorization

factMatDF = pd.DataFrame([{""}])
factMatDF = factMatDF.drop(0)
for path in docDFList:
    wordCountDoc = pd.DataFrame(docDFList[path].words.str.split(expand=True).stack().value_counts())
    wordCountDoc.columns = [path]
    wordCountDoc.to_csv('Res/'+ path +'Count.csv')
    if int(factMatDF.size) > 0 :
        factMatDF = factMatDF.join(wordCountDoc, sort=True)
    else :
        factMatDF = wordCountDoc
factMatDF = factMatDF.fillna(0)
print("Factorized matrix :")
print(factMatDF)
```

En résumé, on sépare en mots la liste des lignes de tous les documents, on compte ensuite le nombre d'occurrence de chaque mot dans chaque document. On effectue finalement une jointure sur les mots des matrices de nombre d'occurrence des mots dans les différents documents.
On obtient ainsi :

```
Factorized matrix :
                 BioScience.txt  Zoological Records.txt  Philosophy.txt  ...
aalesund                      2                     0.0             0.0  ...
aanii                         1                     0.0             0.0  ...
aaohn                         2                     0.0             0.0  ...
aarsberet                     6                     0.0             0.0  ...
aarsberetning                 4                     0.0             0.0  ...
...                         ...                     ...             ...  ...
zur                           8                    44.0             0.0  ...
zusetsu                       2                     0.0             0.0  ...
zweijahresber                 3                     0.0             0.0  ...
zweijahresbericht             3                     0.0             0.0  ...
zygon                         2                     0.0             2.0  ...
```

On effectue ensuite la factorisation NMF et LSA comme suit :

```
model = NMF(n_components=2, init='random', random_state=0)
modelLSA  = TruncatedSVD(n_components=2, n_iter=7, random_state=42)
W = model.fit_transform(factMatNP)
H = model.components_
U = modelLSA.fit_transform(factMatNP)
V = modelLSA.components_
```

On a donc en input la matrice factorisée TF : *factMatNP*
On obtient donc avec NMF les matrices *W*(words) et *H*(docs), et les matrices *U*(words) et *V*(docs) avec LSA.

Comparaison quantitative des méthodes

On reconstruit ensuite la matrice avec pour LSA $D = U.V$ et pour NMF $D = W.H$ comme suit :

```
UsV = U.dot(V) WH= W.dot(H)
```

On peut ensuite calculer la $MSE = \frac{1}{n}\sum(y - \hat{y})^2$ où $\hat{y}$ correspond à l'étiquette et donc la matrice factorisée TF et y la matrice générée D :

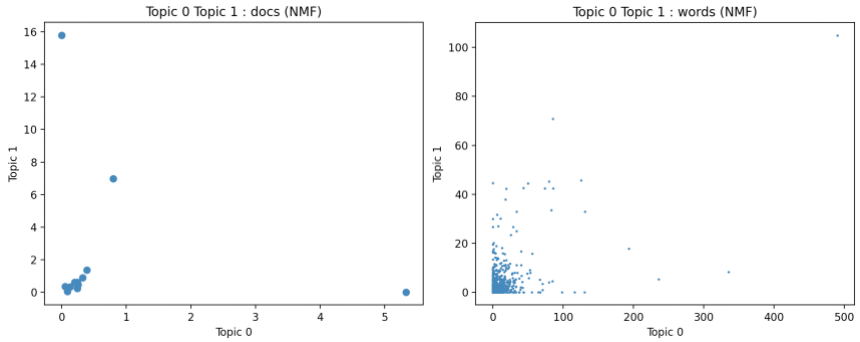
```
print("MSE NMF : ",((factMatNP-WH)**2).mean())
print("MSE LSA : ",((factMatNP-UsV)**2).mean())
```

Et on obtient :

```
MSE NMF :  33.63096214480806
MSE LSA :  33.54724781751873
```

Comparaison qualitative des méthodes

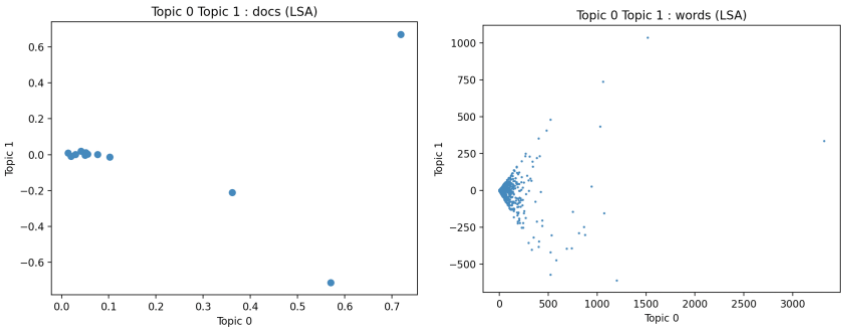
On retrouve ci-dessous le graphique des documents et des mots obtenu à partir de la factorisation NMF :



D'après le graphique de répartition des documents selon les topics 0 et 1, on observe que 3 documents se distinguent et montrent une appartenance à l'un des topics. Si l'on regarde plus en détail, on peut observer que les documents 'BioScience.txt' et 'Zoological Records.txt' se rapprochent du Topic 1 alors que le document 'Medical.txt' se rapproche du Topic 0. On peut aussi observer un gros cluster contenant tous les documents vers 0.
Le graphique des mots permet d'observer que la majorité des mots sont situés aux alentours de 0 avec quelques exceptions.

```
Matrice H :
 [[ 0.79774489  0.          0.13055966  5.32839114  0.24111314  0.2020128
   0.32264998  0.24252111  0.2547742   0.05547532  0.08727169  0.38733792]
 [ 6.97332639 15.79976396  0.31814067  0.          0.2330192   0.61435469
   0.88263753  0.61411013  0.46359423  0.34588975  0.05075391  1.36808147]]
PathList :  ['BioScience.txt', 'Zoological Records.txt', 'Philosophy.txt', 'Medical.txt',
```

On retrouve ci-dessous le graphique des documents et des mots obtenu à partir de la factorisation LSA :



D'après le graphique des documents, on peut faire les mêmes observations que précédemment quant à l'appartenance au Topic 0 et Topic 1 et l'existence d'un cluster. On remarque toutefois qu'avec LSA, la différence quant à l'appartenance au Topic 0 est moins marqué alors qu'elle l'est plus pour le Topic 1.
Le fait que l'on puisse avoir des valeures négatives augmente les écarts entre les valeurs discriminées. On remarque toutefois qu'il n'y a uniquement des valeurs négatives sur l'axe x.

```
Matrice V :
 [[ 3.61344545e-01  5.70216274e-01  2.92532524e-02  7.18884822e-01
   4.11509755e-02  4.97089435e-02  7.57981696e-02  5.51849422e-02
   5.13786027e-02  2.00587579e-02  1.36625372e-02  1.02217806e-01]
 [-2.09653110e-01 -7.11776342e-01  1.98909774e-03  6.69777822e-01
   1.93308882e-02 -2.32333383e-03  6.86300846e-04  2.67877014e-03
   1.08147644e-02 -8.58236965e-03  8.52744997e-03 -1.27930927e-02]]
```

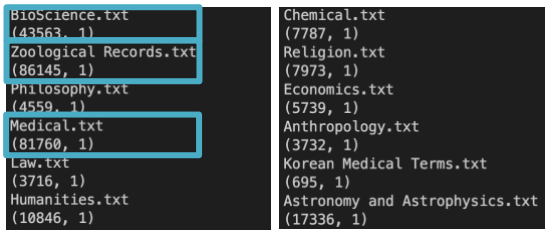
Discussion

Dans notre cas applicatif, LSA obtient une meilleure loss de $\frac{33.6309-33.5472}{33.6309} = 0.2\%$ ce qui signifie que la factorisation est plus proche de l'originale. Toutefois, le fait qu'on ait des chiffres négatifs perturbe l'interprétation humaine.
De son côté, les échelles de la factorisation NMF sont plus raisonnables ce qui améliore son utilisation.

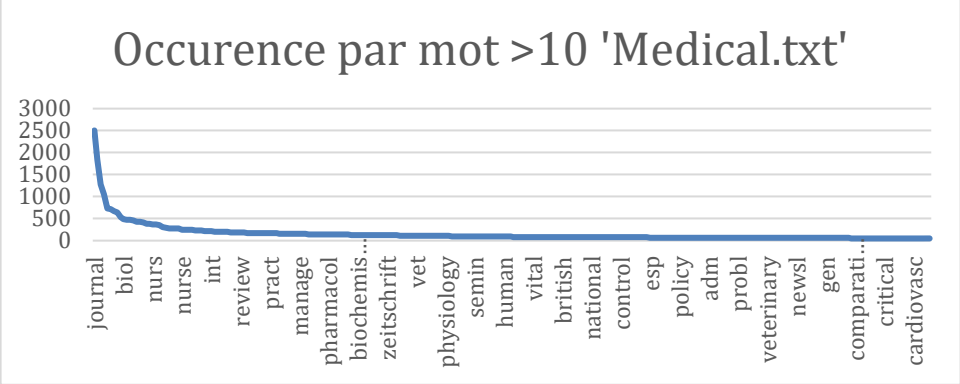
On remarque que les documents 'Philosophy.txt', 'Law.txt', 'Humanities.txt', 'Chemical.txt', 'Religion.txt', 'Economics.txt', 'Anthropology.txt', 'Korean Medical Terms.txt', 'Astronomy and Astrophysics.txt' forment un cluster qui n'est pas très discriminé entre le Topic 0 et 1. On remarque aussi que les documents 'BioScience.txt' et 'Zoological Records.txt' se rapprochent du Topic 1 et le document 'Medical.txt' se rapproche du Topic 0. On a donc deux classes qui commencent à s'identifier et se discriminer, les autres restent assez flous.

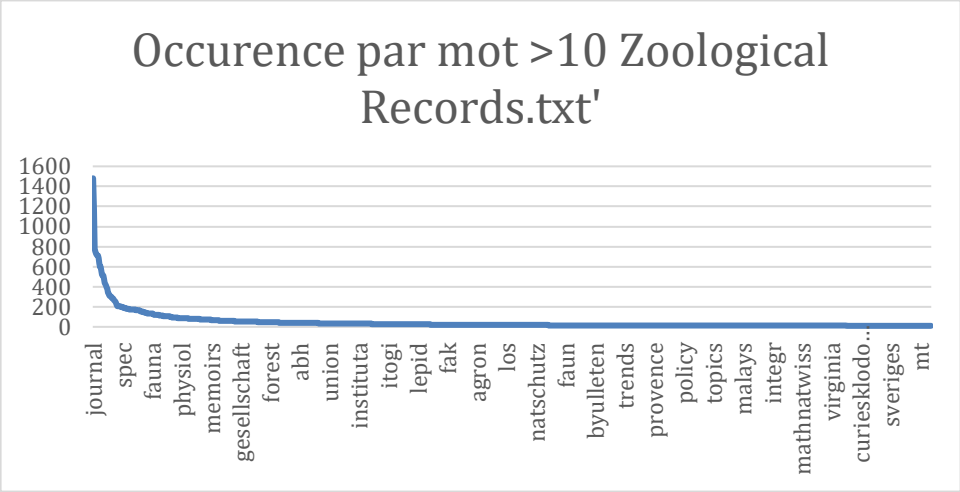
Même si corrélation ne veut pas dire causalité, on remarque que les trois classes discriminées sont aussi celles qui ont le plus de mots. On peut donc en déduire que le nombre de mots permet d'augmenter la certitude d'appartenance à une classe. Une autre hypothèse pourrait-être que du fait que le cluster/le dataset est déséquilibré (plus de mots dans certains documents *voir liste ci-dessous*), il est impossible de les rassembler en topic/classe, en effet certaines entrées sont ridiculement petites par rapport aux autres. D'après moi, la raison pour laquelle 9 documents ne sont pas classés est un mélange de ces deux hypothèses. On peut donc penser qu'augmenter et/ou égaliser le nombre de mots par document serait intéressant.

Le nettoyage du dataset est intéressant et utile puisqu'il permet d'une part de réduire la loss de moitié (MSE = 65 sans nettoyage, MSE = 27 avec nettoyage).



On peut aussi s'intéresser à la variété des mots. Par exemple si on regarde les documents 'Zoological Records.txt' et 'Medical.txt' on voit qu'avec une factorisation NMF 'Medical.txt' est discriminé à 5,5 pour le Topic 0 alors que 'Zoological Records.txt' est discriminé à 15,5 pour le Topic 1. On voit pourtant qu'ils ont approximativement le même nombre total de mots différents. Or quand on regarde la répartition des occurrences on voit qu'il y a 259 mots qui apparaissent plus de 10 fois dans 'Medical.txt', alors qu'il y en a 1251 dans 'Zoological Records.txt'. De plus, si on regarde les graphiques ci-dessous, on voit aussi qu'il y a beaucoup plus de mots qui apparaissent plus souvent. On peut donc interpréter ça en disant qu'il y a un vocabulaire moins varié dans 'Zoological records.txt' et qu'on peut donc le classer avec plus de certitude.





On peut choisir de changer la définition de document en transformant notre définition actuelle qui est qu'un document est un fichier texte en un document est une ligne de chacun de ces fichiers et donc un document est un article avec son auteur.
Ainsi on pourrait changer le nombre de classe/topic à 12 et retrouver nos 12 catégories ou alors laisser le nombre de classe/topic inchangé pour potentiellement trouver des liens entre nos 12 catégories (en mettant de la même couleur les titres d'article d'une de nos 12 catégories, on pourrait ainsi voir un Amat de couleur).