TP Réduction

Youna Froger, Alice Gyde, Mathis Quais

15 Novembre 2024

1 Exercice du TP

Voici le lien du GitHub : https://github.com/younafroger/TAL$_H NM2$

1.1 Questions

1. What is dimensionality reduction?

La réduction de dimensionnalité est une étape dans l'étude statistique de données qui consiste à réduire le nombre de variables, afin de rendre la visualisation des résultats plus claire et lisible.

2. Why and when should dimensionality reduction be applied?
La réduction de dimensionnalité intervient lorsque le nombre de variables et de données est très élévé. Par conséquent, la génération de graphique devient compliqué et le résultat est difficilement compréhensible par l'oeil humain. Elle permet également de réduire le bruit qui peut être présent, et d'améliorer les performances de traitement par la machine.

3. What is latent semantic analysis (LSA) ?
C'est une méthode de réduction de dimensionnalité qui utilise la technique SVD (Singular value decomposition) consistant en la factorisation des données. La méthode LSA priorise la mise en avant des concepts dans un texte, afin de réduire les variables à celles qui sont les plus significatives.

1.2 Instructions

Dans ce TP, le but était d'importer des données sur le notebook, les convertir en groupes de mots, appliquer la méthode LSA pour obtenir une réduction de dimension et afficher les données sous plusieurs dimensions (1D, 2D, 3D).

1.3 Explications du code

Dans un premier temps, le code permet de charger et préparer les données textuelles importées de la librairie sklearn.

```
[ ]  from sklearn.datasets import fetch_20newsgroups

     categories = ['sci.med', 'sci.space', 'sci.electronics']
     newsgroups = fetch_20newsgroups(subset ='train', categories = categories )
     documents = newsgroups.data
     targets = newsgroups.target
     target_names = [ newsgroups.target_names [ i ] for i in targets ]

     def explain_sample(sample_index):
         print("Content:", documents[sample_index])
         print("Target:", targets[sample_index])
         print("Target Name:", target_names[targets[sample_index]])
     explain_sample(1)
```

On les stocke dans `documents`, et leurs catégories réelles dans `targets` (indices) et `target_name` (noms). La fonction `explain_sample` a pour objectif de donner, pour un document : son contenu et sa catégorie.

Dans la seconde partie du code, on va traiter les données pour extraire les mots qui les composent.

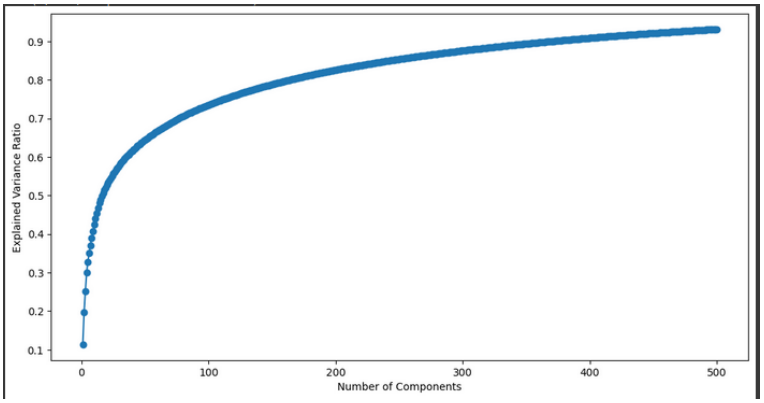
```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(analyzer="word", max_features=5000 , stop_words ='english')
doc_term_matrix = vectorizer.fit_transform(documents)

print(doc_term_matrix)
print("Shape of doc_term_matrix :", doc_term_matrix.shape) # Should show (1778 , 5000)
```

`Analyzer` indique qu'on applique notre travail sur un genre de données en particulier, ici les mots. `Max_features` limite à 5000 le nombre de *features* (sans doublon). Et `Stop-word` indique la liste des mots vides à enlever en fonction de la langue. Cette fonction permet de transformer les documents textuels chargés précédemment en une représentation sous forme de tableau.

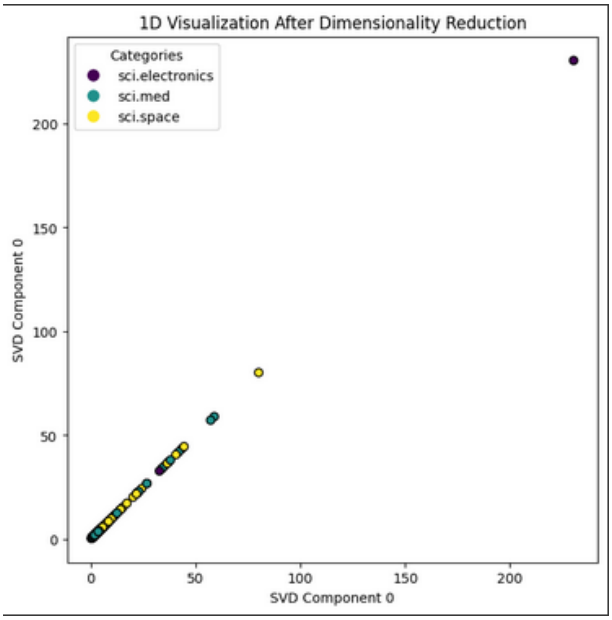
La troisième partie va nous permettre de simplifier la projection des données sur une échelle plus petite grâce à `svd_model` et réduire alors le nombre de dimensions à 500 au lieu de 5000.

```
from sklearn.decomposition import TruncatedSVD

# Initialize Truncated SVD with desired number of components (e.g. , 500)
n_components = 500
svd_model = TruncatedSVD(n_components, random_state=42)

# Fit and transform the document - term matrix with Truncated SVD
reduced_matrix = svd_model.fit_transform(doc_term_matrix)

print("Shape of reduced_matrix :", reduced_matrix.shape ) # Should show (1778 , 500)
print(reduced_matrix)
```

Grâce aux lignes de code suivantes, on peut voir le graphique associé à la variance expliquée et donc en déduire qu'à partir de 400 dimensions, on dépasse les 90 % d'informations conservées. Cela permet donc d'avoir des résultats suffisants.



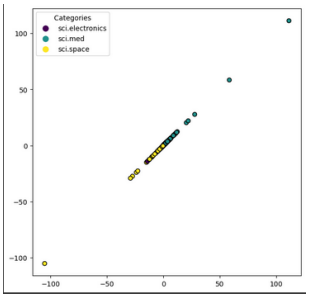
La graphique d'après montre les résultats après la réduction de dimensionnalité.



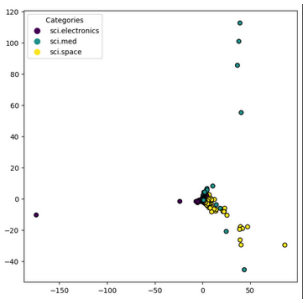
1.4 Figures générées

Visualisation 1D :

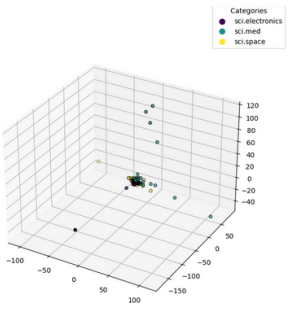
On observe grâce à ce graphique la variabilité des données sur une seule composante principale, ce qui est limité, car beaucoup d'informations se retrouvent compressées au même endroit.



Visualisation 2D : Cette fois, deux composantes principales permettent une visualisation un peu plus nette.



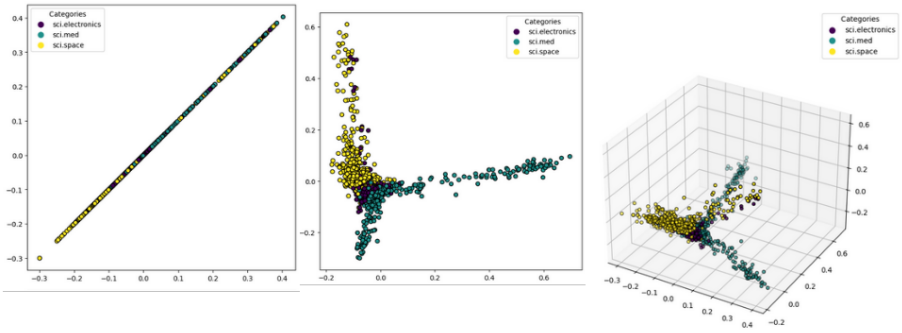
Visualisation 3D : Ici, on a une représentation en perspective, plus claire pour une compréhension humaine.



La partie concernant la matrice TF-IDF permet d'améliorer la qualité de notre étude, en gardant les termes significatifs de nos sources textuelles.

```
from sklearn.feature_extraction.text import TfidfTransformer

# Calculer la matrice TF-IDF
tfidf_transformer = TfidfTransformer()
tfidf_matrix = tfidf_transformer.fit_transform(doc_term_matrix)


svd_model = TruncatedSVD(n_components=500, random_state=42)
reduced_matrix = svd_model.fit_transform(tfidf_matrix)

plot_1D(3)
plot_2D(1,2)
plot_3D(3,1,2)
```

`TfidfTransformer()` calcule la pondération selon la méthode TF-IDF, pour chaque terme, ce qui sert à fournir, grâce à `fit_transform()`, une matrice pondérée et donc une analyse plus pertinente puisque la pertinence des mots est prise en compte. Les graphiques générés en 1D, 2D et 3D sont donc différents de ceux générés précédemment sans cette méthode de pondération.



2 Conclusion

Grâce à ce TP, nous avons vu les différentes étapes de la réduction de dimensionnalité dans le cadre d'analyse de données textuelles. Pour cela, nous avons utilisé la librairie `scikit-learn`. La réduction de dimensionnalité ici a servi à simplifier la représentation des données textuelles tout en conservant les caractéristiques essentielles à leur compréhension. Malgré le passage de 5000 à 500 dimensions, 90 % des informations sont conservées. Enfin, l'utilisation de la méthode TF-IDF rend nos résultats plus pertinents puisqu'elle permet de mettre en avant, dans nos dimensions, les mots les plus significatifs à propos des sujets des données textuelles analysées.