Travaux Pratiques
Apprentissage Automatique I

TP #6 – Clustering

Professeur

Abdelouahab Moussaoui

Partie 1 — Manipulation du Clustering

Tout d'abord, permettez-moi de préciser que le clustering TP fait référence à l'analyse de clustering, qui est une technique utilisée dans l'apprentissage automatique non supervisé pour regrouper des points de données similaires en fonction de leurs attributs. *K-means* et Hierarchical Clustering Analysis (*HCA*) sont deux algorithmes de clustering populaires.

Exercice 1 — Clustering des données clients d'un centre commercial par K-Means

Maintenant, utilisons le jeu de données Mall_Customers.csv comme exemple pour effectuer le clustering *K-means* et *HCA*.

L'ensemble de données Mall_Customers.csv contient des informations sur les clients d'un centre commercial, notamment leur *sexe*, leur *âge*, leur *revenu annuel* et leur *score de dépenses*. Nous utiliserons ces variables pour effectuer l'analyse de clustering.

Voici les étapes pour effectuer le clustering K-means :

1. Importez les bibliothèques nécessaires et chargez le jeu de données :

```
# Importer les bibliothèques nécessaires
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

# Charger le dataset
data = pd.read_csv('Mall_Customers.csv')
X = data.iloc[:, [3, 4]].values
```

2. Déterminer le nombre optimal de grappes à l'aide de la méthode du coude :

```
wcss = []

for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
```

```
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')

plt.show()
```

Cela produira un graphique de la somme des carrés intra-cluster (WCSS) pour différentes valeurs de k. Nous voulons sélectionner la valeur de k au « coude » du graphique, où le taux de diminution du WCSS commence à ralentir.

3. *Effectuez un clustering K-means avec le nombre de clusters sélectionné :*

```
kmeans = KMeans(n_clusters = 5, init = 'k-means++', random_state = 42)
y_kmeans = kmeans.fit_predict(X)
```

Cela affectera chaque point de données à l'un des cinq clusters.

4. *Visualisez les clusters :*

```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red',
                label = 'Cluster 1')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 100, c = 'blue',
                label = 'Cluster 2')
plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'green',
                label = 'Cluster 3')
plt.scatter(X[y_kmeans == 3, 0], X[y_kmeans == 3, 1], s = 100, c = 'cyan',
                label = 'Cluster 4')
plt.scatter(X[y_kmeans == 4, 0], X[y_kmeans == 4, 1], s = 100, c = 'magenta',
                label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
                s = 300, c = 'yellow', label = 'Centroids')

plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

Exercice 2 — Clustering des données clients d'un centre commercial par HCA

Maintenant, faites la même chose mais en utilisant le même le jeu de données Mall_Customers.csv avec l'algorithme *HCA*.

Partie 2 — TP en Python

Exercice 1 — Détection de fraudes par carte de crédit

Objectif

L'objectif de ce TP est d'appliquer des algorithmes de clustering (K-means, DBSCAN, HCA, etc.) pour détecter les fraudes par carte de crédit à partir d'un ensemble de données.

Questions

1. Charger le jeu de données de détection de fraude par carte de crédit *"creditcard.csv"*.
2. Effectuer une analyse de données exploratoire (*Exploratory Data Analysis : EDA*) pour comprendre la distribution des transactions et des variables pertinentes.
3. Prétraiter les données pour les rendre appropriées pour la clustering (par exemple, standardiser les variables numériques).
4. Appliquer des algorithmes de clustering tels que *K-means*, *DBSCAN*, *HCA*, etc., pour identifier des groupes de transactions similaires.
5. Évaluer les résultats du clustering en utilisant des métriques telles que le coefficient de silhouette ou l'inertie.
6. Interpréter les résultats et identifier les transactions suspectes.

Ressources

- Tutoriel sur la détection de fraude par carte de crédit avec K-means : https://www.analyticsvidhya.com/blog/2019/02/outlier-detection-python-pyod/
- Tutoriel sur la détection de fraude par carte de crédit avec DBSCAN : https://towardsdatascience.com/fraud-detection-using-clustering-and-density-based-algorithms-23bfe9425c52

Exercice 2 — Détection d'URL malveillantes

Objectif

L'objectif de ce TP est d'appliquer des algorithmes de clustering L'objectif de ce TP est d'utiliser des algorithmes de clustering pour détecter les URL malveillantes à partir d'un ensemble de données.

Questions

1. Charger le jeu de données de détection de fraude par carte de crédit *"malicious_phish.csv"*.
2. Effectuer une EDA pour comprendre la distribution des URL et des variables pertinentes.
3. Prétraiter les données pour les rendre appropriées pour la clustering (par exemple, utiliser TF-IDF pour représenter les URL sous forme de vecteurs).
4. Appliquer des algorithmes de clustering tels que *K-means*, *DBSCAN*, *HCA*, etc., pour identifier des groupes d'URL similaires.
5. Évaluer les résultats du clustering en utilisant des métriques telles que le coefficient de silhouette ou l'inertie.
6. Interpréter les résultats et identifier les URL suspectes.

Ressources

- Tutoriel sur la détection d'URL malveillantes avec K-means : https://towardsdatascience.com/finding-malicious-urls-using-machine-learning-clustering-aad4a4dcb281
- Tutoriel sur la détection d'URL malveillantes avec DBSCAN : https://towardsdatascience.com/unsupervised-machine-learning-for-detection-of-malicious-websites-ff3d1be715b2

Partie 3 — Mini-Projets en Python

Mini-Projet 1 — Clustering MNIST dataset with k-means & HCA

Ici on veut regrouper les chiffres décimaux de la base MNIST en utilisant les deux algorithmes K-means et HCA.

Questions :

1. Charger les données du *MNIST.*
2. Afficher les informations concernant les données.
3. Proposer un modèle de clustering basé sur les K-means.
4. Quel est la valeur optimale de **K** (utliser la méthode *Elbow* et *Silhouette*).
5. Refaire la même chose avec HCA.
6. Comparer ces deux modèles.
7. Visualiser les résultats comparatifs graphiquement.
8. Conclusion

Mini-Projet 2 — Clustering IRIS dataset with k-means & HCA

Même question que le Mini-projet 1 mais avec IRIS Dataset cette fois-ci.

Mini-Projet 3 — Clustering IRIS dataset with k-means & HCA

On veut découvrir les habitudes des habitantes de la ville de New York City Bike (https://ride.citibikenyc.com/system-data).

1. Données système

Où roulent les Citi Bikers ? Quand roulent-ils ? Jusqu'où vont-ils ? Quelles stations sont les plus populaires ? Quels jours de la semaine sont effectués le plus de trajets ? Nous avons entendu toutes ces questions et plus encore de votre part, et nous sommes heureux de fournir les données pour vous aider à découvrir les réponses à ces questions et plus

encore. Nous invitons les développeurs, ingénieurs, statisticiens, artistes, universitaires et autres membres du public intéressés à utiliser les données que nous fournissons pour l'analyse, le développement, la visualisation et tout ce qui vous émeut.

Ces données sont fournies conformément à la politique d'utilisation des données de Citi Bike (https://ride.citibikenyc.com/data-sharing-policy).

2. Histoires de voyages à vélo à Citi

Nous publions des fichiers téléchargeables de données de trajet Citi Bike (https://s3.amazonaws.com/tripdata/index.html). Les données comprennent :

- ID de course
- Type roulant
- Commencé à
- Terminé à
- Nom de la station de départ
- ID de la station de départ
- Nom de la station finale
- Identifiant de la station terminale
- Latitude de départ
- Longitude de départ
- Latitude de fin
- Longitude de fin
- Membre ou trajet occasionnel

Format de données auparavant :

- Durée du voyage (secondes)
- Heure et date de début
- Heure et date d'arrêt
- Nom de la station de départ
- Nom de la station finale
- Identifiant de la station
- Station Lat/Long
- ID de vélo
- Type d'utilisateur (Client = utilisateur du pass 24 heures ou 3 jours ; Abonné = membre annuel)
- Sexe (zéro=inconnu ; 1=homme ; 2=femme)
- Année de naissance

Ces données ont été traitées pour supprimer les trajets effectués par le personnel lors de l'entretien et de l'inspection du système, les trajets effectués vers/depuis l'une de nos stations « d'essai » (que nous utilisions davantage en juin et juillet 2013) et tout les trajets d'une durée inférieure à 60 secondes (potentiellement de faux départs ou des utilisateurs essayant de reconnecter un vélo pour s'assurer qu'il est sécurisé).

Veuillez tenir compte des limitations de lignes de votre logiciel lorsque vous visualisez les données. De nombreux fichiers CSV contiennent plus d'un million de lignes. Après le téléchargement, vous devrez utiliser un outil / visualiseur de données volumineux (comme Tableau, Alteryx, R ou autres) pour afficher et analyser les ensembles de données complets.

Télécharger les données de l'historique des trajets Citi Bike (https://s3.amazonaws.com/tripdata/index.html)

3. Données en temps réel

Citi Bike publie des données système en temps réel au format General Bikeshare Feed Specification (https://github.com/NABSA/gbfs/blob/master/gbfs.md).

Obtenez le flux GBFS ici (http://gbfs.citibikenyc.com/gbfs/gbfs.json) .

4. Rapports d'exploitation mensuels

Consultez les rapports d'exploitation mensuels (https://ride.citibikenyc.com/system-data/operating-reports) que nous fournissons au département des transports de New York.

5. Ressources additionnelles

- Les données cyclables (http://www.nyc.gov/html/dot/html/about/datafeeds.shtml#Bikes) de la ville de New York
- Un groupe de développeurs de logiciels et d'explorateurs de données travaillant avec des flux de données du système de partage de vélos de NYC et d'autres données sur les vélos maintiennent ce groupe Google (https://groups.google.com/forum/#!aboutgroup/citibike-hackers) (remarque : Citi Bike n'est pas responsable de ce groupe - il est géré et maintenu par un groupe de citoyens privés intéressés)

Questions :

1. Télacharger une datasets selon votre choix (Télécharger les données de l'historique des trajets Citi Bike)— https://s3.amazonaws.com/tripdata/index.html.
2. Regrouper et visualiser les données à l'aide des algorithmes de clustering spatiales (DBSCAN).
3. Comparer les résultats avec d'autres modèles basés également sur la densité.