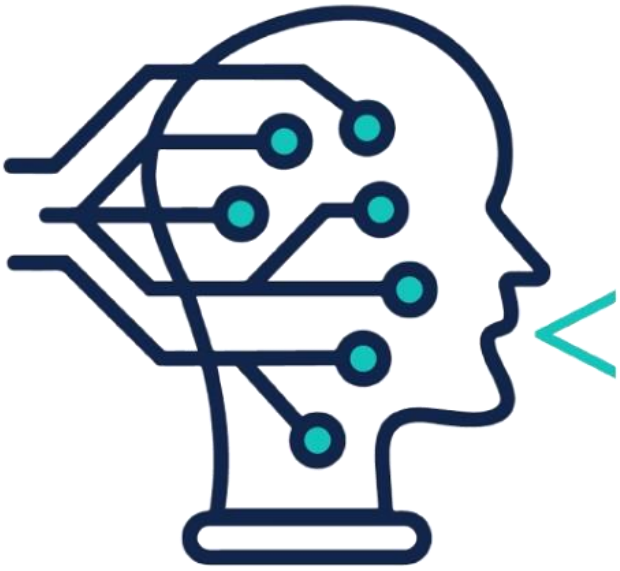
Lab 4 : Clustering on IOT-detection and NSL-KDD

Big Data et IA pour les réseaux et la sécurité



6 OCTOBRE 2024

Ilhem OUASSINI
I2-APP RS1

PARIS PANTHÉON-ASSAS UNIVERSITÉ

Table des matières

Introduction 3

Partie 1 : Introduction au clustering 4

Partie 2 : Chargement des données 5

Partie 3 : K-means Clustering 6

Partie 4 : Hierarchical Clustering (CAH) 9

Partie 5 : Questions à répondre 11

Partie 6 : Report and Submit Findings 18

Introduction

Vous trouverez dans le notebook nommé « TP4_OUASSINI_Ilhem.ipynb » tous les commentaires et exécutions du code, et dans ce rapport, les analyses des résultats.

L'objectif de ce laboratoire est d'acquérir des compétences dans l'application de deux algorithmes de clustering essentiels : **K-means** et le **Clustering Hiérarchique (CAH)**. Nous analyserons deux ensembles de données, à savoir NSL-KDD, relatif à la détection d'intrusions dans les réseaux, et IoT-detection, qui se concentre sur la détection des activités des dispositifs.

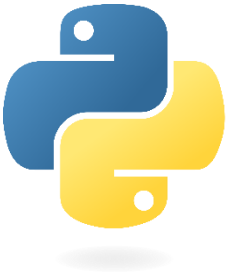
Au cours de ce laboratoire, nous aborderons différentes étapes qui nous permettront d'effectuer une analyse approfondie et d'interpréter les résultats obtenus.

Outils :

- VSC
- Jupyter

Langage de programmation :

- Python



Partie 1 : Introduction au clustering

Dans cette partie, nous allons présenter le concept de clustering. Le clustering est une technique qui permet de regrouper des points de données similaires. Dans notre étude, nous utiliserons le dataset NSL-KDD pour classer le trafic en deux catégories : le comportement "normal" et l'activité "suspecte". De la même manière, il est possible de regrouper les appareils IoT selon leurs modèles d'activité. Nous allons nous concentrer sur deux algorithmes principaux : **K-means** et le **Clustering Hiérarchique (CAH)**, qui sont des méthodes d'apprentissage non supervisé utilisées pour analyser et regrouper des données sans étiquettes prédéfinies.

Définitions des algorithmes :

- **K-means** : Cet algorithme divise les données en un nombre prédéfini de groupes, en cherchant à rassembler les points similaires au sein de chaque groupe.
- **Clustering Hiérarchique (CAH)** : Cette méthode crée des clusters progressivement, en fusionnant les points de données similaires les uns avec les autres.

Partie 2 : Chargement des données

Nous allons travailler avec deux ensembles de données :

- **NSL-KDD** : Cet ensemble est utilisé pour détecter les intrusions dans les réseaux, notamment pour identifier les attaques.
- **IoT-detection** : Ce dataset contient des informations sur l'activité des appareils IoT, que nous pouvons regrouper en fonction de leur comportement.

Nous allons reprendre les étapes des laboratoires 2 et 3, qui incluent les tâches suivantes : charger les fichiers CSV, encoder les caractéristiques, convertir certaines colonnes, et appliquer un standard scaler pour normaliser les caractéristiques.

Dans le notebook de code, vous trouverez en détail les instructions pour le traitement des deux fichiers CSV, à savoir IoT et NSL-KDD.

Ensuite, nous sauvegarderons chaque datasets prétraité dans de nouveaux fichiers CSV, qui nous seront très utiles pour la suite du laboratoire.

Pour IOT :

```
# Sauvegarde du dataset prétraité en un nouveau fichier CSV
df.to_csv('iot_dataset_pre-processed.csv', index=False)

# Recharger le dataset prétraité
df_pretraite = pd.read_csv('iot_dataset_pre-processed.csv')

# Afficher les premières lignes du dataset pour vérifier
df_pretraite.head()
```

Pour NSL-KDD :

```
# Sauvegarder le DataFrame d'entraînement et le DataFrame de test en fichiers CSV
train_df.to_csv('nsl_kdd_simple.csv', index=False)

# Recharger le dataset prétraité
df_pretraite = pd.read_csv('nsl_kdd_simple.csv')

# Afficher les premières lignes du dataset pour vérifier
df_pretraite.head()
```

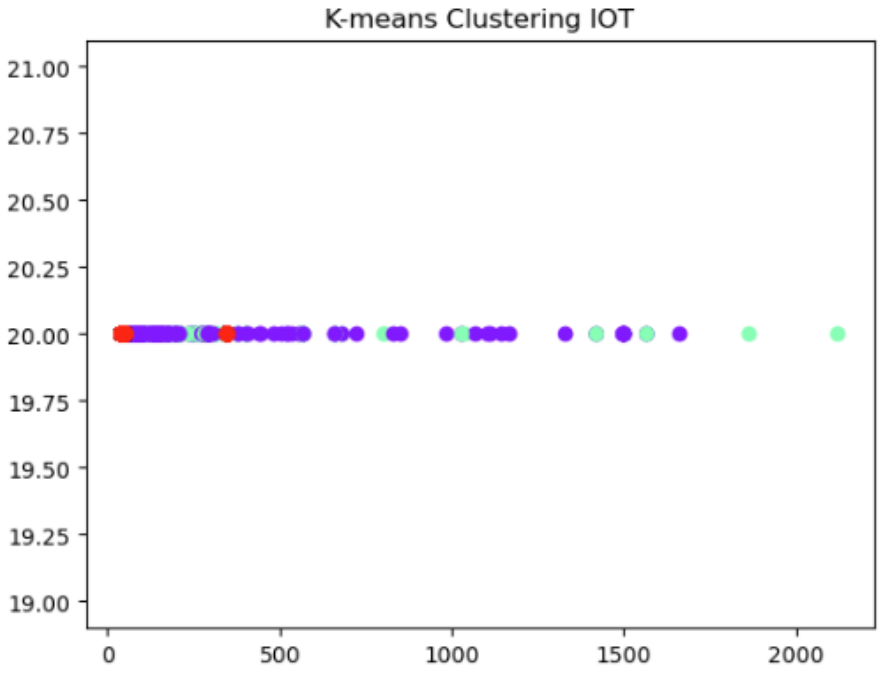
Partie 3 : K-means Clustering

Dans cette section, nous allons appliquer l'algorithme K-means, qui divise les données en un certain nombre de groupes. Cet algorithme cherche à identifier des groupes où les points de données sont proches les uns des autres. Nous allons donc procéder à l'exécution de l'algorithme K-means.

Pour IOT :

```
import pandas as pd
from sklearn.cluster import KMeans
from matplotlib import pyplot as plt
# Load the dataset (NSL-KDD or IoT-detection)
data = pd.read_csv('iot_dataset_pre-processed.csv')
# K-means Clustering with 3 clusters
kmeans = KMeans(n_clusters=3)
kmeans_labels = kmeans.fit_predict(data)
# Plotting the clusters using the first two features
plt.scatter(data.iloc[:, 0], data.iloc[:, 1], c=kmeans_labels, cmap='rainbow')
plt.title('K-means Clustering IOT')
plt.show()
```

Résultat

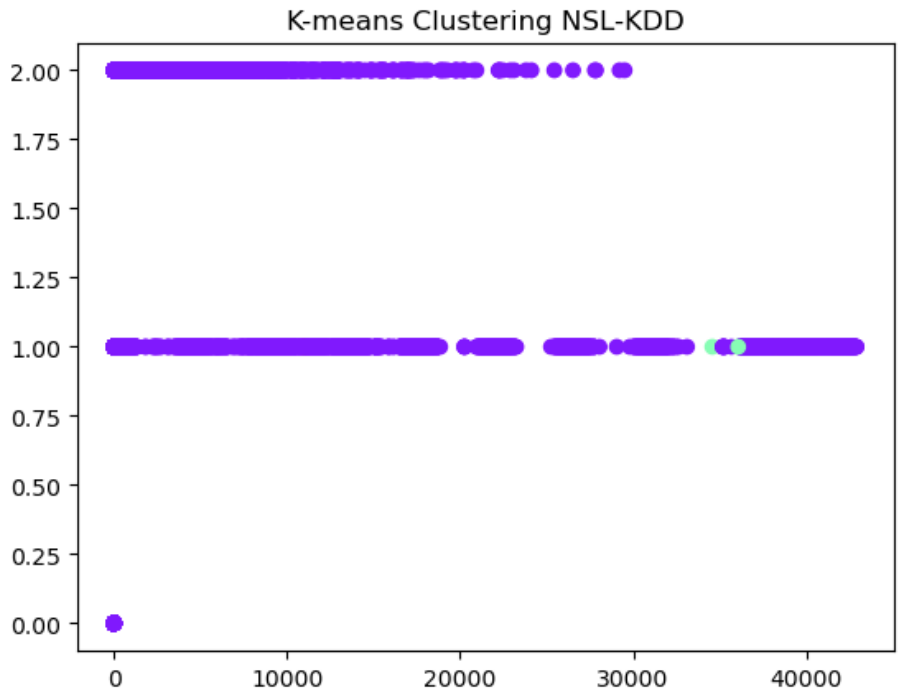


Pour l'IOT, le graphique affiche des points de trois couleurs différentes : rouge, violet et vert, ce qui indique que l'algorithme K-means a généré trois clusters. Cela correspond au paramètre défini dans le code, où nous avons spécifié le nombre de clusters à 3 : kmeans = KMeans(n_clusters=3).

Pour NSL-KDD :

```
import pandas as pd
from sklearn.cluster import KMeans
from matplotlib import pyplot as plt
# Load the dataset (NSL-KDD or IoT-detection)
data = pd.read_csv('nsl_kdd_simple.csv')
# K-means Clustering with 3 clusters
kmeans = KMeans(n_clusters=3)
kmeans_labels = kmeans.fit_predict(data)
# Plotting the clusters using the first two features
plt.scatter(data.iloc[:, 0], data.iloc[:, 1], c=kmeans_labels, cmap='rainbow')
plt.title('K-means Clustering NSL-KDD')
plt.show()
```

Résultat



Pour le NSL-KDD, le graphique affiche deux couleurs distinctes : violet et vert. Cependant, la couleur violette domine largement. On remarque également que, sur le graphique, les

points pour les valeurs y=1 et y=2 forment une ligne constante, tandis qu'il y a un point violet isolé à la position y=0, x=0. Cette répartition résulte de l'algorithme K-means, où nous avons défini le nombre de clusters à 3 : kmeans = KMeans(n_clusters=3). Je pense que deux des clusters partagent des caractéristiques très similaires, ce qui donne l'impression que les données sont dominées par une seule couleur. Cela peut s'expliquer par le fait que le dataset NSL-KDD classe le trafic en deux catégories : comportement normal ou suspect.

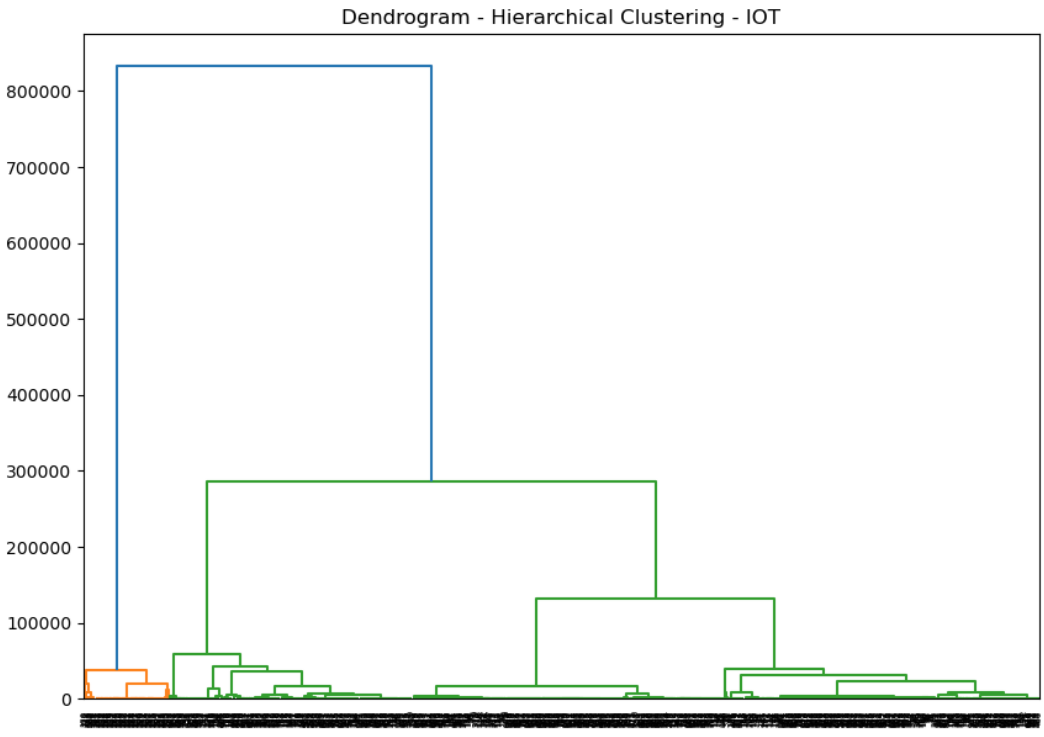
Partie 4 : Hierarchical Clustering (CAH)

Dans cette section, nous allons utiliser l'algorithme de Clustering Hiérarchique (CAH), qui fusionne progressivement les points de données les plus similaires en groupes. Pour visualiser ce processus, nous nous appuierons sur un dendrogramme, qui illustre la formation des groupes étape par étape. Nous allons donc exécuter l'algorithme CAH, en utilisant également la méthode de Ward pour regrouper les points similaires.

Pour IOT :

```
from scipy.cluster.hierarchy import dendrogram, linkage
from matplotlib import pyplot as plt
data = pd.read_csv('iot_dataset_pre-processed.csv', nrows=1000)
# Create the linkage matrix using the Ward method
linked = linkage(data, method='ward')
# Plot the dendrogram
plt.figure(figsize=(10, 7))
dendrogram(linked)
plt.title('Dendrogram - Hierarchical Clustering - IOT')
plt.show()
```

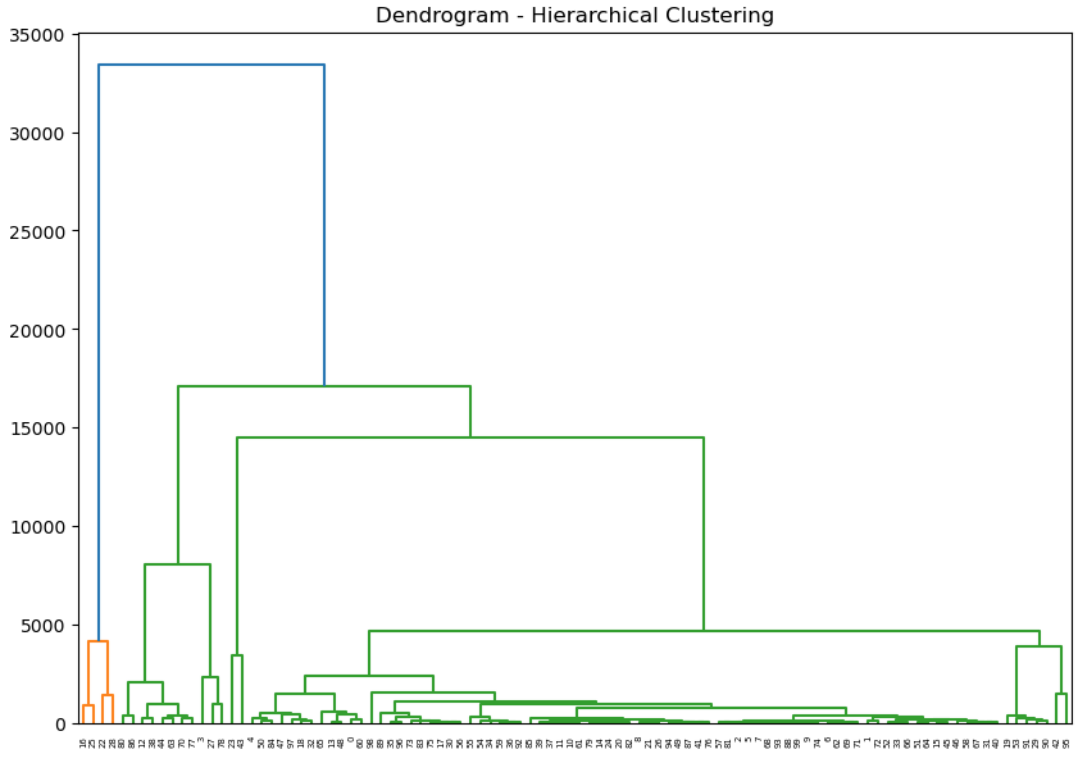
Résultat



Pour NSL-KDD :

```
from scipy.cluster.hierarchy import dendrogram, linkage
from matplotlib import pyplot as plt
data = pd.read_csv('nsl_kdd_simple.csv', nrows=100)
# Create the linkage matrix using the Ward method
linked = linkage(data, method='ward')
# Plot the dendrogram
plt.figure(figsize=(10, 7))
dendrogram(linked)
plt.title('Dendrogram - Hierarchical Clustering - NSL-KDD')
plt.show()
```

Résultat



Partie 5 : Questions à répondre

Après avoir exécuté les algorithmes K-means et Hierarchical Clustering, nous allons répondre aux questions suivantes :

1. K-means Clustering

Combien de clusters les K-moyens ont-ils créées ?

Pour IOT :

Les couleurs dans le graphique représentent les trois clusters créés par l'algorithme K-means. Chaque couleur (rouge, violet et vert) correspond à un groupe de points de données similaires, basés sur les caractéristiques du dataset IoT.

L'algorithme K-means a permis de regrouper les appareils IoT en fonction de ces caractéristiques, ce qui facilite la distinction entre comportements réseau normaux et potentiellement anormaux. Cette segmentation est particulièrement utile pour surveiller les anomalies et identifier des patterns spécifiques de communication entre les dispositifs IoT.

En définissant le nombre de clusters à 3 dans le code, l'algorithme a ainsi segmenté les données en trois groupes distincts, ce qui simplifie l'analyse et l'interprétation des résultats.

Pour NSL-KDD :

D'après le graphique, il semble y avoir deux clusters, identifiés par des différences de couleurs, principalement du violet avec un peu de vert. Cependant, la majorité des points appartiennent au cluster violet. Je pense que cela est dû au dataset NSL-KDD, qui classe le trafic réseau en deux catégories de comportement : normal ou suspect.

Pouvez-vous voir une séparation claire entre les clusters sur le graphique ?

Pour IOT :

On constate que le graphique montre une forte superposition des points près de l'axe des abscisses, surtout entre 0 et 500 sur l'axe des x. Dans cette zone, il n'y a pas de séparation claire entre les clusters car les points de différentes couleurs sont mélangés. En revanche, à partir ~1000 sur l'axe des x, les points commencent à être mieux séparés, et on peut voir une certaine distinction même si la séparation n'est pas encore parfaite.

Cette forte superposition des points entre 0 et 500 peut être due à plusieurs raisons. Tout d'abord, les appareils IoT dans cette zone peuvent avoir des comportements ou des caractéristiques très similaires, ce qui rend difficile la séparation des clusters. De plus, le choix de 3 clusters peut ne pas suffire pour capturer la complexité des données, ce qui entraîne un chevauchement. À partir de 1000 sur l'axe des x, les différences entre les appareils deviennent plus marquées et facilitant ainsi la séparation visuelle des clusters.

Pour NSL-KDD :

La séparation entre les clusters n'est pas clairement visible. Les points sont majoritairement alignés sur deux lignes horizontales (à 1 et 2 sur l'axe des y), avec beaucoup de chevauchements, rendant difficile la distinction entre les clusters.

2. Hierarchical Clustering (CAH)

Regardez le dendrogramme. Combien de groupes pensez-vous qu'il y a sur la base du dendrogramme ?

Pour IOT :

En observant le dendrogramme, on remarque que la coupure idéale se situe généralement là où il y a le plus grand saut vertical avant que les branches ne se divisent. Dans ce cas, un grand saut se produit autour de 300 000, ce qui suggère l'existence de 2 clusters principaux. Si l'on coupe le dendrogramme à ce niveau, on formerait deux grands groupes.

Ainsi, bien qu'il soit possible de discerner jusqu'à 3 clusters, les deux premiers sont les plus clairement définis d'après la structure du dendrogramme.

Les deux clusters principaux du dendrogramme peuvent s'expliquer par plusieurs facteurs. Les données IoT se divisent naturellement en deux groupes à cause de caractéristiques similaires. La majorité des points appartient à ces deux groupes denses, tandis qu'un troisième cluster est plus dispersé et moins prononcé. L'algorithme CAH regroupe les points par similarité, et le saut dans le dendrogramme montre une séparation nette entre les deux clusters. Un troisième cluster est présent, mais moins marqué.

Pour NSL-KDD :

En observant le dendrogramme, on remarque un grand saut vertical dans la distance à ~10 000. Cela indique qu'il pourrait y avoir environ 3 clusters principaux si nous faisons une coupure à ce niveau. Toutefois, plusieurs petites branches en bas pourraient également représenter d'autres clusters, bien qu'elles soient moins distinctes.

Le dendrogramme suggère 3 clusters principaux en raison de plusieurs éléments. D'abord, un grand saut vertical dans la distance à environ 10 000 indique une grande différence entre ces clusters, ce qui signifie qu'ils sont plus similaires entre eux en dessous de cette coupure. De plus, la structure des données NSL-KDD pourrait naturellement se diviser en différentes catégories de trafic, reflétant des comportements normaux ou suspects. Enfin, bien que des petites branches soient visibles, elles représentent des regroupements moins significatifs. Ainsi, le dendrogramme montre une séparation claire en 3 clusters principaux tout en laissant la possibilité de clusters supplémentaires moins distincts.

En quoi la classification hiérarchique diffère-t-elle de la classification K-means en ce qui concerne le regroupement des données ?

Pour IOT :

Voici un tableau qui compare les deux modèles.

Critère	**K-means**	**Classification Hiérarchique (CAH)**
Nombre de clusters	Le nombre de clusters doit être défini à l'avance (K).	Pas besoin de le définir à l'avance, les clusters sont déterminés par le dendrogramme.
Visualisation	Affiche les clusters sans relation hiérarchique visible.	Génère un dendrogramme qui montre les relations hiérarchiques entre les données.
Flexibilité	Moins flexible, car il génère des clusters fixes.	Très flexible, permet d'explorer plusieurs niveaux de regroupement en fonction de la granularité souhaitée.
Vitesse de calcul	Plus rapide et efficace pour de grandes quantités de données IoT.	Plus lent, surtout avec des ensembles de données volumineux comme ceux générés par les dispositifs IoT.
Sensibilité aux valeurs incohérentes	Très sensible aux valeurs incohérentes, qui peuvent perturber les clusters formés.	Moins sensible aux valeurs aberrantes, ces données peuvent être mieux absorbées dans la structure hiérarchique.
Complexité des données	Moins efficace pour les données IoT complexes ou avec des anomalies.	Adapté pour des données IoT complexes, car il révèle les relations entre les capteurs ou les mesures.
Utilisation pour IoT	Pratique pour des analyses rapides avec des clusters simples et bien séparés.	Utile pour découvrir des relations et des patterns cachés dans les données IoT.

Pour les données IoT, la classification hiérarchique est idéale pour analyser les relations complexes entre les capteurs et les différents types de données, surtout lorsqu'il s'agit d'explorer divers niveaux de regroupement. En revanche, K-means est mieux adapté aux

environnements où une vitesse rapide de traitement est nécessaire, mais il peut être limité en raison de sa sensibilité aux anomalies et de la nécessité de définir un nombre fixe de clusters à l'avance.

Pour NSL-KDD :

Voici un tableau qui compare les deux modèles.

Critère	**K-means**	**Classification Hiérarchique (CAH)**
Nombre de clusters	Nécessite de fixer le nombre de clusters à l'avance (K).	Pas besoin de le définir à l'avance -> les clusters peuvent être observés dans le dendrogramme.
Visualisation	Offre une visualisation plus directe des clusters, mais sans montrer les relations entre eux.	Produit un dendrogramme qui montre la structure et les relations entre les données à différents niveaux.
Flexibilité	Moins flexible, car il fixe les clusters et ne permet pas de voir la hiérarchie.	Permet une meilleure flexibilité, utile pour explorer des niveaux de regroupement variés.
Vitesse de calcul	Plus rapide et adapté aux grands ensembles de données.	Plus lent, surtout pour de grands ensembles de données comme NSL-KDD.
Sensibilité aux valeurs incohérentes	Très sensible aux valeurs incohérentes, ce qui peut perturber les clusters.	Moins sensible aux valeurs aberrantes, ces données peuvent être mieux intégrées.
Complexité des données	Convient mieux aux données plus simples et bien séparées.	Convient mieux aux données complexes, car il montre les relations profondes entre elles.
Utilisation pour NSL-KDD	Rapide pour des clusters simples, mais moins adapté aux comportements complexes ou irréguliers des données NSL-KDD.	Permet d'analyser des attaques ou des comportements anormaux à différents niveaux de détail.

Les données NSL-KDD sont complexes et incluent des attaques ou comportements anormaux. La classification hiérarchique permet une exploration plus approfondie et flexible. Elle montre bien les relations entre les clusters. En revanche, K-means est plus rapide et utile pour des regroupements simples. Mais il est sensible aux valeurs aberrantes, ce qui peut nuire à la qualité des résultats.

Conclusion

Dans ce travail pratique, nous avons étudié et comparé deux méthodes de regroupement de données : la classification hiérarchique ascendante (CAH) et K-means, en appliquant ces algorithmes sur deux datasets distincts, IoT et NSL-KDD.

La CAH est adaptée pour les données complexes comme IoT. Elle est flexible et robuste face aux anomalies, sans besoin de définir le nombre de clusters à l'avance. K-means est plus rapide, ce qui est utile pour des grands ensembles de données comme NSL-KDD. Cependant, il est sensible aux valeurs aberrantes et exige de connaître le nombre de clusters à l'avance. Le choix entre les deux dépend des besoins en flexibilité ou en rapidité.

Voici un tableau qui présente les avantages et les inconvénients des deux modèles étudiés dans ce TP, à savoir la CAH et K-means.

Modèle	**Avantages**	**Inconvénients**
CAH (Classification Hiérarchique)	- Pas besoin de définir le nombre de clusters à l'avance - Bonne visualisation des relations entre les données - Moins sensible aux valeurs aberrantes	- Plus lent avec de grands ensembles de données - Complexité accrue pour des grands volumes
K-means	- Rapide et efficace pour de grands volumes de données - Simple à implémenter	- Nécessite de fixer le nombre de clusters à l'avance - Sensible aux valeurs aberrantes

Partie 6 : Report and Submit Findings

Pour consulter le code exécuté et les analyses en détail, veuillez télécharger le fichier zip nommé LAB4_OUASSINI_Ilhem.zip, qui contient :

- Notebook : « TP4_OUASSINI_Ilhem.ipynb »
- CSV : « iot_dataset_pre-processed.csv»
- CSV : « nsl_kdd_simple.csv »
- Rapport : « Lab4_clustering_OUASSINI.pdf »