Royaume du Maroc
Université Mohammed Premier
École Supérieure De Technologie-Oujda
Département : Informatique
Filière : LP Informatique Décisionnelle

Analyse De Données

TP7 : Clustering (Regroupement)

Réalisé par : **HNIOUA Abdessamad**

Encadré par : **M. Mounir Grari**

Année Universitaire : 2023/2024

Introduction

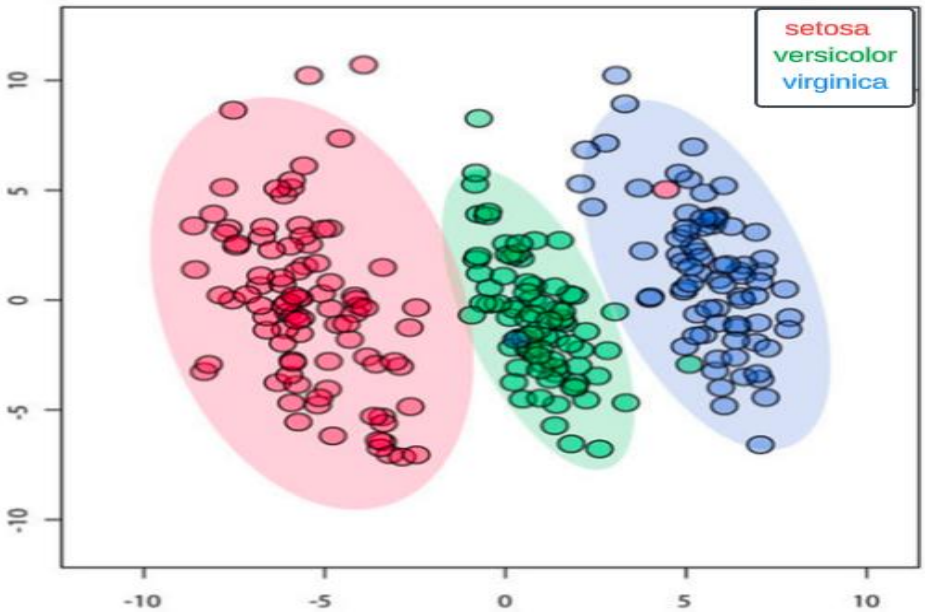
Le clustering est une technique d'apprentissage automatique non supervisée essentielle pour découvrir des regroupements naturels au sein de données non étiquetées. Par le biais d'algorithmes tels que **K-means**, **DBSCAN** et **le clustering hiérarchique**, cette approche vise à maximiser la similarité au sein des clusters tout en accentuant les différences entre eux.

Technique d'apprentissage automatique non supervisée : est une méthode où l'algorithme apprend et trouve des motifs dans des données sans être guidé par des réponses spécifiques. Il s'agit de découvrir des structures cachées, comme regrouper des données similaires, sans avoir de catégories ou d'étiquettes préétablies. Cela permet de révéler des regroupements naturels, des anomalies ou des associations au sein des données, offrant des insights précieux lorsqu'on ne connaît pas à l'avance les résultats ou les classifications des données analysées.

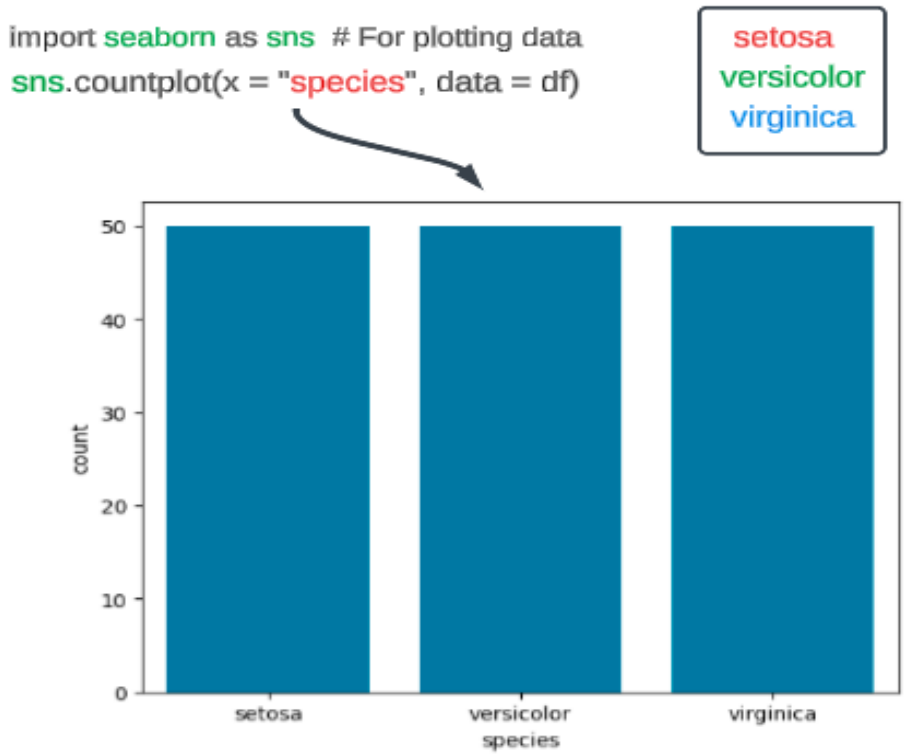
L'objectif de ce rapport est de démontrer l'application de l'algorithme de clustering K-means sur le jeu de données Iris, incluant l'importation, la préparation des données, l'exécution de l'analyse de clustering, et la visualisation des résultats. Nous visons à illustrer comment le clustering peut révéler des groupements naturels au sein des données et évaluer l'efficacité de K-means pour classer les échantillons en clusters significatifs basés sur leurs attributs.

Algorithmes K-means :

L'algorithme K-means est une méthode de clustering non supervisée qui divise un ensemble de données en K clusters en minimisant la variance intra-cluster et maximisant la variance inter-cluster, en utilisant des itérations d'affectation et de mise à jour des centroïdes.



- Visualiser la distribution des différentes catégories de la variable "species" dans le DataFrame df, ce qui aide à comprendre la répartition des données avant d'appliquer des techniques de clustering.



- La variable cible 'species' est assignée à y, tandis que X contient les caractéristiques des observations sans la classe 'species', afin de préparer les données pour le clustering.

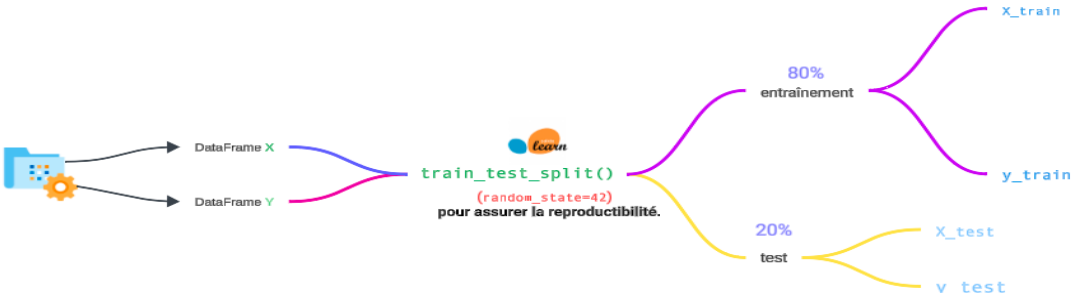
```
y = df.species
# Removes the y column from df
X = df.drop('species', axis=1)
```

- Cette partie du code normalise les données dans X à l'aide de la standardisation, puis crée un nouveau DataFrame pour stocker ces données standardisées, prêt à être utilisé dans le processus de clustering. L'affichage des cinq premières lignes du DataFrame X vérifie que la transformation s'est effectuée correctement.

```
X = pd.DataFrame(StandardScaler().fit_transform(X),
columns=X.columns)
# Displays the first 5 rows of df
X.head()
```

- Répartition des données en entraînement et test :

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```



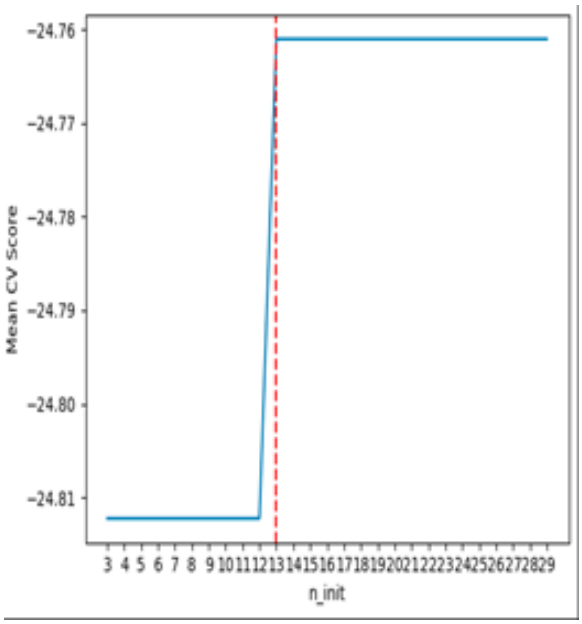
- Cette partie du code instancie un objet KMeans avec des paramètres spécifiés, ajuste ce modèle aux données d'entraînement et affiche les paramètres du modèle ajusté, offrant ainsi une vue détaillée des paramètres utilisés pour former le modèle K-means.

```
# Sets up the kMeans object
km = KMeans(n_clusters=3, random_state=42, n_init=10)

# Fits the model to the data
km.fit(X_train)

# Displays the parameters of the fitted model
km.get_params()
```

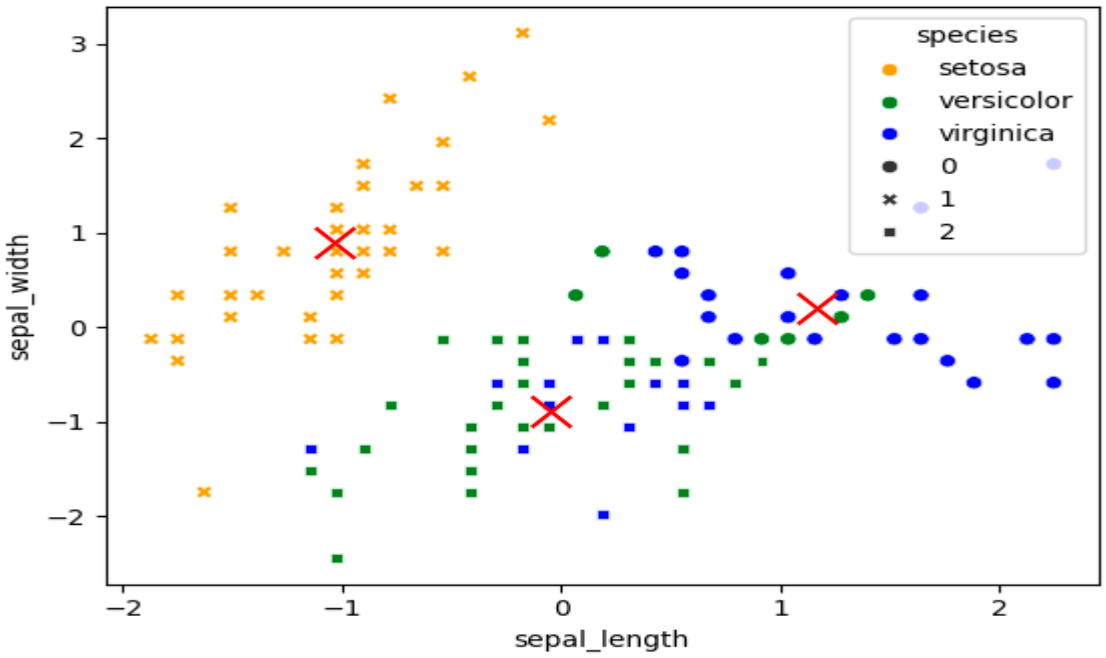
```
{'algorithm': 'lloyd',
 'copy_x': True,
 'init': 'k-means++',
 'max_iter': 300,
 'n_clusters': 3,
 'n_init': 13,
 'random_state': 42,
 'tol': 0.0001,
 'verbose': 0}
```



Ce code utilise `GridSearchCV` pour trouver la meilleure valeur du paramètre `n_init` pour le modèle k-means, puis imprime les meilleurs paramètres trouvés.

```
# Creates a scatter plot
sns.scatterplot(
    x='sepal_length',
    y='sepal_width',
    data=X_train,
    hue=y_train,
    style=km.labels_,
    palette=["orange", "green", "blue"])

# Adds cluster centers to the same plot
plt.scatter(
    km.cluster_centers_[:,0],
    km.cluster_centers_[:,1],
    marker='x',
    s=200,
    c='red')
```



Ce code crée un graphique de dispersion (scatter plot) en utilisant les données d'apprentissage `X_train`, où les coordonnées des points sont représentées par la longueur et la largeur des sépales (`sepal_length` et `sepal_width`). Les points sont colorés en fonction des étiquettes de classe `y_train`, et différents styles sont utilisés pour représenter les clusters identifiés par le modèle `km`. De plus, les centres de cluster calculés par le modèle `km` sont ajoutés au même graphique, marqués par des croix rouges.