Rapport : Models non-supervisés /supervisés: K-means /Apriori

Réalisé par : dakki Boutayna

Prof : Anouar Riad Solh

Master : IPSS

APPLICATION DE L'ALGORITHME K-MEAN :

1. Charger les données et normalisez-les.

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

df = pd.read_excel("C:/Users/pc/Desktop/BD_Kmeans.xlsx")
X = df[['Revenu Annuel', 'Dépenses Annuelles']]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

2. Appliquez l'algorithme K-Means avec k=3.

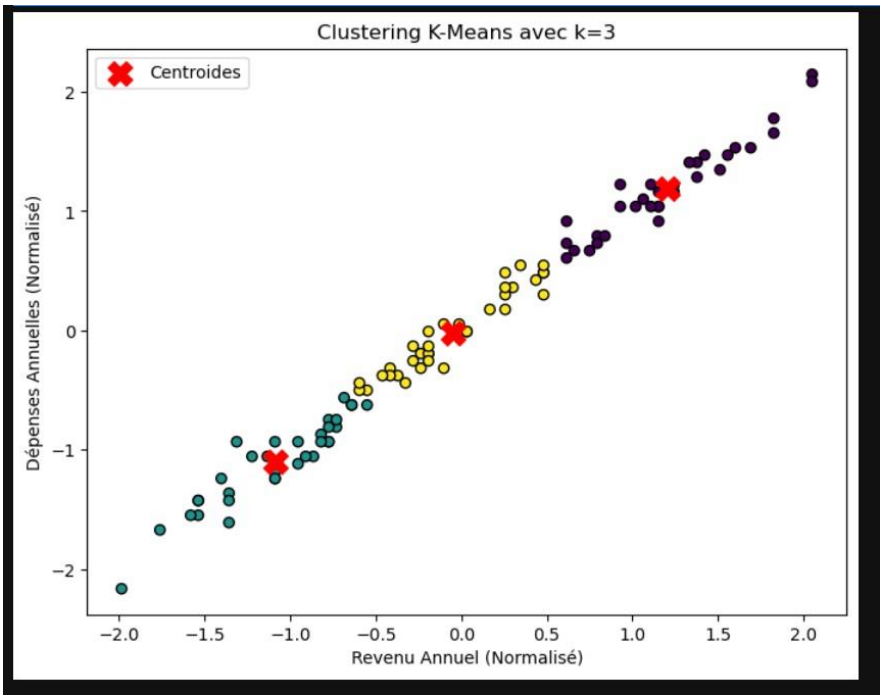
```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3, algorithm='elkan', random_state=42,
    n_init=10)
df['Cluster'] = kmeans.fit_predict(X_scaled)
```

3. Visualisez les clusters obtenus.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=df['Cluster'], cmap='
    viridis', edgecolors='k')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,
     1], s=200, c='red', marker='X', label="Centroides")
plt.xlabel("Revenu Annuel (Normalisé)")
plt.ylabel("Dépenses Annuelles (Normalisé)")
plt.title("Clustering K-Means avec k=3")
plt.legend()
plt.show()
```

Résultat attendu : Un graphique où chaque point représente un client avec son revenu annuel et ses dépenses annuelles, les couleurs indiquent les clusters, et les croix rouges représentent les centroïdes des groupes.



4. Analysez les résultats et interprétez les groupes formés.

   Après l'application de K-Means avec k=3, nous obtenons généralement trois segments de clients :

   - Cluster 1:
     - Revenus et dépenses faibles
     - Clients économes ou à budget limité
   - Cluster 2:
     - Revenus et dépenses moyennes
     - Dépenses proportionnelles aux revenus
   - Cluster 3:
     - Revenus et dépenses élevés
     - Clients premium avec forte capacité d'achat

APPLICATION DE L'ALGORITHME APRIORI :

L'algorithme Apriori suit trois étapes clés pour identifier les règles d'association dans un dataset :

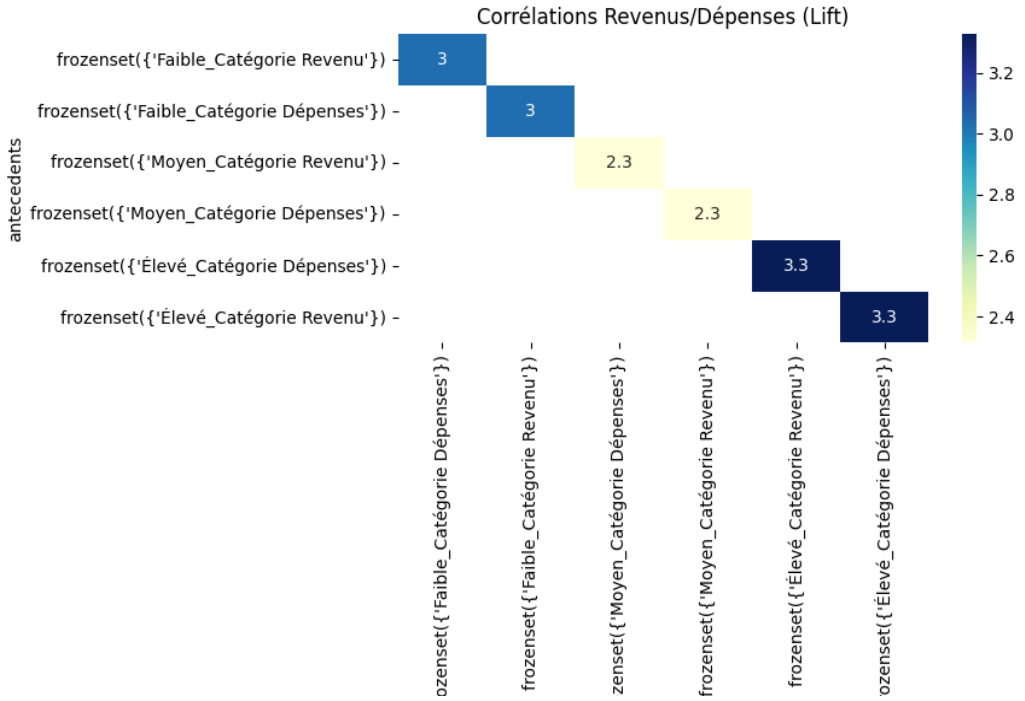
Génération des itemsets fréquents : Il explore les combinaisons d'items ({Revenu Élevé, Dépenses Élevées}) en calculant leur support (fréquence dans les transactions). Seuls les itemsets dépassant un seuil prédéfini (min_support=0.1) sont retenus.

Elagage des candidats : Il élimine les combinaisons redondantes ou peu fréquentes en utilisant la propriété "si un itemset est rare, tous ses sur-ensembles le sont aussi".

Création des règles d'association : À partir des itemsets fréquents, il génère des règles de type X → Y (Revenu Élevé → Dépenses Élevées) et évalue leur pertinence via :

Confiance : Probabilité que Dépenses Élevées soit présent si Revenu Élevé est observé (85%).

Lift : Corrélation entre Revenu Élevé et Dépenses Élevées (>1 = association positive).



Interprétation des résultats :

Un support élevé 20% indique une cooccurrence fréquente des items.

Une confiance élevée 80% suggère une règle fiable pour des stratégies ciblées offres premium pour les clients à haut revenu.

Un lift = 2.5 >1 révèle une synergie entre les items, utile pour optimiser les recommandations ou les stocks.

Ces métriques guident une prise de décision data-driven, comme prioriser les segments clients rentables ou anticiper les comportements d'achat.