


Module : Analyse et fouille de données	
Responsable du Cours : Bouaziz Souhir Enseignants TP : Bedoui Abir	Auditoire : D-LSI-ADBD A-U: 2024-2025
Révision	

Exercice 1 : Analyse de données patients

Nous avons récupéré les données de la base « **patients.csv** » dans une DataFrame appelée **df** afin de pouvoir effectuer les traitements nécessaires.

Il s'agit de résumer l'information contenue dans un fichier décrivant (**n = 10**) patients ayant suivi un bilan médical à l'aide de (**p = 7**) variables.

Un aperçu des données est donné :

Patient	Age	Poids (kg)	Taille (cm)	Tension_systole (mmHg)	Tension_diastole (mmHg)	Cholesterol (mg/dL)	Groupe_risque
1	45	NAN	175	140	90	220	élevé
2	34	65	168	120	80	190	moyen
3	50	90	180	150	95	250	élevé
4	29	70	172	110	70	180	faible
5	40	75	178	130	85	200	moyen

- 1) Télécharger dans votre notebook la base de données 'patients.csv'
- 2) Effectuer le chargement des données à partir d'un fichier CSV dans une DataFrame appelée **df**
- 3) Afficher les 6 premier lignes du df
- 4) Afficher les 4 derniers lignes du df
- 5) Remplacer les NAN par la moyenne de la variable
- 6) Standardiser (Centrer et Réduire) les données
- 7) Pourquoi est-il important de centrer et réduire ces données avant l'ACP ?
- 8) Vérifier que notre matrice des données obtenue est bien centrée et réduite
- 9) Réalisé un ACP normé avec PCA et donner la présentation graphique de la cumulation des variances expliquées
- 10) Quelles sont les composantes à retenir afin de conserver 98% de l'information ?
- 11) Donner le cercle de corrélation
 - a) Que représente l'axe des abscisses de la cercle ?
 - b) Quelles sont les deux variables les plus corrélées à l'axe des abscisses ?
 - c) Quelles sont les deux variables les moins corrélées entre eux ?

12) Appliquer K-means pour diviser les données en 3 groupes

13) Donner une définition de k-means

Annexe :

```
df = df.fillna(df.mean())

# 2. Normaliser les données
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)
data_scaled

# 3. Appliquer l'ACP
pca = PCA()
pca.fit(data_scaled)

# 4. Calculer la variance cumulée expliquée
explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)

# 5. Graphique de la variance cumulée expliquée
plt.figure(figsize=(8, 5))
plt.plot(range(1, len(cumulative_variance) + 1), cumulative_variance, marker='o',
color='r', alpha=0.7)
plt.xlabel('Nombre de composantes principales')
plt.ylabel('Variance cumulée expliquée')
plt.title('Variance cumulée expliquée par les composantes principales')
plt.grid(True)
plt.show()

# Récupérer les loadings
loadings = pca.components_

# Cercle de corrélation
plt.figure(figsize=(8, 8))
ax = plt.gca()

# Tracer les flèches et les étiquettes
for i, var in enumerate(data.columns):
    ax.arrow(0, 0, loadings[0, i], loadings[1, i], head_width=0.03, fc='r', ec='r')
    ax.text(loadings[0, i]*1.1, loadings[1, i]*1.1, var, ha='center', va='center')
```

```

# Tracer le cercle unité
ax.add_artist(plt.Circle((0, 0), 1, color='b', fill=False, linestyle='--'))
# Configurer l'affichage
plt.xlim(-1.2, 1.2)
plt.ylim(-1.2, 1.2)
plt.axhline(0, color='black', linewidth=0.5)
plt.axvline(0, color='black', linewidth=0.5)
plt.gca().set_aspect('equal')
plt.title('Cercle de corrélation')
plt.grid(True)
plt.show()

# 3. Appliquer K-means
kmeans = KMeans(n_clusters=3, random_state=42) # Choix de 3 clusters
kmeans.fit(data_scaled)

# 4. Extraire les labels (appartenance aux clusters) et les centres des clusters (centroïdes)
labels = kmeans.labels_
centroids = kmeans.cluster_centers_

# 5. Visualisation des clusters et des points
plt.figure(figsize=(6, 6))

# Tracer les points de données avec différentes couleurs selon le cluster
for i in range(3):
    # Sélectionner les points qui appartiennent à chaque cluster
    cluster_data = data_scaled[labels == i]
    # Tracer chaque cluster avec une couleur différente
    plt.scatter(cluster_data[:, 0], cluster_data[:, 1], label=f'Cluster {i + 1}', s=100,
alpha=0.6)

# Tracer les centres des clusters
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', marker='X', s=200, label='Centres des
clusters')

# Ajouter des étiquettes aux points
for i, txt in enumerate(data.index):
    plt.annotate(txt, (data_scaled[i, 0], data_scaled[i, 1]), fontsize=12, alpha=0.7)

# Ajouter des légendes, titres et axes

```

```
plt.title('Clustering K-means avec les centres des clusters et étiquettes des points',
fontsize=16)
plt.xlabel('Variable 1 (ex: Age)', fontsize=14)
plt.ylabel('Variable 2 (ex: Poids)', fontsize=14)
plt.legend(loc='upper right')
plt.grid(True)
plt.show()
```

Exercice2 : K-means & CAH

K-means

On compte représenter 8 objets {A,B,C,D,E,F,G,H} en se basant sur 2 critères X et Y tels que représentés dans le tableau suivant:

1/ Appliquer K-means pour diviser les objets en 3 groupes, ci-dessous l'initialisation des clusters :

$C1=\{A,C,E,G\}$

$C2=\{B,D\}$

$C3=\{F,H\}$

Objets	X	Y
A	2	10
B	2	5
C	8	4
D	5	8
E	7	5
F	6	4
G	1	2
H	4	9

CAH

1/ Dressez la matrice des distances

2/ Utiliser la méthode de classification ascendante hiérarchique (CAH) pour tracer le dendrogramme en se basant sur la méthode d'agrégation de saut minimum.