UNIVERSITE ABDELMALEK ESSAADI
FACULTE DES SCIENCES ET TECHNIQUES DE TANGER
DEPARTEMENT GENIE INFORMATIQUE

PROJET DE FIN DE MODULE

"EM-Algorithme"

Réalisé par :

- **SAMADY Ahmed**
- **FAKHRE-EDDINE MOHAMED AMINE**
- **CHIBANI Fahd**

Encadré par :

- **Pr. Abdelaziz Assadouq**
- **Pr. Tarik Amtout**

Année universitaire 2023/2024

Table des matières

Sigles et Abréviations

Sigles	Significations
GMM	Gaussian Mixture Model
EMV	Estimateur de Maximum Vraisemblance
EM	Expectation-Maximization
MAP	Maximum A Posteriori

Tables des Figures

1. INTRODUCTION :

L'algorithme Expectation-Maximization (EM) est une méthode pour estimer les paramètres dans des modèles avec des variables non observées. Des exemples classiques d'applications se trouvent dans le clustering basé sur des modèles de mélange Gaussien. EM signifie Expectation-Maximization, et il décrit une méthode itérative qui maximise une valeur attendue à chaque itération.

2. MODÈLE DE MÉLANGE GAUSSIEN :

Lorsqu'une valeur est observée, une variable latente implicite x_i détermine de quelles gaussiennes (neighbourhood) elle provient.

Dans cet exemple de fixation du prix des logements, décrivons le prix de chaque immobilier comme une variable aléatoire à valeur réelle x_i et le quartier non observé dont il appartient comme une variable aléatoire à valeur discrète z_i. Dans l'hypothèse de k quartiers, z_i peut être représenté comme une distribution catégorielle avec un paramètre $\pi = [\pi_1, \dots, \pi_k]$, et la distribution des prix du $k^{éme}$ quartier comme une gaussienne $\mathcal{N}(\mu_k, \sigma_k^2)$ avec une moyenne μ_k et une variance σ_k^2. La densité des x_i est donc donnée par :

$$p(x_i|\theta) = \sum_{k=1}^{K} p(z_i = k)p(x_i|z_i = k, \mu_k, \sigma_k^2)$$

$$x_i|z_i \sim \mathcal{N}(\mu_k, \sigma_k^2) \qquad (\mathbf{1})$$

$$z_i \sim \text{Categorical}(\pi)$$

θ représente les paramètres gaussiens μ_k, σ_k^2 et les variables catégorielles π. Les poids de mélange précédents des quartiers, désignés par π, indiquent leur pourcentage relatif si aucune autre information n'est connue. Pour obtenir la densité des variables observées x_i , nous devons marginaliser les variables non observées z_i. Pour simplifier l'équation 1, nous modélisons la distribution des prix de chaque logement comme une combinaison linéaire ("modèle de mélange") de nos K gaussiennes (quartiers).

Nous disposons à présent d'une poignée de problèmes d'inférence pertinents, compte tenu de différentes hypothèses :

1. Pour calculer la responsabilité d'une cluster K par rapport à un point i, utilisez la formule :

$$r_{ik} = p(z_i = k|x_i, \theta).$$

Cette formule vous indique dans quelle mesure un lieu est "susceptible" ou "proche" d'un cluster connu. Ce calcul peut être appliqué aux classificateurs GMM pour déterminer la classe la plus probable pour x_i, ainsi qu'à la méthode EM.

2. Estimation des paramètres gaussiens μ_k, σ_k^2et des variables catégorielles π à l'aide d'informations fournies :

a. Uniquement les points observés x_i;

b. Les positions observées x_i et les valeurs des variables latentes z_i.

3. L'ALGORITHME-EM :

L'algorithme de maximisation des attentes (EM) est une méthode itérative permettant de trouver l'estimation EMV ou MAP pour les modèles à variables latentes. Voici une description du fonctionnement de l'algorithme à 10 000 pieds d'altitude :

0. ***Initialisation :*** Obtenir une estimation initiale des paramètres θ^0 (par exemple, tous les paramètres μ_k, σ_k^2et π). Dans de nombreux cas, il peut s'agir d'une initialisation aléatoire.
1. ***Étape d'espérance :*** Supposons que les paramètres (θ^{t-1}) de l'étape précédente sont fixes, calculer les valeurs attendues des variables latentes (ou plus souvent une fonction des valeurs attendues des variables latentes).
2. ***Étape de maximisation :*** Étant donné les valeurs calculées à la dernière étape (essentiellement des valeurs connues pour les variables latentes), estimez de nouvelles valeurs pour θ^t qui maximisent une variante de la fonction de vraisemblance.
3. ***Condition de sortie :*** Si la vraisemblance des observations n'a pas beaucoup changé, quitter ; sinon, revenir à l'étape 1.

Les étapes 2 et 3 présentent l'avantage d'être faciles à calculer de manière séquentielle, car nous n'essayons pas de déterminer les variables latentes et les paramètres du modèle en même temps. Nous montrerons plus tard que chaque itération de l'algorithme augmentera la fonction de vraisemblance, mais comme elle n'est pas convexe, nous ne sommes assurés que d'approcher un maxima local. Une façon de contourner ce problème consiste à exécuter l'algorithme pour plusieurs valeurs initiales afin d'obtenir une couverture plus large de l'espace des paramètres.

4. MODELE MATHEMATIQUE DE L'ALGORITHME:

4.1 Log-vraisemblance et fonction auxiliaire :

Rappelons que l'objectif global de l'algorithme EM est de trouver une estimation EMV (ou MAP) dans un modèle avec des variables latentes non observées. Par définition, les estimations EMV tentent de maximiser la fonction de vraisemblance. Dans le cas général, avec des observations x_i et des variables latentes z_i nous avons la log-vraisemblance suivante :

$$l(\theta) = \sum_{i=1}^{N} \log p(x_i|\theta) = \sum_{i=1}^{N} \log \sum_{z_i} p(x_i, z_i|\theta) \qquad (2)$$

La première expression est simplement la définition de la fonction de vraisemblance (la probabilité que les données correspondent à un ensemble donné de paramètres). La deuxième expression montre que nous devons marginaliser (intégrer si elle était continue) la variable latente non observée z_i . Malheureusement, cette expression est difficile à optimiser car nous ne pouvons pas "pousser" le log

dans la sommation. L'algorithme EM contourne ce problème en définissant une quantité appelée fonction de log-vraisemblance des données complètes (nous expliquerons pourquoi cela fonctionne plus tard) :

$$l_c(\theta) = \sum_{i=1}^{N} \log p(x_i, z_i|\theta) = \sum_{i=1}^{N} \log[p(z_i|\theta)p(x_i|z_i, \theta)] \quad (\mathbf{3})$$

Là encore, il n'est pas possible de le calculer car nous n'observons jamais les valeurs z_i. Cependant, nous pouvons prendre la valeur attendue de l'équation 3 par rapport à la distribution conditionnelle des z_i compte tenu des données et de notre valeur précédente des paramètres, θ^{t-1}. C'est un peu long, alors laissez-moi vous expliquer d'une autre manière.

Prendre l'espérance est utile parce qu'à chaque fois que nous avons besoin de connaître explicitement la valeur de z_i non observé, nous pouvons utiliser sa valeur attendue. Ainsi, toutes les valeurs inconnues de z_i sont "remplies" et ce qui nous reste est une fonction uniquement des valeurs que nous voulons maximiser, c'est-à-dire θ. Maintenant, la mise en garde est que lors du calcul de la valeur attendue de z_i nous utilisons la distribution conditionnelle, $p(z_i|x_i, \theta^{t-1})$ sur les données et les valeurs précédentes des paramètres. C'est ainsi que nous entrons dans la nature itérative de l'algorithme EM. Jetons un coup d'œil aux mathématiques et décomposons-les.

Nous prenons d'abord l'espérance de l'équation 3 par rapport à la distribution conditionnelle des z_i compte tenu des données et de notre valeur précédente des paramètres, que nous définissons comme la fonction auxiliaire $Q(\theta, \theta^{t-1})$:

$$Q(\theta, \theta^{t-1}) = E[l_c(\theta)|\mathcal{D}, \theta^{t-1}] = \sum_{i=1}^{N} E[\log [p(z_i|\theta)p(x_i|z_i, \theta)]] \quad (\mathbf{4})$$

où $\mathcal{D}$ représente toutes nos données (en supprimant le conditionnement dans la deuxième expression pour rendre la notation un peu plus claire). Rappelons que nous prenons l'espérance sur la probabilité conditionnelle, ce qui se traduit par $E[l_c(\theta)|\mathcal{D}, \theta^{t-1}] = \sum_{k'=1}^{K} l_c(\theta) \cdot p(z_i = k'|\mathcal{D}, \theta^{t-1})$. Il est important de s'en souvenir car la notation va devenir un peu déroutante et nous devons garder à l'esprit quels termes sont constants par rapport à l'espérance.

À ce stade, la manière dont l'espérance est évaluée n'est pas claire du tout. Il existe une petite astuce pratique que nous pouvons utiliser, en supposant que nos variables latentes z_i sont discrètes (ce qui est généralement le cas lorsque nous utilisons l'algorithme EM). D'après l'équation 4 :

$$\begin{aligned} Q(\theta, \theta^{t-1}) &= \sum_{i=1}^{N} E[\log[p(z_i|\theta)p(x_i|z_i, \theta)]] \\ &= \sum_{i=1}^{N} E[\log \Pi_{k=1}^{K}[p(z_i = k|\theta)p(x_i|z_i = k, \theta)]^{I(z_i=k)}] \\ &= \sum_{i=1}^{N} E[\sum_{k=1}^{K} \log\big[p(z_i = k|\theta)p(x_i|z_i = k, \theta)\big]^{I(z_i=k)}] \end{aligned}$$

(5)

où $I(z_i = k)$ est la fonction indicatrice. Cette petite astuce est un peu compliquée, mais elle n'est pas si difficile à comprendre.

Rappelons que dans l'équation 3, nous ne pouvions évaluer aucune des valeurs z_i directement parce qu'elles n'ont pas été observées. En d'autres termes, nous ne savions pas si $z_i = 1$ ou $z_1 = 2$ etc., ce qui serait connu s'il était observé. Cette astuce utilise la fonction d'indicateur pour agir comme un "filtre" pour les produits sur k, en éliminant la valeur exacte de z_i (si elle était connue). La raison pour laquelle nous utilisons cette astuce est qu'elle décompose les variables non observées z_i en probabilités (par exemple $p(z_i = k|\theta), p(x_i|z_i = k, \theta)$ qui peuvent être évaluées et une simple fonction d'une variable aléatoire $I(z_i = k)$. Les premières seront uniquement des fonctions de nos paramètres à maximiser (par exemple θ), tandis que les secondes peuvent être considérées comme des espérances.

Nous pouvons simplifier un peu plus l'équation 5 :

$$
\begin{aligned}
Q(\theta, \theta^{t-1}) &= \sum_{i=1}^{N} E[\sum_{k=1}^{K} \log [p(z_i = k|\theta)p(x_i|z_i = k, \theta)]^{I(z_i=k)}] \\
&= \sum_{i=1}^{N} E[\sum_{k=1}^{K} I(z_i = k) \log [p(z_i = k|\theta)p(x_i|z_i = k, \theta)]] \\
&= \sum_{i=1}^{N} \sum_{k=1}^{K} E[I(z_i = k)] \log [p(z_i = k|\theta)p(x_i|z_i = k, \theta)] \\
&= \sum_{i=1}^{N} \sum_{k=1}^{K} p(z_i = k|\mathcal{D}, \theta^{t-1}) \log [p(z_i = k|\theta)p(x_i|z_i = k, \theta)]
\end{aligned}
\tag{6}
$$

Remarquez que l'espérance n'est effectuée que sur la fonction indicatrice, alors que les énoncés de probabilité dans le logarithme sont facilement évalués en fonction des seuls paramètres.

En résumé, la boucle EM vise à maximiser la log-vraisemblance attendue des données complètes, ou fonction auxiliaire $Q(\theta, \theta^{t-1})$ en deux étapes :

1. Étant donné les paramètres θ^{t-1} de l'itération précédente, évaluer la fonction Q de façon qu'elle soit uniquement en termes de θ.
2. Maximiser cette fonction Q simplifiée en termes de θ. Ces paramètres deviennent le point de départ de l'itération suivante.

Nous verrons comment cela se passe explicitement avec les GMM dans la section suivante.

L'autre question que vous vous posez peut-être est de savoir pourquoi nous définissons cette fonction $Q(\theta, \theta^{t-1})$? Il s'avère que l'amélioration de la fonction Q n'entraînera jamais de perte dans notre fonction de vraisemblance. Par conséquent, la boucle EM devrait toujours améliorer notre fonction de vraisemblance (jusqu'à un maximum local).

4.2 EM pour les modèles de mélanges gaussiens :

En partant de l'équation 6, nous obtenons la plupart des résultats de l'EM pour le GMM, en procédant à quelques réarrangements :

$$\begin{aligned} Q(\theta, \theta^{t-1}) &= \sum_{i=1}^{N} \sum_{k=1}^{K} p(z_i = k|\mathcal{D}, \theta^{t-1}) \log [p(z_i = k|\theta) p(x_i|z_i = k, \theta)] \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} [r_{ik} \log\, p(z_i = k|\theta) + r_{ik} \log\, p(x_i|z_i = k, \theta)]] \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} [r_{ik} \log\, \pi_k + r_{ik} \log [\frac{1}{\sqrt{2\sigma_k^2 \pi}} \exp (\frac{-(x_i - \mu_k)^2}{2\sigma_k^2})]] \end{aligned} \tag{7}$$

Remarquez que l'équation 7 ne s'exprime qu'en termes de nos paramètres, π_k, μ_k, σ_k. L'algorithme EM pour les GMM se résume donc à calculer d'abord r_{ik} (en utilisant les paramètres de l'itération précédente θ^{t-1}) afin de définir notre fonction $Q(\theta, \theta^{t-1})$ soit définie, puis de la maximiser :

$$\begin{aligned} \theta^t &= \underset{\theta}{\text{argmax}}\ Q(\theta, \theta^{t-1}) \\ &= (\boldsymbol{\pi}^t, \boldsymbol{\mu}^t, \boldsymbol{\sigma}^t) \end{aligned} \tag{8}$$

Vous pouvez dériver des expressions pour celles-ci à partir des premiers principes de l'équation 8 en examinant simplement les estimations de l'EMV pour la distribution multinomiale (avec $n = 1$) et la distribution gaussienne. La distribution normale devrait être une simple application de la prise du gradient, mais la distribution multinomiale est un peu plus compliquée en raison de la contrainte supplémentaire selon laquelle les $\sum_k \pi_k = 1$.

5. AVANTAGES ET INCONVENIENTS DE EM:

5.1 Avantages :

- Augmentation garantie de la vraisemblance $P_{\theta_t}(x) \geq P_{\theta_{t-1}}(x)$.
- Soft Clustering.

5.2 Inconvénients :

- Aucune garantie qu'il donne θ_{EMV} .
- Il se peut qu'il converge vers des maximas locaux.
- La convergence peut être lente (dépend de la base des exemples).
- Dépendance sur l'initialisation des paramètres : nécessite souvent plusieurs répétitions à partir de différents points d'initialisation.

6. ALGORITHME K-MEANS:

0. ***Initialisation :*** Nombre de clusters à former (K) La bases des exemples d'apprentissage
1. ***Début :*** Prendre K exemples aléatoirement des exemples de la base comme des centroïdes initiaux.
2. ***Répéter :*** Affecter chaque exemple au cluster du vecteur prototype le plus proche, et recalculer le vecteur prototype lié à chaque cluster Jusqu'à convergence (centroïdes fixes ou nombre d'itérations maximum atteint).

7. COMPARAISON ENTRE K-MEANS ET EM :

Nous allons faire une petite comparaison entre l'algorithme EM et l'algorithme K-means, car les deux algorithmes effectuent un apprentissage non supervisé (clustering). Le K-means est reconnu comme l'un des algorithmes les plus efficaces pour le clustering. De plus, il est important de noter que le K-means effectue un hard clustering tandis que l'EM effectue un soft clustering.

7.1 Datasets en question :

- ***Iris :*** Le jeu de données comprend 50 échantillons de chacune des trois espèces d'iris (Iris ***setosa***, Iris ***virginica*** et Iris ***versicolor***). Quatre caractéristiques ont été mesurées à partir de chaque échantillon : la longueur et la largeur des sépales et des pétales, en centimètres.

Les vrais clusters de ***Iris*** :

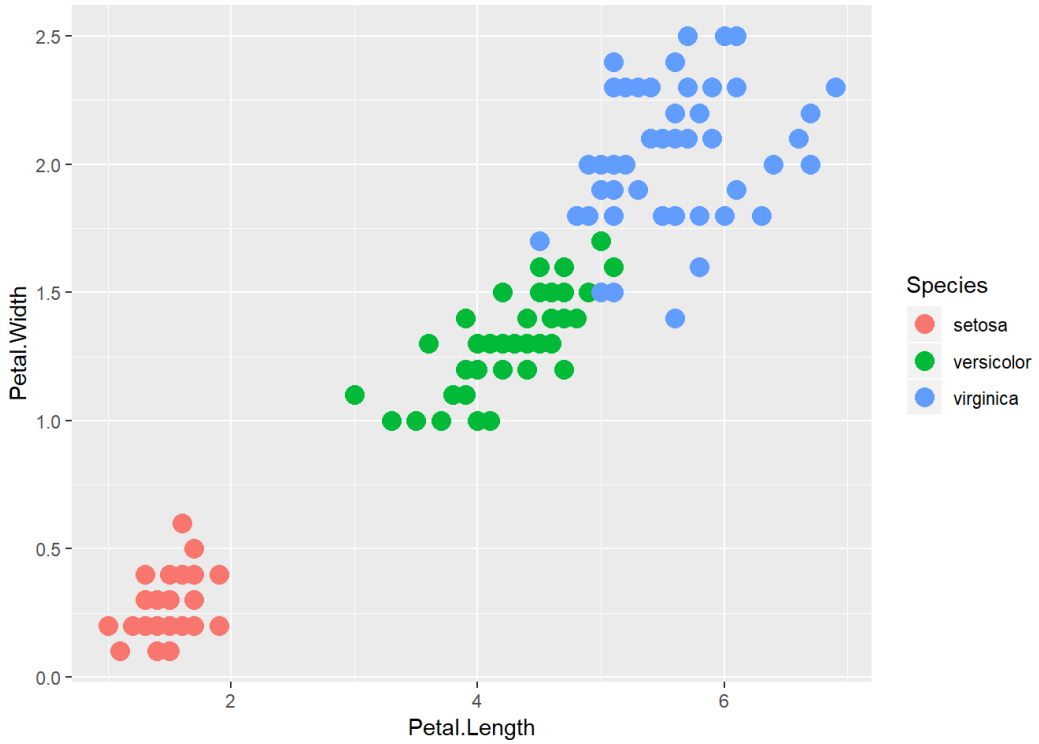

Figure 1: Les vraies cluster de Iris (PetalWidth vs PetalLength).

- ***Mall Customers :*** Vous êtes propriétaire d'un supermarché et, grâce aux cartes de membre, vous disposez de quelques données de base sur vos clients, telles que ***CustomerID***, ***age***, ***gender***, ***annual income*** et ***spending score***. Le ***spending score*** est une donnée que vous attribuez au client sur la base de paramètres définis tels que le comportement du client et les données d'achat.

7.2 Résultats :

Pour tous les tests, nous avons testé les algorithmes K-means et EM en R et en Python. Nous devons également noter que le nombre de centroïdes choisis est de 4 et 3 pour l'ensemble de données du centre commercial des clients et l'ensemble de données iris respectivement.

7.2.1 Iris clustering avec EM :

L'algorithme EM donne des résultats assez décents pour le clustering de l'ensemble de données iris, comme on peut le voir dans la **Figure 2**.

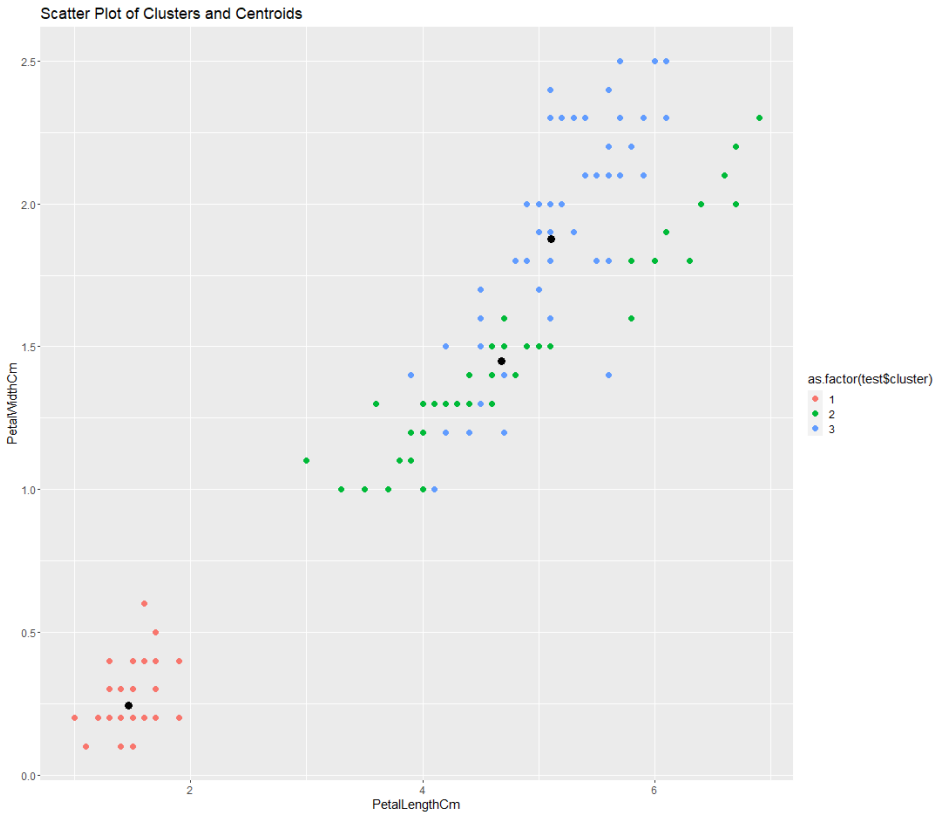

Figure 2: Clustering sur dataset iris avec EM

7.2.2 Iris clustering avec K-means:

L'algorithme K-means donne de très bons résultats pour le clustering de l'ensemble de données iris, comme on peut le voir dans la **Figure 3.**

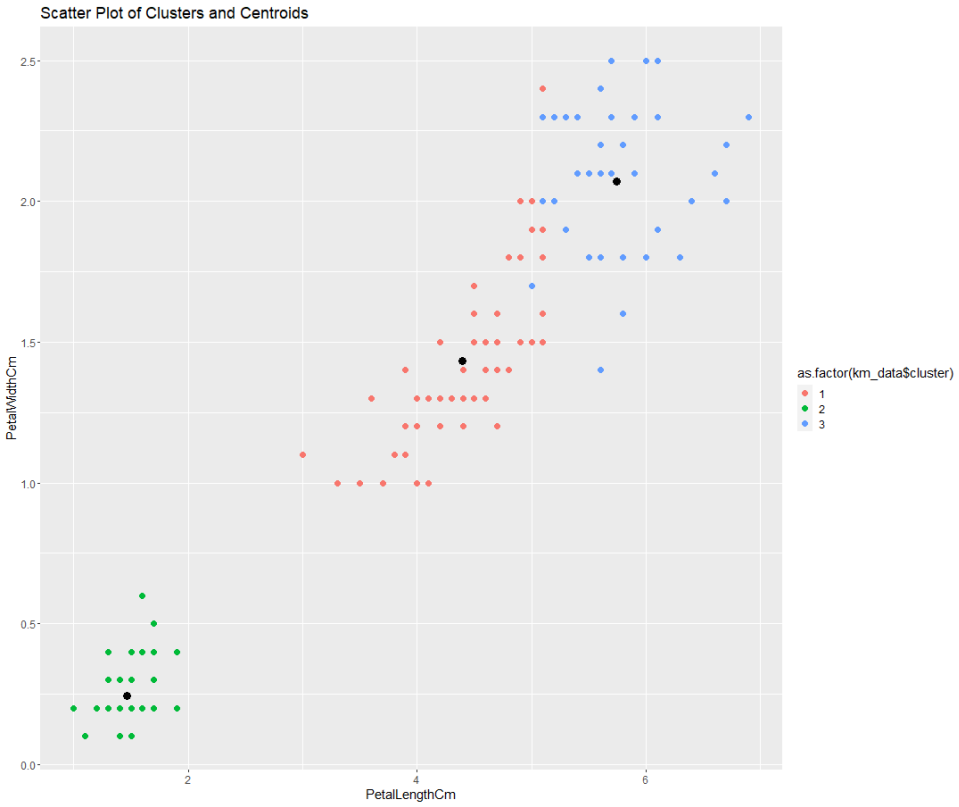

Figure 3: Clustering sur dataset iris avec K-means.

7.2.3 Mall Customers clustering avec EM:

Comme on peut le voir dans la **Figure 4**, l'algorithme EM ne parvient pas à trouver correctement les clusters pour l'ensemble de données de mall customers.

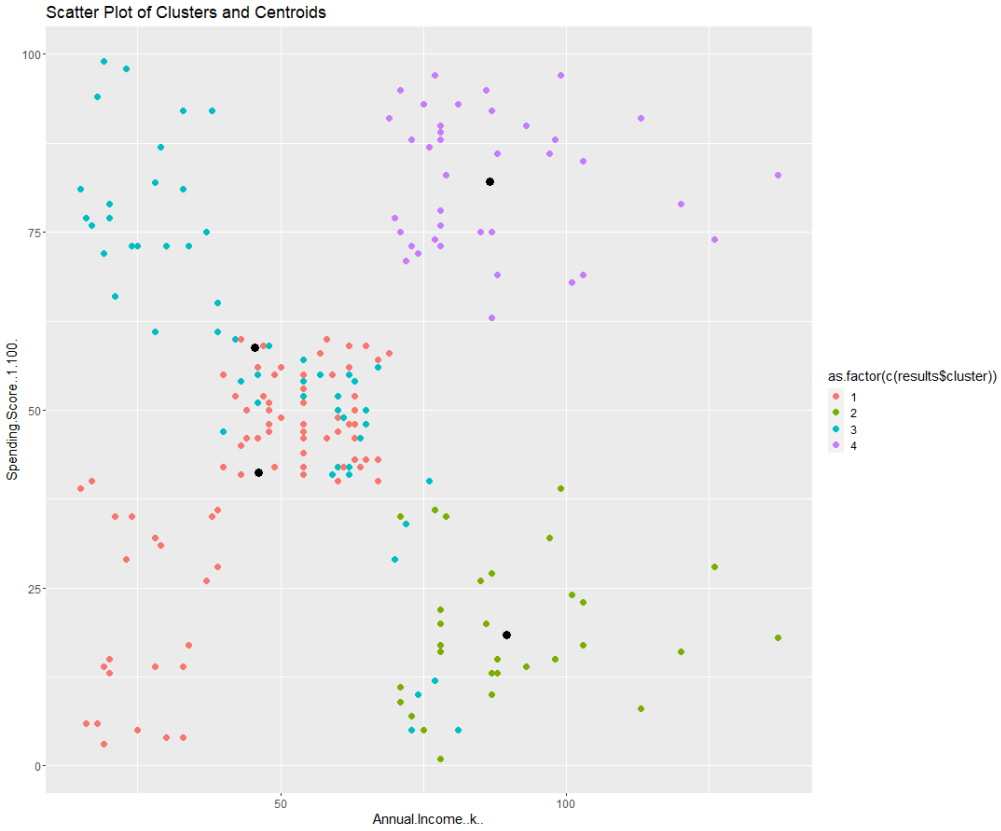

Figure 4: Clustering sur dataset mall avec EM.

7.2.4 Mall Customers clustering avec K-means:

L'algorithme K-means donne des résultats raisonnables pour le clustering de l'ensemble de données de mall customers, comme on peut le voir dans la **Figure 5**.

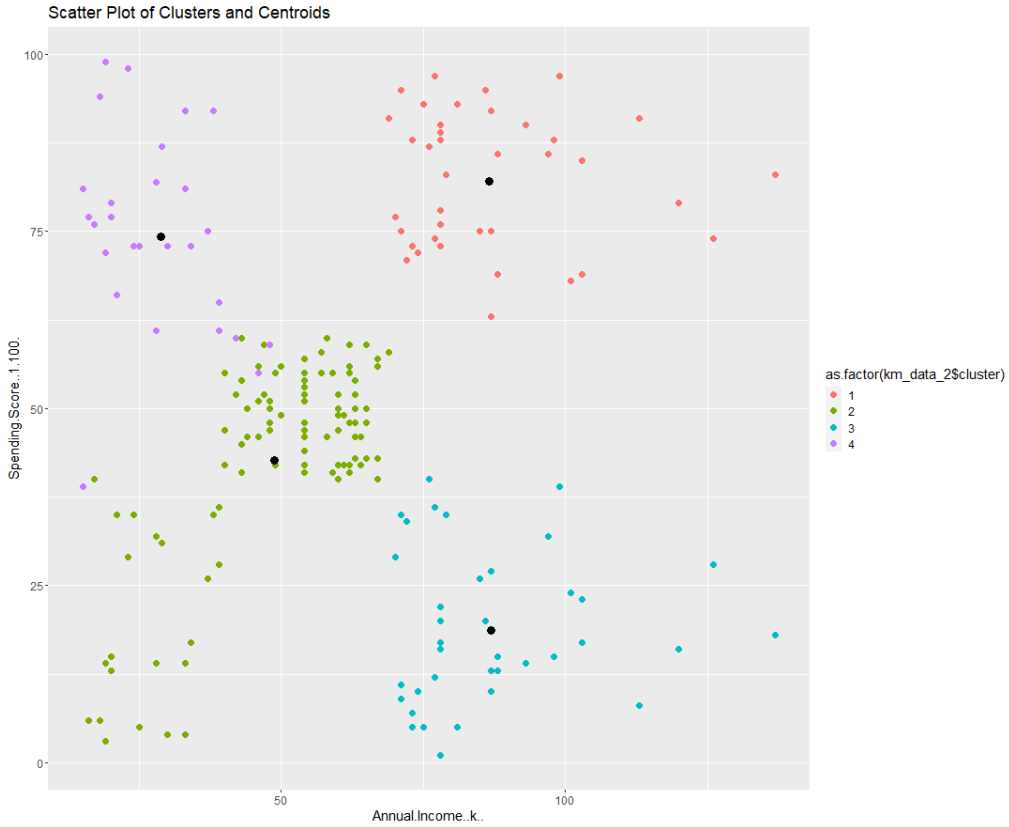

Figure 5: Clustering sur dataset mall avec K-means.

8. CONCLUSION :

L'algorithme EM (Expectation-Maximization) est une méthode probabiliste pour l'apprentissage non supervisé qui utilise des modèles de mélange gaussien. Dans notre analyse des clusters du jeu de données Iris, nous avons constaté que l'algorithme EM nécessite une bonne initialisation des paramètres. Cela confirme les inconvénients que nous avons mentionnés précédemment à propos de l'algorithme EM. Par conséquent, dans notre comparaison, l'algorithme K-means semble être plus précis pour le clustering que l'algorithme EM. Cela prouve les limites de l'algorithme EM qui est probablement dû à la convergence vers des maxima locaux.

REFERENCES ET BIOGRAPHIE

- *Martin Haugh. The EM Algorithm. Published 2015. https://www.columbia.edu/~mh2078/MachineLearningORFE/EM_Algorithm.pdf*
- *Henrik Hult. Lecture 8. https://www.math.kth.se/matstat/gru/Statistical%20inference/Lecture8.pdf*
- *Sean Borman. The Expectation Maximization Algorithm, A short tutorial. Published July 18, 2004. https://www.lri.fr/~sebag/COURS/EM_algorithm.pdf*
- *Tengyu Ma. and Andrew Ng. CS229 Lecture notes. Published May 13, 2019. https://cs229.stanford.edu/notes2020spring/cs229-notes8.pdf*
- *Keng B. The Expectation-Maximization Algorithm. Bounded Rationality. Published October 7, 2016. https://bjlkeng.io/posts/the-expectation-maximization-algorithm/*
- *Samashi. (n.d.). GitHub - Samashi47/EM-algorithm https://github.com/Samashi47/EM-algorithm*