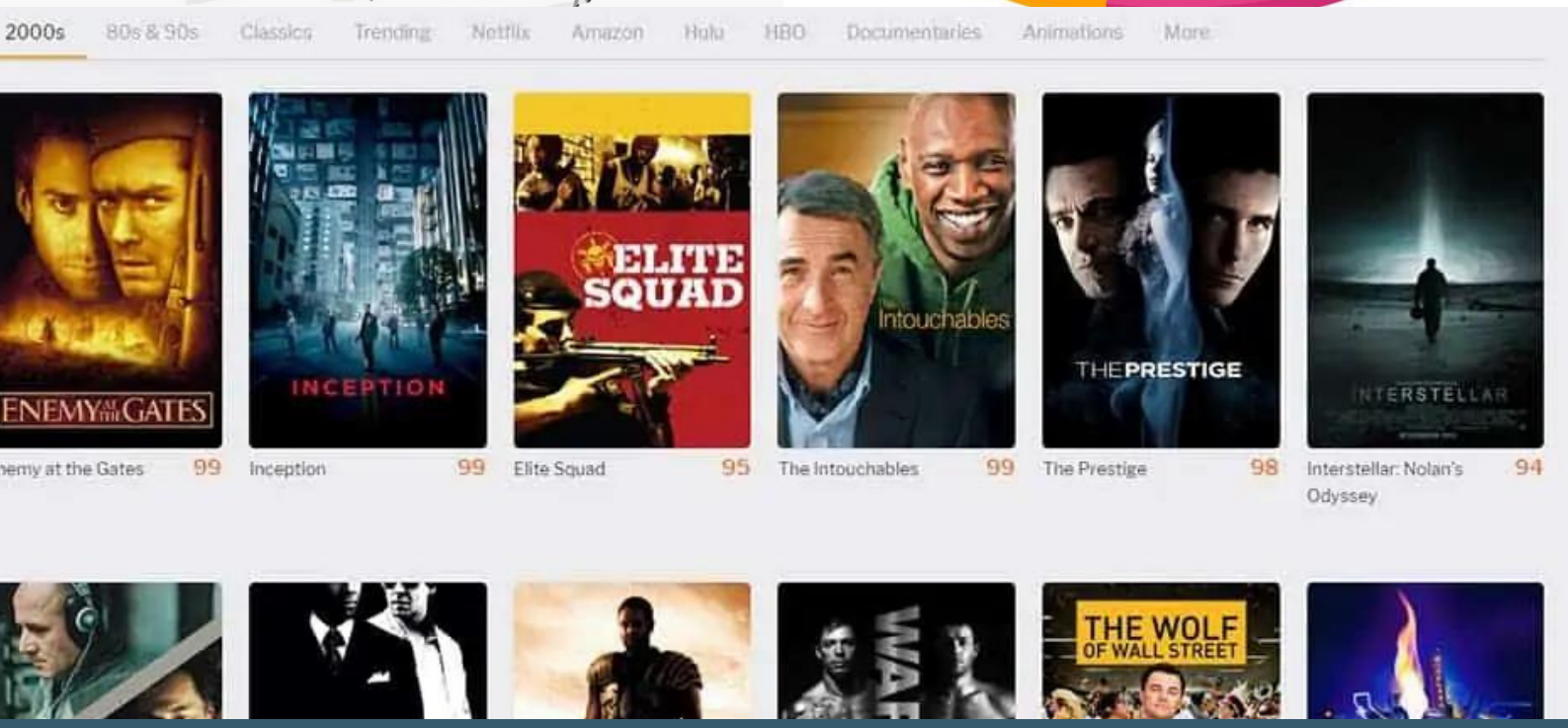



# Projet Algorithmique pour le Big Data

Ulrick BLAVIN - Amélie PSYCHE

2023

# Sommaire

# Introduction

**Aujourd'hui l'informatique est important dans tous les domaines, que ce soit médical, marketing, aéronautique etc. C'est ce que nous allons voir dans un domaine précis qui est le marketing.  Tout d'abord, nous allons faire de la segmentation client, nos données n'étant pas étiquetées nous allons chercher un comportement qui divise les personnes en plusieurs groupes. Afin de vérifier la structure qui en ressortira, nous ferons une classification pour classer une personnes dans l'un des groupes qui aura été défini. Puis, à partir d'une autre base de données nous allons faire de la recommandation de films, nous utiliserons le filtrage collaboratif où l'on pourra recommander des films à un utilisateur à partir de ce qu'il aura regardé avant ainsi que les notes qu'il aura donné.**

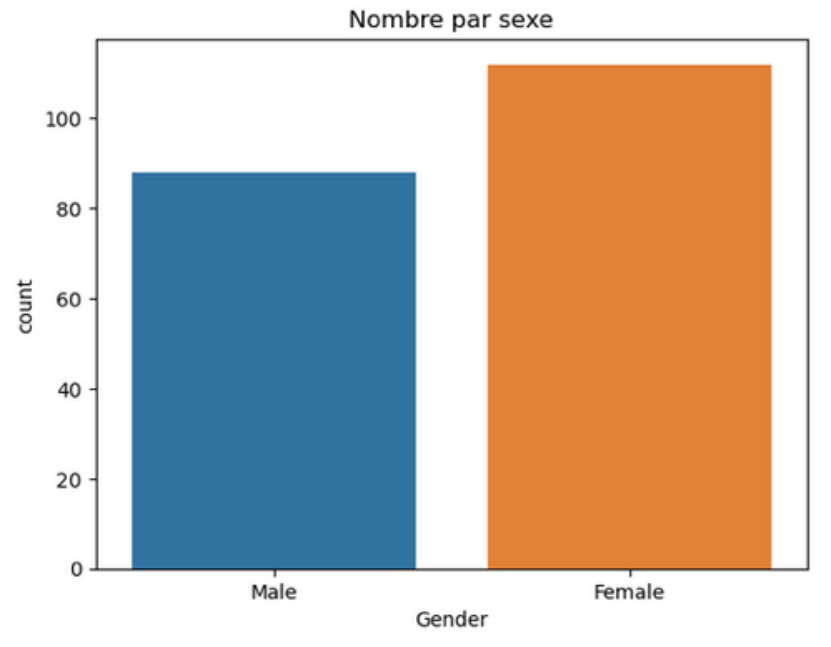
# Segmentation

Premier jeu de données

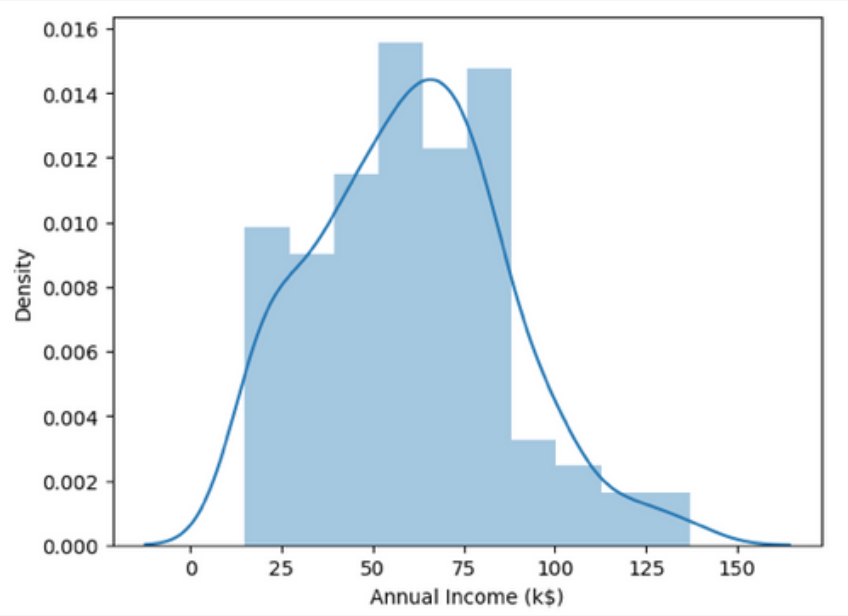
La segmentation client est une stratégie de marketing qui consiste à diviser un grand groupe de clients en sous-groupes plus petits ayant des besoins, des comportements d'achat et des caractéristiques similaires.
Pour ce faire, nous allons utiliser le fichier "Mall_Customers.csv" qui contient 5 colonnes. Nous avons :

- "CustomerID" est l'identifiant unique donnée au client
- "Gender" est le sexe du client (femme ou homme)
- "Age" est l'âge du client
- "Annual Income (k$)" est le revenu annuel du client (en milliers)
- "Spending Score (1-100)" est la note compris entre 1 et 100 que le centre commercial donne au client par rapport à ses dépenses

Pour s'approprier les données, nous faisons des visualisation la répartition.

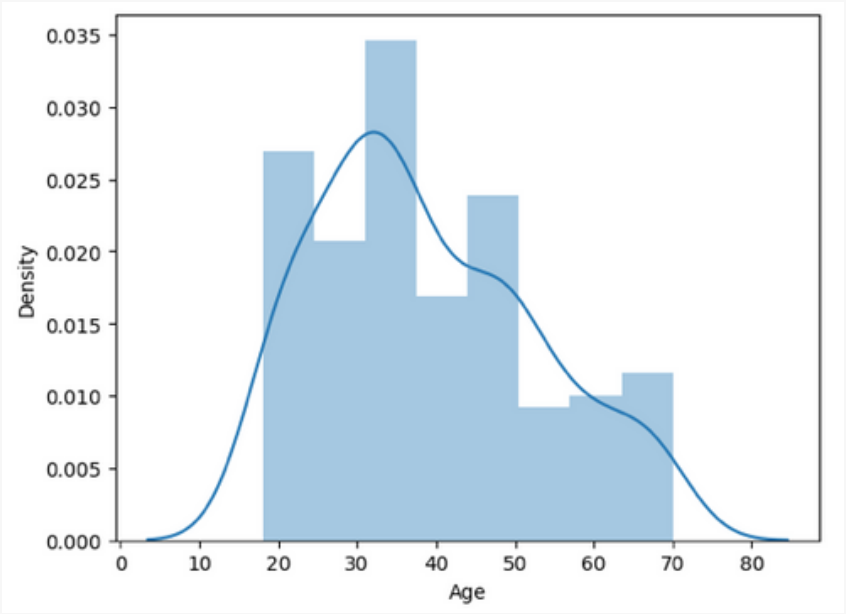


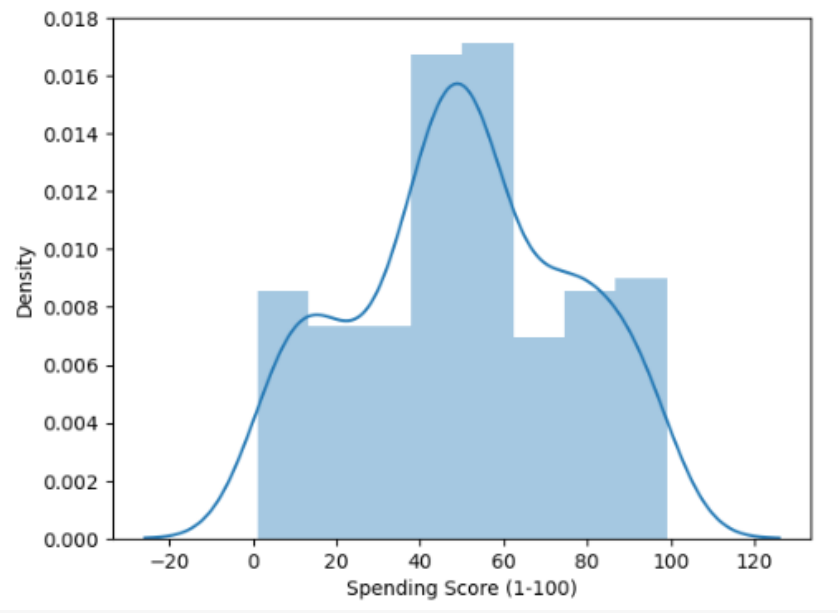

Sur cette visualisation, nous voyons la répartition des clients, et l'on constate qu'il y a plus d'hommes que de femmes. Il y a environ 90 femmes, et environ 110 hommes.



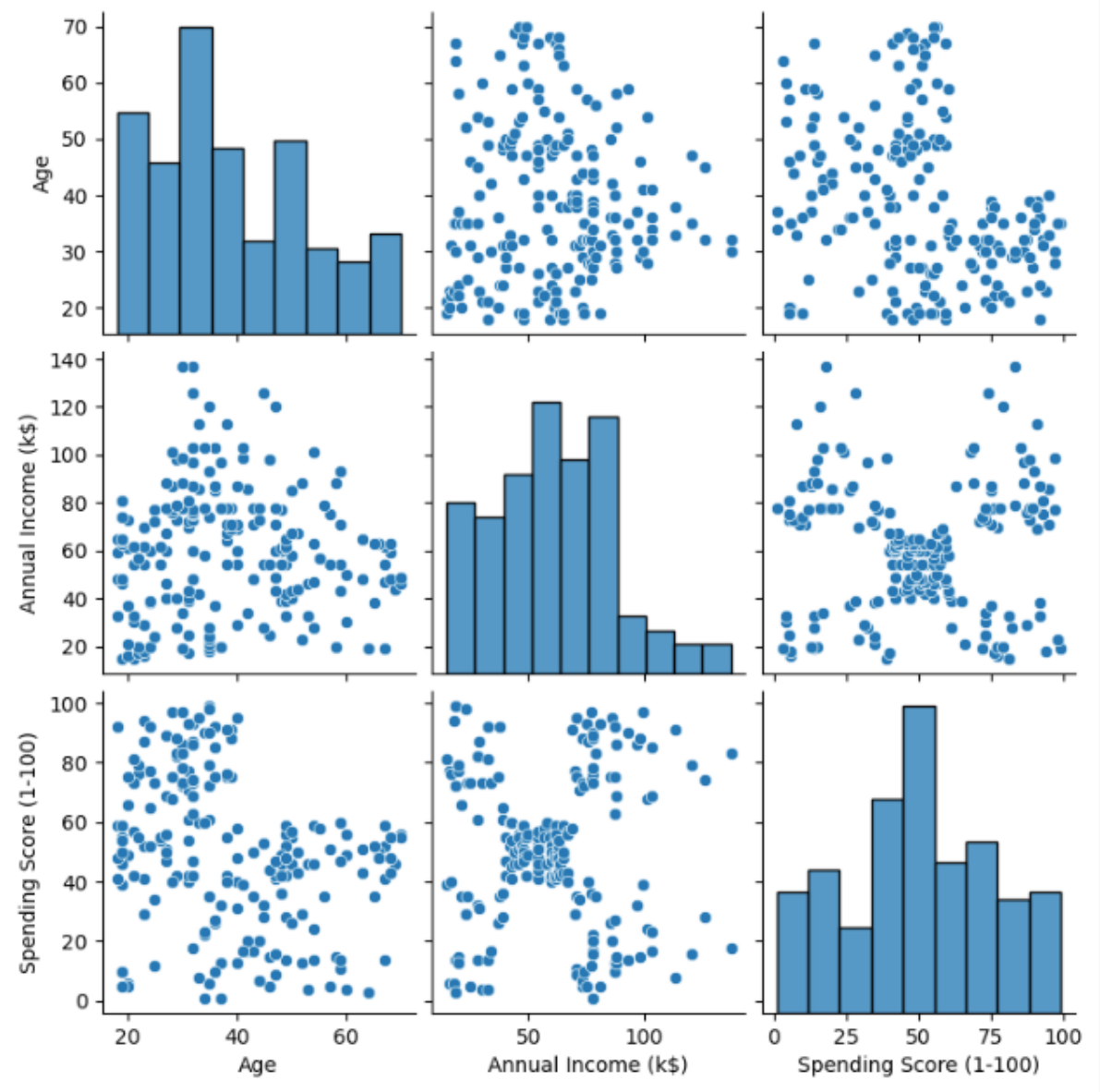

On constate qu'une grande partie des clients ont un revenu annuel 70 000$, cependant nous ne savons pas s'il y a plus de femmes ou d'hommes qui touchent ce revenu.

La majorité des clients ont environ 35 ans, et si l'on veux être plus précis la grande majorité de la base de données se situe entre la tranche d'âge 25 ans et 40 ans.




La plupart des clients sont notés par le centre commercial à 50 donc la moyenne.




Ici, nous comparons les colonnes "Age", "Annual Income" et "Spending Score" les unes avec les autres pour visualiser si des clusters se dégagent naturellement. Sur la diagonale, nous avons des graphiques en barre que nous ne prendrons pas en compte car c'est le croisement de la même colonne.
En croisant la colonne "Annual Income" et "Age" on s'aperçoit que les points sont dispersés partout et qu'aucun groupe de points ne se différencie.
Entre "Spending Score" et "Age" les points aussi sont dispersés mais nous arrivons à différencier 2 groupes de points avec une droite qui passerait à l'origine pour bien séparer les groupes.
Cependant, en regardant le croisement entre "Spending Score" et "Annual Income" des groupes de points se forment, nous arrivons à en compter 5.

Nous commencons à traiter les données et pour cela nous utilisons la fonction convert_data qui va créer la colonne "GenderIndexed" par rapport à la colonne "Gender" en un nombre décimal et que l'on renommera, puis, on supprimera les colonnes dont nous n'avons pas besoin : "CustomerID" et "Gender". "CustomerID" ne nous apporte aucune information. Voici maintenant la structure de notre base :

```
root
 |-- Age: integer (nullable = true)
 |-- Annual Income (k$): integer (nullable = true)
 |-- Spending Score (1-100): integer (nullable = true)
 |-- GenderIndexed: double (nullable = false)
```

Ensuite, nous faisons appel à la fonction convert_vector qui créera un vecteur à partir des colonnes "Annual Income", "Spending Score" et "Age".

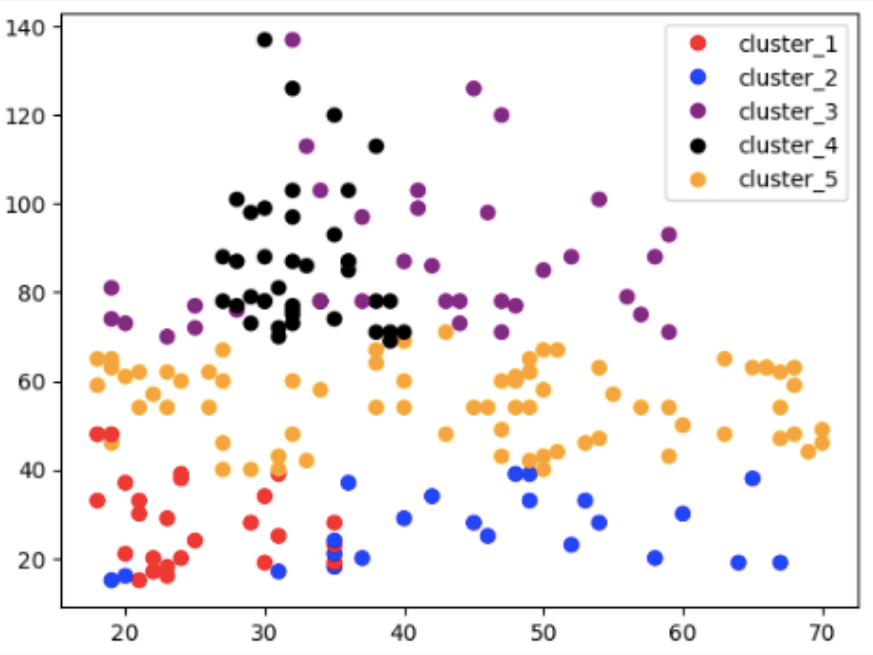
Nous souhaitons savoir la meilleure silhouette, c'est-à-dire, k avec le meilleur résultat pour savoir en combien de clusters s'est maximisé. Nous décidons de calculer k de 2 à 8. Pour cela, on se sert de BestKmeanwithSil.

Après cette fonction, nous récupérons k clusters et nous l'utilisons pour faire l'algorithme Kmeans que l'on a mis en fonction kmeans_algo. Dans notre cas, la meilleure valeur est 5, ce dont nous avons fait l'hypothèse auparavant.
Enfin, nous affichons l'extrait du résultat.

| prediction | count |
|---|---|
| 0 | 25 |
| 1 | 23 |
| 2 | 37 |
| 3 | 39 |
| 4 | 76 |

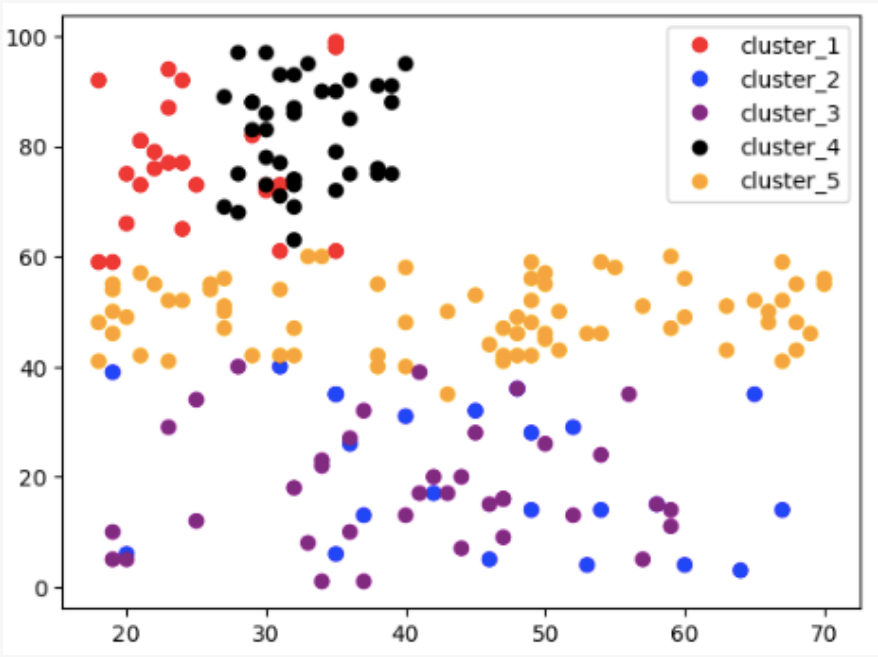
Nous pouvons voir les clusters avec le nombre de points, les clients par cluster.
Donc dans les clusters de 0 à 4, nous avons respectivement 25, 23, 37, 39 et 76 clients.

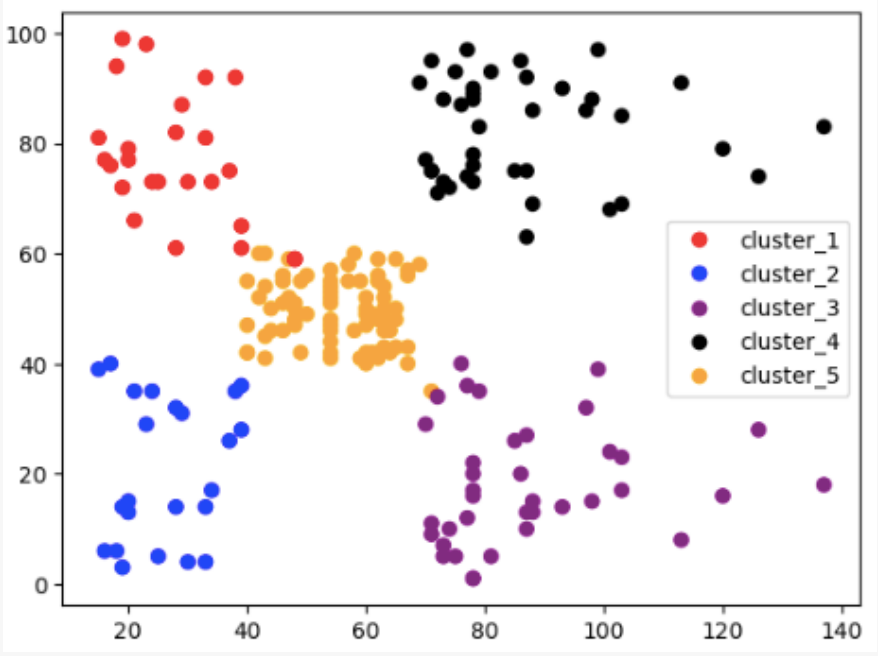
Cependant, nous ne savons pas encore comment ils sont dispersés, c'est pourquoi nous allons croiser les colonnes où chaque cluster ressortira en couleur pour savoir les similitudes.




Nous affichons les clusters où la colonne "Age" est en abscisse et "Annual Income" en ordonnée. Nous voyons les clusters mais le cluster 3 et 4 sont confondus et un peu le cluster 1 et 2. Donc ce n'est pas le revenu annuel fonction de l'âge.

Nous affichons l'âge en fonction de la note attribuée par le centre commercial en fonction des dépenses. On remarque que les clusters ici aussi sont confondus tels que le cluster 3 et le cluster 4. Donc ce n'est pas par rapport à l'âge et la note.




Nous visualisons le revenu annuel en fonction de la note attribuée et les clusters sont bien séparés, nous arrivons bien à distinguer le cluster de 1 à 5.

Enfin, nous faisons appel à la fonction classificateur qui renvoie l'accuracy qui est à 1 et nous utilisons la régression logistique et qui renvoie l'accuracy, le recall ou encore la précision. Et l'on voit qu'ils sont tous à 1, ce qui signifie que cela est bien.

# Conclusion

Nous pouvons en conclure que les groupes sont formés par la note des dépenses en fonction de leur revenu.
Donc les 5 groupes que nous distinguons et que l'on pourrait nommer est :

- cluster 1 : un faible revenu effectue des dépenses élevées
- cluster 2 : un faible revenu effectue peu de dépenses
- cluster 3 : un revenu élevé effectue peu de dépenses
- cluster 4 : un revenu élevé effectue des dépenses élevées
- cluster 5 : un revenu moyen effectue des dépenses moyennes

# Segmentation

Second jeu de données

Nous importons le fichier "small-mall-dataset.csv" et nous faisons le même principe que pour le premier jeu de données, la différence est que ce fichier contient plus de colonnes.

```
root
 |-- Invoice ID: string (nullable = true)
 |-- Branch: string (nullable = true)
 |-- City: string (nullable = true)
 |-- Customer type: string (nullable = true)
 |-- Gender: string (nullable = true)
 |-- Product line: string (nullable = true)
 |-- Unit price: double (nullable = true)
 |-- Quantity: integer (nullable = true)
 |-- Tax 5%: double (nullable = true)
 |-- Total: double (nullable = true)
 |-- Date: string (nullable = true)
 |-- Time: timestamp (nullable = true)
 |-- Payment: string (nullable = true)
 |-- cogs: double (nullable = true)
 |-- gross margin percentage: double (nullable = true)
 |-- gross income: double (nullable = true)
 |-- Rating: double (nullable = true)
```

Nous appelons le fonction transformeD qui va indexer les colonnes suivantes : "Gender", "Branch", "City", "Customer type", "Payment" .
Elles se renommera de la même manière mais en ajoutant *Indexed* à la fin du nom.
Puis, on supprime les colonnes que l'on a cité juste au dessus en plus des colonnes "Invoice ID", "Tax 5%", "Date", "Time", "cogs", "gross income", "Rating" .
Et nous finissons en retournant un vecteur avec les colonnes indexés.

Nous utilisons la fonction BestKmeanwithSil et avec la sortie nous continuons avec l'exécution de l'algorithme Kmeans. Nous renommons la colonne prediction en prediction_Kmeans, on divise les données en données d'entraînement et de test avant de faire appel au classificateur et à la régression logistique pour afficher l'accuracy, la précision ou encore le recall.

# Conclusion

Nous pouvons en conclure qu'avec l'accuracy on peux déterminer que les résultat sont encourageant et que cet algorithme est efficace pour la segmentation.

# Recommandation de films

Les algorithmes peuvent nous recommander des films en se basant sur ce que l'on a vu et aimé par rapport aux notes mises, ou bien par le genre de films comme action, aventure, science-fiction ou encore par des profils similaires que le nôtre. Il est possible de s'appuyer sur des profils similaires ayant les mêmes goûts que nous, qui ont vu des films que l'on a pas vu et puisse nous le recommander. Tout cela sont des algorithmes de recommandation de films qui portent des noms : Modèles de contenu, modèle de mémoire de matrice factorisée ou encore filtrage collaboratif.
Nous allons voir en détail un de ces algorithmes, mais avant voici les jeux de données que nous allons utilisés pour réaliser ceci : "movies.csv" et "ratings.csv"

Tout d'abord, nous voyons la structure de "movies.csv" :

```
root
 |-- movieId: string (nullable = true)
 |-- title: string (nullable = true)
 |-- genres: string (nullable = true)
```

Ce fichier est composé de 3 colonnes :

- "movieId" qui représente l'id du film
- "title" qui est le titre du film
- "genres" qui est le genre du film

Puis, la structure de "ratings.csv" :

```
root
 |-- userId: string (nullable = true)
 |-- movieId: string (nullable = true)
 |-- rating: string (nullable = true)
 |-- timestamp: string (nullable = true)
```

Nous faisons appel à la fonction rating_cast qui va nous permettre de supprimer l colonne timestamp car dans notre cas, nous nous en servirons pas et de transformer les colonnes qui sont en string en entier (integer) sauf la colonne rating que l'on va mettre en float car une note peut être décimal.

Ensuite, nous utilisons la fonction ALS_recommandation qui récupère les données avec les colonnes "movieId", "userId" et "rating" du fichier "ratings.csv" et les séparent en données d'entraînement et de test, et exécute l'algorithme ALS qui est un algorithme de filtrage collaboratif. Il renverra les 10 meilleurs films avec la note pour un utilisateur.

```
DataFrame[userId: int, recommendations: array<struct<movieId:int,rating:float>>]
```

Voici ce que nous renvoie ALS_recommandation ci-dessus, c'est ainsi, que nous utilisons la fonction recommandation_tab qui convertira les recommandations dans un format interprétable.

| userId | movieId | rating |
|---|---|---|
| 1 | 203086 | 5.804782 |
| 1 | 173871 | 5.759889 |
| 1 | 151989 | 5.718561 |
| 1 | 194434 | 5.6865563 |
| 1 | 203882 | 5.600602 |
| 1 | 183947 | 5.4728765 |
| 1 | 193257 | 5.4701 |
| 1 | 194334 | 5.4418344 |
| 1 | 192089 | 5.2483006 |
| 1 | 207640 | 5.2422843 |
| 6 | 194434 | 6.324586 |
| 6 | 203086 | 6.080991 |
| 6 | 203882 | 5.9627953 |
| 6 | 151989 | 5.783994 |
| 6 | 183947 | 5.7063117 |
| 6 | 173871 | 5.703575 |
| 6 | 193257 | 5.616068 |
| 6 | 157789 | 5.570483 |
| 6 | 157791 | 5.570483 |
| 6 | 194334 | 5.527264 |

Après, nous faisons une jointure pour relier le fichier "movies.csv" et la sortie de recommandation_tab, et nous filtrons pour l'utilisateur dont l'id est 100 en brut pour afficher les films qu'on lui recommande.

| movieId | title | genres | userId | rating |
|---|---|---|---|---|
| 193257 | Familie Brasch (2... | Documentary | 100 | 5.6120677 |
| 203086 | Truth and Justice... | Drama | 100 | 5.5448117 |
| 194434 | Adrenaline (1990) | (no genres listed) | 100 | 5.469316 |
| 203882 | Dead in the Water... | Horror | 100 | 5.4482026 |
| 194334 | Les Luthiers: El ... | (no genres listed) | 100 | 5.445797 |
| 173871 | I'll Take You The... | Comedy\|Drama\|Romance | 100 | 5.3621836 |
| 151989 | The Thorn (1971) | Comedy | 100 | 5.213784 |
| 183947 | NOFX Backstage Pa... | (no genres listed) | 100 | 5.18946 |
| 192089 | National Theatre ... | Comedy | 100 | 5.175419 |
| 155923 | Sing (1989) | (no genres listed) | 100 | 5.001781 |

*Les 10 premiers films pour l'utilisateur avec l'id 100*

De plus, pour simplifier tout ceci, nous avons joint les fichiers "movies.csv" et "ratings.csv" à l'aide "movieId" car dans le fichier "ratings.csv" il nous manquait les informations présentes dans le fichier "movies.csv" soit le genre et le titre du film

| movieId | userId | rating | timestamp | title | genres |
|---|---|---|---|---|---|
| 296 | 1 | 5.0 | 1147880044 | Pulp Fiction (1994) | Comedy\|Crime\|Dram... |
| 306 | 1 | 3.5 | 1147868817 | Three Colors: Red... | Drama |
| 307 | 1 | 5.0 | 1147868828 | Three Colors: Blu... | Drama |
| 665 | 1 | 5.0 | 1147878820 | Underground (1995) | Comedy\|Drama\|War |
| 899 | 1 | 3.5 | 1147868510 | Singin' in the Ra... | Comedy\|Musical\|Ro... |
| 1088 | 1 | 4.0 | 1147868495 | Dirty Dancing (1987) | Drama\|Musical\|Rom... |
| 1175 | 1 | 3.5 | 1147868826 | Delicatessen (1991) | Comedy\|Drama\|Romance |
| 1217 | 1 | 3.5 | 1147878326 | Ran (1985) | Drama\|War |
| 1237 | 1 | 5.0 | 1147868839 | Seventh Seal, The... | Drama |
| 1250 | 1 | 4.0 | 1147868414 | Bridge on the Riv... | Adventure\|Drama\|War |
| 1260 | 1 | 3.5 | 1147877857 | M (1931) | Crime\|Film-Noir\|T... |
| 1653 | 1 | 4.0 | 1147868097 | Gattaca (1997) | Drama\|Sci-Fi\|Thri... |
| 2011 | 1 | 2.5 | 1147868079 | Back to the Futur... | Adventure\|Comedy\|... |
| 2012 | 1 | 2.5 | 1147868068 | Back to the Futur... | Adventure\|Comedy\|... |
| 2068 | 1 | 2.5 | 1147869044 | Fanny and Alexand... | Drama\|Fantasy\|Mys... |
| 2161 | 1 | 3.5 | 1147868609 | NeverEnding Story... | Adventure\|Childre... |
| 2351 | 1 | 4.5 | 1147877957 | Nights of Cabiria... | Drama |
| 2573 | 1 | 4.0 | 1147878923 | Tango (1998) | Drama\|Musical |
| 2632 | 1 | 5.0 | 1147878248 | Saragossa Manuscr... | Adventure\|Drama\|M... |
| 2692 | 1 | 5.0 | 1147869100 | Run Lola Run (Lol... | Action\|Crime |

# Conclusion

La recommandation de films permet de faire plein de recommandation en fonction de ce que l'on souhaite, c'est-à-dire, par genre, par note ou par profil similaire. Les résultats que nous avons obtenu sont satisfaisant et permet de bien prédire les meilleurs films à voir pour un utilisateur en fonction de ce qu'il a pu voir avant ainsi qu'avec les notes qu'il a mis.

# Conclusion générale

Ce projet nous a permis de voir la segmentation ainsi que les algorithmes qui nous permettent de l'effectuer. La visualisation est une étape importante pour émettre des hypothèses et avoir une vision des données. La segmentation a permis de confirmer ce que l'on a pu voir et en effectuant des tests, se rendre compte que l'accuracy était élevé.
De plus, la recommandation de films nous montre qu'il est possible de faire des recommandations en fonction de plein d'éléments et pouvoir satisfaire une majorité de personnes en se basant sur qu'elle souhaite regarder et en limitant à 10 films.

- Mall_Customers.csv qui provient de : https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python
- small-mall-dataset.csv qui provient de : https://www.kaggle.com/datasets/jonhsmith/smallmalldataset
- movies.csv et ratings.csv qui proviennent de : https://grouplens.org/datasets/movielens/25m/
- Github : https://github.com/ulrickamelie/RecommandationF-Segmentation