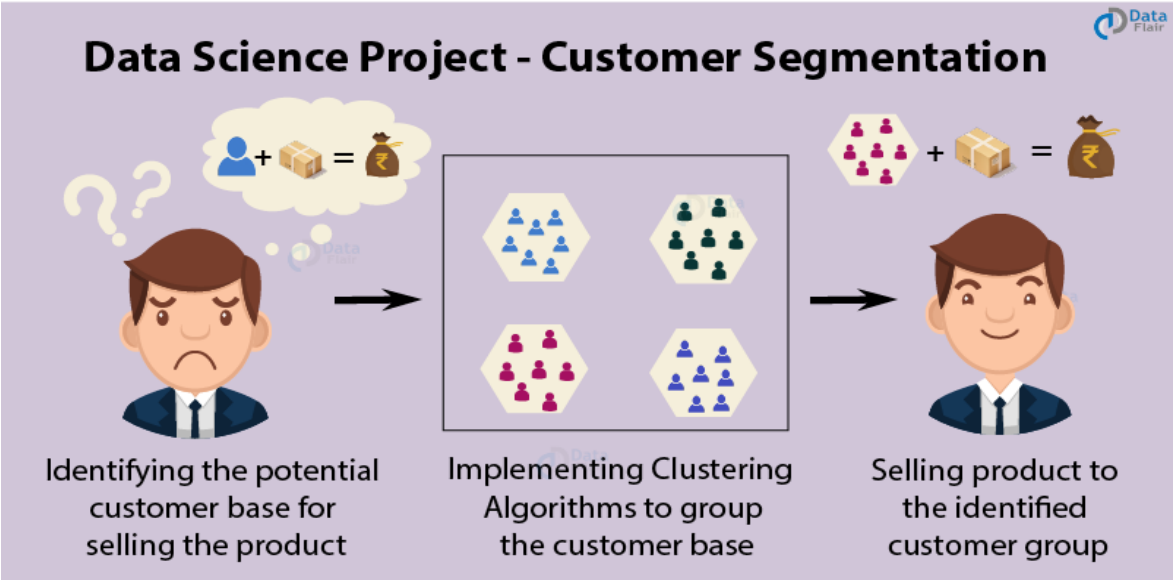
Project – Customer Segmentation using Machine Learning in R

Dans ce projets Data Science en R, nous réaliserons l'une des applications les plus essentielles du machine learning: la segmentation client. Dans ce projet, nous mettrons en œuvre la segmentation client en R. Chaque fois que vous avez besoin de trouver votre meilleur client, la segmentation client est la méthodologie idéale.

Dans ce projet d'apprentissage automatique, on fourni le contexte de la segmentation client. Ensuite, nous explorerons les données sur lesquelles nous allons construire notre modèle de segmentation. De plus, dans ce projet de science des données, nous verrons l'analyse descriptive de nos données, puis implémenterons plusieurs versions de l'algorithme K-means. Suivez donc l'intégralité du projet de segmentation des clients en science des données à l'aide de l'apprentissage automatique dans R

Projet de segmentation client en R

La segmentation client est l'une des applications les plus importantes de l'apprentissage non supervisé. En utilisant des techniques de clustering, les entreprises peuvent identifier les différents segments de clients leur permettant de cibler la base d'utilisateurs potentiels. Dans ce projet d'apprentissage automatique, nous utiliserons le *clustering K-means* qui est l'algorithme essentiel pour grouper un ensemble de données non étiqueté. Avant d'aller plus loin dans ce projet, découvrez ce qu'est réellement la segmentation client



Qu'est-ce que la segmentation client?

La segmentation de la clientèle est le processus de division de la clientèle en plusieurs groupes d'individus qui partagent une similitude de différentes manières pertinentes pour le marketing, telles que le sexe, l'âge, les intérêts et les habitudes de dépenses diverses.

Les entreprises qui déploient la segmentation de la clientèle pensent que chaque client a des exigences différentes et requiert un effort marketing spécifique pour y répondre de manière appropriée. Les entreprises visent à acquérir une approche plus approfondie du client qu'elles ciblent. Par conséquent, leur objectif doit être spécifique et doit être adapté pour répondre aux exigences de chaque client individuel. En outre, grâce aux données collectées, les entreprises peuvent acquérir une compréhension plus approfondie des préférences des clients ainsi que des exigences pour découvrir des segments précieux qui leur procureraient un profit maximal. De cette façon, ils peuvent élaborer des stratégies de leurs techniques de marketing plus efficacement et minimiser la possibilité de risque pour leur investissement.

La technique de segmentation de la clientèle dépend de plusieurs différenciateurs clés qui divisent les clients en groupes à cibler. Les données relatives à la démographie, la géographie, la situation économique ainsi que les modèles de comportement jouent un rôle crucial dans la détermination de l'orientation de l'entreprise vers les différents segments.

Vous pouvez télécharger l'ensemble de données pour le projet de segmentation client:
https://www.kaggle.com/shwetabh123/mall-customers

Comment implémenter la segmentation client en R?

Dans la première étape de ce projet de science des données, nous effectuerons l'exploration des données. Nous importerons les packages essentiels requis pour ce rôle, puis lirons nos données. Enfin, nous allons parcourir les données d'entrée pour obtenir les informations nécessaires à ce sujet.

Code:

```
customer_data = read.csv("/home/richmond/Mall_Customers.csv")
str(customer_data)

names(customer_data)
```

Capture d'écran de sortie:

```
customer_data=read.csv("/home/dataflair/Mall_Customers.csv")
str(customer_data)
```

```
## 'data.frame':    200 obs. of  5 variables:
##  $ CustomerID            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Gender                : Factor w/ 2 levels "Female","Male": 2 2 1 1 1 1 1 1 2 1
...
##  $ Age                   : int  19 21 20 23 31 22 35 23 64 30 ...
##  $ Annual.Income..k..    : int  15 15 16 16 17 17 18 18 19 19 ...
##  $ Spending.Score..1.100.: int  39 81 6 77 40 76 6 94 3 72 ...
```

```
names(customer_data)
```

```
## [1] "CustomerID"             "Gender"
## [3] "Age"                    "Annual.Income..k.."
## [5] "Spending.Score..1.100."
```

Nous allons maintenant afficher les six premières lignes de notre ensemble de données à l'aide de la fonction head () et utiliser la fonction summary () pour en sortir un résumé.

Code:

```
1. head(customer_data)
2. summary(customer_data$Age)
```

Capture d'écran de sortie:

```
head(customer_data)
```

```
##   CustomerID Gender Age Annual.Income..k.. Spending.Score..1.100.
## 1          1   Male  19                 15                     39
## 2          2   Male  21                 15                     81
## 3          3 Female  20                 16                      6
## 4          4 Female  23                 16                     77
## 5          5 Female  31                 17                     40
## 6          6 Female  22                 17                     76
```

```
summary(customer_data$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   28.75   36.00   38.85   49.00   70.00
```

Code:

```
1. sd(customer_data$Age)
2. summary(customer_data$Annual.Income..k..)
3. sd(customer_data$Annual.Income..k..)
4. summary(customer_data$Age)
```

Capture d'écran de sortie:

```
sd(customer_data$Age)
```

```
## [1] 13.96901
```

```
summary(customer_data$Annual.Income..k..)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.00   41.50   61.50   60.56   78.00  137.00
```

```
sd(customer_data$Annual.Income..k..)
```

```
## [1] 26.26472
```

```
summary(customer_data$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   28.75   36.00   38.85   49.00   70.00
```

Code:

```
1. sd(customer_data$Spending.Score..1.100.)
```

Capture d'écran de sortie:

```
sd(customer_data$Spending.Score..1.100.)
```

```
## [1] 25.82352
```

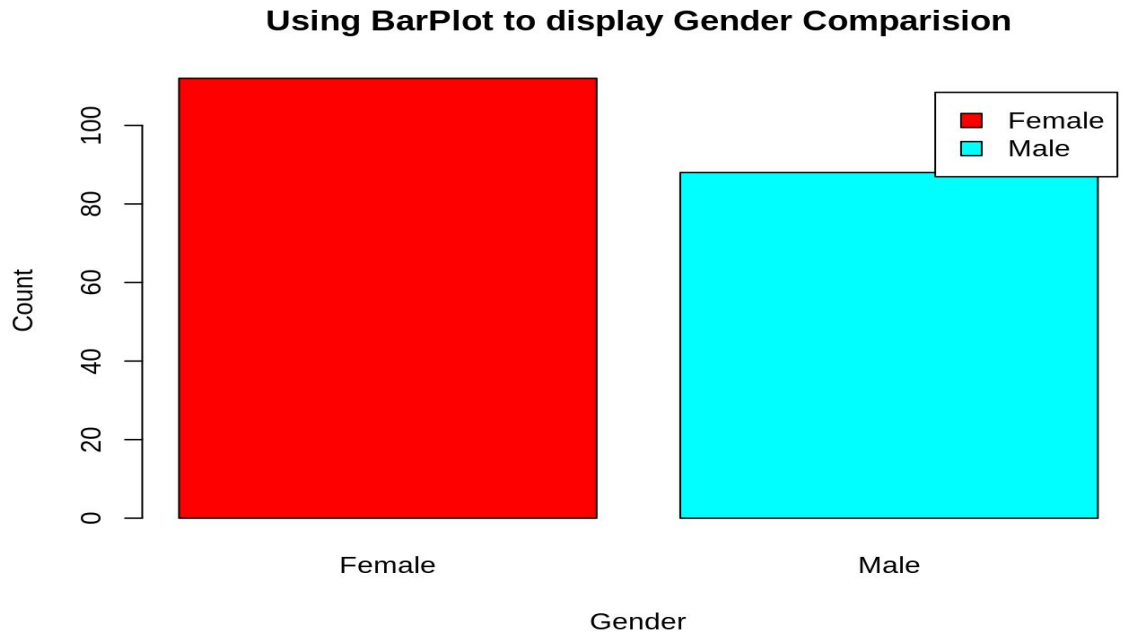
Visualisation du genre du client

Dans ce document, nous allons créer un diagramme à barres et un diagramme à pièces pour montrer la répartition par sexe dans notre ensemble de données customer_data.

Code:

```
a=table(customer_data$Gender)
barplot(a,main="Using BarPlot to display Gender Comparision",
        ylab="Count",
        xlab="Gender",
        col=rainbow(2),
        legend=rownames(a))
```

Output:

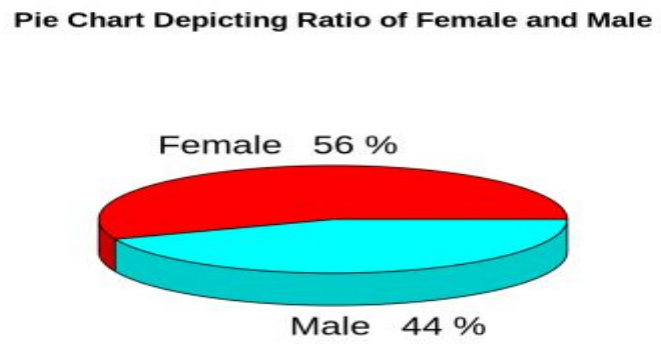


À partir du diagramme à barres ci-dessus, nous observons que le nombre de femelles est supérieur à celui des mâles. Maintenant, visualisons un graphique circulaire pour observer le rapport de la distribution masculine et féminine.

Code:

```
pct=round(a/sum(a)*100)
lbs=paste(c("Female","Male")," ",pct,"%",sep=" ")
library(plotrix)
pie3D(a,labels=lbs,
     main="Pie Chart Depicting Ratio of Female and Male")
```

Output:



À partir du graphique ci-dessus, nous concluons que le pourcentage de femmes est de 56% , tandis que le pourcentage d'hommes dans l'ensemble de données clients est de 44% .

Code:

```
1. summary(customer_data$Age)
```

Capture d'écran de sortie:

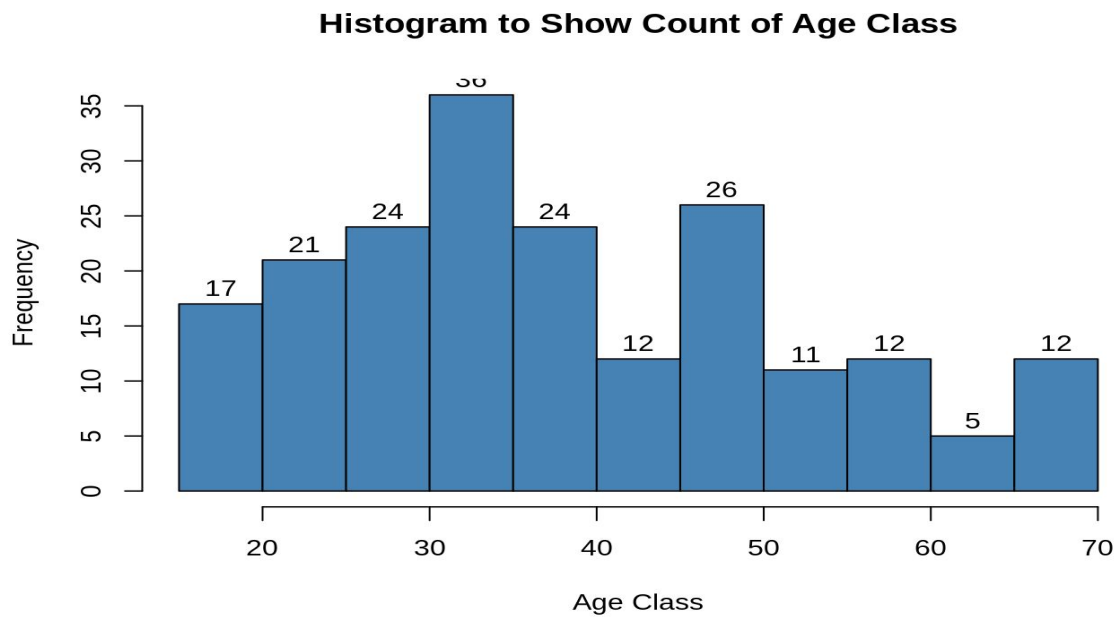
```
summary(customer_data$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   28.75   36.00   38.85   49.00   70.00
```

Code:

```
hist(customer_data$Age,
    col="blue",
    main="Histogram to Show Count of Age Class",
    xlab="Age Class",
    ylab="Frequency",
    labels=TRUE)
```

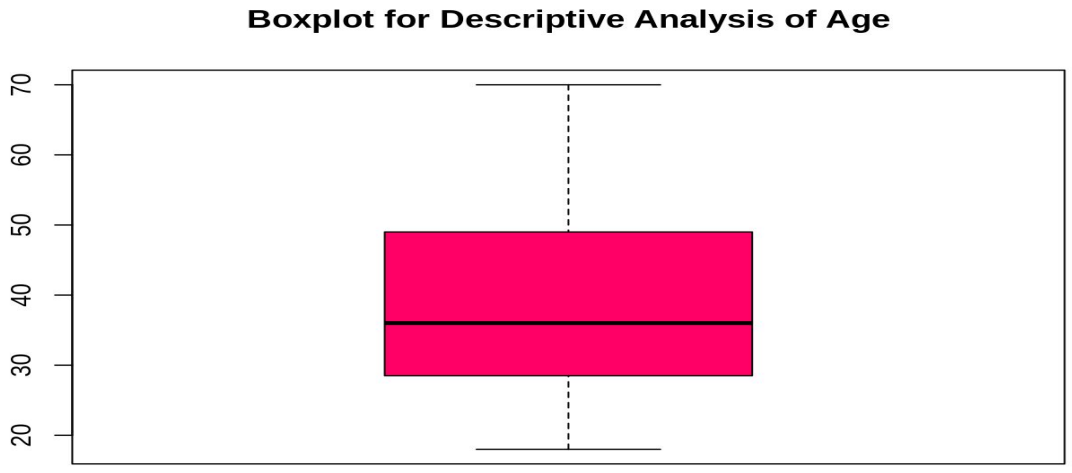
Output:



Code:

```
boxplot(customer_data$Age,
        col="#ff0066",
        main="Boxplot for Descriptive Analysis of Age")
```

Output:



À partir des deux visualisations ci-dessus, nous concluons que l'âge maximum des clients se situe entre 30 et 35 ans. L'âge minimum des clients est de 18 ans, alors que l'âge maximum est de 70 ans.

Analyse du revenu annuel des clients

Dans cette section du projet R, nous allons créer des visualisations pour analyser le revenu annuel des clients. Nous tracerons un histogramme puis nous procéderons à l'examen de ces données à l'aide d'un tracé de densité.

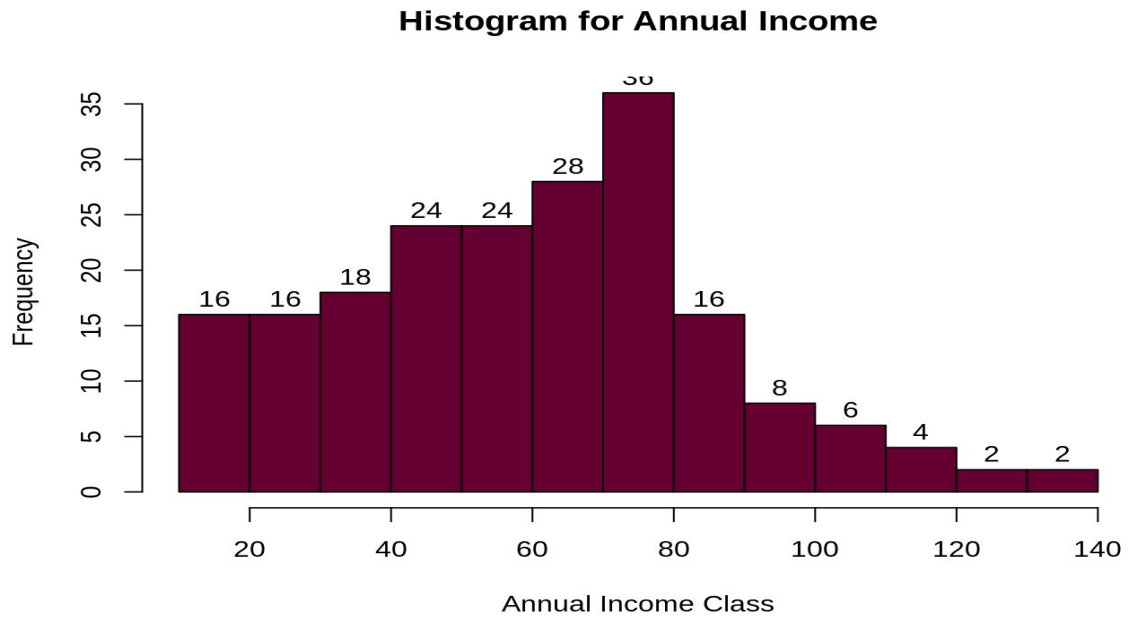
Code:

```
summary(customer_data$Annual.Income..k..)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.00   41.50   61.50   60.56   78.00  137.00
```

```
hist(customer_data$Annual.Income..k..,
     col="#660033",
     main="Histogram for Annual Income",
     xlab="Annual Income Class",
     ylab="Frequency",
     labels=TRUE)
```

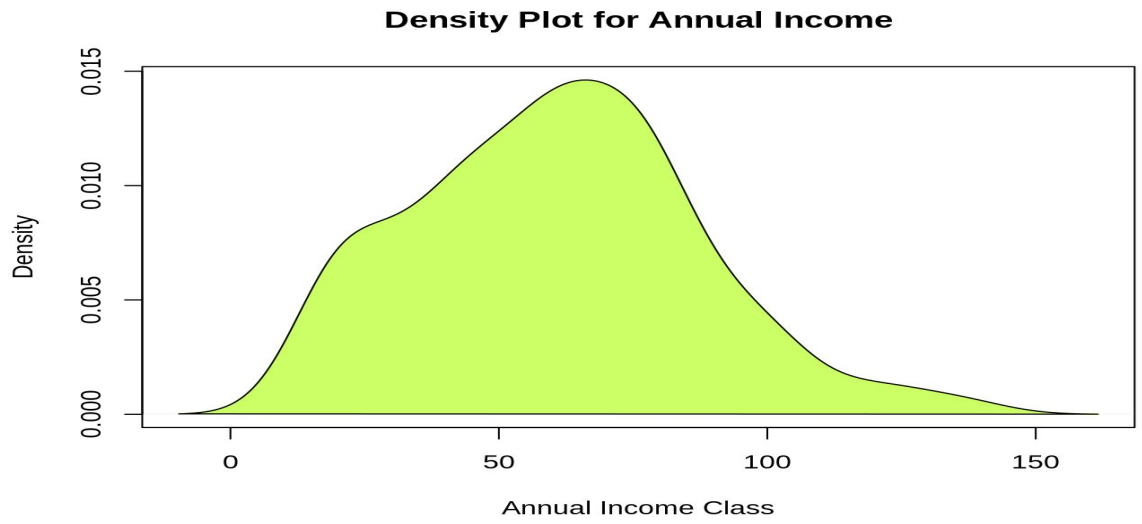
Output:



Code:

```
plot(density(customer_data$Annual.Income..k..),
     col="yellow",
     main="Density Plot for Annual Income",
     xlab="Annual Income Class",
     ylab="Density")
polygon(density(customer_data$Annual.Income..k..),
        col="#ccff66")
```

Production:



De l'analyse descriptive ci-dessus, nous concluons que le revenu annuel minimum des clients est de 15 et le revenu maximum est de 137. Les personnes gagnant un revenu moyen de 70 ont le nombre de fréquences le plus élevé dans notre distribution d'histogramme. Le salaire moyen de tous les clients est de 60,56. Dans le graphique de densité de noyau que nous avons affiché ci-dessus, nous observons que le revenu annuel a une *distribution normale* .

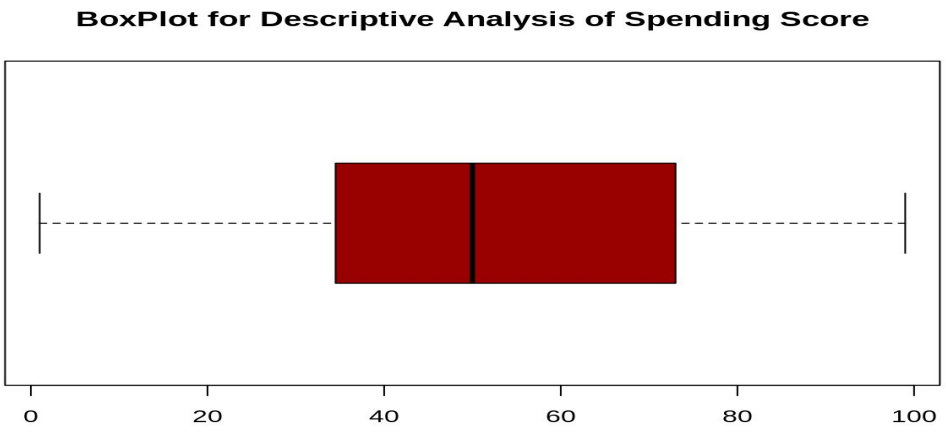
Analyse du score de dépenses des clients

```
summary(customer_data$Spending.Score..1.100.)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   34.75   50.00   50.20   73.00   99.00
```

```
boxplot(customer_data$Spending.Score..1.100.,
        horizontal=TRUE,
        col="#990000",
        main="BoxPlot for Descriptive Analysis of Spending Score")
```

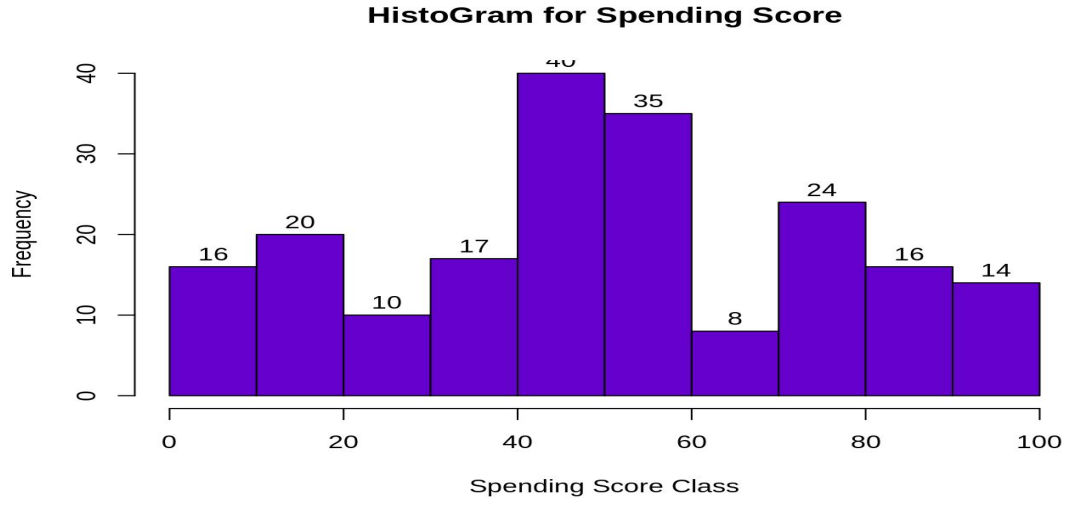
Output:



Code:

```
hist(customer_data$Spending.Score..1.100.,
     main="HistoGram for Spending Score",
     xlab="Spending Score Class",
     ylab="Frequency",
     col="#6600cc",
     labels=TRUE)
```

Output:



Le score minimum de dépenses est de 1, le maximum est de 99 et la moyenne est de 50,20. Nous pouvons voir que l'analyse descriptive du score de dépenses est que Min est 1, Max est 99 et moy. est 50,20. À partir de l'histogramme, nous concluons que les clients entre les classes 40 et 50 ont le score de dépenses le plus élevé parmi toutes les classes.

Algorithme K-means

En utilisant l'algorithme de clustering k-means, la première étape consiste à indiquer le nombre de clusters (k) que nous souhaitons produire dans la sortie finale. L'algorithme commence par sélectionner au hasard k objets dans l'ensemble de données qui serviront de centres initiaux pour nos clusters. Ces objets sélectionnés sont les moyens de cluster, également appelés centroïdes. Ensuite, les objets restants ont une affectation du centroïde le plus proche. Ce centroïde est défini par la distance euclidienne présente entre l'objet et la moyenne du cluster. Nous appelons cette étape «affectation de cluster». Une fois l'affectation terminée, l'algorithme procède au calcul de la nouvelle valeur moyenne de chaque cluster présent dans les données. Après le recalcul des centres, les observations sont vérifiées si elles sont plus proches d'un cluster différent. En utilisant la moyenne de cluster mise à jour, les objets subissent une réaffectation. Cela continue à plusieurs reprises à travers plusieurs itérations jusqu'à ce que les affectations de cluster cessent de changer. Les clusters présents dans l'itération en cours sont les mêmes que ceux obtenus dans l'itération précédente.

Résumant le clustering K-means -

- Nous spécifions le nombre de clusters que nous devons créer.
- L'algorithme sélectionne k objets au hasard dans l'ensemble de données. Cet objet est le cluster ou la moyenne initiale.
- Le centroïde le plus proche obtient l'affectation d'une nouvelle observation. Nous basons cette affectation sur la distance euclidienne entre l'objet et le centroïde.
- k grappes dans les points de données mettent à jour le centroïde par le calcul des nouvelles valeurs moyennes présentes dans tous les points de données de la grappe. Le centroïde du kième groupe a une longueur de p qui contient la moyenne de toutes les variables pour les observations dans le groupe k. On note le nombre de variables avec p.
- Minimisation itérative du total dans la somme des carrés. Ensuite, grâce à la minimisation itérative de la somme totale du carré,

l'affectation cesse de vaciller lorsque nous atteignons l'itération maximale. La valeur par défaut est 10 que le logiciel R utilise pour les itérations maximales.

Détermination des grappes optimales

Lorsque vous travaillez avec des clusters, vous devez spécifier le nombre de clusters à utiliser. Vous souhaitez utiliser le nombre optimal de clusters. Pour vous aider à déterminer les grappes optimales, il existe trois méthodes populaires -

- Méthode du coude
- Méthode silhouette
- Statistiques d'écart

Méthode du coude

L'objectif principal des méthodes de partitionnement de cluster comme k-means est de définir les clusters de telle sorte que la variation intra-cluster reste minimale.

minimize (sum W (Ck)), k = 1... k

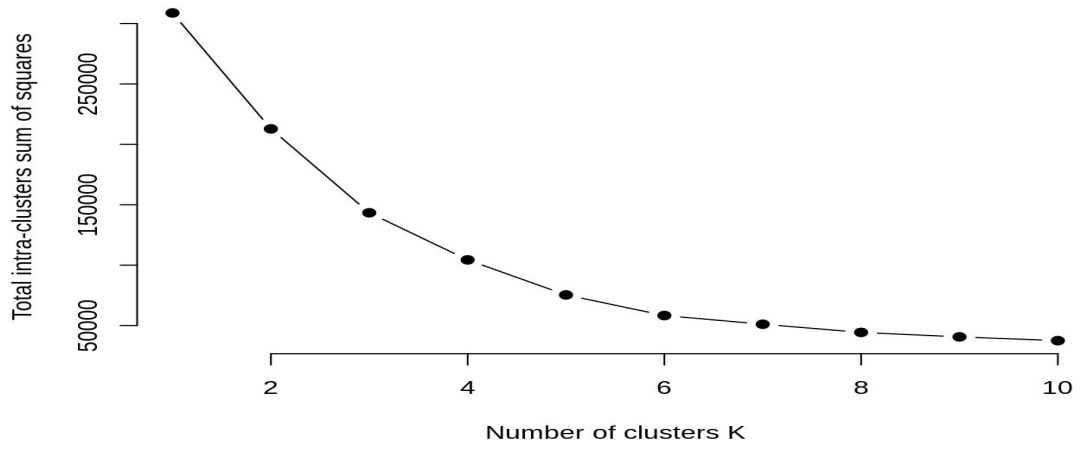
Où Ck représente le kème cluster et W (Ck) désigne la variation intra-cluster. Avec la mesure de la variation totale intra-cluster, on peut évaluer la compacité de la frontière de clustering. Nous pouvons ensuite procéder pour définir les clusters optimaux comme suit -

Tout d'abord, nous calculons l'algorithme de clustering pour plusieurs valeurs de k. Cela peut être fait en créant une variation à l'intérieur de k de 1 à 10 grappes. Nous calculons ensuite la somme totale intra-cluster du carré (iss). Ensuite, nous procédons au tracé de Iss en fonction du nombre de k grappes. Ce graphique indique le nombre approprié de grappes requis dans notre modèle. Dans l'intrigue, l'emplacement d'un coude ou d'un genou indique le nombre optimal de grappes. Implémentons ceci dans R comme suit -

Code:

```
library(purrr)
set.seed(123)
# function to calculate total intra-cluster sum of square
iss <- function(k) {
  kmeans(customer_data[,3:5],k,iter.max=100,nstart=100,algorithm="Lloyd" )$tot.withins
s
}

k.values <- 1:10


iss_values <- map_dbl(k.values, iss)

plot(k.values, iss_values,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total intra-clusters sum of squares")
```

Output:



À partir du graphique ci-dessus, nous concluons que 4 est le nombre approprié de grappes car il semble apparaître au virage du tracé du coude.

Méthode de silhouette moyenne

À l'aide de la méthode de la silhouette moyenne, nous pouvons mesurer la qualité de notre opération de clustering. Avec cela, nous pouvons déterminer dans quelle mesure au sein du cluster est l'objet de données. Si nous obtenons une largeur de silhouette moyenne élevée, cela signifie que nous avons un bon regroupement. La méthode de silhouette moyenne calcule la moyenne des observations de silhouette pour différentes valeurs de k. Avec le nombre optimal de k grappes, on peut maximiser la silhouette moyenne sur des valeurs significatives pour k grappes.

En utilisant la fonction silhouette dans le package de cluster, nous pouvons calculer la largeur de silhouette moyenne en utilisant la fonction kmean. Ici, le cluster optimal possédera la moyenne la plus élevée.

Code:

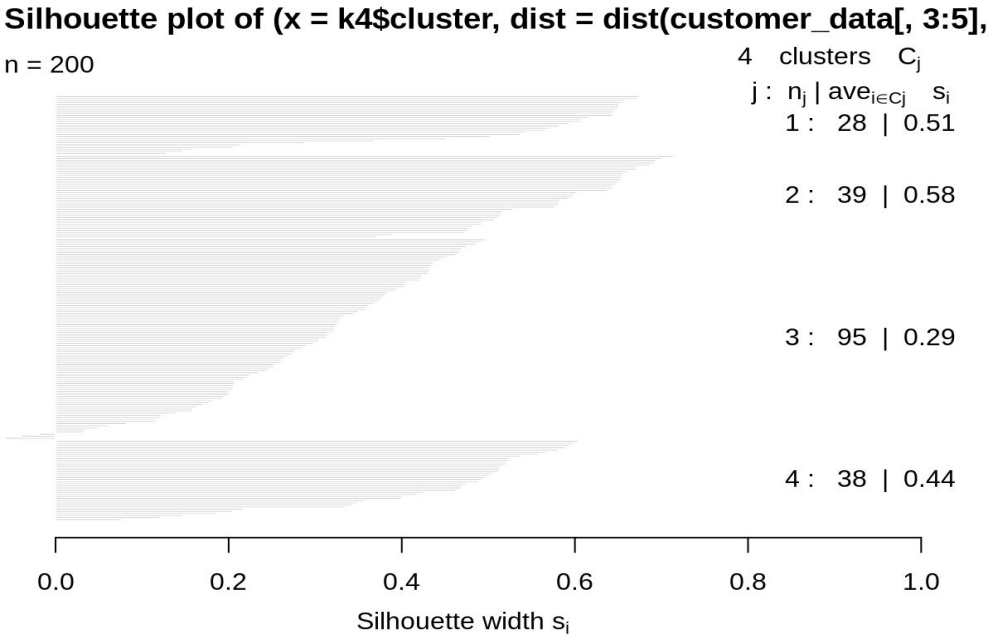
```
library(cluster)
library(gridExtra)
library(grid)


k2<-kmeans(customer_data[,3:5],2,iter.max=100,nstart=50,algorithm="Lloyd")
s2<-plot(silhouette(k2$cluster,dist(customer_data[,3:5],"euclidean")))
```

Output:

Code:

```
k3<-kmeans(customer_data[,3:5],3,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

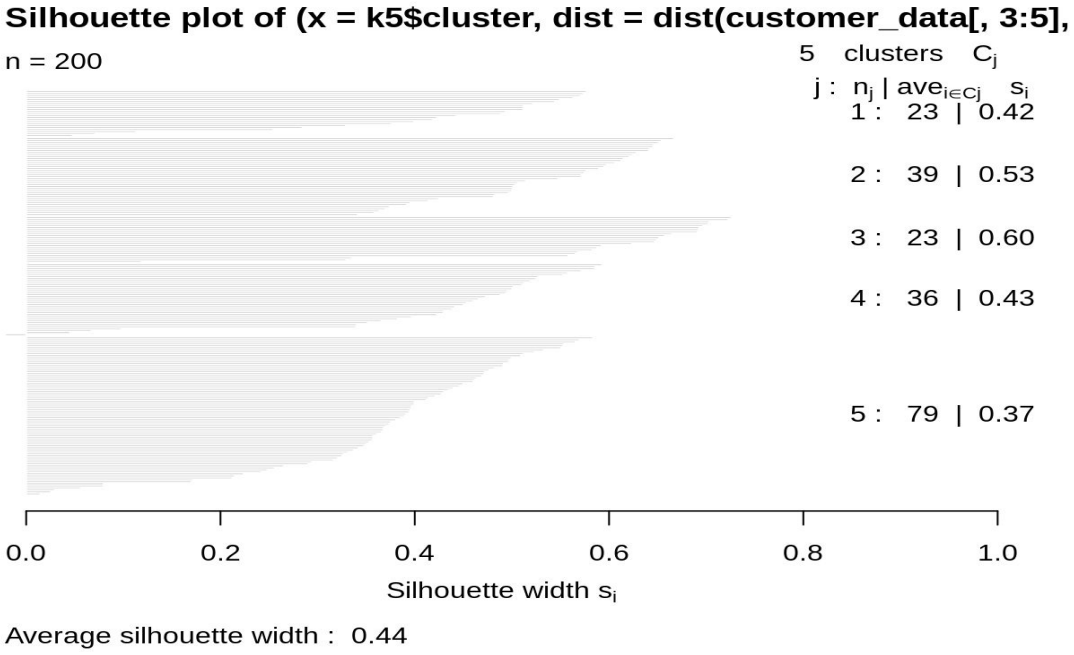
Output:



Code:

```
k5<-kmeans(customer_data[,3:5],5,iter.max=100,nstart=50,algorithm="Lloyd")
s5<-plot(silhouette(k5$cluster,dist(customer_data[,3:5],"euclidean")))
```

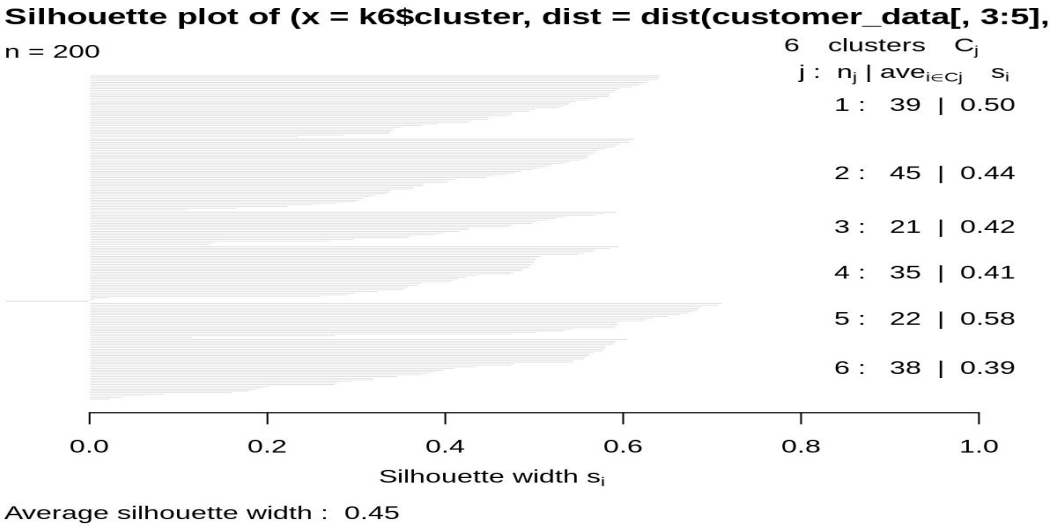
Output:



Code:

```
k6<-kmeans(customer_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
s6<-plot(silhouette(k6$cluster,dist(customer_data[,3:5],"euclidean")))
```

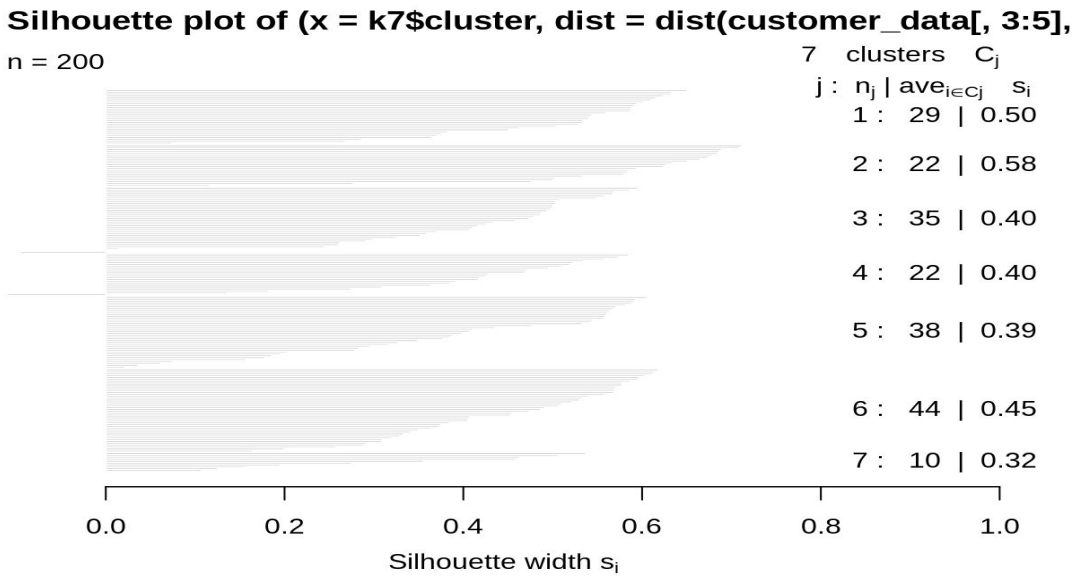
Output:



Code:

```
k7<-kmeans(customer_data[,3:5],7,iter.max=100,nstart=50,algorithm="Lloyd")
s7<-plot(silhouette(k7$cluster,dist(customer_data[,3:5],"euclidean")))
```

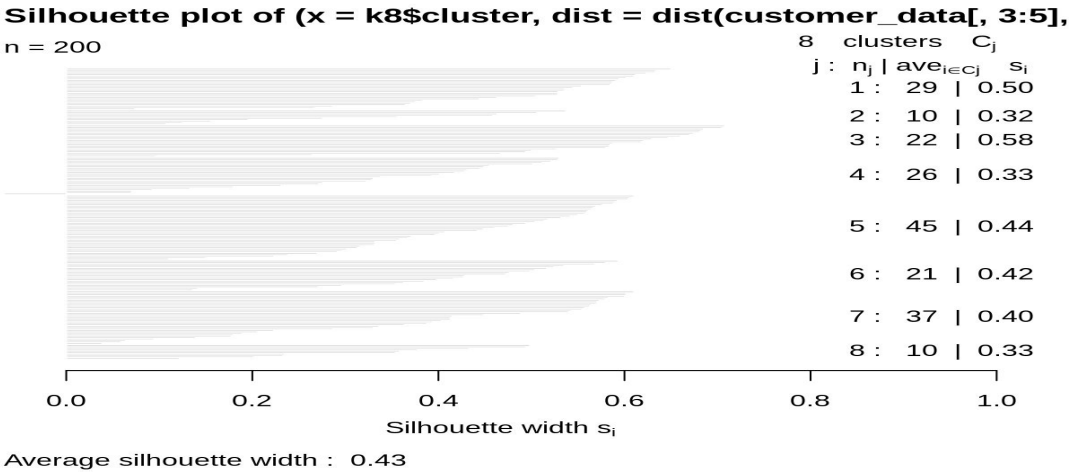
Output:



Code:

```
k8<-kmeans(customer_data[,3:5],8,iter.max=100,nstart=50,algorithm="Lloyd")
s8<-plot(silhouette(k8$cluster,dist(customer_data[,3:5],"euclidean")))
```

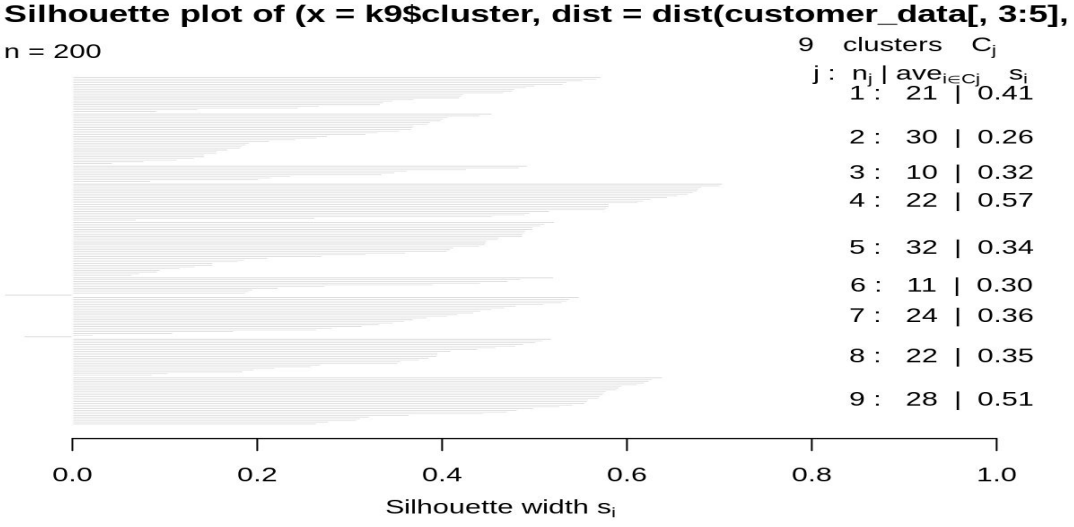
Output:



Code:

```
k9<-kmeans(customer_data[,3:5],9,iter.max=100,nstart=50,algorithm="Lloyd")
s9<-plot(silhouette(k9$cluster,dist(customer_data[,3:5],"euclidean")))
```

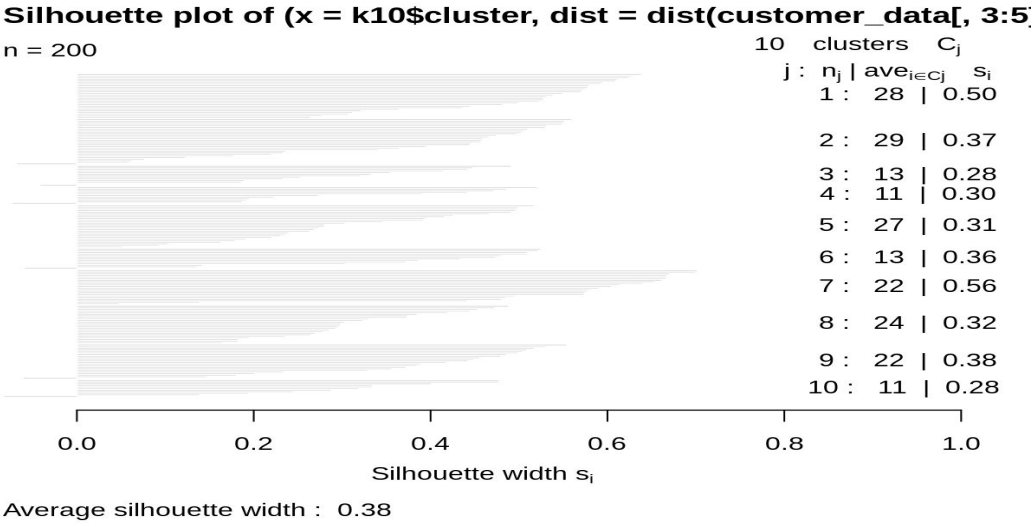
Output:



Code:

```
k10<-kmeans(customer_data[,3:5],10,iter.max=100,nstart=50,algorithm="Lloyd")
s10<-plot(silhouette(k10$cluster,dist(customer_data[,3:5],"euclidean")))
```

Output:



Maintenant, nous utilisons la fonction fviz_nbclust () pour déterminer et visualiser le nombre optimal de clusters comme suit -

Code:

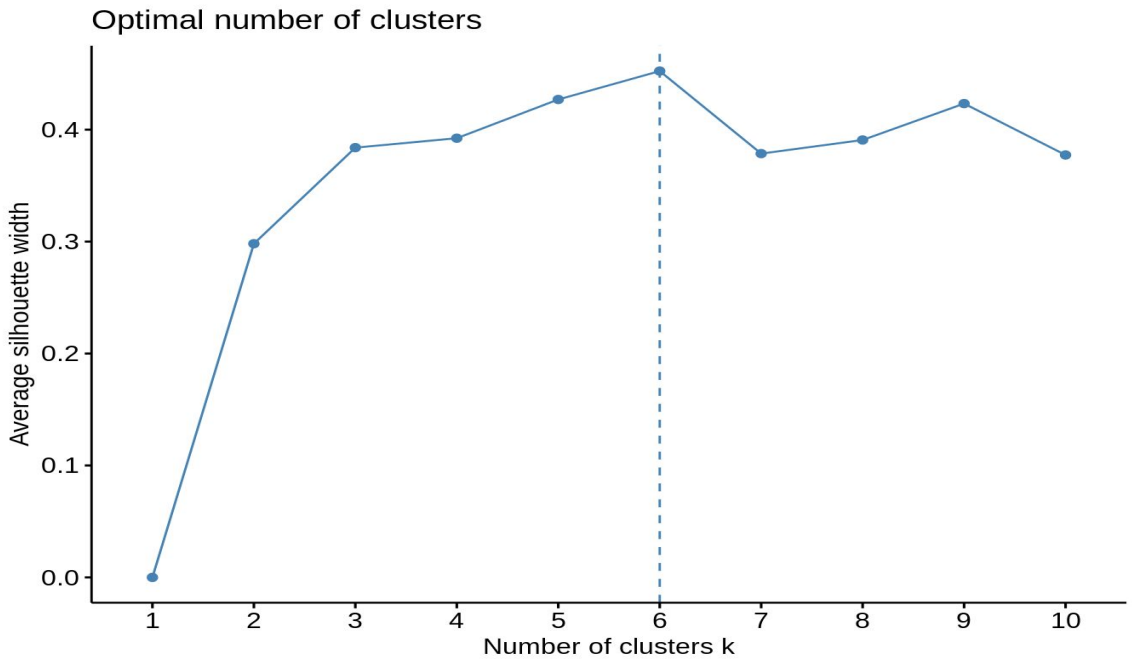
```
library(NbClust)
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.g
l/13EFCZ
```

```
fviz_nbclust(customer_data[,3:5], kmeans, method = "silhouette")
```

Output:



Méthode statistique d'écart

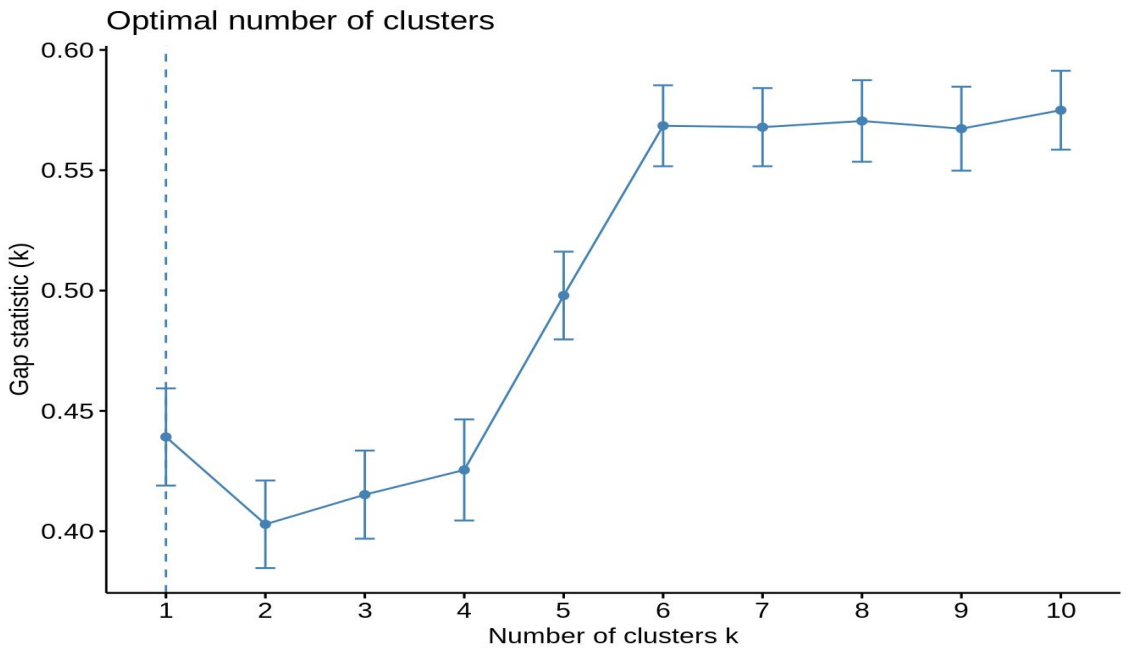
En 2001, des chercheurs de l'Université de Stanford - R. Tibshirani, G.Walther et T. Hastie ont publié la méthode statistique Gap. Nous pouvons utiliser cette méthode pour n'importe quelle méthode de clustering comme les moyennes K, le clustering hiérarchique, etc. En utilisant la statistique de l'écart, on peut comparer la variation totale intracluster pour différentes valeurs de k avec leurs valeurs attendues sous la distribution de référence nulle des données. À l'aide de simulations de Monte Carlo , on peut produire l'échantillon de données. Pour chaque variable de l'ensemble de données, nous pouvons calculer la plage entre min (xi) et max (xj) à travers laquelle nous pouvons produire des valeurs uniformément de l'intervalle inférieur à la limite supérieure.

Pour calculer la méthode des statistiques d'écart, nous pouvons utiliser la fonction clusGap pour fournir des statistiques d'écart ainsi qu'une erreur standard pour une sortie donnée.

Code:

```
# compute gap statistic
set.seed(123)
gap_stat <- clusGap(customer_data[,3:5], FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
fviz_gap_stat(gap_stat)
```

Output:



Maintenant, prenons k = 6 comme cluster optimal -

Code:

```
# compute gap statistic
k6<-kmeans(customer_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
k6
```

```
## K-means clustering with 6 clusters of sizes 45, 22, 21, 38, 35, 39
##
## Cluster means:
##        Age Annual.Income..k.. Spending.Score..1.100.
## 1 56.15556           53.37778                49.08889
## 2 25.27273           25.72727                79.36364
## 3 44.14286           25.14286                19.52381
## 4 27.00000           56.65789                49.13158
## 5 41.68571           88.22857                17.28571
## 6 32.69231           86.53846                82.12821
##
## Clustering vector:
##   [1] 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3
##  [36] 2 3 2 3 2 1 2 1 4 3 2 1 4 4 4 1 4 4 1 1 1 1 1 4 1 1 4 1 1 1 4 1 1 4 4
##  [71] 1 1 1 1 1 4 1 4 4 1 1 4 1 1 4 1 1 4 4 1 1 4 1 4 4 4 1 4 1 4 4 1 1 4 1
```

- **cluster** - C'est un vecteur de plusieurs entiers qui dénote le cluster qui a une allocation de chaque point.
- **totss** - Cela représente la somme totale des carrés.
- **centers** - Matrice comprenant plusieurs centres de cluster
- **withinss** - Il s'agit d'un vecteur représentant la somme intra-cluster de carrés ayant une composante par cluster.
- **tot.withinss** - Ceci dénote la somme totale intra-cluster des carrés.
- **betweenss** - C'est la somme des carrés entre les grappes.
- **size** – Le nombre total de points que chaque cluster détient.

Visualisation des résultats du clustering à l'aide des deux premiers composants principaux

Code:

```
pcclust=prcomp(customer_data[,3:5],scale=FALSE)  #principal component analysis
summary(pcclust)
```

```
## Importance of components:
##                             PC1     PC2     PC3
## Standard deviation     26.4625 26.1597 12.9317
## Proportion of Variance  0.4512  0.4410  0.1078
## Cumulative Proportion   0.4512  0.8922  1.0000
```

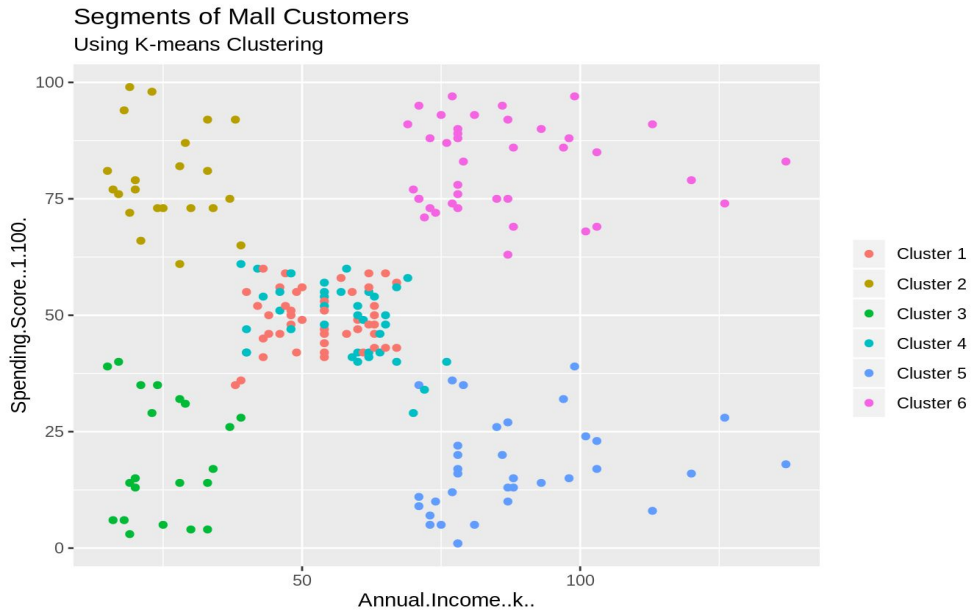
```
pcclust$rotation[,1:2]
```

```
##                               PC1        PC2
## Age                     0.1889742 -0.1309652
## Annual.Income..k..     -0.5886410 -0.8083757
## Spending.Score..1.100. -0.7859965  0.5739136
```

Code:

```
## VISULIASE THE CLUSTERS
set.seed(1)
ggplot(customer_data, aes(x =Annual.Income..k.., y = Spending.Score..1.100.)) +
    geom_point(stat = "identity", aes(color = as.factor(k6$cluster))) +
    scale_color_discrete(name=" ",
                         breaks=c("1", "2", "3", "4", "5","6"),
                         labels=c("Cluster 1", "Cluster 2", "Cluster 3", "Cluster 4",
"Cluster 5","Cluster 6")) +
    ggtitle("Segments of Mall Customers", subtitle = "Using K-means Clustering")
```

Output:



De la visualisation ci-dessus, nous observons qu'il existe une distribution de 6 grappes comme suit -

Cluster 6 et 4 - Ces clusters représentent les données client avec le salaire moyen ainsi que la dépense annuelle moyenne de salaire.

Cluster 1 - Ce cluster représente les données clients ayant un revenu annuel élevé ainsi que des dépenses annuelles élevées.

Cluster 3 - Ce cluster désigne les données client avec un faible revenu annuel ainsi que de faibles dépenses annuelles de revenu.

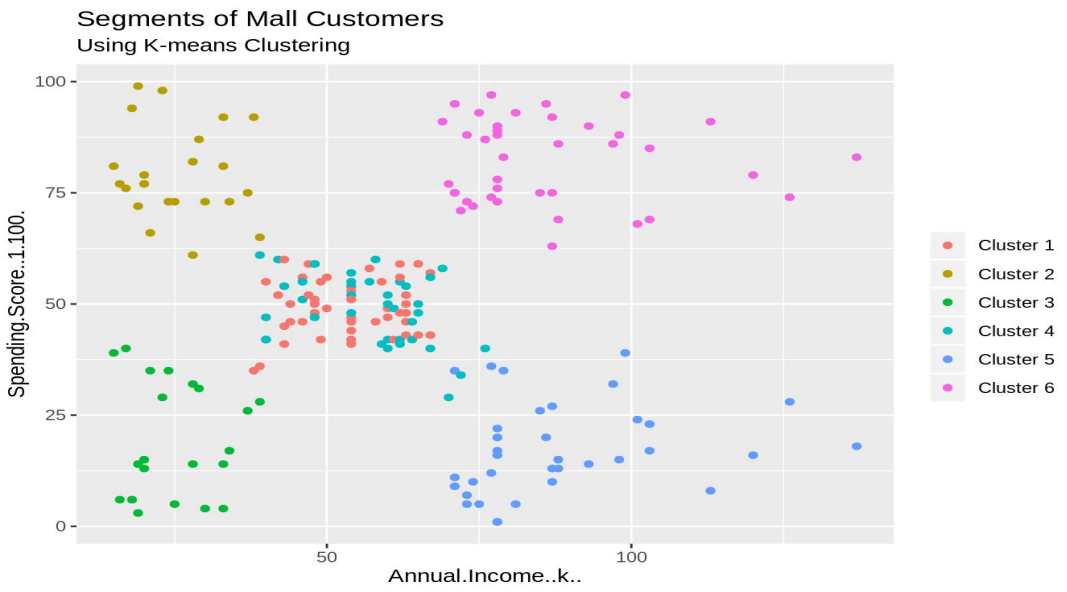
Groupe 2 - Ce groupe dénote un revenu annuel élevé et des dépenses annuelles faibles.

Groupe 5 - Ce groupe représente un faible revenu annuel mais ses dépenses annuelles élevées.

Code:

```
ggplot(customer_data, aes(x =Spending.Score..1.100., y =Age)) +
    geom_point(stat = "identity", aes(color = as.factor(k6$cluster))) +
    scale_color_discrete(name=" ",
                         breaks=c("1", "2", "3", "4", "5","6"),
                         labels=c("Cluster 1", "Cluster 2", "Cluster 3", "Cluster 4",
"Cluster 5","Cluster 6")) +
    ggtitle("Segments of Mall Customers", subtitle = "Using K-means Clustering")
```

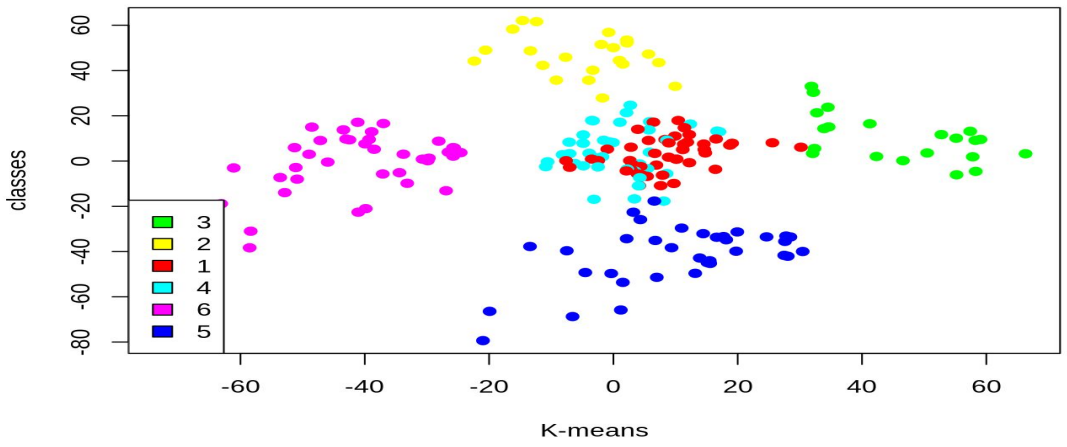
Output:



Code:

```
kCols=function(vec){cols=rainbow (length (unique (vec)))
return (cols[as.numeric(as.factor(vec))])}

digCluster<-k6$cluster; dignm<-as.character(digCluster);    # K-means clusters

plot(pcclust$x[,1:2], col =kCols(digCluster),pch =19,xlab ="K-means",ylab="classes")
legend("bottomleft",unique(dignm),fill=unique(kCols(digCluster)))
```

Output:



Cluster 4 et 1 - Ces deux clusters sont constitués de clients avec un score PCA1 et PCA2 moyen.
Cluster 6 - Ce cluster représente les clients ayant un PCA2 élevé et un PCA1 faible.
Cluster 5 - Dans ce cluster, il y a des clients avec un PCA1 moyen et un score PCA2 faible.
Cluster 3 - Ce cluster comprend des clients avec un revenu PCA1 élevé et un PCA2 élevé.
Cluster 2 - Cela comprend les clients avec un PCA2 élevé et une dépense annuelle moyenne de revenus.
Avec l'aide du clustering, nous pouvons mieux comprendre les variables, nous incitant à prendre des décisions prudentes. Avec l'identification des clients, les entreprises peuvent lancer des produits et services qui ciblent les clients en fonction de plusieurs paramètres comme le revenu, l'âge, les habitudes de dépenses, etc. De plus, des modèles plus complexes comme les avis sur les produits sont pris en considération pour une meilleure segmentation.

Sommaire

Dans ce projet de science des données, nous sommes passés par le modèle de segmentation client. Nous l'avons développé à l'aide d'une classe d'apprentissage automatique appelée apprentissage non supervisé. Plus précisément, nous avons utilisé un algorithme de clustering appelé clustering K-means. Nous avons analysé et visualiser les données, puis nous avons mis en œuvre notre algorithme.