Projet PUBG

Templier Timothée et Paul Le Breton

1. Comprendre et pré-traiter les données

1.1 Chargement et exploration des données

```
##                  WinRatio TimeSurvived RoundsPlayed Wins Top10s Top10Ratio
## Escapist             4.26     35776.89           47    2      6       12.8
## inkop                0.00      2567.00            2    0      0        0.0
## Shanks              13.33     34681.05           30    4      7       23.3
## Ace_Finessed_It      0.00     10618.52           11    0      3       27.3
## MasterOfTheEnder     8.57     36369.04           35    3      5       14.3
## MassacreX479        18.18     41830.97           33    6     15       45.5
##                  Losses DamagePg HeadshotKillsPg HealsPg KillsPg MoveDistancePg
## Escapist             45   235.91            0.30    0.66    1.96        2845.92
## inkop                 2   130.78            0.00    5.00    1.50        2106.94
## Shanks               26   392.68            1.07    0.83    3.37        3350.91
## Ace_Finessed_It      11   230.63            0.27    1.18    2.36        3017.29
## MasterOfTheEnder     32   188.18            0.11    0.91    1.66        2260.34
## MassacreX479         27   345.49            0.52    2.15    3.21        2790.69
##                  TimeSurvivedPg Kills Assists HeadshotKills LongestTimeSurvived
## Escapist                 761.21    92       7            14              2027.78
## inkop                   1283.50     3       0             0              1433.77
## Shanks                  1156.04   101       5            32              1972.75
## Ace_Finessed_It          965.32    26       0             3              1740.82
## MasterOfTheEnder        1039.12    58       0             4              1953.06
## MassacreX479            1267.61   106       8            17              1967.35
##                  WalkDistance RideDistance MoveDistance AvgWalkDistance
## Escapist             63798.99     69959.20    133758.20         1357.42
## inkop                 4213.88         0.00      4213.88         2106.94
## Shanks               45673.62     54853.70    100527.32         1569.02
## Ace_Finessed_It      18966.67     14223.50     33190.17         1724.24
## MasterOfTheEnder     57317.87     21793.88     79111.75         1638.66
## MassacreX479         48168.15     43924.51     92092.66         1599.53
##                  AvgRideDistance Heals Boosts DamageDealt
## Escapist                 1488.49    31     57    11087.66
## inkop                       0.00    10      5      261.55
## Shanks                   1742.89    25     53    11780.34
## Ace_Finessed_It          1293.05    13     18     2536.93
## MasterOfTheEnder          836.82    32     32     6586.29
## MassacreX479             1493.30    71     98    11401.22
```

1.2 Nettoyage/modification des données

Il n'y a ni valeur manquante, ni doublon dans le jeu de données.

Modification de variables

```
# Assists --> AssistsPg nombre moyen d'assistances par partie
data$AssistsPg <- round(data$Assists/data$RoundsPlayed, 2)

# Boosts --> BoostsPg nombre moyen de boosts utilisés par partie
data$BoostsPg <- round(data$Boosts/data$RoundsPlayed, 2)

data <- subset(data, select = -c(Assists, Boosts))
```

Suppression des variables inutiles

```
data <- subset(data, select = -c(LongestTimeSurvived, Top10s, Losses, HeadshotKills, WalkDistance, Ride
colnames(data)
```

```
##  [1] "WinRatio"        "RoundsPlayed"    "Top10Ratio"      "DamagePg"
##  [5] "HeadshotKillsPg" "HealsPg"         "KillsPg"         "MoveDistancePg"
##  [9] "TimeSurvivedPg"  "AvgWalkDistance" "AvgRideDistance" "AssistsPg"
## [13] "BoostsPg"
```

```
data <- data[data$RoundsPlayed>=50,]
```

Pour élaborer un profil de joueur, nous privilégions l'analyse des moyennes statistiques par partie, telles que le pourcentage de victoires, afin d'éviter un biais potentiel lié au temps de jeu. Dans cette optique, nous éliminons les variables suivantes :

_ Les victoires, car nous disposons déjà du ratio de victoires.

_ Les top 10, car nous avons déjà le ratio de top 10.

_ Les éliminations, car nous disposons du nombre moyen d'éliminations par partie.

_ Les soins prodigués, car nous avons le nombre moyen de soins par partie.

_ La distance parcourue en déplacement, car nous avons la distance moyenne parcourue par partie.

_ Le temps de survie, car nous avons le temps moyen de survie par partie.

_ Les dégâts infligés, car nous avons les dégâts moyens infligés par partie.

_ Les éliminations par coup à la tête, car nous avons le nombre moyen d'éliminations par coup à la tête par partie.

_ La distance parcourue en véhicule, car nous avons la distance moyenne parcourue en véhicule par partie.

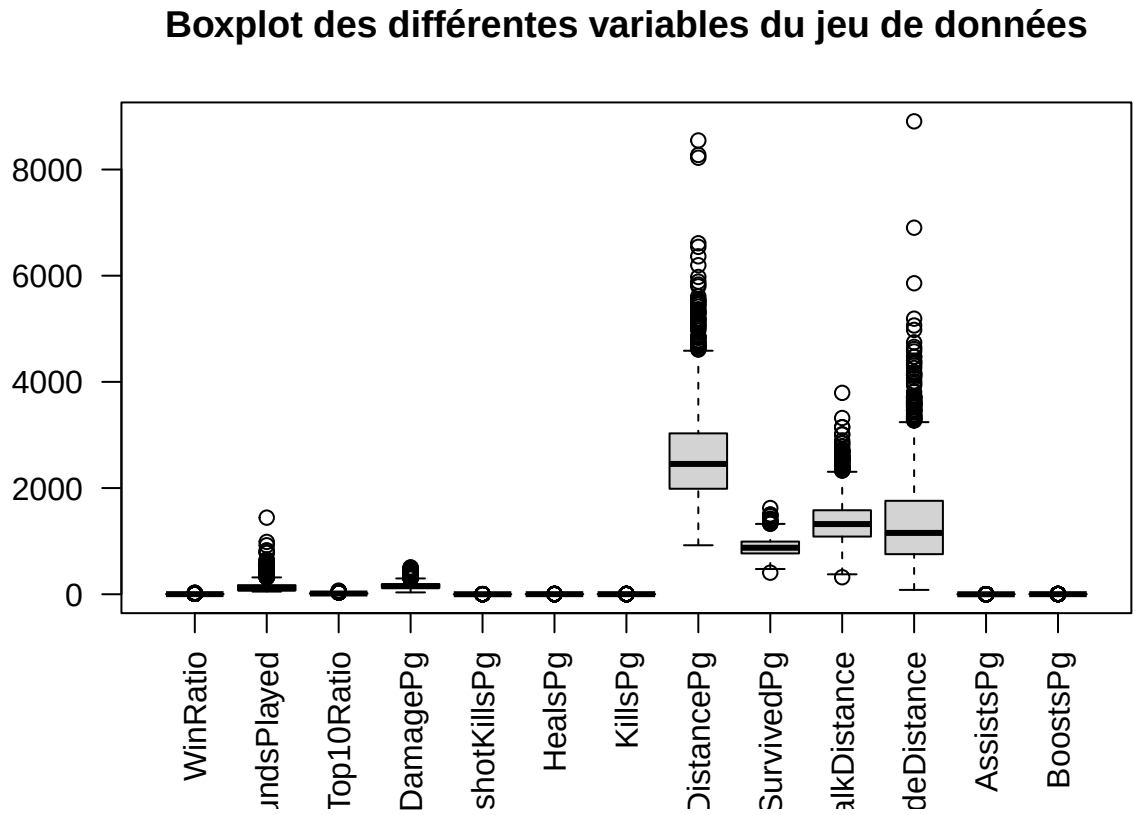
_ La distance parcourue à pied, car nous avons la distance moyenne parcourue à pied par partie.

_ Le temps maximum que le joueur a survécu dans une partie, car si un joueur se cache durant une partie, cela biaise le résultat et donc cela n'aide pas à savoir le style du joueur ou s'il est bon ou non.

Pour éviter tout biais dans les données, nous conservons uniquement les individus ayant participé à au moins 50 parties. Une personne avec un faible nombre de parties aura des statistiques qui auront moins de chances de représenter son niveau réel.

1.3 Analyse des variables

Boxplots



On peut remarquer que certaines colonnes ont des valeurs beaucoup plus élevées que d'autres. Pour éviter que cela impacte trop nos algorithmes, nous allons standardiser les données. De plus, nous avons différents types de données, notamment des temps, des nombres et des distances. C'est pourquoi nous devons les standardiser afin de pouvoir les comparer entre elles de manière cohérente.

Standardisation des données

```
data_sc <- scale(data)
head(data_sc)
```

```
##                  WinRatio RoundsPlayed    Top10Ratio   DamagePg HeadshotKillsPg
## Toffees        -0.8837199   0.12807625  0.0009977722 -0.2595887     -0.42232812
## Hugobugo        0.2896321   0.02613028  0.6732084247  0.1015465      0.09623328
## Robotnutkicker  0.3032362  -0.52067079 -0.1765673058  0.3200431     -0.05933514
## Scoep           0.9800392  -0.64115238  2.7405732614  1.1956503      0.82221923
## Tarz-an        -0.6082372   2.12992423 -0.3160827242 -0.7312690     -0.31861584
## Rag3            2.5173004  -0.57627767  0.7366245240  3.3874236      2.22233500
##                     HealsPg    KillsPg MoveDistancePg TimeSurvivedPg
## Toffees        -0.008480097 -0.2783392     -0.1789070    0.888129384
## Hugobugo        2.716940713 -0.2958953      2.3726024    1.338421823
## Robotnutkicker  0.158382402  0.3361247     -0.5116187   -0.805010840
## Scoep           1.363500447  1.3543792      0.6025964    2.161945236
## Tarz-an         0.010060181 -0.7699103     -0.6033104    0.003985082
## Rag3            1.270799059  3.5313371      0.6595743    0.982647751
##                AvgWalkDistance AvgRideDistance  AssistsPg   BoostsPg
## Toffees          -0.481869500      -1.0947338  1.2577689  0.2672024
```

```
## Hugobugo            3.421370622       2.5868158  1.5117045  0.4433534
## Robotnutkicker     -0.589010275      -0.3925045 -0.7737161 -0.3317111
## Scoep               0.180808001      -0.3547440 -1.0276518  2.4162450
## Tarz-an            -0.009110228      -0.2945714 -0.2658449 -0.5607075
## Rag3                0.489675142       0.5129131  1.5117045  2.0639429
```

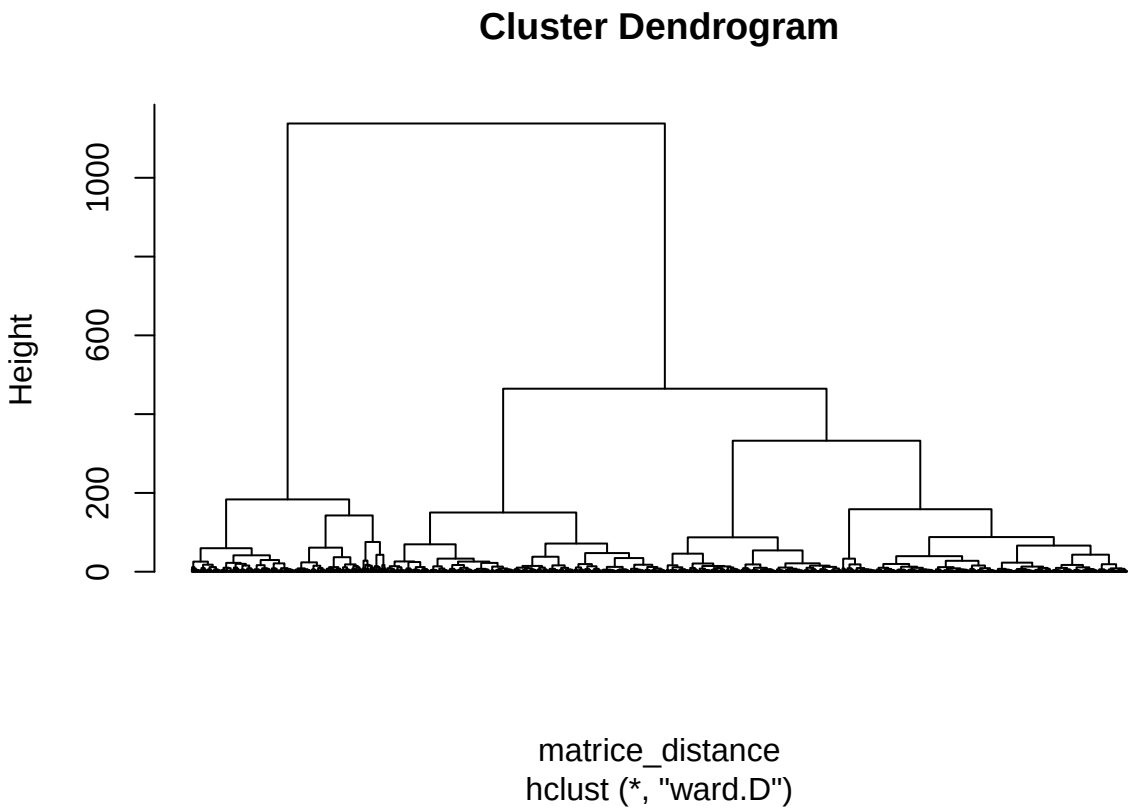
2. Classifier les joueurs en utilisant les différents algorithmes abordés en cours

2.1 Méthode Classification Ascendante Hiérarchique (CAH)

CAH avec distance euclidienne et la stratégie d'agrégation de Ward

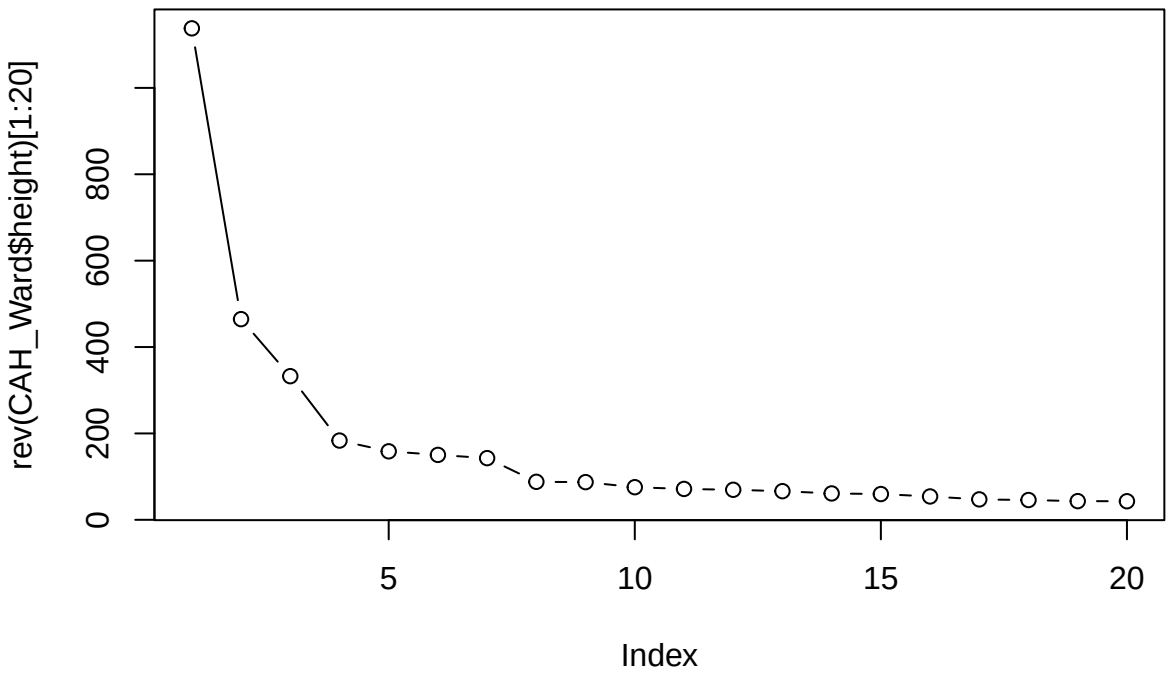
```
matrice_distance <- dist(data_sc, method="euclidean")
CAH_Ward <- hclust(matrice_distance, method="ward.D")
```

Construction dendrogramme



Le dendrogramme nous montre que la plupart des groupes se forment à faible hauteur.

Choix du nombre de classes
Méthode du coude:
Nous allons observer la courbe de perte d'inertie intra-classes.



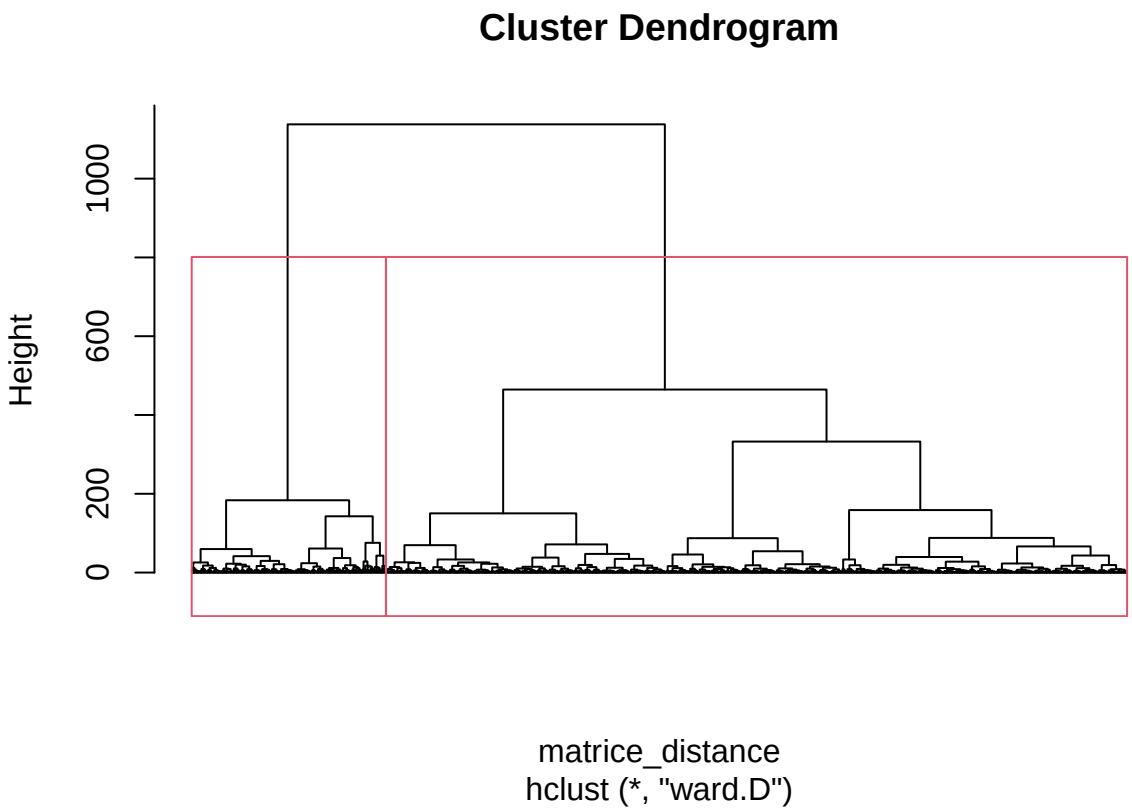
En utilisant la méthode du coude, la courbe nous indique de choisir une partition en 3 groupes.

Méthode avec NbClust:
On peut aussi regarder les critères automatiques calculés dans le package NbClust:

```
NbClust(data_sc, min.nc=2, max.nc=5, method="ward.D", index="all")
```

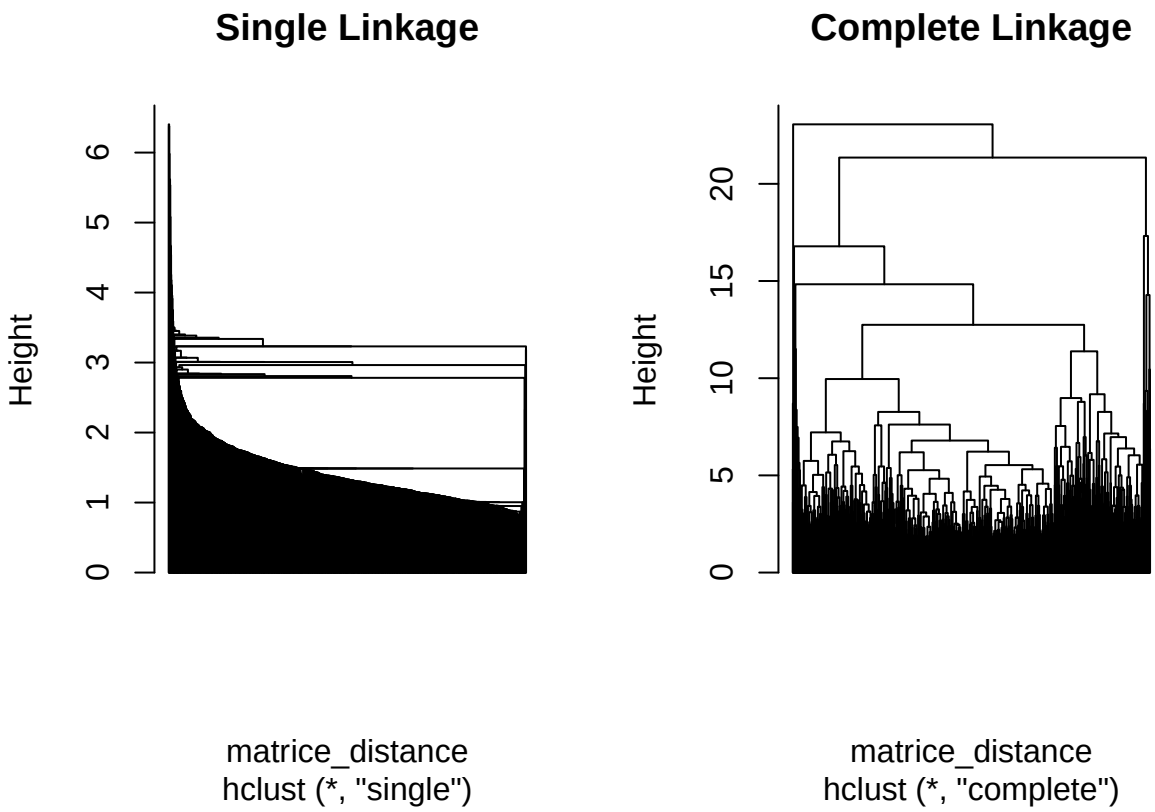
"According to the majority rule, the best number of clusters is 2." Le résultat de cette méthode donne une partition en 2 groupes pour nos données.

Ajout des classes sur le dendrogramme



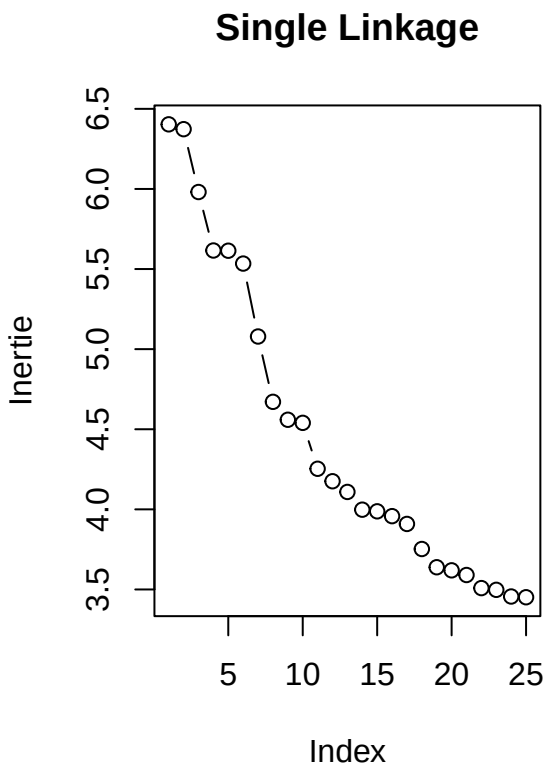
CAH avec d'autres stratégies d'agrégation
Avec les méthodes de la distance du saut minimal et maximal :

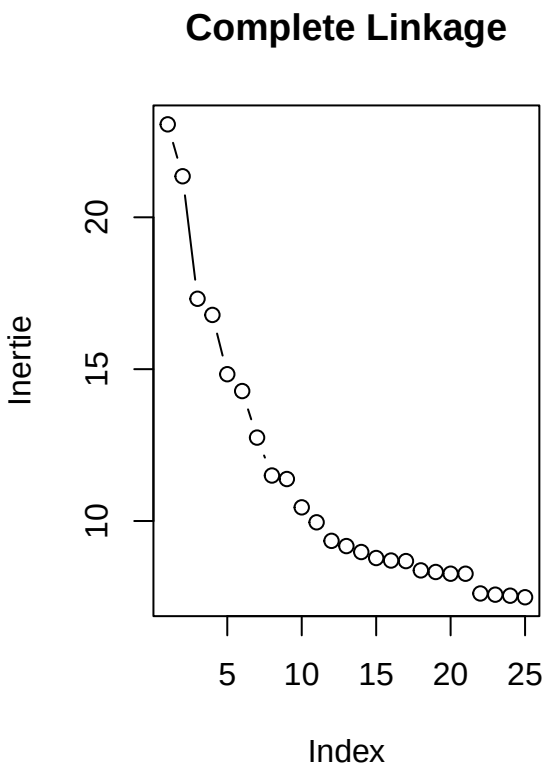
Leurs dendrogrammes :



Ils sont peu informatifs et difficiles à interpréter. On remarque néanmoins une hauteur plus faible que pour la distance de Ward.

Courbes d'intertie:





Avec la méthode de la distance du saut minimal, on aurait tendance à choisir 5 groupes selon la méthode du coude. Pour celle de la distance du saut maximal, on choisrait 2 ou 4 groupes.

On va vérifier ces choix avec la fonction NbClust :

```
NbClust(data_sc, min.nc = 2, max.nc = 10, method = "single", index="all")
```

"According to the majority rule, the best number of clusters is 2."

```
NbClust(data_sc, min.nc = 2, max.nc = 10, method = "complete", index="all")
```

"According to the majority rule, the best number of clusters is 3."

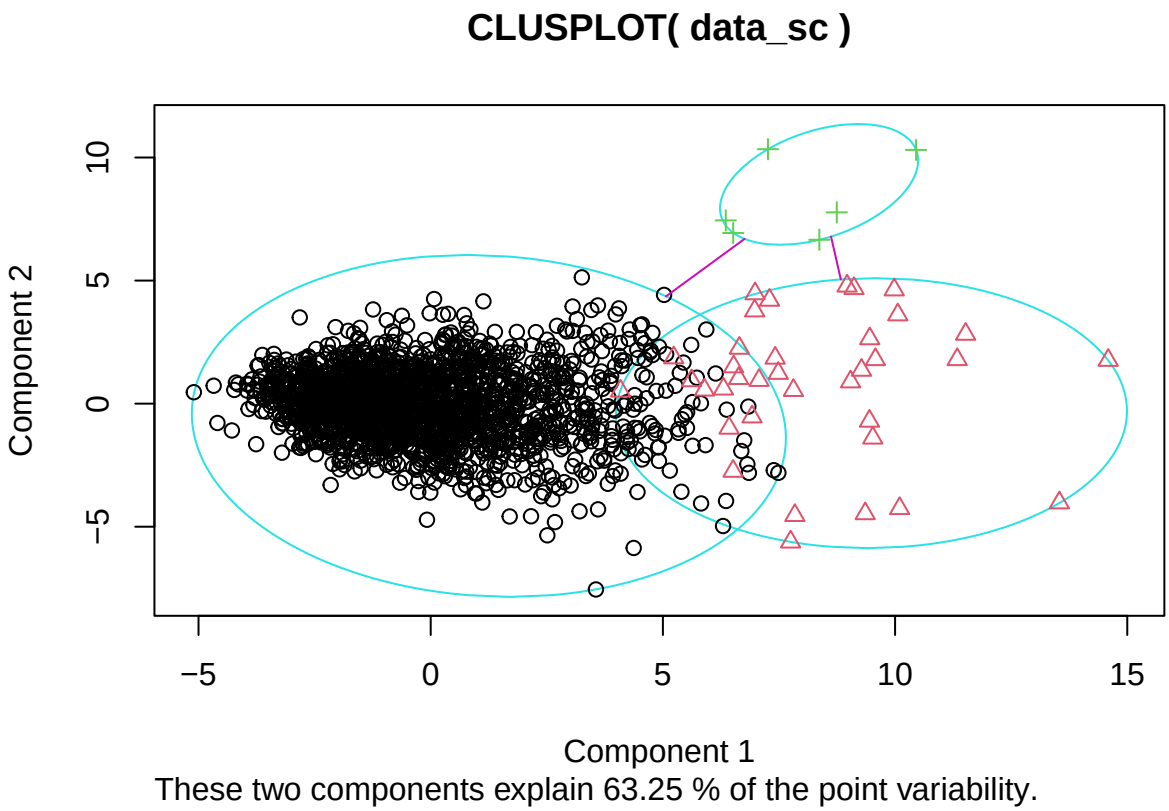
Pour la méthode de la distance du saut minimal et pour celle du saut maximal, on obtient 2 et 3 groupes.

Face à ces résultats, nous privilégions la méthode de classification basée sur la distance maximale. Dans notre jeu de données, il peut être avantageux d'avoir des groupes avec des nombres d'individus différents (joueurs moins performants aux joueurs de niveau intermédiaire, et jusqu'aux joueurs très compétitifs). Ainsi, la méthode basée sur la distance maximale permet de former des clusters distincts avec des profils de joueurs variés, ce qui est cohérent avec la diversité des compétences et des styles de jeu dans le jeu PUBG.

Comparativement, l'utilisation de la distance minimale pourrait conduire à la formation d'un grand cluster avec un grand nombre d'individus, et un autre cluster avec très peu d'individus, ce qui ne représenterait pas bien la diversité des observations dans nos données. De même, avec la méthode de Ward, nous observons une inertie intra-groupe élevée, ainsi que des groupes homogènes qui pourraient ne pas être optimaux pour nos données.

Analalyse des groupes avec la distances de Wards:

```r
gpe.complete <- cutree(CAH_Complete, k = 3)
clusplot(data_sc, gpe.complete, col.p = as.numeric(gpe.complete))
```



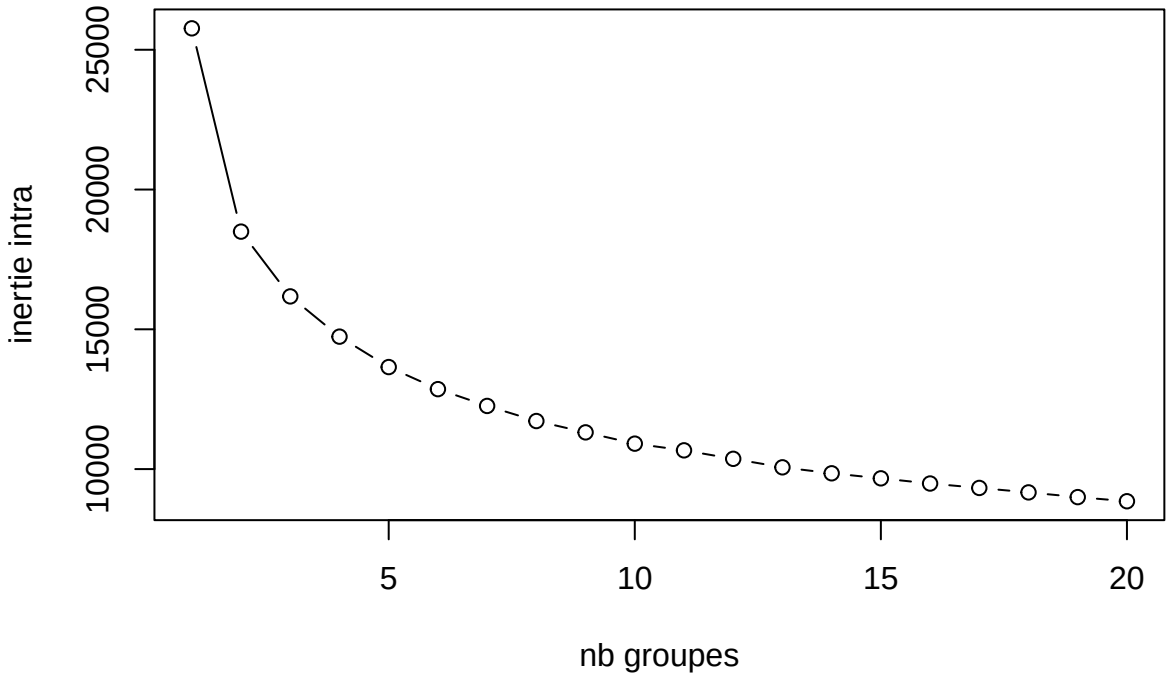
Ce graphe correspond à la représentation des groupes sur les deux premiers axes principaux d'une ACP. De plus, des ellipses de contour autour des groupes sont tracées.

Cela signifie que les deux composantes principales (axes) du graphique de la CAH représentent ensemble 63.25% de la variance totale des données. (expliquent 63.25% de l'information contenue dans les données dans un espace en deux dimensions)

Les 3 groupes sont reconnaissables mais superposés à certains endroits, particulièrement au centre du graphique pour les groupes rouge et noir.

2.2 Méthode Centres Mobiles (K-means)

Choix du nombre de groupes optimal
Méthode du coude:



A l'aide de la méthode du coude et d'une observation de la courbe de perte d'inertie intra, on aurait tendance ici à choisir 2 à 3 groupes. Cependant, le choix étant compliqué, on va lancer l'algorithme pour différentes valeurs de K et comparer les résultats.

```
kmeans_2 <- kmeans(data_sc, centers = 2, nstart = 50, iter.max = 500)
kmeans_3 <- kmeans(data_sc, centers = 3, nstart = 50, iter.max = 500)
```

Effectifs des clusters pour les différentes valeurs de K choisi:

```
table(kmeans_2$cluster)
```

```
##
##    1    2
##  535 1448
```

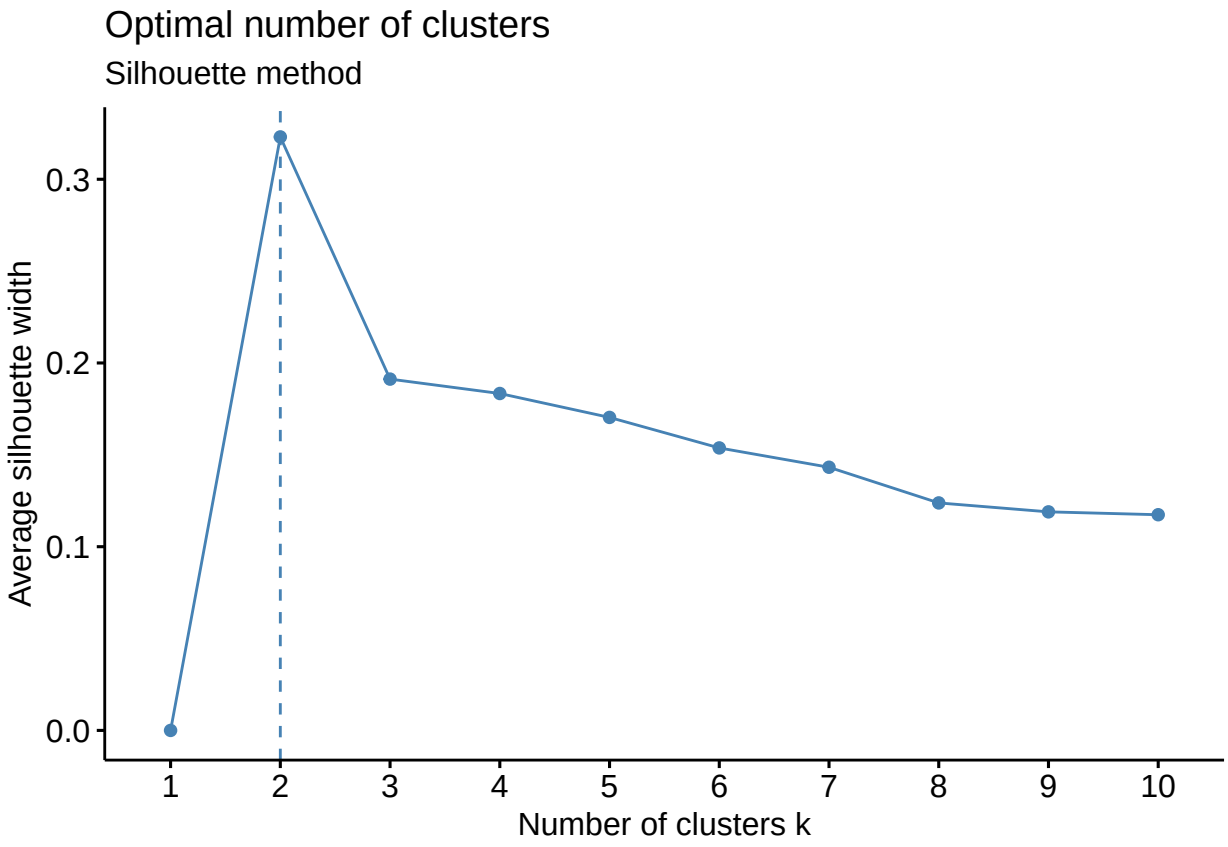
```
table(kmeans_3$cluster)
```

```
##
##   1   2   3
## 232 764 987
```

On peut voir que dans le cas ou il y a 3 clusters choisi, on observe que l'un des groupes est sous-représenté avec 232 individus. On aura donc tendance à choisir 2 groupes pour cette methode.

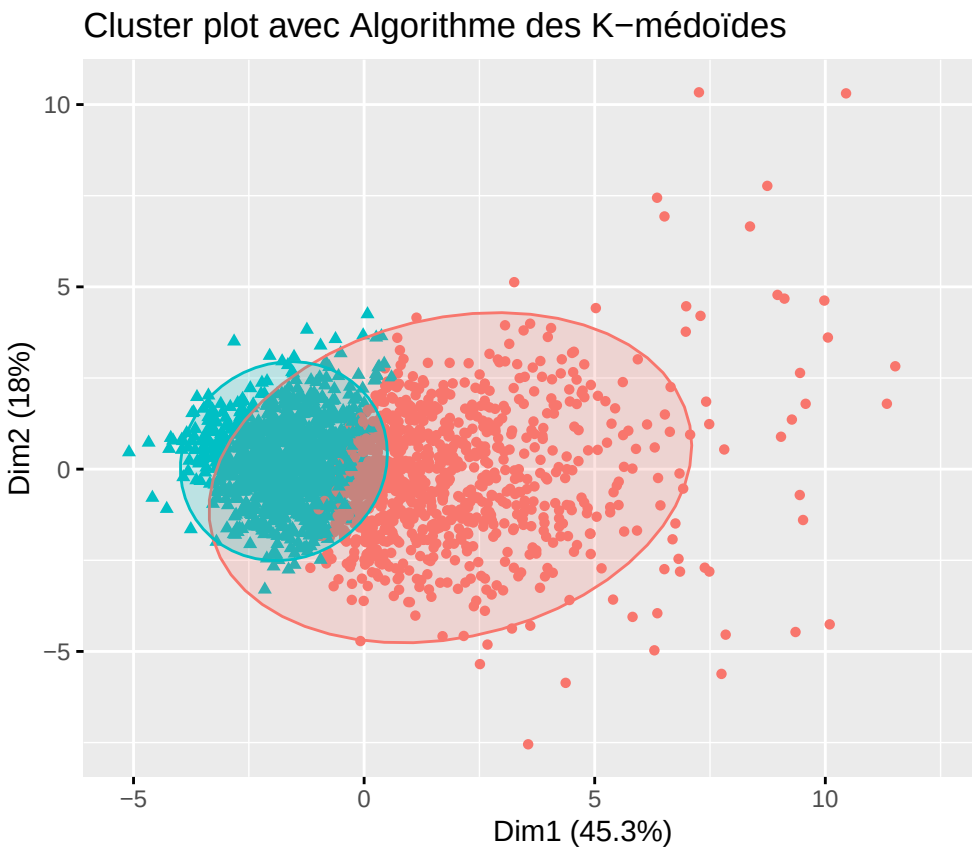
Vérifions si cette réflexion est bonne à l'aide de la méthode d'indice Silhouette moyen.

Méthode Silhouette moyen:



Comme le montre bien ce graphique, le nombre optimal de clusters est donc 2.

Cependant, on sait uqe l'algorithme des K-means est peu robuste aux valeurs extrêmes et aux grands jeu de données. On va donc mettre en place des algorithmes variants à celui des centres mobiles pour les comparer celui-ci.



2.2.1 Algorithme des K-médoïdes

On va donc comparer les groupes formés entre les 2 algorithmes.

```
all.equal(kmedoids_2$clustering, kmeans_2$cluster)
```

```
## [1] "Mean relative difference: 1"
```

On obtient une différence moyenne de 0.6 entre les affectations de cluster des 2 algorithmes ce qui montre une différence dans les regroupements obtenus pour nos données.

On va donc comparer les performances des 2 ensembles de clusters en utilisant l'indice silhouette pour choisir quel est le meilleur résultat:

```
## Moyenne de l'indice de silhouette pour k-means: 0.3230519
```

```
## Moyenne de l'indice de silhouette pour k-medoids: 0.2300202
```

Une moyenne plus élevée pour la méthode des K-means de base indique une meilleure séparation entre les clusters. Ainsi, on va retenir cet algorithme car c'est celui qui a généré les clusters avec la meilleure cohésion interne.

2.3 Modèles de mélange

On va utiliser un modèle de mélange gaussien car les distributions des caractéristiques des joueurs de PUBG pourraient être représentées par des clusters qui ne suivent pas nécessairement des distributions simples ou uniformes. Ce choix permet donc de capturer cette compléxité et de classifier efficacement les joueurs.

Création du modèle de mélange gaussien

```r
mclust_model <- Mclust(data_sc)

summary(mclust_model)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 4
## components:
##
##  log-likelihood    n  df       BIC       ICL
##       -13348.85 1983 419 -29878.91 -30074.22
##
## Clustering table:
##   1   2   3   4
## 612 813 446 112
```

Le modèle a été ajusté avec 4 composantes, ce qui indique une complexité relativement élevée de la structure de nos données. Les valeurs log-vraisemblance, BIC et ICL fournissent des indications sur l'adéquation du modèle, avec des valeurs de BIC et ICL plus basses indiquant un meilleur ajustement. Enfin, les effectifs dans chaque groupe suggèrent une répartition inégale des observations entre les différents groupes étudiés.

Extraction du nombre optimal de groupes identifié par la fonction Mclust

```r
best_model <- Mclust(data_sc)
best_model$G
```

```
## [1] 4
```

Le modèle de mélange gaussien a identifié 4 comme étant le nombre optimal de groupes.

3. Comparer les résultats obtenus avec les différentes méthodes

3.1 Evaluation quantitative

Indice silhouette moyen

```
## Indice silhouette moyen pour la CAH: 0.5551178
```

```
## Indice silhouette moyen pour les K-means: 0.3230519
```

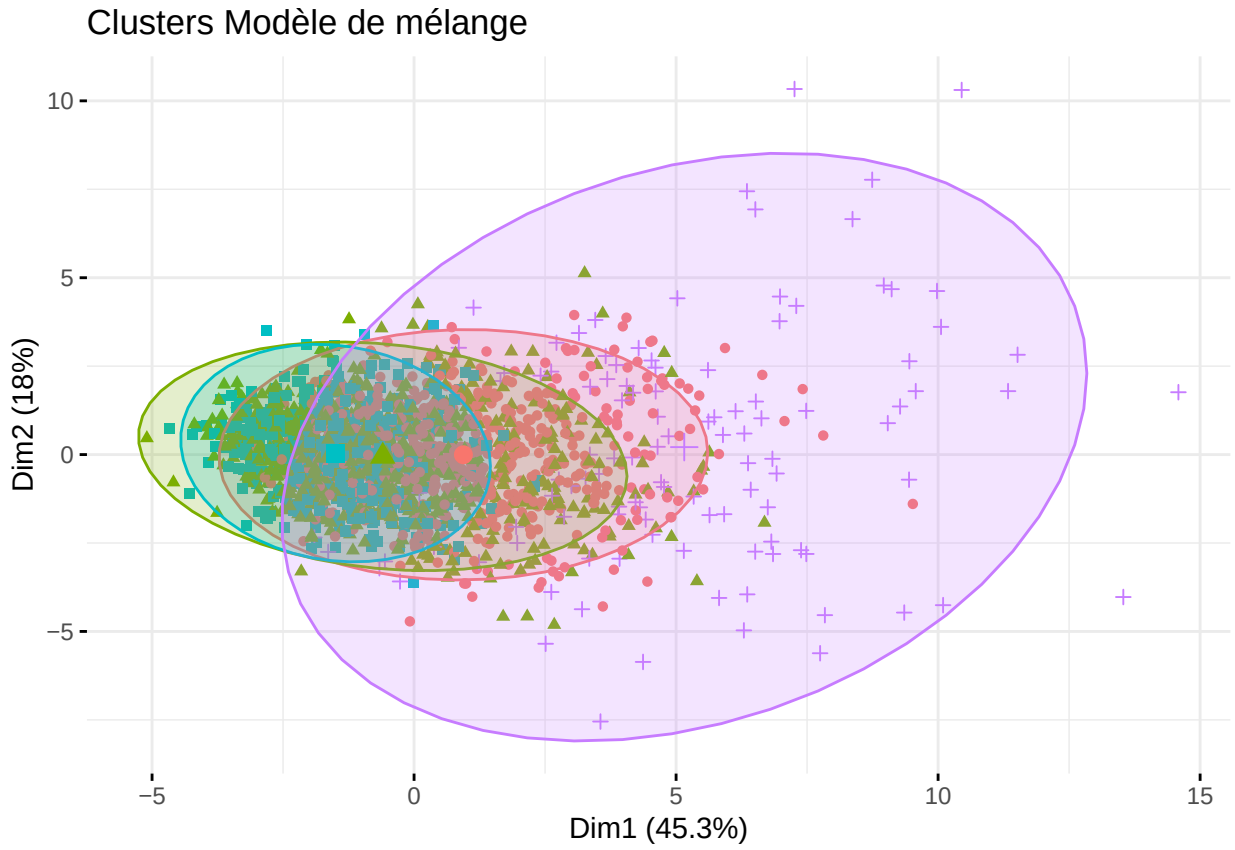
```
## Indice silhouette moyen pour le modèle de mélange gaussien: -0.002808339
```

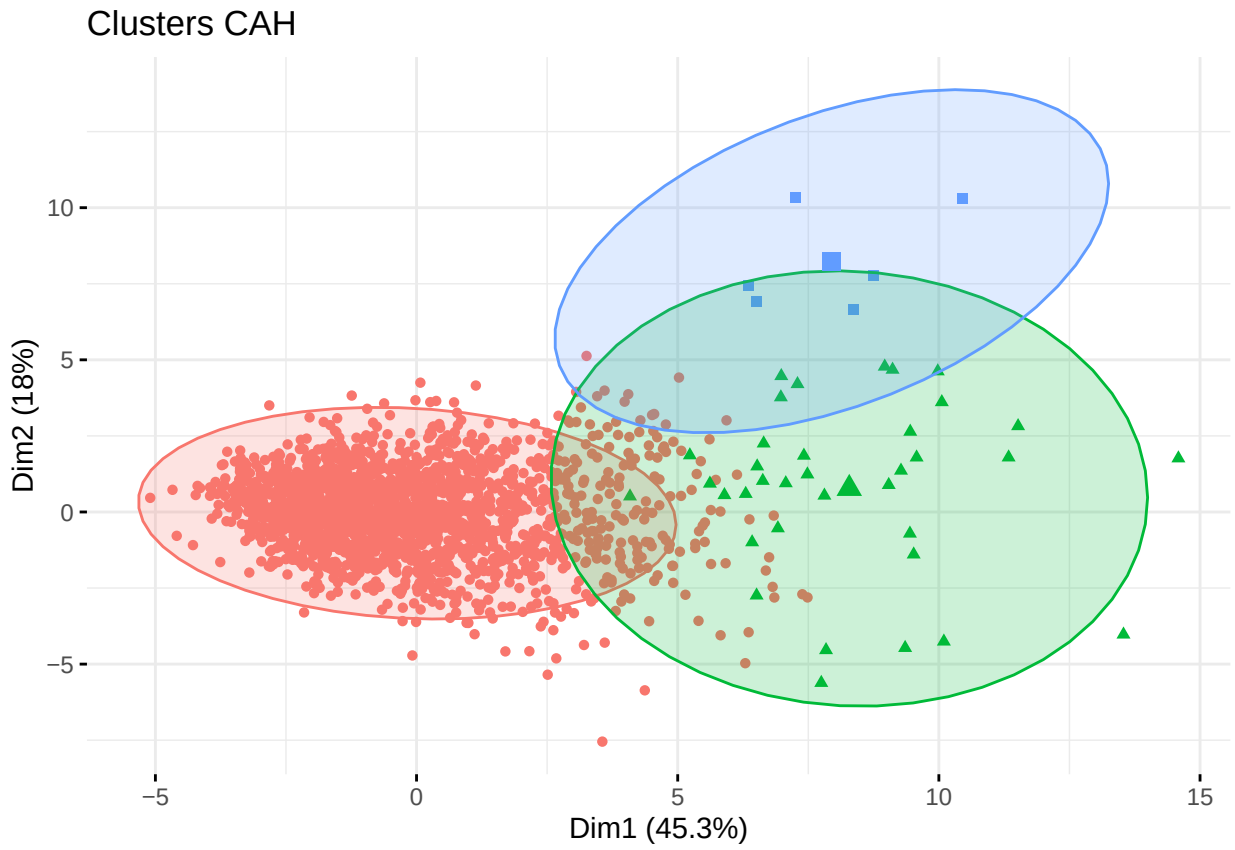
Les résultats montrent que l'algorithme CAH a obtenu la meilleure séparation entre les clusters.

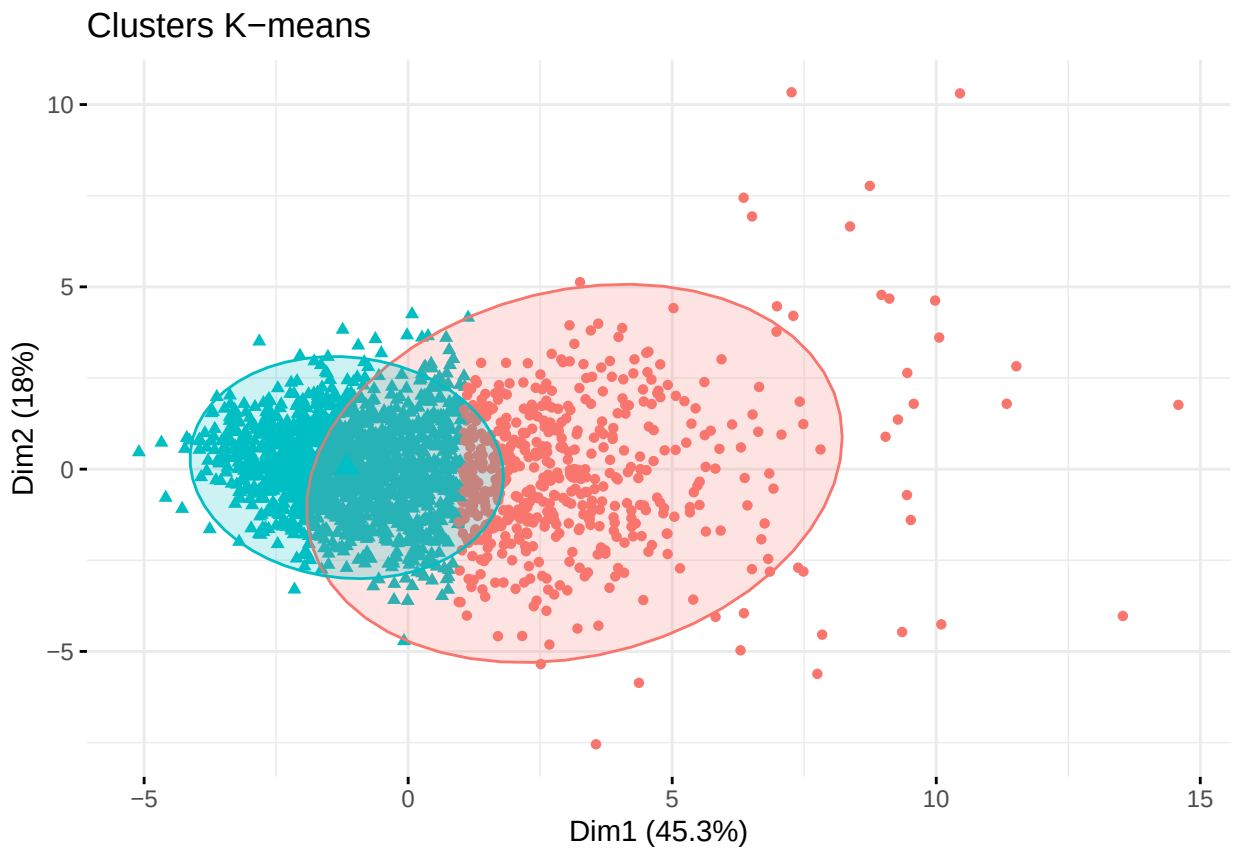
On peut aussi voir que l'indice silhouette moyen pour le modèle de mélange gaussien est négatif, cela suggère une très mauvaise qualité de classification.

L'indice silhouette de l'algorithme des K-means n'est pas très éloigné de celui de la CAH, ce qui pourrait indiquer une performance relativement similaire entre ces deux méthodes.

3.2 Visualisation des clusters







Les graphiques illustrant les clusters semblent montrer que la méthode des K-means et la méthode CAH offrent une séparation nette. Pour l'analyse des différents groupes d'individus, nous allons donc choisir qu'elle est la meilleur méthode de classification entre ces 2 algorithmes.

3.3 Choix de la meilleur méthode algorithme CAH et K-means

Comparaison des affectations de clusters entre la CAH et les K-means

```
## [1] "Mean relative difference: 0.4517658"
```

Une différence moyenne de 45.17% entre les affectations de clusters des 2 méthodes CAH et K-means montre une divergence notable dans la manière dont les observations sont regroupées entre la CAH et les K-means.

Calcul et analyse des inerties

```
## Inertie intra-cluster pour la CAH: 4315.094
```

```
## Inertie inter-cluster pour la CAH: 4292.033
```

```
## Inertie intra-cluster pour les K-means: 18494.91
```

```
## Inertie inter-cluster pour les K-means: 11.27888
```

Les résultats montrent que la CAH offre une répartition des clusters avec une cohésion et une séparation équilibrées, tandis que K-means semble avoir une dispersion plus importante des points à l'intérieur des clusters. En conséquence, pour des clusters plus cohérents et distincts, la CAH peut être préférée, tandis que K-means peut être plus adapté si la dispersion des points est acceptable ou recherchée.

4. Conclure vis à vis des choix effectués

Après avoir exploré et appliqué les différentes méthodes de clustering sur les données des joueurs de PUBG, nous pouvons tirer les conslusions suivantes:

En comparant les différentes évaluations telles que l'indice silhouette moyen et l'inertie intra-cluster, nous avons constaté que les CAH a produit des clusters plus cohérents et mieux séparés parmi les méthodes testées. Cela suggère que la méthode des CAH est la plus appropriée pour notre analyse des joueurs de PUBG.
On va donc retenir la méthode des CAH pour la description des différents profils types de joueurs. En effet, bien que la méthode k-means ait produit des résultats intéressants, les CAH ont démontré une meilleure performance globale en termes de séparation des clusters et de cohésion interne.

5. Suggérer une description des différents profils types de joeuurs

Analyse statistiques descriptives

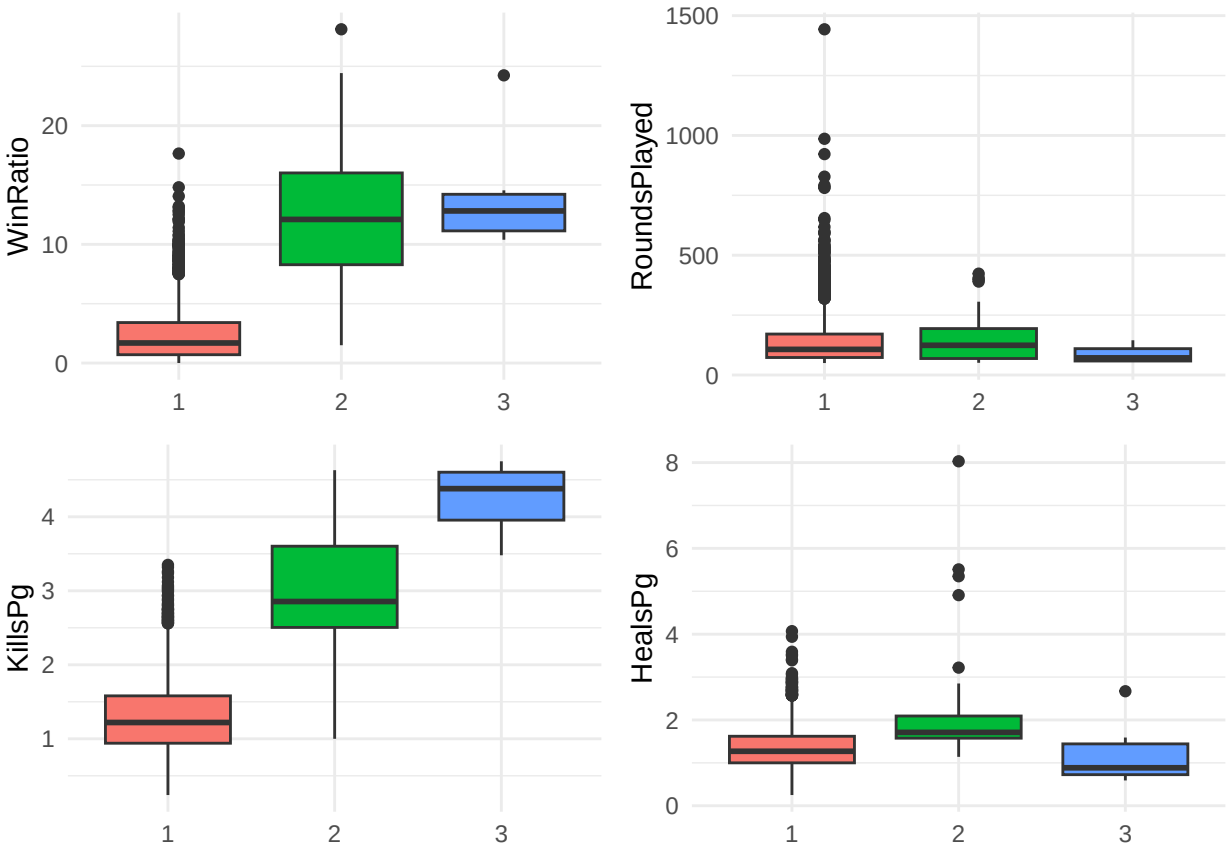
```
## # A tibble: 3 x 14
##   Group_CAH WinRatio RoundsPlayed Top10Ratio DamagePg HeadshotKillsPg HealsPg
##       <int>    <dbl>        <dbl>      <dbl>    <dbl>           <dbl>   <dbl>
## 1         1     2.37        142.        15.0     157.           0.278    1.35
## 2         2    12.9         156.        37.0     337.           0.721    2.25
## 3         3    14.3          86.7       27.5     457.           2.21     1.22
## # i 7 more variables: KillsPg <dbl>, MoveDistancePg <dbl>,
## #   TimeSurvivedPg <dbl>, AvgWalkDistance <dbl>, AvgRideDistance <dbl>,
## #   AssistsPg <dbl>, BoostsPg <dbl>
```

En analysant les moyennes des variables en fonction de chacun des 3 groupes, on peut constater que le groupe 3 se démarque comme étant le meilleur, affichant un ratio de victoire supérieur. Cette différence pourrait être expliquée par le fait que le troisième groupe présente de meilleures statistiques dans les catégories suivantes : Top10Ratio, DamagePg, HeadshotKillsPg, KillsPg, et TimeSurvivedPg.

Ces variables semblent être les plus discriminantes pour identifier un joueur de haut niveau par rapport à un joueur ordinaire.

Pour la différence entre le groupe un et deux, on remarque que le groupe deux, composé de joueurs intermédiaires, affiche de meilleures statistiques dans l'ensemble, notamment en ce qui concerne leur mobilité et leur capacité à effectuer des éliminations par partie. Ces deux variables semblent jouer un rôle important dans la distinction entre un joueur de faible niveau et un joueur intermédiaire

Analyse graphique



Conclusion

L'analyse des trois groupes de joueurs de PUBG révèle des profils distincts en termes de performances et de stratégies de jeu. Le groupe 1 se caractérise par des performances plus modérées, présentant moins de kills et une utilisation moins fréquente de soins et de boosts. En revanche, le groupe 3 se distingue par des performances élevées, affichant un nombre de kills par partie supérieur et un recours moins fréquent aux soins, tandis que le deuxième groupe présente des chiffres légèrement supérieurs en termes de kills et de soins par rapport au groupe 1. Ces différences suggèrent une disparité dans le niveau de compétence entre les trois groupes, bien que le groupe 3 ait un nombre total de parties jouées inférieur.

Enfin, il convient de noter que les résultats peuvent être biaisés par le fait que de nombreux joueurs du groupe 3 ont peu de parties jouées, ce qui pourrait entraîner des résultats moyens globalement meilleurs pour ce groupe que pour les autres, malgré le critère de sélection imposant un minimum de 50 parties jouées.