Characterization of professional football players based on performance data using Gaussian mixture clustering models

César Soto Valero

January 10, 2017

Introduction

This is a **R** clustering analysis of the ultimate soccer database for data analysis and machine learning. We use principal components and the **mclust** package for classifing players according to the SOFIFA scores. For more details about the data source see https://github.com/hugomathien/football-data-collection/tree/master/footballData.

Loading required packages

```
library(RSQLite)
library(tidyr)
library(dplyr)
library(data.table)
library(ggplot2)
library(mclust)
library(xgboost)
library(Matrix)
library(psych)
library(caret)
library(xlsx)
```

Loading data

```
con <- dbConnect(SQLite(), dbname="C:/database.sqlite")
```

Data exploration and tidying

Browsing data

```
#list all tables
dbListTables(con)
```

```
## [1] "Country"         "League"          "Match"           "Player"
## [5] "Player_Stats"    "Team"            "sqlite_sequence"
```

```
#list all field on table
dbListFields(con, "Player")
```

```
## [1] "id"                 "player_api_id"      "player_name"
## [4] "player_fifa_api_id" "birthday"           "height"
## [7] "weight"
```

```
dbListFields(con, "Player_Stats")
```

```
##  [1] "id"                  "player_fifa_api_id"  "player_api_id"
##  [4] "date_stat"           "overall_rating"      "potential"
##  [7] "preferred_foot"      "attacking_work_rate" "defensive_work_rate"
## [10] "crossing"            "finishing"           "heading_accuracy"
## [13] "short_passing"       "volleys"             "dribbling"
## [16] "curve"               "free_kick_accuracy"  "long_passing"
## [19] "ball_control"        "acceleration"        "sprint_speed"
## [22] "agility"             "reactions"           "balance"
## [25] "shot_power"          "jumping"             "stamina"
## [28] "strength"            "long_shots"          "aggression"
## [31] "interceptions"       "positioning"         "vision"
## [34] "penalties"           "marking"             "standing_tackle"
## [37] "sliding_tackle"      "gk_diving"           "gk_handling"
## [40] "gk_kicking"          "gk_positioning"      "gk_reflexes"
```

Import tables of interest

```
player <- dbGetQuery(con, "SELECT * FROM player")
player_stats <- dbGetQuery(con, "SELECT * FROM Player_Stats")
```

Merge tables

```
merged_table <- merge(player, player_stats, by = "player_api_id")

#descriptive statistics of the players dataset
summary(merged_table)
```

```
##  player_api_id         id.x       player_name        player_fifa_api_id.x
##  Min.   :  2625   Min.   :    1   Length:183253      Min.   :     2
##  1st Qu.: 34967   1st Qu.: 2754   Class :character   1st Qu.:155877
##  Median : 78411   Median : 5532   Mode  :character   Median :183518
##  Mean   :136270   Mean   : 5525                      Mean   :165799
##  3rd Qu.:191616   3rd Qu.: 8257                      3rd Qu.:199895
##  Max.   :750584   Max.   :22127                      Max.   :234141
##
##    birthday             height          weight           id.y
##  Length:183253      Min.   :157.5   Min.   :117.0   Min.   :     1
##  Class :character   1st Qu.:177.8   1st Qu.:159.0   1st Qu.: 46006
##  Mode  :character   Median :182.9   Median :168.0   Median : 92004
```

```
##                    Mean   :181.9   Mean   :168.8   Mean   : 92084
##                    3rd Qu.:185.4   3rd Qu.:179.0   3rd Qu.:138003
##                    Max.   :208.3   Max.   :243.0   Max.   :367597
##
##  player_fifa_api_id.y  date_stat         overall_rating   potential
##  Min.   :     2       Length:183253      Min.   :33.0   Min.   :39.00
##  1st Qu.:155885       Class :character   1st Qu.:64.0   1st Qu.:69.00
##  Median :183540       Mode  :character   Median :69.0   Median :74.00
##  Mean   :165834                          Mean   :68.6   Mean   :73.46
##  3rd Qu.:199912                          3rd Qu.:73.0   3rd Qu.:78.00
##  Max.   :234141                          Max.   :94.0   Max.   :97.00
##                                          NA's   :85     NA's   :85
##  preferred_foot     attacking_work_rate defensive_work_rate
##  Length:183253      Length:183253       Length:183253
##  Class :character   Class :character    Class :character
##  Mode  :character   Mode  :character    Mode  :character
##
##
##
##
##     crossing       finishing     heading_accuracy short_passing
##  Min.   : 1.00   Min.   : 1.00   Min.   : 1.00    Min.   : 3.00
##  1st Qu.:45.00   1st Qu.:34.00   1st Qu.:49.00    1st Qu.:57.00
##  Median :59.00   Median :53.00   Median :60.00    Median :65.00
##  Mean   :55.09   Mean   :49.92   Mean   :57.27    Mean   :62.43
##  3rd Qu.:68.00   3rd Qu.:65.00   3rd Qu.:68.00    3rd Qu.:72.00
##  Max.   :95.00   Max.   :97.00   Max.   :98.00    Max.   :97.00
##  NA's   :85      NA's   :85      NA's   :85       NA's   :85
##     volleys        dribbling         curve       free_kick_accuracy
##  Min.   : 1.00   Min.   : 1.00   Min.   : 2.00   Min.   : 1.00
##  1st Qu.:35.00   1st Qu.:52.00   1st Qu.:41.00   1st Qu.:36.00
##  Median :52.00   Median :64.00   Median :56.00   Median :50.00
##  Mean   :49.47   Mean   :59.18   Mean   :52.97   Mean   :49.38
##  3rd Qu.:64.00   3rd Qu.:72.00   3rd Qu.:67.00   3rd Qu.:63.00
##  Max.   :93.00   Max.   :97.00   Max.   :94.00   Max.   :97.00
##  NA's   :1962    NA's   :85      NA's   :1962    NA's   :85
##   long_passing    ball_control    acceleration    sprint_speed
##  Min.   : 3.00   Min.   : 5.00   Min.   :10.00   Min.   :12.00
##  1st Qu.:49.00   1st Qu.:58.00   1st Qu.:61.00   1st Qu.:62.00
##  Median :59.00   Median :67.00   Median :69.00   Median :69.00
##  Mean   :57.07   Mean   :63.39   Mean   :67.66   Mean   :68.05
##  3rd Qu.:67.00   3rd Qu.:73.00   3rd Qu.:77.00   3rd Qu.:77.00
##  Max.   :97.00   Max.   :97.00   Max.   :97.00   Max.   :97.00
##  NA's   :85      NA's   :85      NA's   :85      NA's   :85
##     agility        reactions       balance        shot_power
##  Min.   :11.00   Min.   :17.0   Min.   :12.00   Min.   : 2.00
##  1st Qu.:58.00   1st Qu.:61.0   1st Qu.:58.00   1st Qu.:54.00
##  Median :68.00   Median :67.0   Median :67.00   Median :65.00
##  Mean   :65.97   Mean   :66.1   Mean   :65.19   Mean   :61.81
##  3rd Qu.:75.00   3rd Qu.:72.0   3rd Qu.:74.00   3rd Qu.:73.00
##  Max.   :96.00   Max.   :96.0   Max.   :96.00   Max.   :97.00
##  NA's   :1962    NA's   :85     NA's   :1962    NA's   :85
##     jumping         stamina         strength       long_shots
##  Min.   :14.00   Min.   :10.00   Min.   :10.00   Min.   : 1.00
```

```
##  1st Qu.:60.00   1st Qu.:61.00   1st Qu.:60.00   1st Qu.:41.00
##  Median :68.00   Median :69.00   Median :69.00   Median :58.00
##  Mean   :66.97   Mean   :67.04   Mean   :67.43   Mean   :53.34
##  3rd Qu.:74.00   3rd Qu.:76.00   3rd Qu.:76.00   3rd Qu.:67.00
##  Max.   :96.00   Max.   :96.00   Max.   :96.00   Max.   :96.00
##  NA's   :1962    NA's   :85      NA's   :85      NA's   :85
##    aggression    interceptions    positioning        vision
##  Min.   : 6.00   Min.   : 1.00   Min.   : 2.00   Min.   : 1.00
##  1st Qu.:51.00   1st Qu.:34.00   1st Qu.:45.00   1st Qu.:49.00
##  Median :64.00   Median :57.00   Median :60.00   Median :60.00
##  Mean   :60.95   Mean   :52.01   Mean   :55.79   Mean   :57.87
##  3rd Qu.:73.00   3rd Qu.:68.00   3rd Qu.:69.00   3rd Qu.:69.00
##  Max.   :97.00   Max.   :96.00   Max.   :96.00   Max.   :97.00
##  NA's   :85      NA's   :85      NA's   :85      NA's   :1962
##    penalties     marking      standing_tackle sliding_tackle
##  Min.   : 2   Min.   : 1.00   Min.   : 1.00   Min.   : 2
##  1st Qu.:45   1st Qu.:25.00   1st Qu.:29.00   1st Qu.:25
##  Median :57   Median :50.00   Median :56.00   Median :53
##  Mean   :55   Mean   :46.77   Mean   :50.35   Mean   :48
##  3rd Qu.:67   3rd Qu.:66.00   3rd Qu.:69.00   3rd Qu.:67
##  Max.   :96   Max.   :96.00   Max.   :95.00   Max.   :95
##  NA's   :85   NA's   :85      NA's   :85      NA's   :1962
##    gk_diving     gk_handling      gk_kicking gk_positioning
##  Min.   : 1.0   Min.   : 1.00   Min.   : 1   Min.   : 1.00
##  1st Qu.: 7.0   1st Qu.: 8.00   1st Qu.: 8   1st Qu.: 8.00
##  Median :10.0   Median :11.00   Median :12   Median :11.00
##  Mean   :14.7   Mean   :16.06   Mean   :21   Mean   :16.13
##  3rd Qu.:13.0   3rd Qu.:15.00   3rd Qu.:15   3rd Qu.:15.00
##  Max.   :94.0   Max.   :93.00   Max.   :97   Max.   :96.00
##  NA's   :85     NA's   :85      NA's   :85   NA's   :85
##   gk_reflexes
##  Min.   : 1.00
##  1st Qu.: 8.00
##  Median :11.00
##  Mean   :16.44
##  3rd Qu.:15.00
##  Max.   :96.00
##  NA's   :85
```

```
dim(merged_table)
```

```
## [1] 183253     48
```

```
colnames(merged_table)
```

```
##  [1] "player_api_id"        "id.x"                 "player_name"
##  [4] "player_fifa_api_id.x" "birthday"             "height"
##  [7] "weight"               "id.y"                 "player_fifa_api_id.y"
## [10] "date_stat"            "overall_rating"       "potential"
## [13] "preferred_foot"       "attacking_work_rate"  "defensive_work_rate"
## [16] "crossing"             "finishing"            "heading_accuracy"
## [19] "short_passing"        "volleys"              "dribbling"
## [22] "curve"                "free_kick_accuracy"   "long_passing"
```

```
## [25] "ball_control"         "acceleration"         "sprint_speed"
## [28] "agility"              "reactions"            "balance"
## [31] "shot_power"           "jumping"              "stamina"
## [34] "strength"             "long_shots"           "aggression"
## [37] "interceptions"        "positioning"          "vision"
## [40] "penalties"            "marking"              "standing_tackle"
## [43] "sliding_tackle"       "gk_diving"            "gk_handling"
## [46] "gk_kicking"           "gk_positioning"       "gk_reflexes"
```

```
str(merged_table)
```

```
## 'data.frame':    183253 obs. of  48 variables:
##  $ player_api_id       : int  2625 2625 2625 2625 2625 2625 2625 2625 2625 2625 ...
##  $ id.x                : int  8368 8368 8368 8368 8368 8368 8368 8368 8368 8368 ...
##  $ player_name         : chr  "Patryk Rachwal,18" "Patryk Rachwal,18" "Patryk Rachwal,18" "Patryk Ra
##  $ player_fifa_api_id.x: int  148544 148544 148544 148544 148544 148544 148544 148544 148544 148544
##  $ birthday            : chr  "1981-01-27 00:00:00" "1981-01-27 00:00:00" "1981-01-27 00:00:00" "198
##  $ height              : num  175 175 175 175 175 ...
##  $ weight              : int  154 154 154 154 154 154 154 154 154 154 ...
##  $ id.y                : int  139756 139752 139753 139754 139755 139757 139758 139759 139760 139761
##  $ player_fifa_api_id.y: int  148544 148544 148544 148544 148544 148544 148544 148544 148544 148544
##  $ date_stat           : chr  "2013-03-08 00:00:00" "2015-01-16 00:00:00" "2014-11-07 00:00:00" "201
##  $ overall_rating      : int  61 61 61 61 61 58 58 58 58 59 ...
##  $ potential           : int  61 61 61 61 61 58 58 58 58 63 ...
##  $ preferred_foot      : chr  "right" "right" "right" "right" ...
##  $ attacking_work_rate : chr  "medium" "medium" "medium" "medium" ...
##  $ defensive_work_rate : chr  "medium" "medium" "medium" "medium" ...
##  $ crossing            : int  50 50 50 50 50 52 52 52 52 52 ...
##  $ finishing           : int  47 47 47 47 47 47 47 47 47 47 ...
##  $ heading_accuracy    : int  46 46 46 46 46 46 46 46 46 46 ...
##  $ short_passing       : int  52 52 52 52 52 53 53 53 53 63 ...
##  $ volleys             : int  39 39 39 39 39 37 37 37 37 37 ...
##  $ dribbling           : int  50 50 50 50 50 56 56 56 56 56 ...
##  $ curve               : int  51 51 51 51 51 49 49 49 49 49 ...
##  $ free_kick_accuracy  : int  50 50 50 50 50 50 50 50 50 50 ...
##  $ long_passing        : int  64 64 64 64 64 66 66 66 66 66 ...
##  $ ball_control        : int  62 62 62 62 62 58 58 58 58 58 ...
##  $ acceleration        : int  67 67 67 67 67 67 67 67 66 66 ...
##  $ sprint_speed        : int  63 63 63 63 63 63 63 63 63 63 ...
##  $ agility             : int  74 74 74 74 74 74 74 60 58 58 ...
##  $ reactions           : int  49 49 49 49 49 49 49 49 49 49 ...
##  $ balance             : int  76 76 76 76 76 76 76 76 75 48 ...
##  $ shot_power          : int  68 68 68 68 68 68 68 68 68 68 ...
##  $ jumping             : int  63 63 63 63 63 63 55 55 55 55 ...
##  $ stamina             : int  77 77 77 77 77 77 77 77 77 77 ...
##  $ strength            : int  56 56 56 56 56 56 56 55 55 55 ...
##  $ long_shots          : int  54 54 54 54 54 58 58 58 58 58 ...
##  $ aggression          : int  71 71 71 71 71 71 71 71 71 71 ...
##  $ interceptions       : int  64 64 64 64 64 64 64 64 60 70 ...
##  $ positioning         : int  49 49 49 49 49 49 49 49 49 49 ...
##  $ vision              : int  55 55 55 55 55 55 55 55 55 55 ...
##  $ penalties           : int  66 66 66 66 66 66 66 66 66 66 ...
##  $ marking             : int  62 62 62 62 62 63 63 63 63 63 ...
##  $ standing_tackle     : int  63 63 63 63 63 63 63 63 63 63 ...
```

```
##  $ sliding_tackle      : int  54 54 54 54 54 52 52 52 52 62 ...
##  $ gk_diving           : int  12 12 12 12 12 12 12 12 12 12 ...
##  $ gk_handling         : int  11 11 11 11 11 11 11 11 11 11 ...
##  $ gk_kicking          : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ gk_positioning      : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ gk_reflexes         : int  8 8 8 8 8 8 8 8 8 8 ...
```

Remove NAs

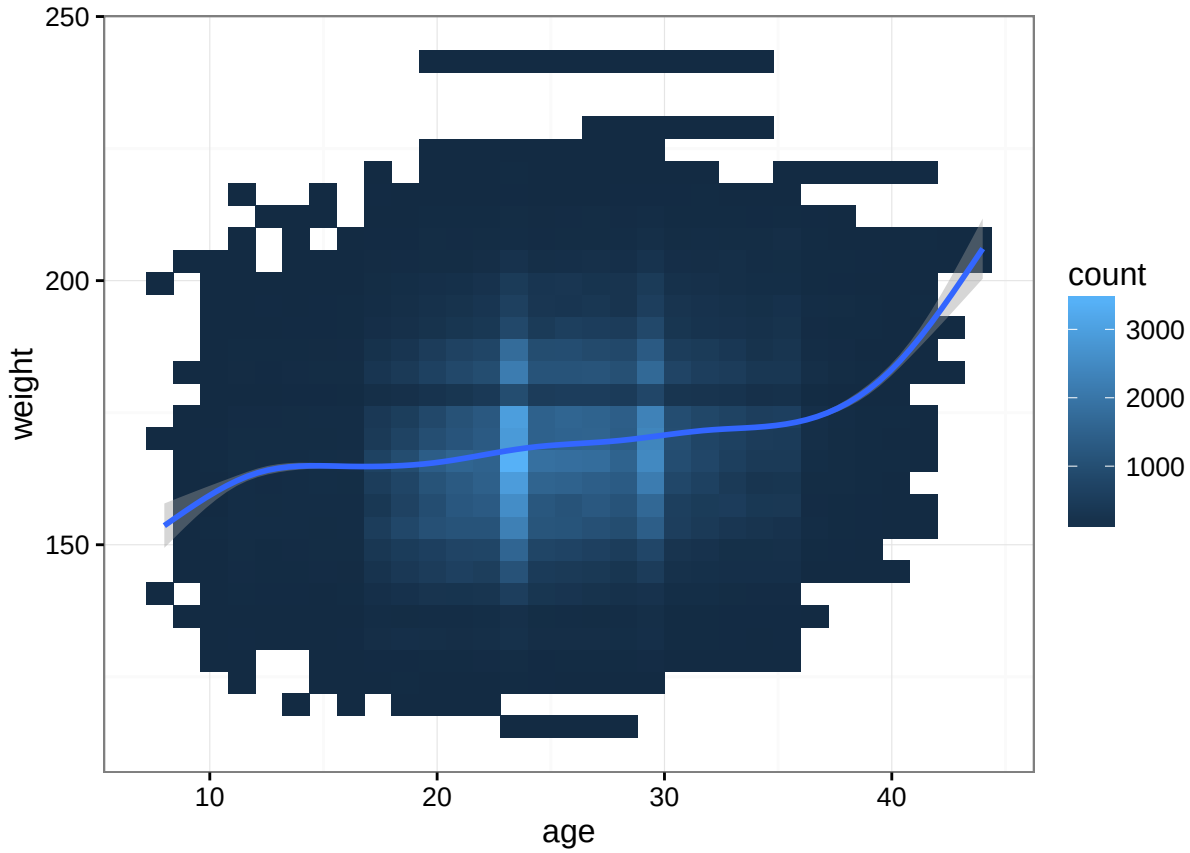
```
merged_table <- na.omit(merged_table)
```

Tidying the data

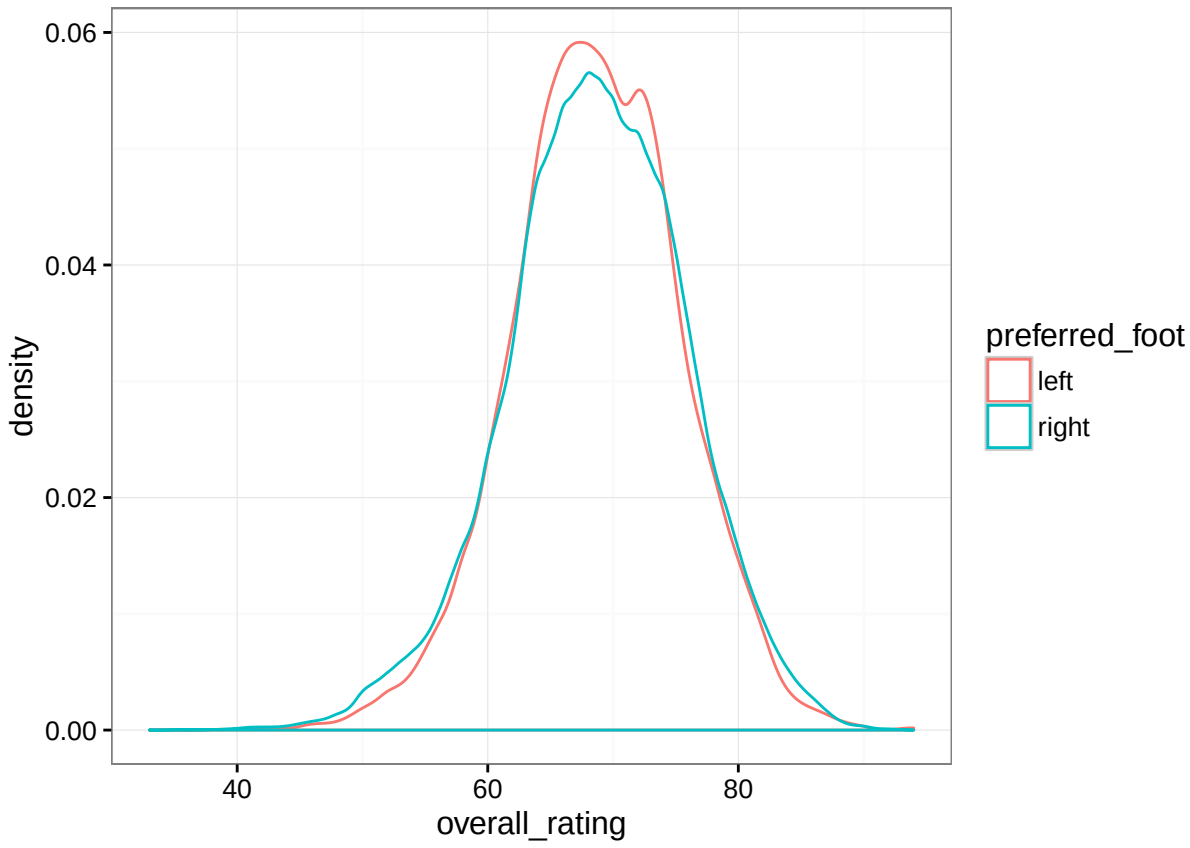
```
merged_table <-
  mutate(merged_table, age = as.integer(substr(merged_table$date_stat, 1, 4)) - as.integer(substr(merge

 merged_table <- mutate(merged_table, bmi = (weight * 0.45359237) / ((height/100) * (height/100)))
```

Some visualizations

```
ggplot(merged_table, aes(age, weight)) +
  geom_bin2d() +
  geom_smooth()+
  theme_bw()
```




```
ggplot(merged_table, aes(overall_rating, color = preferred_foot)) +
  geom_density() +
  theme_bw()
```



Retain columns of interest

```
stats <- merged_table[,-c(1,2,4,5,8,9,10,13,14,15)]
```

Filtering players

To have more robust stats, we remove players whose score is available for < 10 matches.

```
bad_stats <- names(which(table(stats$player_name) < 10))
stats <- stats[-which(stats[, 1] %in% bad_stats),]
```

Compute average scores for each player with > 10 matches.

```
avg_scores <- apply(stats[, 2:ncol(stats)], 2, function(x)
  tapply(x, stats$player_name, mean))

avg_scores_df <- as.data.frame(avg_scores)
summary(avg_scores_df)
```

```
##      height          weight      overall_rating    potential
##  Min.   :157.5   Min.   :117.0   Min.   :49.81   Min.   :52.80
##  1st Qu.:177.8   1st Qu.:159.0   1st Qu.:64.40   1st Qu.:69.33
```

```
##  Median :182.9   Median :168.0   Median :67.95   Median :73.00
##  Mean   :181.9   Mean   :168.9   Mean   :68.19   Mean   :73.06
##  3rd Qu.:185.4   3rd Qu.:179.0   3rd Qu.:71.81   3rd Qu.:76.69
##  Max.   :208.3   Max.   :243.0   Max.   :92.19   Max.   :95.23
##     crossing        finishing      heading_accuracy short_passing
##  Min.   : 8.133   Min.   : 7.538   Min.   : 9.50    Min.   :11.82
##  1st Qu.:45.778   1st Qu.:33.944   1st Qu.:50.23    1st Qu.:57.47
##  Median :58.158   Median :51.600   Median :59.54    Median :64.28
##  Mean   :54.318   Mean   :48.946   Mean   :56.78    Mean   :61.74
##  3rd Qu.:65.929   3rd Qu.:64.125   3rd Qu.:67.50    3rd Qu.:70.15
##  Max.   :89.357   Max.   :92.231   Max.   :93.11    Max.   :95.18
##     volleys         dribbling          curve        free_kick_accuracy
##  Min.   : 5.231   Min.   : 8.333   Min.   : 8.385   Min.   : 8.00
##  1st Qu.:34.625   1st Qu.:51.000   1st Qu.:40.920   1st Qu.:36.62
##  Median :51.000   Median :62.706   Median :54.562   Median :49.95
##  Mean   :48.281   Mean   :58.244   Mean   :51.929   Mean   :48.43
##  3rd Qu.:62.000   3rd Qu.:70.048   3rd Qu.:65.000   3rd Qu.:61.05
##  Max.   :90.789   Max.   :96.462   Max.   :92.571   Max.   :90.49
##   long_passing    ball_control    acceleration    sprint_speed
##  Min.   :11.33   Min.   :10.00   Min.   :17.80   Min.   :18.68
##  1st Qu.:48.92   1st Qu.:58.68   1st Qu.:60.44   1st Qu.:61.57
##  Median :58.10   Median :65.88   Median :68.67   Median :69.00
##  Mean   :56.45   Mean   :62.61   Mean   :67.10   Mean   :67.51
##  3rd Qu.:65.53   3rd Qu.:71.56   3rd Qu.:75.36   3rd Qu.:75.47
##  Max.   :94.16   Max.   :95.77   Max.   :95.79   Max.   :95.70
##     agility        reactions        balance        shot_power
##  Min.   :21.00   Min.   :33.20   Min.   :20.00   Min.   : 9.923
##  1st Qu.:58.06   1st Qu.:61.16   1st Qu.:58.58   1st Qu.:54.235
##  Median :66.93   Median :66.05   Median :65.94   Median :64.500
##  Mean   :65.41   Mean   :65.61   Mean   :64.81   Mean   :60.869
##  3rd Qu.:74.05   3rd Qu.:70.70   3rd Qu.:72.32   3rd Qu.:71.486
##  Max.   :94.67   Max.   :92.54   Max.   :94.31   Max.   :93.077
##     jumping         stamina         strength       long_shots
##  Min.   :28.36   Min.   :16.00   Min.   :23.28   Min.   : 9.00
##  1st Qu.:61.00   1st Qu.:61.59   1st Qu.:61.12   1st Qu.:41.00
##  Median :67.67   Median :68.38   Median :68.35   Median :56.75
##  Mean   :66.75   Mean   :66.46   Mean   :67.30   Mean   :52.24
##  3rd Qu.:73.00   3rd Qu.:73.85   3rd Qu.:74.81   3rd Qu.:65.14
##  Max.   :94.31   Max.   :93.18   Max.   :94.00   Max.   :89.88
##    aggression    interceptions    positioning        vision
##  Min.   :13.53   Min.   : 9.95   Min.   : 6.00   Min.   : 8.00
##  1st Qu.:51.88   1st Qu.:36.16   1st Qu.:45.23   1st Qu.:49.00
##  Median :63.46   Median :55.67   Median :58.89   Median :58.94
##  Mean   :60.59   Mean   :51.63   Mean   :54.71   Mean   :57.00
##  3rd Qu.:71.50   3rd Qu.:66.06   3rd Qu.:66.86   3rd Qu.:67.07
##  Max.   :93.00   Max.   :91.04   Max.   :93.20   Max.   :95.68
##    penalties        marking      standing_tackle sliding_tackle
##  Min.   : 9.60   Min.   : 9.50   Min.   : 9.50   Min.   : 8.00
##  1st Qu.:45.14   1st Qu.:25.16   1st Qu.:29.50   1st Qu.:26.00
##  Median :55.53   Median :50.32   Median :56.10   Median :52.63
##  Mean   :54.09   Mean   :46.63   Mean   :50.08   Mean   :47.78
##  3rd Qu.:64.31   3rd Qu.:64.91   3rd Qu.:67.94   3rd Qu.:66.18
##  Max.   :89.57   Max.   :89.67   Max.   :90.20   Max.   :94.37
##    gk_diving       gk_handling       gk_kicking     gk_positioning
```

```
##  Min.   : 1.935   Min.   : 3.261   Min.   : 3.261   Min.   : 3.261
##  1st Qu.: 7.812   1st Qu.: 9.385   1st Qu.:11.357   1st Qu.: 9.357
##  Median :10.214   Median :12.138   Median :16.318   Median :12.129
##  Mean   :15.132   Mean   :16.341   Mean   :20.745   Mean   :16.416
##  3rd Qu.:12.929   3rd Qu.:14.615   3rd Qu.:23.348   3rd Qu.:14.676
##  Max.   :89.864   Max.   :82.900   Max.   :87.133   Max.   :90.161
##   gk_reflexes          age             bmi
##  Min.   : 3.261   Min.   :16.55   Min.   :17.87
##  1st Qu.: 9.333   1st Qu.:22.50   1st Qu.:22.18
##  Median :12.111   Median :25.38   Median :23.10
##  Mean   :16.756   Mean   :25.68   Mean   :23.11
##  3rd Qu.:14.688   3rd Qu.:28.60   3rd Qu.:23.99
##  Max.   :90.955   Max.   :40.82   Max.   :30.87
```

```
sd(avg_scores_df$age)
```

```
## [1] 4.118864
```

Principal component analysis

```
avg_scores_pca <- prcomp(avg_scores, scale. = T)
summary(avg_scores_pca)
```

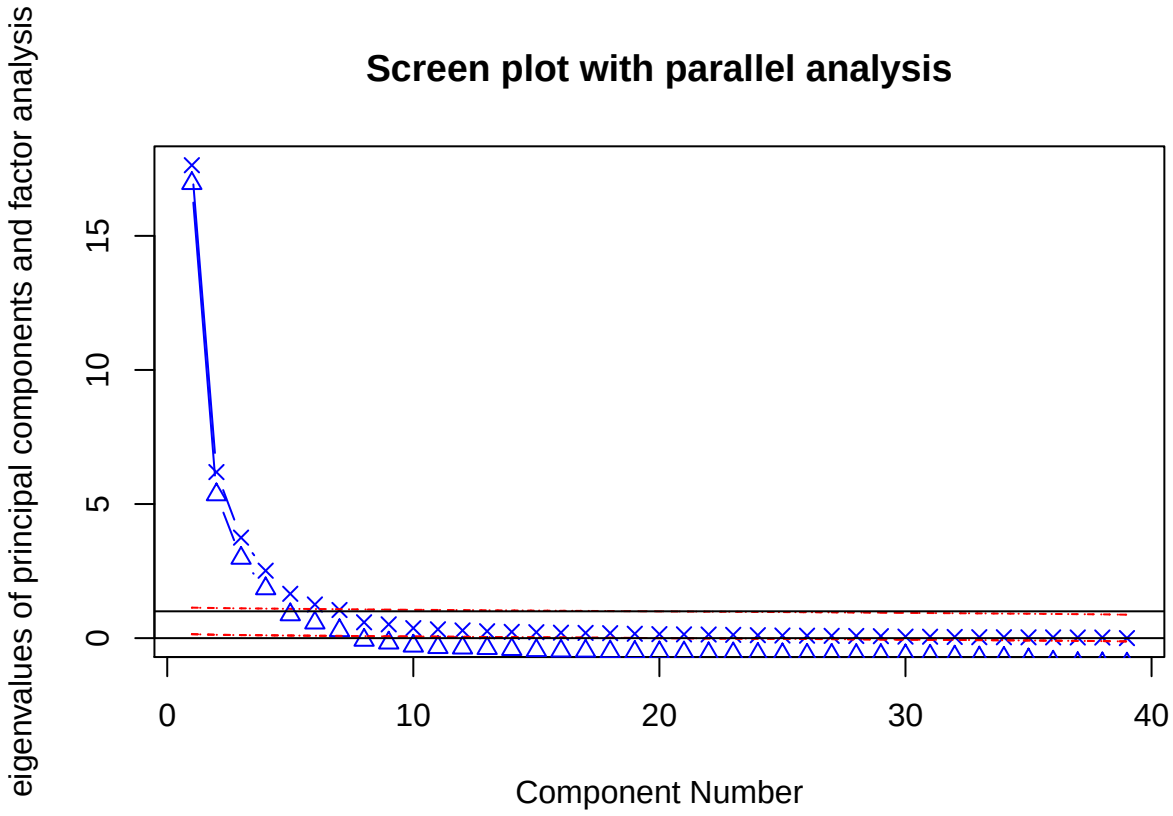
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     4.1993 2.4883 1.93574 1.58565 1.28574 1.11993
## Proportion of Variance 0.4521 0.1588 0.09608 0.06447 0.04239 0.03216
## Cumulative Proportion  0.4521 0.6109 0.70699 0.77146 0.81385 0.84601
##                           PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     1.02401 0.76825 0.71698 0.61016 0.57364 0.53533
## Proportion of Variance 0.02689 0.01513 0.01318 0.00955 0.00844 0.00735
## Cumulative Proportion  0.87290 0.88803 0.90121 0.91076 0.91919 0.92654
##                           PC13    PC14    PC15    PC16   PC17    PC18
## Standard deviation     0.50588 0.48329 0.46956 0.45089 0.4416 0.42965
## Proportion of Variance 0.00656 0.00599 0.00565 0.00521 0.0050 0.00473
## Cumulative Proportion  0.93310 0.93909 0.94475 0.94996 0.9550 0.95969
##                           PC19    PC20    PC21    PC22    PC23    PC24
## Standard deviation     0.40341 0.38637 0.37440 0.36782 0.34949 0.32981
## Proportion of Variance 0.00417 0.00383 0.00359 0.00347 0.00313 0.00279
## Cumulative Proportion  0.96387 0.96769 0.97129 0.97476 0.97789 0.98068
##                           PC25    PC26    PC27    PC28    PC29    PC30
## Standard deviation     0.31187 0.30412 0.29227 0.28278 0.26953 0.23752
## Proportion of Variance 0.00249 0.00237 0.00219 0.00205 0.00186 0.00145
## Cumulative Proportion  0.98317 0.98554 0.98773 0.98978 0.99165 0.99309
##                           PC31   PC32    PC33    PC34    PC35    PC36
## Standard deviation     0.23195 0.2073 0.18588 0.18004 0.17285 0.16707
## Proportion of Variance 0.00138 0.0011 0.00089 0.00083 0.00077 0.00072
## Cumulative Proportion  0.99447 0.9956 0.99646 0.99729 0.99806 0.99877
##                           PC37    PC38    PC39
## Standard deviation     0.15681 0.14868 0.03340
## Proportion of Variance 0.00063 0.00057 0.00003
## Cumulative Proportion  0.99940 0.99997 1.00000
```

```
fa.parallel(avg_scores_df, fa="PC", show.legend=FALSE, main="Screen plot with parallel analysis")
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

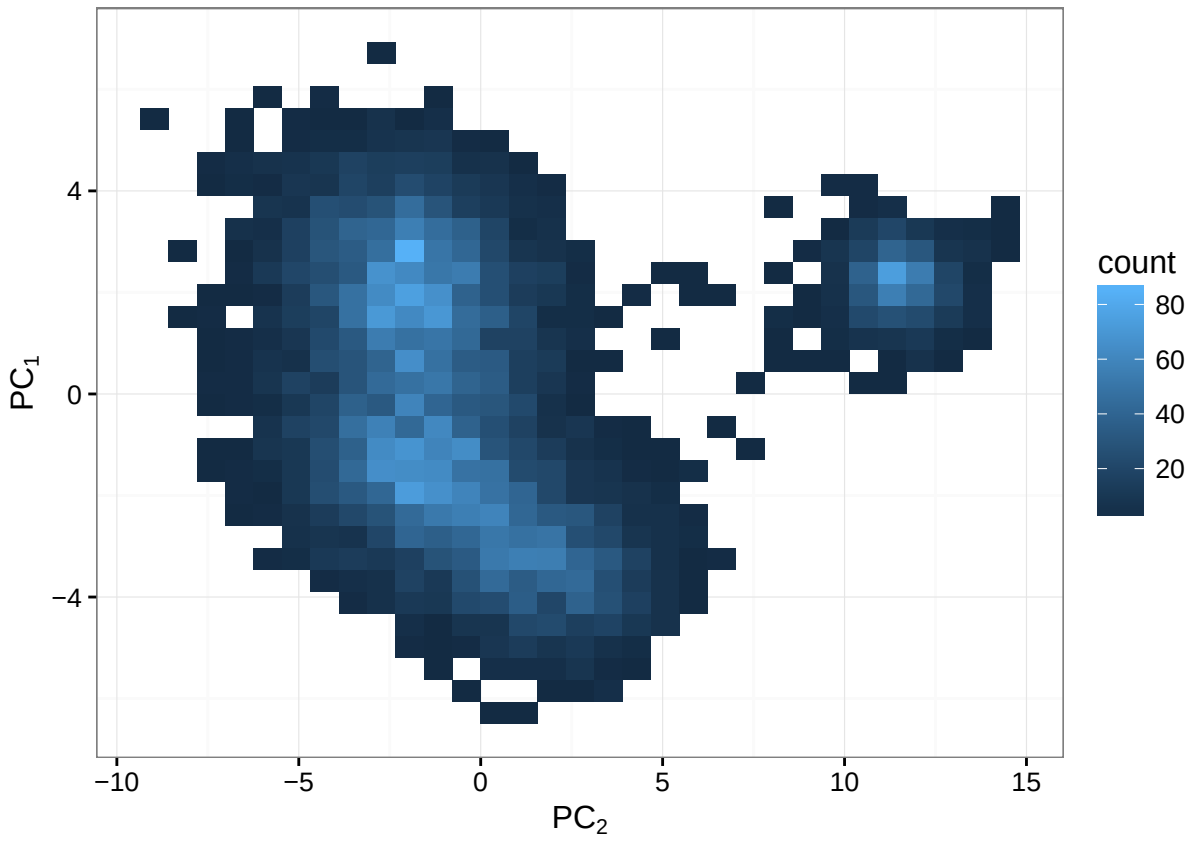
```
## The following object is masked from 'package:dplyr':
##
##     select
```



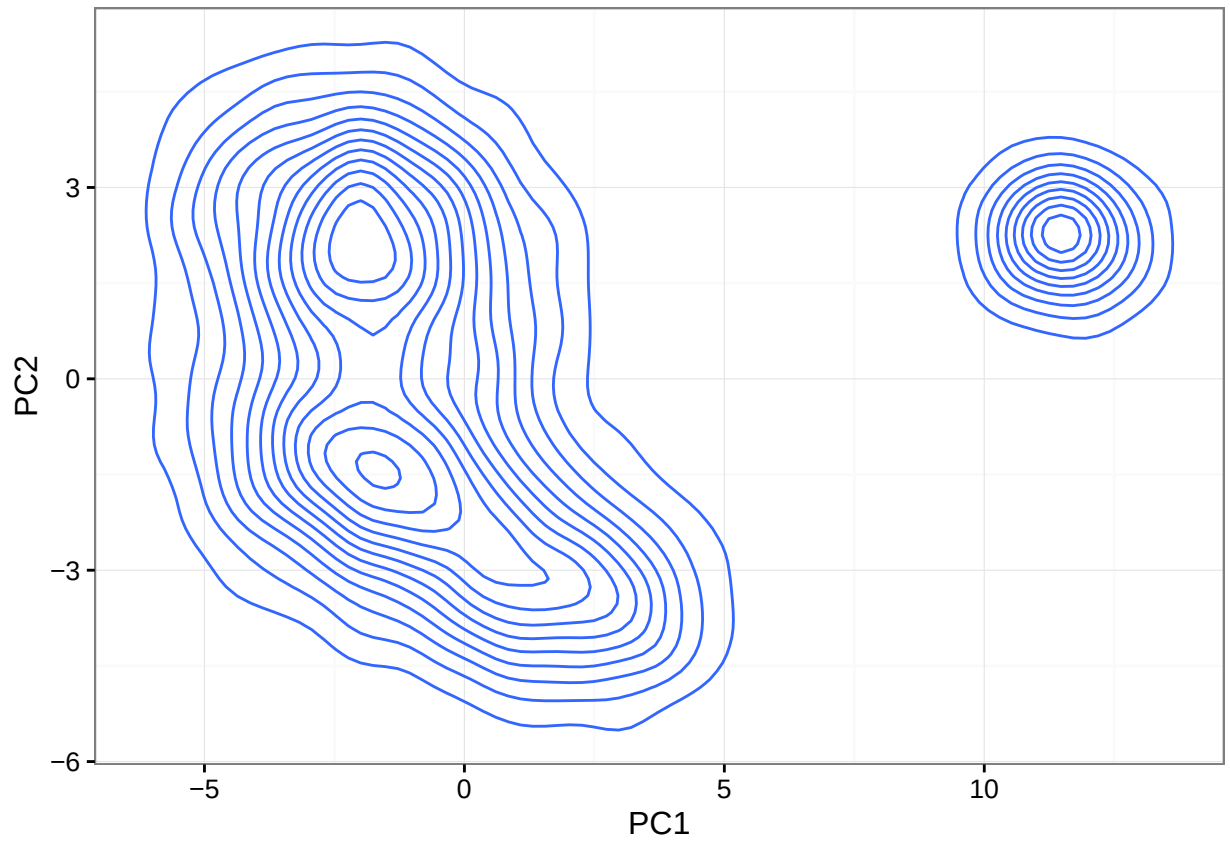

```
## Parallel analysis suggests that the number of factors =  7  and the number of components =  6
```

Using a smooth scatter because too many points prevent from seeing the global picture.

```
pc_df <- as.data.frame(avg_scores_pca$x[, 1:2])
ggplot(pc_df, aes(PC1, PC2)) +
  geom_bin2d(bins = 30) + scale_x_continuous(quote(PC[2])) + scale_y_continuous(quote(PC[1])) +
  theme_bw()
```




```
ggplot(pc_df, aes(PC1, PC2)) +
  geom_density2d() +
  theme_bw()
```



Clustering analysis

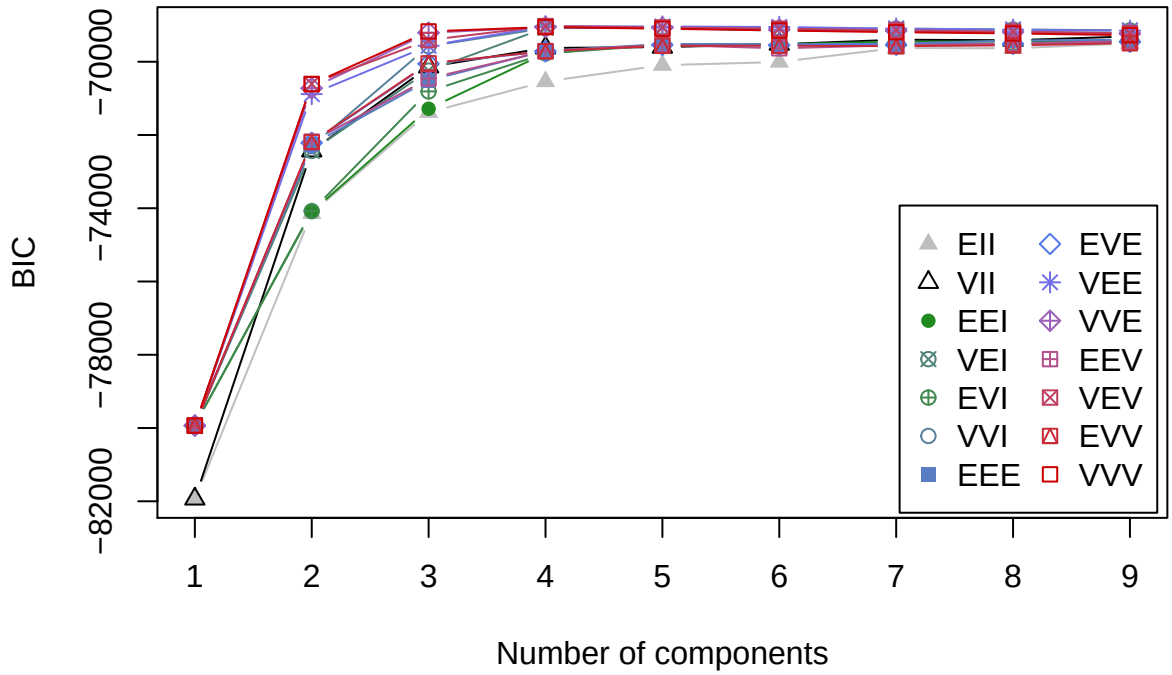
Mclust classification.

```
xyMclust <- Mclust(data.frame (avg_scores_pca$x[,1],avg_scores_pca$x[,2]))
summary(xyMclust, parameters = TRUE)
```

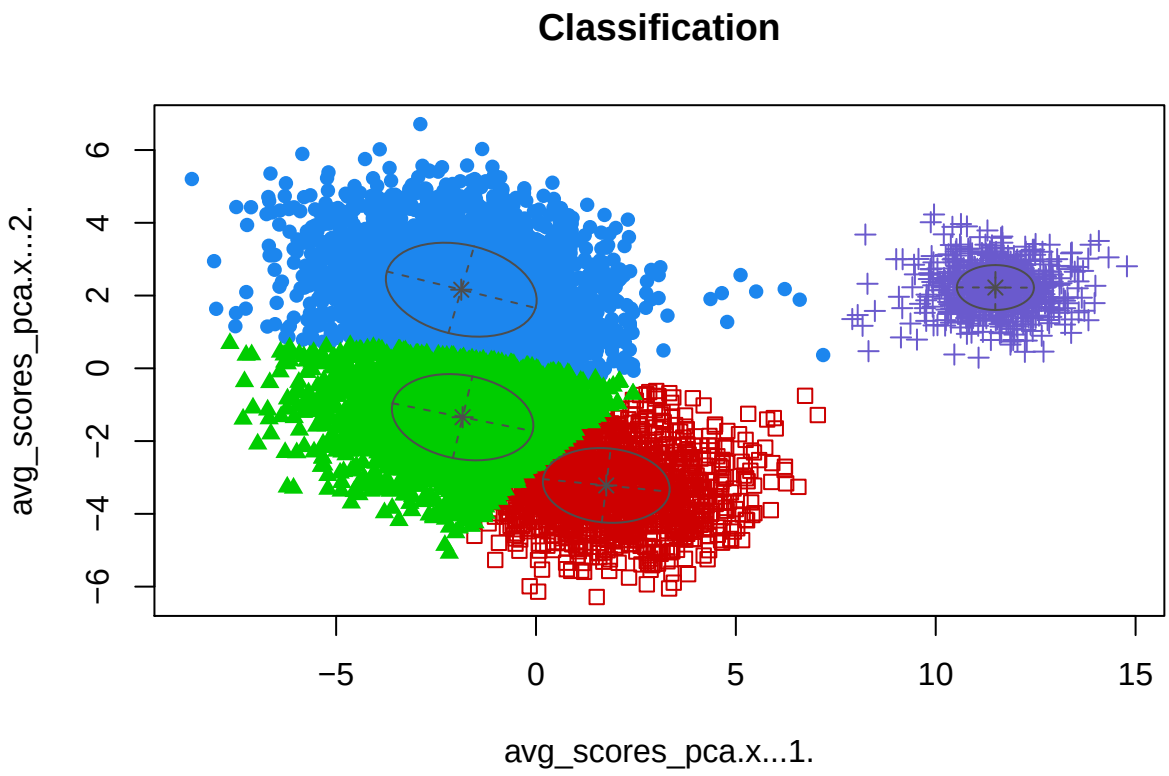
```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEV (ellipsoidal, equal shape) model with 4 components:
##
##  log.likelihood    n df       BIC       ICL
##       -34423.53 7705 20 -69026.06 -70971.13
##
## Clustering table:
##    1    2    3    4
## 3062 1457 2511  675
##
## Mixing probabilities:
##          1          2          3          4
## 0.39573389 0.18963935 0.32701023 0.08761654
##
```

```
## Means:
##                              [,1]      [,2]      [,3]      [,4]
## avg_scores_pca.x...1. -1.869848  1.757691 -1.835760 11.492660
## avg_scores_pca.x...2.  2.161146 -3.219113 -1.343614  2.221127
##
## Variances:
## [,,1]
##                       avg_scores_pca.x...1. avg_scores_pca.x...2.
## avg_scores_pca.x...1.             3.5616652            -0.5497267
## avg_scores_pca.x...2.            -0.5497267             1.6686341
## [,,2]
##                       avg_scores_pca.x...1. avg_scores_pca.x...2.
## avg_scores_pca.x...1.             2.5182484            -0.1625996
## avg_scores_pca.x...2.            -0.1625996             1.0574114
## [,,3]
##                       avg_scores_pca.x...1. avg_scores_pca.x...2.
## avg_scores_pca.x...1.             3.1335726            -0.3955646
## avg_scores_pca.x...2.            -0.3955646             1.4060457
## [,,4]
##                       avg_scores_pca.x...1. avg_scores_pca.x...2.
## avg_scores_pca.x...1.           0.929296683          -0.002052427
## avg_scores_pca.x...2.          -0.002052427           0.380919688
```

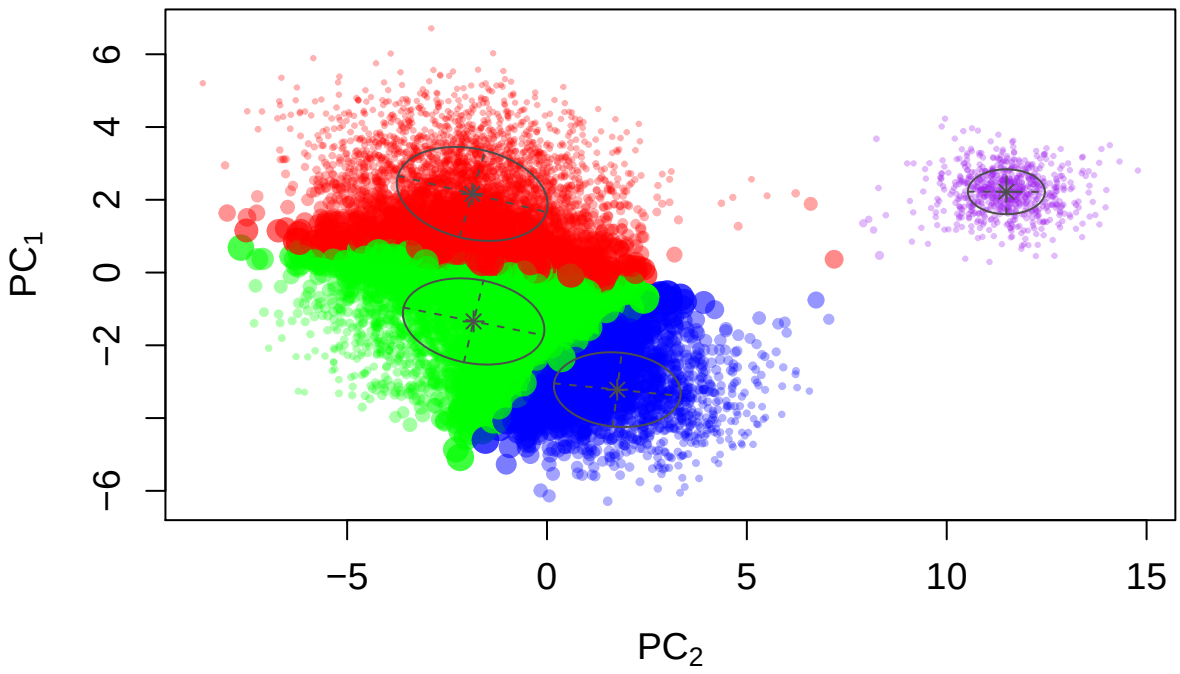
```
plot(xyMclust, what = "BIC")
```



```
plot(xyMclust, what = "classification")
```




```
plot(xyMclust,
     what = "uncertainty",
     xlab = quote(PC[2]),
     ylab = quote(PC[1]), main = NULL, col=c("red","blue","green","purple"), cex.axis = 1.2, cex.lab =
```



Classifing players according to their clustering roles.

```
player_class <- xyMclust$classification

#number of players in each cluster
table(player_class)
```

```
## player_class
##    1    2    3    4
## 3062 1457 2511  675
```

```
table(xyMclust$classification)
```

```
##
##    1    2    3    4
## 3062 1457 2511  675
```

```
clusters <- as.data.frame(table(xyMclust$classification))

#assingning roles to players
player_class[which(player_class == 1)] = "forward "
player_class[which(player_class == 2)] = "midfield"
player_class[which(player_class == 3)] = "defender"
player_class[which(player_class == 4)] = "goalkeeper"
player_class = factor(player_class, levels = c("forward ", "midfield", "defender", "goalkeeper"))
```

```
tmp <- as.data.frame(player_class)

#players from each cluster
defender <- names(player_class[player_class=="forward "][1:clusters[1, 2]])
midfield <- names(player_class[player_class=="midfield"][1:clusters[2, 2]])
forward  <- names(player_class[player_class=="defender"][1:clusters[3, 2]])
goalkeeper <- names(player_class[player_class=="goalkeeper"][1:clusters[4, 2]])

lo <- as.array(forward )
lm <- as.array(midfield)
ld <- as.array(defender)
lg <- as.array(goalkeeper)

#creating a dataset with players' classification
player_class_forward  <- data.frame(name = lo, role = "defender" )
player_class_midfield <- data.frame(name = lm, role = "midfield" )
player_class_defender <- data.frame(name = ld, role = "forward" )
player_class_goalkeeper <- data.frame(name = lg, role = "goalkeeper" )
role_df <- rbind(player_class_forward ,player_class_midfield, player_class_defender, player_class_goalk

#sorting the dataset
role_df <- arrange(role_df, name)

tmp <- avg_scores
tmp <- as.data.frame(avg_scores) %>%
  mutate(name = row.names(tmp))

#the list of player and their classffication (using merge by)
players_classified <- merge(role_df,tmp, by = "name")

#players_classified <- filter(players_classified, role != "goalkeeper")
```

We can find players with any desired proportions of characteristics.

```
#border players (min probabilities)
tot <- data.frame(player_class, xyMclust$z[, 1:4])
colnames(tot)[2:5] <-
  c("prob_forward ",
    "prob_midfield",
    "prob_defender",
    "prob_goalkeeper")

forward  <- tot[which(tot[, 1] == "forward "), ]
midfield <- tot[which(tot[, 1] == "midfield"), ]
defender <- tot[which(tot[, 1] == "defender"), ]
goalkeeper <- tot[which(tot[, 1] == "goalkeeper"), ]

#very offensive
very_forward  <- forward [order(forward [, 2], decreasing = T), ]
#very midfield
very_midfield <- midfield[order(midfield[, 3], decreasing = T), ]
#very defender
very_defender <- defender[order(defender[, 4], decreasing = T), ]
```

```r
#very goalkeeper
very_goalkeeper <- goalkeeper[order(goalkeeper[, 5], decreasing = T), ]

#Border line players  [forward  & midfield]
head(forward [order(forward [,3], decreasing = T), ])
```

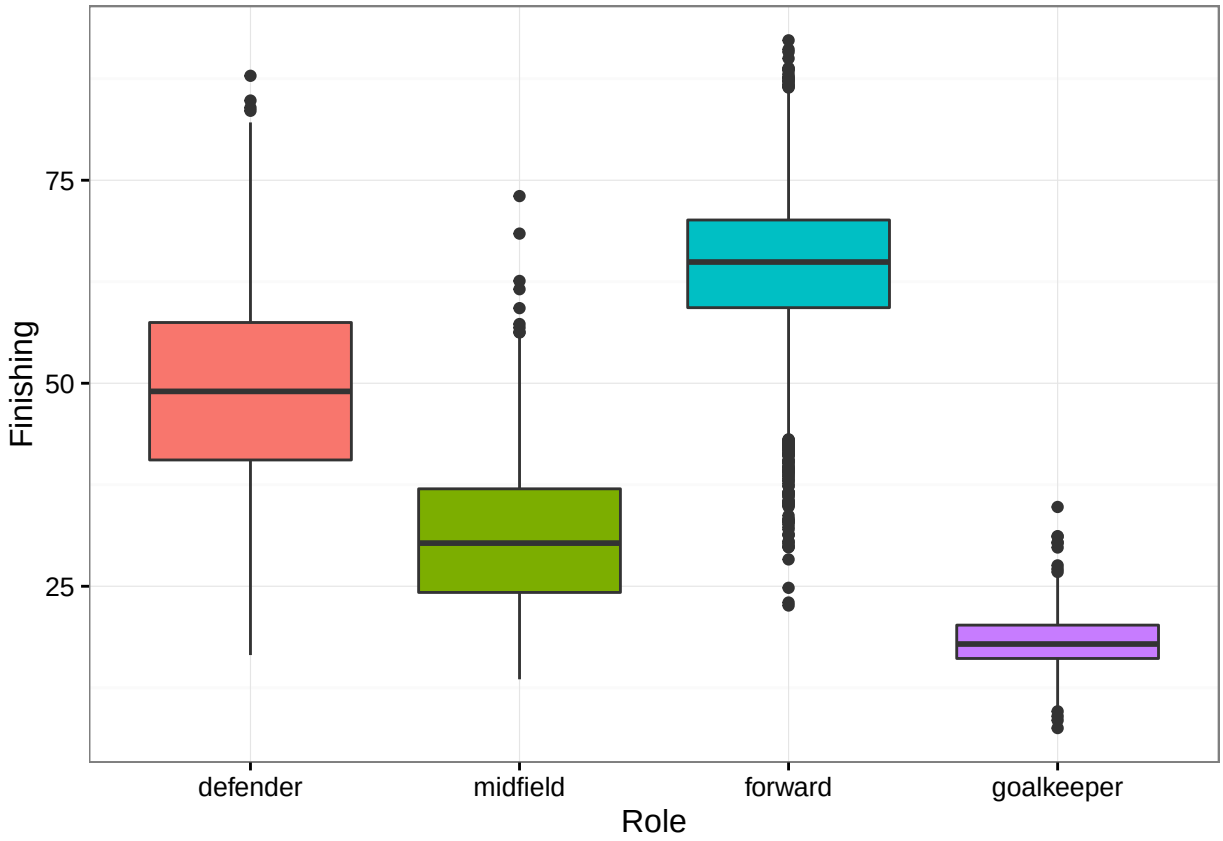
```
##                     player_class prob_forward  prob_midfield prob_defender
## Neto                     forward      0.8435460    0.11077915    0.00906363
## Giuseppe Rossini         forward      0.6333099    0.09206066    0.27462947
## Calvin Zola Makongo      forward      0.4810734    0.07411480    0.44481183
## Lukas Droppa             forward      0.5116378    0.06948598    0.41887619
## Wouter Weghorst          forward      0.6739711    0.06025947    0.26576945
## Nicola Sutter            forward      0.6768708    0.05818825    0.26494097
##                     prob_goalkeeper
## Neto                   3.661127e-02
## Giuseppe Rossini       1.060784e-21
## Calvin Zola Makongo    2.186232e-26
## Lukas Droppa           5.634231e-26
## Wouter Weghorst        1.488274e-22
## Nicola Sutter          1.262270e-22
```

```r
#Border line players  [defender & midfield]
head(defender[order(defender[,3], decreasing = T), ])
```

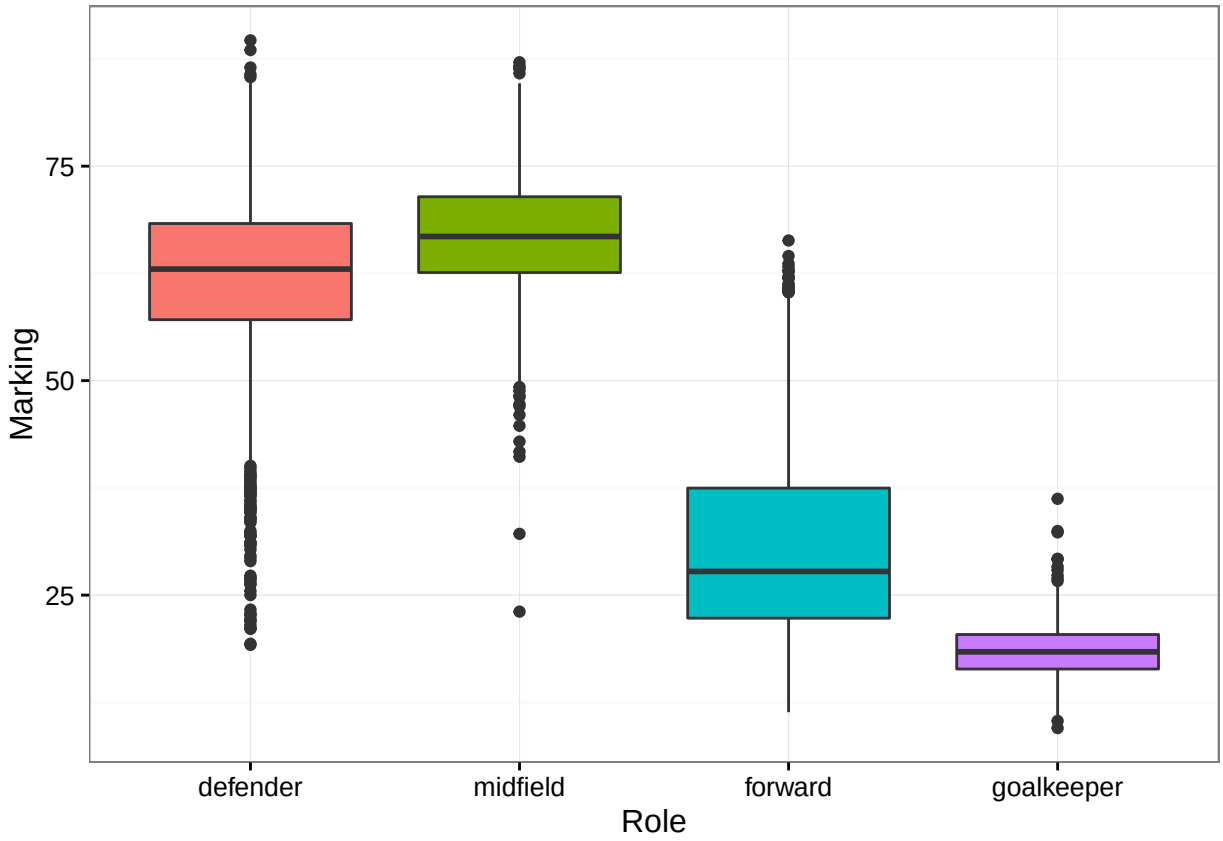
```
##                   player_class prob_forward  prob_midfield prob_defender
## Wim De Decker         defender  2.970182e-03     0.4977282     0.4993017
## Juan Forlin           defender  4.265349e-04     0.4966720     0.5029015
## Sally Sarr            defender  3.732468e-03     0.4963054     0.4999622
## Nicolas Otamendi      defender  4.946020e-05     0.4954459     0.5045047
## Marcos Gullon         defender  1.346470e-03     0.4950691     0.5035845
## Miranda               defender  6.088404e-05     0.4906522     0.5092870
##                   prob_goalkeeper
## Wim De Decker        1.255474e-41
## Juan Forlin          1.786458e-50
## Sally Sarr           1.186991e-40
## Nicolas Otamendi     5.153355e-61
## Marcos Gullon        3.246811e-45
## Miranda              4.496620e-60
```

Some visualizations

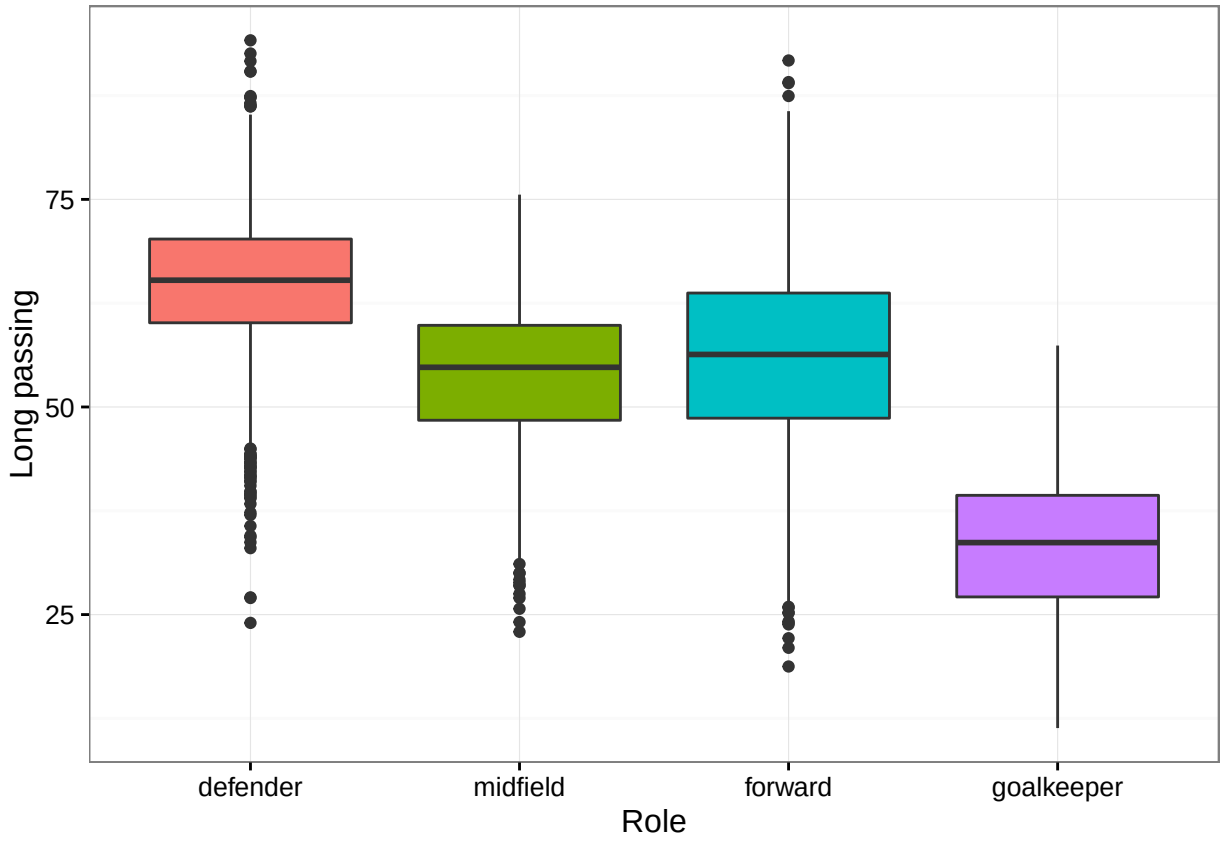
```r
#forward
ggplot(players_classified, aes(role, finishing, fill = role)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none") +
  labs(x = "Role", y = "Finishing")
```



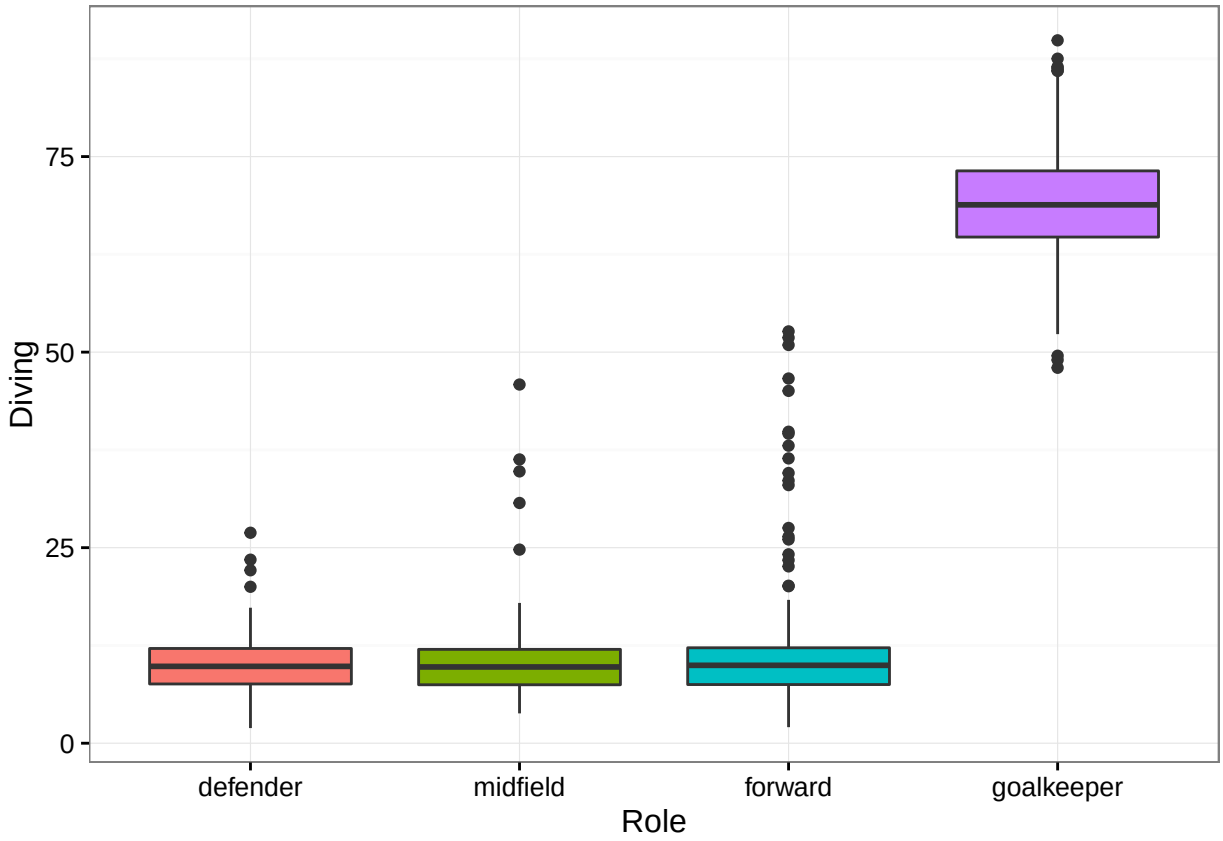

```
#midfield
ggplot(players_classified, aes(role, marking, fill = role)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")+
  labs(x = "Role", y = "Marking")
```




```
#defender
ggplot(players_classified, aes(role, long_passing, fill = role)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")+
  labs(x = "Role", y = "Long passing")
```




```
#goalkeeper
ggplot(players_classified, aes(role, gk_diving, fill = role)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")+
  labs(x = "Role", y = "Diving")
```



Preprocessing data for radial plot of roles by means

```
by_role <- group_by(players_classified, role)
sum_role <- summarise(
  by_role,
  height = mean(height),
  weight = mean(weight),
  overall_rating = mean(overall_rating),
  potential = mean(potential),
  crossing = mean(crossing),
  finishing = mean(finishing),
  heading_accuracy = mean(heading_accuracy),
  short_passing = mean(short_passing),
  volleys = mean(volleys),
  dribbling = mean(dribbling),
  curve = mean(curve),
  free_kick_accuracy = mean(free_kick_accuracy),
  long_passing = mean(long_passing),
  ball_control = mean(ball_control),
  acceleration = mean(acceleration),
  sprint_speed = mean(sprint_speed),
  agility = mean(agility),
  reactions = mean(reactions),
  balance = mean(balance),
```

```
  shot_power = mean(shot_power),
  jumping = mean(jumping),
  stamina = mean(stamina),
  strength = mean(strength),
  long_shots = mean(long_shots),
  aggression = mean(aggression),
  interceptions = mean(interceptions),
  positioning = mean(positioning),
  vision = mean(vision),
  penalties = mean(penalties),
  marking = mean(marking),
  standing_tackle = mean(standing_tackle),
  sliding_tackle = mean(sliding_tackle),
  gk_diving = mean(gk_diving),
  gk_handling = mean(gk_handling),
  gk_kicking = mean(gk_kicking),
  gk_positioning = mean(gk_positioning),
  gk_reflexes = mean(gk_reflexes),
  age = mean(age),
  bmi = mean(bmi)
)
sum_role <- as.data.frame(sum_role)

write.xlsx(sum_role, "c:/sum_role.xlsx")
```

Xgboost

```
#removing players' name from the dataset (xgboost manages only numeric vectors)
data_xgb <- players_classified[,-c(1)]

#creating the sparse matrix for training the model
sparse_matrix <- sparse.model.matrix(role ~ . -1, data = data_xgb)

#this loop is for creating the parameter "label" which is required for taining the xgb model
i <- 1
vector <- as.array(data_xgb$role)
output_vector <- c(1:length(data_xgb$role))
output_vector[] <- as.array(data_xgb$role)
for (i in 1:length(vector)) {
  if(vector[i] == "forward"){
    output_vector[i] <- 1
  }else if(vector[i] == "defender"){
    output_vector[i] <- 2
  }else if(vector[i] == "midfield"){
    output_vector[i] <- 3
  }else{
    output_vector[i] <- 4
  }
}

#run de xgb model
```

```
model_xgb <-
  xgboost(
    data = sparse_matrix,
    label = output_vector,
    max.depth = 10,
    eta = 1,
    nthread = 2,
    nround = 30,
    objective = "reg:linear"
  )
```

```
## [0]  train-rmse:0.172241
## [1]  train-rmse:0.104802
## [2]  train-rmse:0.081733
## [3]  train-rmse:0.058529
## [4]  train-rmse:0.047834
## [5]  train-rmse:0.044528
## [6]  train-rmse:0.036572
## [7]  train-rmse:0.030559
## [8]  train-rmse:0.026160
## [9]  train-rmse:0.023808
## [10] train-rmse:0.019879
## [11] train-rmse:0.017497
## [12] train-rmse:0.014231
## [13] train-rmse:0.011590
## [14] train-rmse:0.010406
## [15] train-rmse:0.008898
## [16] train-rmse:0.007484
## [17] train-rmse:0.006513
## [18] train-rmse:0.005836
## [19] train-rmse:0.004873
## [20] train-rmse:0.004240
## [21] train-rmse:0.003918
## [22] train-rmse:0.003288
## [23] train-rmse:0.002989
## [24] train-rmse:0.002636
## [25] train-rmse:0.002394
## [26] train-rmse:0.002161
## [27] train-rmse:0.002023
## [28] train-rmse:0.001899
## [29] train-rmse:0.001793
```

Testing the Xgboost model using ten-fold cross validation

The pupose of the model we have built is to classify new data. Now we perform the prediction using cross-validation.

```
cv <- xgb.cv(data = sparse_matrix, label = output_vector, nfold = 10,
             nrounds = 100, objective = "reg:linear",
             early.stop.round = 3, maximize = FALSE, stratified = T, prediction = T)
```

```
## [0]  train-rmse:1.242574+0.003153    test-rmse:1.245806+0.028413
```

```
## [1]  train-rmse:0.883018+0.002287    test-rmse:0.891972+0.020626
## [2]  train-rmse:0.632717+0.001693    test-rmse:0.648466+0.015692
## [3]  train-rmse:0.459306+0.001444    test-rmse:0.483212+0.013115
## [4]  train-rmse:0.340910+0.001581    test-rmse:0.374697+0.011134
## [5]  train-rmse:0.261629+0.001493    test-rmse:0.306047+0.010968
## [6]  train-rmse:0.209522+0.001645    test-rmse:0.264720+0.010995
## [7]  train-rmse:0.175889+0.001642    test-rmse:0.240693+0.010443
## [8]  train-rmse:0.154448+0.001573    test-rmse:0.226990+0.010271
## [9]  train-rmse:0.140540+0.001683    test-rmse:0.219764+0.009631
## [10] train-rmse:0.131261+0.001645    test-rmse:0.215413+0.009260
## [11] train-rmse:0.124962+0.001877    test-rmse:0.213124+0.009202
## [12] train-rmse:0.120079+0.001990    test-rmse:0.211652+0.009039
## [13] train-rmse:0.116305+0.001751    test-rmse:0.210614+0.008596
## [14] train-rmse:0.113406+0.001469    test-rmse:0.210199+0.008594
## [15] train-rmse:0.110706+0.001382    test-rmse:0.209866+0.008487
## [16] train-rmse:0.108518+0.001499    test-rmse:0.209486+0.008545
## [17] train-rmse:0.106797+0.001863    test-rmse:0.209306+0.008351
## [18] train-rmse:0.105187+0.001683    test-rmse:0.209117+0.008501
## [19] train-rmse:0.103745+0.001890    test-rmse:0.209268+0.008517
## [20] train-rmse:0.102414+0.002083    test-rmse:0.209267+0.008526
## [21] train-rmse:0.100988+0.002083    test-rmse:0.209259+0.008414
## Stopping. Best iteration: 19
```

```
pred <- cv$pred
pred_role <- c(1:length(data_xgb$role))


for (i in 1:length(pred)) {
  if(pred[i] < 1.5){
    pred_role[i] <- 1
  }else if(pred[i] > 1.5 &&  pred[i] <= 2.5){
    pred_role[i] <- 2
  }else if(pred[i] > 2.5 &&  pred[i] <= 3.5){
    pred_role[i] <- 3
  }else{
    pred_role[i] <- 4
  }
}

confusionMatrix(data = output_vector, pred_role)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    2    3    4
##          1 2964   97    1    0
##          2  124 2308   79    0
##          3    1  106 1350    0
##          4    0    2    3  670
##
## Overall Statistics
##
##                Accuracy : 0.9464
##                  95% CI : (0.9411, 0.9513)
```

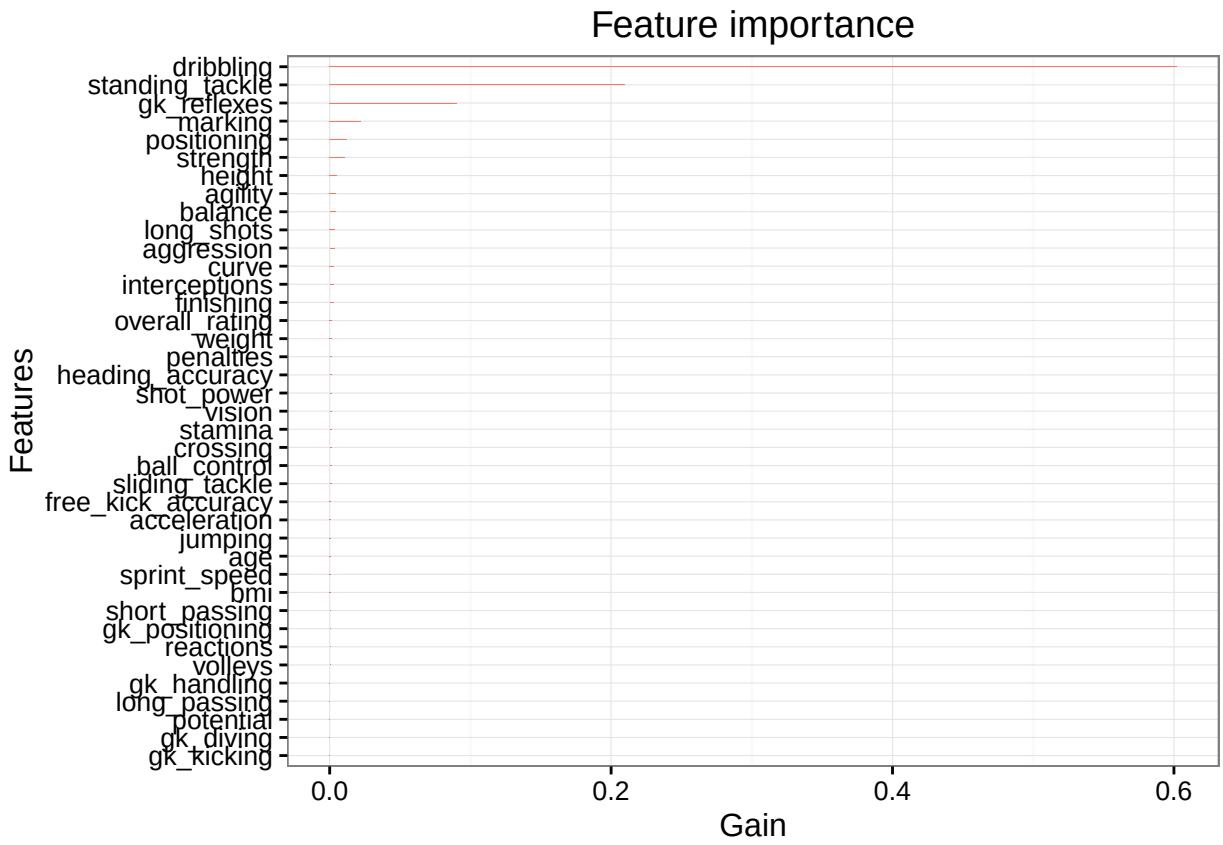
```
##     No Information Rate : 0.4009
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9225
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity            0.9595   0.9184   0.9421  1.00000
## Specificity            0.9788   0.9609   0.9829  0.99929
## Pos Pred Value         0.9680   0.9192   0.9266  0.99259
## Neg Pred Value         0.9731   0.9605   0.9867  1.00000
## Prevalence             0.4009   0.3262   0.1860  0.08696
## Detection Rate         0.3847   0.2995   0.1752  0.08696
## Detection Prevalence   0.3974   0.3259   0.1891  0.08761
## Balanced Accuracy      0.9692   0.9397   0.9625  0.99964
```

Attribute importance

```
#feature importance using xgb
importance <-
  xgb.importance(sparse_matrix@Dimnames[[2]], model = model_xgb)
head(importance)
```

```
##            Feature       Gain      Cover  Frequence
## 1:       dribbling 0.60197473 0.02433470 0.02547489
## 2: standing_tackle 0.20992711 0.03798827 0.02091003
## 3:     gk_reflexes 0.09026424 0.03447383 0.02311883
## 4:         marking 0.02234285 0.01974199 0.01737594
## 5:     positioning 0.01232959 0.02081496 0.01870122
## 6:        strength 0.01072716 0.03091257 0.02429686
```

```
#Plot of feature importance: Feature have automatically been divided in 2 clusters: the interesting fea
xgb.plot.importance(importance, numberOfClusters = c(1:1))  + theme_bw() + theme(legend.position = "non
```


Feature importance
dribbling
standing_tackle
gk_reflexes
marking
positioning
strength
height
agility
balance
long_shots
aggression
curve
interceptions
finishing
overall_rating
weight
penalties
heading_accuracy
shot_power
vision
stamina
crossing
ball_control
sliding_tackle
free_kick_accuracy
acceleration
jumping
age
sprint_speed
bmi
short_passing
gk_positioning
reactions
volleys
gk_handling
long_passing
potential
gk_diving
gk_kicking
Features
0.0
0.2
0.4
0.6
Gain