

# **Práctica: Trabajo AEM**

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Introducción al conjunto de datos</b>                | <b>1</b>  |
| <b>2</b> | <b>Tratamiento de los datos</b>                         | <b>1</b>  |
| <b>3</b> | <b>Análisis de los datos</b>                            | <b>5</b>  |
| 3.1      | Análisis inicial de los datos . . . . .                 | 5         |
| 3.2      | PCA (Análisis de las Componentes Principales) . . . . . | 11        |
| 3.3      | Análisis Discriminante . . . . .                        | 24        |
| 3.4      | Análisis cluster . . . . .                              | 30        |
| 3.4.1    | Cluster con K-means . . . . .                           | 30        |
| 3.4.2    | Cluster jerárquico . . . . .                            | 31        |
| 3.4.3    | Número óptimo de clusters . . . . .                     | 33        |
| 3.5      | Regresión . . . . .                                     | 35        |
| 3.5.1    | Regresión lineal múltiple . . . . .                     | 35        |
| 3.5.2    | Regresión Multinomial . . . . .                         | 39        |
| <b>4</b> | <b>Conclusión</b>                                       | <b>49</b> |

## **1 Introducción al conjunto de datos**

Este dataset abarca las estadísticas de los jugadores de la ACB desde la temporada 1983-1984 a la temporada 2022-2023 y consta de 10742 filas y 37 columnas. Los datos se han recopilado a nivel de temporada, es decir, cada fila representa el rendimiento general del jugador en la temporada en cuestión. Además, se incluyen diversos datos personales o biográficos del jugador, como su nombre completo, altura, fecha y lugar de nacimiento. En cuanto a los datos estadísticos, estos resumen el desempeño del jugador a lo largo de la temporada, incluyendo, entre otros, el número de partidos que ha jugado, el total de minutos, puntos, asistencias, rebotes, etc.

## **2 Tratamiento de los datos**

Primero de todo vamos a importar nuestros datos:

```
acb <- read.csv("../data/acb_datos_jugadores_1983_2022.csv")
```

Si observamos los nombres de las columnas del dataset

```
tail(colnames(acb))
```

```
## [1] "Número.de.faltas.realizadas" "Número.de.faltas.recibidas"  
## [3] "X..." "Valoración"  
## [5] "Partidos.ganados" "Partidos.perdidos"
```

podemos observar que hay muchos campos diferentes para analizar y de hecho hay una columna cuyo nombre es “X...” y no se explica por ningún lado su significado, así que la vamos a eliminar

```
acb <- acb[,-34]  
print(colnames(acb))
```

Pasamos a ver el tipo de datos que tenemos en las variables con:

```
str(head(acb))
```

```
## 'data.frame': 6 obs. of 36 variables:  
## $ Nombre : chr "Fabien Causseur" "Fabien Causseur"  
## $ Equipo : chr "Caja Laboral" "Laboral Kutxa Baskonia"  
## $ Posición : chr "Alero" "Alero" "Alero" "Alero" ...  
## $ Altura : chr "1,96 m" "1,96 m" "1,96 m" "1,96 m" ...  
## $ Lugar.de.nacimiento : chr "Brest, Francia" "Brest, Francia" "Brest, Francia" ...  
## $ Fecha.de.nacimiento : chr "16/06/1987 (35 años)" "16/06/1987 (35 años)" ...  
## $ Nacionalidad : chr "Francia" "Francia" "Francia" "Francia" ...  
## $ Temporada : chr "12-13" "13-14" "14-15" "15-16" ...  
## $ Número.de.partidos : int 35 29 37 24 38 40  
## $ Total.minutos.jugados : num 722 715 957 635 677 611  
## $ Número.de.partidos.en.alineación.inicial: int 21 21 32 24 24 17  
## $ Total.puntos : num 315 255 355 215 277 289  
## $ Máximo.de.puntos : int 22 17 25 21 21 20  
## $ Tiros.de.3.puntos.anotados : int 28 22 37 30 32 31  
## $ Tiros.de.3.puntos.intentados : int 87 60 92 86 71 80  
## $ Porcentaje.de.acierto.tiros.de.3.puntos : chr "32,2%" "36,7%" "40,2%" "34,9%" ...  
## $ Tiros.de.2.puntos.anotados : int 86 69 88 39 72 81  
## $ Tiros.de.2.puntos.intentados : int 148 115 149 80 120 135  
## $ Porcentaje.de.acierto.tiros.de.2.puntos : chr "58,1%" "60,0%" "59,1%" "48,8%" ...  
## $ Tiros.libres.anotados : int 59 51 68 47 37 34  
## $ Tiros.libres.intentados : int 78 64 92 66 48 45  
## $ Porcentaje.de.acierto.tiros.libres : chr "75,6%" "79,7%" "73,9%" "71,2%" ...  
## $ Número.de.rebotes.ofensivos : int 22 23 13 13 12 18  
## $ Número.de.rebotes.defensivos : int 51 64 96 49 59 36  
## $ Total.rebotes : int 73 87 109 62 71 54  
## $ Total.asistencias : int 47 35 88 63 56 48  
## $ Número.de.balones.perdidos : int 31 25 51 35 21 22  
## $ Número.de.balones.recuperados : int 44 40 50 34 36 38  
## $ Número.de.tapones.realizados : int 1 7 6 3 4 2  
## $ Numero.de.tapones.recibidos : int 13 3 12 4 8 11
```

```

## $ Número.de.mates : int 2 1 1 0 0 0
## $ Número.de.faltas.realizadas : int 71 66 74 58 54 37
## $ Número.de.faltas.recibidas : int 98 68 130 77 57 57
## $ Valoración : num 297 271 463 243 290 272
## $ Partidos.ganados : int 25 15 20 18 33 33
## $ Partidos.perdidos : int 10 14 17 6 5 7

```

Podemos observar que las características que son de porcentaje son de tipo “chr”, así que las pasaremos a “num”. De igual manera, “Total.puntos” es de tipo “num” así que lo vamos a pasar a “int” y “Altura” es de tipo “chr” así que lo vamos a pasar a “num”:

```

library(dplyr)

acb <- acb |>
  mutate_at(vars(starts_with("Porcentaje")),
            funs(as.numeric(gsub("[%,]", "", .))/1000))

acb$Total.puntos <- as.integer(acb$Total.puntos)

acb <- acb |>
  mutate_at(vars(starts_with("Altura")),
            funs(as.numeric(gsub("[m]", "", .))/100))

```

Finalmente podemos ver que la columna de los minutos no tiene mucha coherencia, pues los jugadores con muchos partidos tienen uno y pico minutos jugados, mientras que hay otros que tienen novecientos. Esto ocurre porque el que creó el dataset los introdujo mal, así que para arreglarlo podemos darnos cuenta de que cuando se juegan menos de 1000 minutos se pone en formato “número\_minutos.000” mientras que si se han jugado 1000 o más minutos se pone como si el número de minutos se hubiera dividido entre mil.

Para arreglarlo hacemos lo siguiente:

```

# Aplicar la condición: si número de partidos >= 25 y minutos están entre 1 y 2,
# multiplicar por 1000; de lo contrario, mantener el valor original
acb$Total.minutos.jugados <- ifelse(acb$Número.de.partidos >= 25 & acb$Total.minutos.jugados <= 2,
                                         acb$Total.minutos.jugados * 1000,
                                         acb$Total.minutos.jugados)

```

Nótese que hemos utilizado el hecho de que para jugar 1000 minutos, mínimo hemos tenido que jugar 25 partidos (40 minutos completos cada partido). En realidad este número podría variar, porque puede haber prórrogas, pero mirando el dataset parece haber funcionado.

Me he dado cuenta de otro error del dataset, cuando estaba ya terminando el trabajo. Así que cuando lleguemos a la parte de regresión mencionaré cómo me he dado cuenta. Pero el problema es básicamente que las columnas de “Número.de.balones.perdidos” y “Número.de.balones.recuperados” están intercambiadas. Así que simplemente, intercambiaremos los nombres de las columnas así:

```

acb <- acb |>
  rename(
    Número.de.balones.perdidos = Número.de.balones.recuperados,
    Número.de.balones.recuperados = Número.de.balones.perdidos
  )

```

Finalmente vamos a quitar los jugadores que tienen altura cero, pues no es una gran cantidad del total de datos y nos permitirá hacer mejores regresiones:

```
acb <- acb[acb$Altura != 0, ]
```

Ahora vamos a dividir el dataset en varios diferentes, para poder manejarlo mejor y tener pequeños datasets más “especializados”:

```
acb_jugador_hist <- acb |>
  group_by(Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento,
            Nacionalidad) |>
  summarise(total_partidos = sum(Número.de.partidos),
             total_minutos = sum(Total.minutos.jugados),
             total_titular = sum(Número.de.partidos.en.alineación.inicial),
             total_puntos = sum(Total.puntos),
             max_puntos = max(Máximo.de.puntos),
             total_triple_anotados = sum(Tiros.de.3.puntos.anotados),
             total_triple_intentados = sum(Tiros.de.3.puntos.intentados),
             porcentaje_triple = mean(Porcentaje.de.acierto.tiros.de.3.puntos),
             total_dos_anotados = sum(Tiros.de.2.puntos.anotados),
             total_dos_intentados = sum(Tiros.de.2.puntos.intentados),
             porcentaje_dos = mean(Porcentaje.de.acierto.tiros.de.2.puntos),
             total_tl_anotados = sum(Tiros.libres.anotados),
             total_tl_intentados = sum(Tiros.libres.intentados),
             porcentaje_tl = mean(Porcentaje.de.acierto.tiros.libres),
             total_rbo = sum(Número.de.rebotes.ofensivos),
             total_rbd = sum(Número.de.rebotes.defensivos),
             total_rb = sum(Total.rebotes),
             total_asis = sum(Total.asistencias),
             total_bal_per = sum(Número.de.balones.perdidos),
             total_bal_rec = sum(Número.de.balones.recuperados),
             total_tapones_real = sum(Número.de.tapones.realizados),
             total_tapones_rec = sum(Numero.de.tapones.recibidos),
             total_mates = sum(Número.de.mates),
             total_faltas_real = sum(Número.de.faltas.realizadas),
             total_faltas_rec = sum(Número.de.faltas.recibidas),
             valoracion_pro = mean(Valoración),
             total_ganados = sum(Partidos.ganados),
             total_perdidos = sum(Partidos.perdidos)
  )
```

Vamos a ver si se ha realizado bien la transformación:

```
head(acb_jugador_hist)
```

```
## # A tibble: 6 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [6]
##   Nombre    Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>     <chr>    <dbl> <chr>           <chr>           <chr>
## 1 " John D~ Base      1.8  Kansas City (Misso~ 28/10/1988 (34 año~ Bosnia y He~
```

```

## 2 "Aaron A~ Alero      1.96 Douglas (Georgia),~ 09/01/1971 (52 año~ EE.UU.
## 3 "Aaron E~ Píivot     2.06 Tuscaloosa (Alabama~ 26/07/1993 (29 año~ EE.UU.
## 4 "Aaron J~ Píivot     2.08 Dekalb, Illinois, ~ 10/01/1977 (46 año~ EE.UU.
## 5 "Aaron L~ Base       1.93 Hartford (Connecticut~ 06/05/1986 (36 año~ EE.UU.
## 6 "Aaron M~ Base       1.86 Portland, Oregon, ~ 13/04/1983 (40 año~ EE.UU.
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...

```

Así, tenemos ya terminada el primer conjunto de datos que utilizaremos para hacer el análisis, que es básicamente un histórico desde la temporada 1983-1984 hasta la temporada 2022-2023 de todos los jugadores de la ACB.

Ahora vamos a construir un segundo conjunto de datos que tendrá los datos de la última temporada registrada (la 2022-2023):

```

acb_2022_2023 <- acb[acb$Temporada == "22-23", ]
head(acb_2022_2023)

```

## 3 Análisis de los datos

### 3.1 Análisis inicial de los datos

Primero calculamos varias medidas descriptivas sobre las variables:

```
summary(acb_jugador_hist)
```

```

##      Nombre           Posición          Altura      Lugar.de.nacimiento
##  Length:3398      Length:3398      Min.   :1.670  Length:3398
##  Class :character  Class :character  1st Qu.:1.920  Class :character
##  Mode  :character  Mode  :character  Median :2.000  Mode  :character
##                                         Mean   :1.985
##                                         3rd Qu.:2.050
##                                         Max.   :2.240
##      Fecha.de.nacimiento Nacionalidad      total_partidos  total_minutos
##  Length:3398      Length:3398      Min.   : 1.00  Min.   :  0.0
##  Class :character  Class :character  1st Qu.: 9.00  1st Qu.: 114.2
##  Mode  :character  Mode  :character  Median :31.00  Median : 560.5
##                                         Mean   :74.94  Mean   :1591.0
##                                         3rd Qu.:88.00  3rd Qu.:1799.8
##                                         Max.   :824.00  Max.   :20212.0
##      total_titular      total_puntos      max_puntos      total_triple_anotados
##  Min.   : 0.00  Min.   : 0.00  Min.   : 0.00  Min.   :  0.00
##  1st Qu.: 0.00  1st Qu.: 31.25  1st Qu.: 9.00  1st Qu.:  0.00
##  Median : 8.00  Median :208.00  Median :18.00  Median :  6.00
##  Mean   :32.87  Mean   :631.78  Mean   :17.93  Mean   : 50.97

```

```

## 3rd Qu.: 35.00 3rd Qu.: 733.75 3rd Qu.:26.00 3rd Qu.: 46.00
## Max. :525.00 Max. :9759.00 Max. :54.00 Max. :1233.00
## total_triple_intentados porcentaje_triple total_dos_anotados
## Min. : 0.0 Min. :0.0000 Min. : 0.0
## 1st Qu.: 2.0 1st Qu.:0.0000 1st Qu.: 8.0
## Median : 23.0 Median :0.2500 Median : 56.0
## Mean : 142.1 Mean :0.2209 Mean : 182.3
## 3rd Qu.: 135.0 3rd Qu.:0.3480 3rd Qu.: 207.8
## Max. :3077.0 Max. :1.0000 Max. :3277.0
## total_dos_intentados porcentaje_dos total_tl_anotados total_tl_intentados
## Min. : 0.0 Min. :0.0000 Min. : 0.0 Min. : 0.0
## 1st Qu.: 19.0 1st Qu.:0.3911 1st Qu.: 5.0 1st Qu.: 8.0
## Median : 114.0 Median :0.4860 Median : 35.0 Median : 49.0
## Mean : 345.6 Mean :0.4383 Mean : 118.9 Mean : 162.5
## 3rd Qu.: 397.8 3rd Qu.:0.5429 3rd Qu.: 132.0 3rd Qu.: 183.0
## Max. :5564.0 Max. :1.0000 Max. :2289.0 Max. :3349.0
## porcentaje_tl total_rbo total_rbd total_rb
## Min. :0.0000 Min. : 0.00 Min. : 0.0 Min. : 0.0
## 1st Qu.:0.5000 1st Qu.: 4.00 1st Qu.: 10.0 1st Qu.: 14.0
## Median :0.6670 Median : 21.00 Median : 55.0 Median : 78.0
## Mean : 0.5820 Mean : 73.11 Mean : 166.7 Mean : 239.8
## 3rd Qu.:0.7709 3rd Qu.: 77.00 3rd Qu.: 187.0 3rd Qu.: 268.8
## Max. :1.0000 Max. :1982.00 Max. :3068.0 Max. :4725.0
## total_asis total_bal_per total_bal_rec total_tapones_real
## Min. : 0.00 Min. : 0 Min. : 0.00 Min. : 0.00
## 1st Qu.: 3.00 1st Qu.: 8 1st Qu.: 3.00 1st Qu.: 0.00
## Median : 22.00 Median : 36 Median : 19.00 Median : 4.00
## Mean : 94.89 Mean : 100 Mean : 63.03 Mean : 20.77
## 3rd Qu.: 84.00 3rd Qu.: 115 3rd Qu.: 66.75 3rd Qu.: 18.00
## Max. :2760.00 Max. :1443 Max. :1164.00 Max. :738.00
## total_tapones_rec total_mates total_faltas_real total_faltas_rec
## Min. : 0.0 Min. : 0.00 Min. : 0.0 Min. : 0.0
## 1st Qu.: 0.0 1st Qu.: 0.00 1st Qu.: 15.0 1st Qu.: 3.0
## Median : 5.0 Median : 0.50 Median : 64.5 Median : 35.0
## Mean : 16.1 Mean : 13.21 Mean : 167.9 Mean : 139.2
## 3rd Qu.: 18.0 3rd Qu.: 9.00 3rd Qu.: 194.0 3rd Qu.: 141.8
## Max. :391.0 Max. :732.00 Max. :2181.0 Max. :3262.0
## valoracion_pro total_ganados total_perdidos
## Min. : -22.0 Min. : 0.00 Min. : 0.00
## 1st Qu.: 12.0 1st Qu.: 3.00 1st Qu.: 5.00
## Median : 93.0 Median : 13.00 Median : 17.00
## Mean : 148.1 Mean : 37.59 Mean : 37.32
## 3rd Qu.: 234.0 3rd Qu.: 42.75 3rd Qu.: 46.00
## Max. : 889.0 Max. :601.00 Max. :349.00

```

Obtenemos toda esta retahíla de datos, pero si nos fijamos en algunos de ellos podemos ver primero de todo que entre la temporada 1983-1984 y la 2022-2023 han jugado un total de 3431 jugadores en la ACB (aparecen 3398 porque hemos filtrado los que no tenían altura).

Otro dato interesante es la media de altura, que confirma que para los jugadores de baloncesto son bastante altos, con una altura media de 1.985 metros.

También vemos que las variables van en rangos muy distintos, esto nos servirá más tarde cuando

tengamos que emplear distintas técnicas para las cuales estandarizaremos los datos.

El máximo de minutos que ha jugado un jugador entre estas temporadas ha sido de 20212 minutos. Lo que son unas 336,86 horas. Podemos buscar este jugador con “View(acb\_jugador\_hist)” y ordenando. Vemos que este jugador es Alberto Oliver Campos. Así podemos buscar también sus estadísticas:

```
jug_mas_min <- acb_jugador_hist %>%
  filter(total_minutos == 20212)

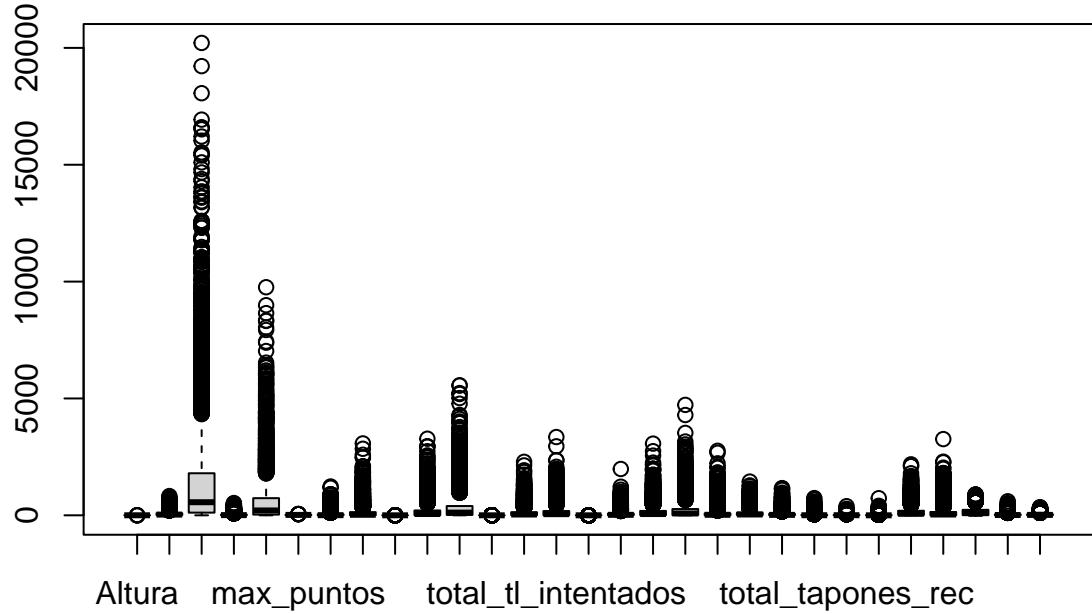
print(jug_mas_min)

## # A tibble: 1 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [1]
##   Nombre   Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>     <chr>    <dbl> <chr>           <chr>           <chr>
## 1 Joan Cre~ Base      1.76 Ripollet (Barcelon~ 24/11/1956 (66 año~ España
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...
```

Vemos también que la media del total de partidos jugados es 74.94 y que la media del total de puntos es 631.78, así que la media de puntos por partido es  $631.78/74.94 = 8,4305$  puntos/partido.

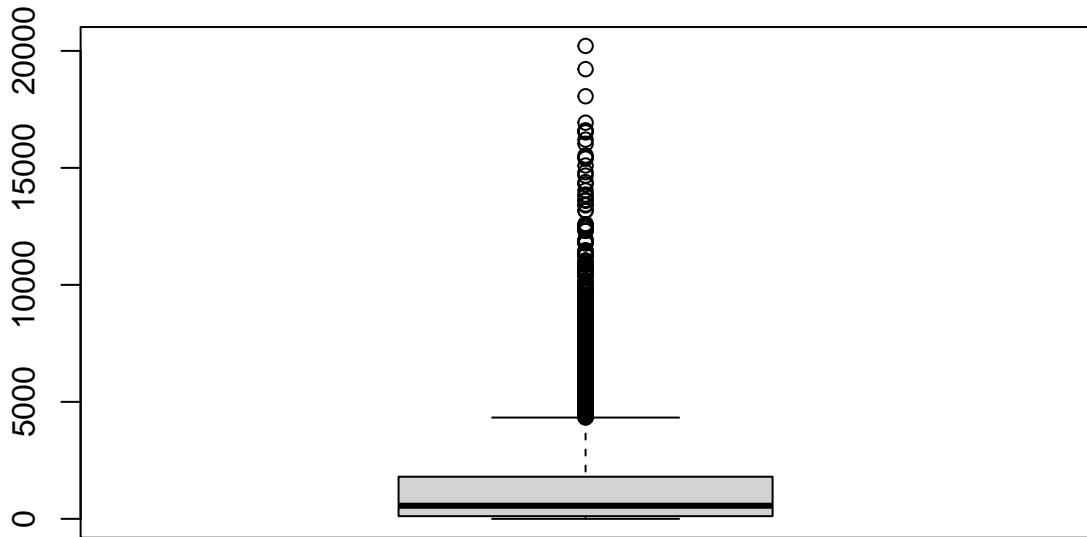
Podemos hacer también un diagrama de caja-bigotes para ver los rangos en los que se mueven nuestras variables:

```
boxplot(acb_jugador_hist[, -c(1,2,4:6)])
```



No podemos decir mucho de las variables en particular, pero lo que sí que podemos decir es que en general hay una grandísima cantidad de datos atípicos. Parece indicar que hay muchos jugadores que se salen de la norma “por arriba”, estos son los grandes jugadores. Vamos a ver el diagrama de la tercera variable que aparece:

```
boxplot(acb_jugador_hist[,8])
```



Esta variable es el total de minutos jugados, la linea central de la caja es la mediana, que si buscamos en el “summary” de arriba podemos ver que es 550. A partir del 5000 (más o menos), los datos ya se consideran outliers y vemos que hay un gran número de jugadores que superan este número, de hecho podemos calcular cuantos jugadores superan los 5000 minutos:

```
mas_5000 <- sum(acb_jugador_hist$total_minutos > 5000)

cat("El número de jugadores con más de 5000 minutos jugados es:", mas_5000)

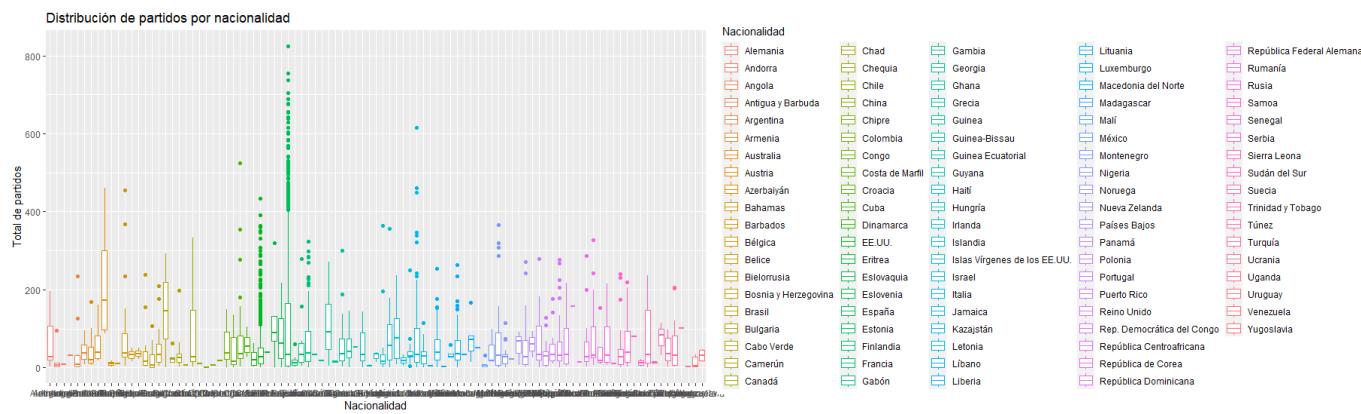
## El número de jugadores con más de 5000 minutos jugados es: 316
```

Teniendo en cuenta que tenemos 3398 jugadores, el número de datos atípicos en esta variable representa más o menos el 9.3% de los datos.

Podemos también graficar el número de puntos totales por altura:

```
library(tidyverse)

acb_jugador_hist |>
  ggplot(aes(x = Nacionalidad, y = total_partidos)) +
  geom_boxplot(aes(color = Nacionalidad)) +
  labs(title = "Distribución de partidos por nacionalidad",
       x = "Nacionalidad",
       y = "Total de partidos")
```



Ocurre una cosa curiosa, y es que el país cuya caja-bigotes está más alta es la de Azerbaiyán, luego también podemos observar que hay dos países con muchos atípicos, el que tiene menos es EEUU y el que más tiene es España.

Esto nos indica que los jugadores que más partidos tienen a sus espaldas son los españoles como era de esperar, pero los siguientes son los estadounidenses, lo cuál es normal, ya que en EEUU están los mejores jugadores de baloncesto y es normal que los equipos busquen a estos jugadores.

Antes de seguir vamos a ver los jugadores de Azerbaiyán para ver qué tienen de especial:

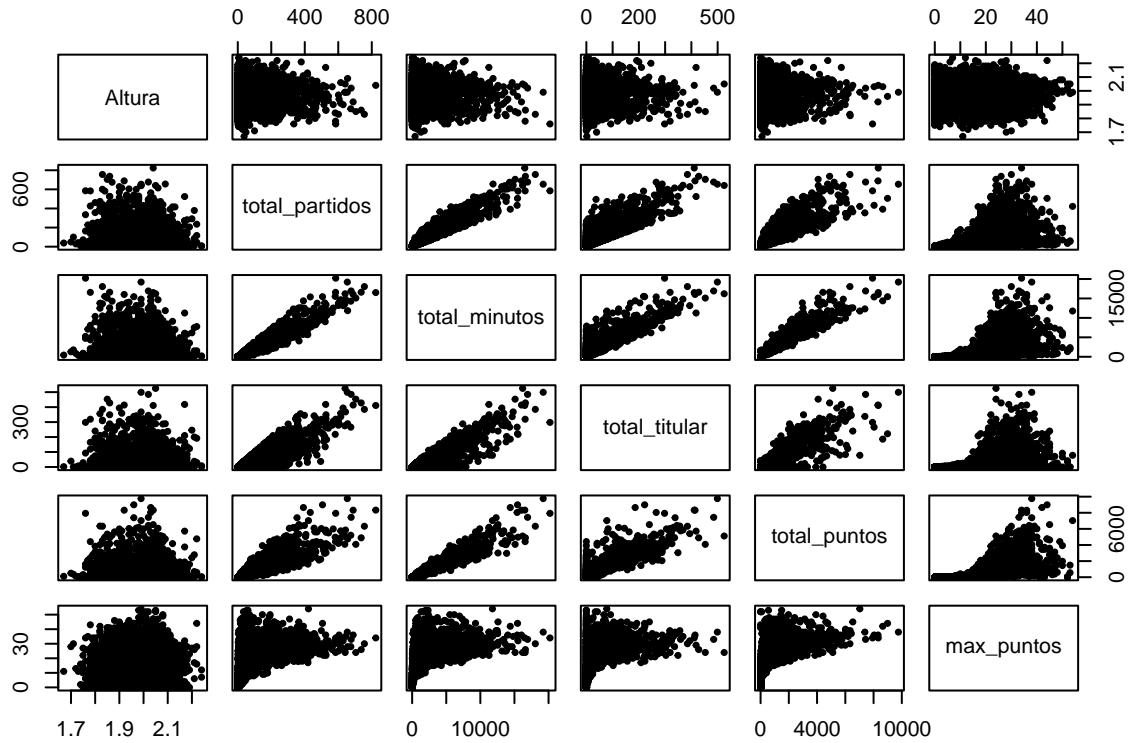
```
azerbaiyanos <- acb_jugador_hist[acb_jugador_hist$Nacionalidad == "Azerbaiyán", ]  
print(azerbaiyanos)
```

```
## # A tibble: 4 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [4]
##   Nombre   Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>     <chr>    <dbl>  <chr>           <chr>           <chr>
## 1 "Jaycee ~ Escolta    1.88 Laramie (Wyoming),~ 16/04/1983 (40 año~ Azerbaiyán
## 2 "Jordan ~ Escolta    1.88 Las Vegas (Nevada)~ 06/06/1997 (25 año~ Azerbaiyán
## 3 "Niklas ~ Ala-pív~  2.03 Beverly (Massachus~ 20/10/1983 (39 año~ Azerbaiyán
## 4 "Spencer~ Ala-pív~  2.03 Pocatello (Idaho),~ 11/07/1980 (42 año~ Azerbaiyán
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...
```

Vemos que son 4 jugadores nacidos en EEUU, nacionalizados con Azerbaiyán con bastantes partidos en ACB. Ahora sí que cuadra lo que mostraba la gráfica: parece que la selección de Azerbaiyán busca jugadores estadounidenses en ACB para “ficharlos”.

Vamos a probar hacer un plot para ver cómo se correlacionan los datos, pero no de todas las variables porque sería muy costoso computacionalmente y difícil de visualizar:

```
plot(acb_jugador_hist[,c(3,7:11)],pch=20,cex=0.8)
```

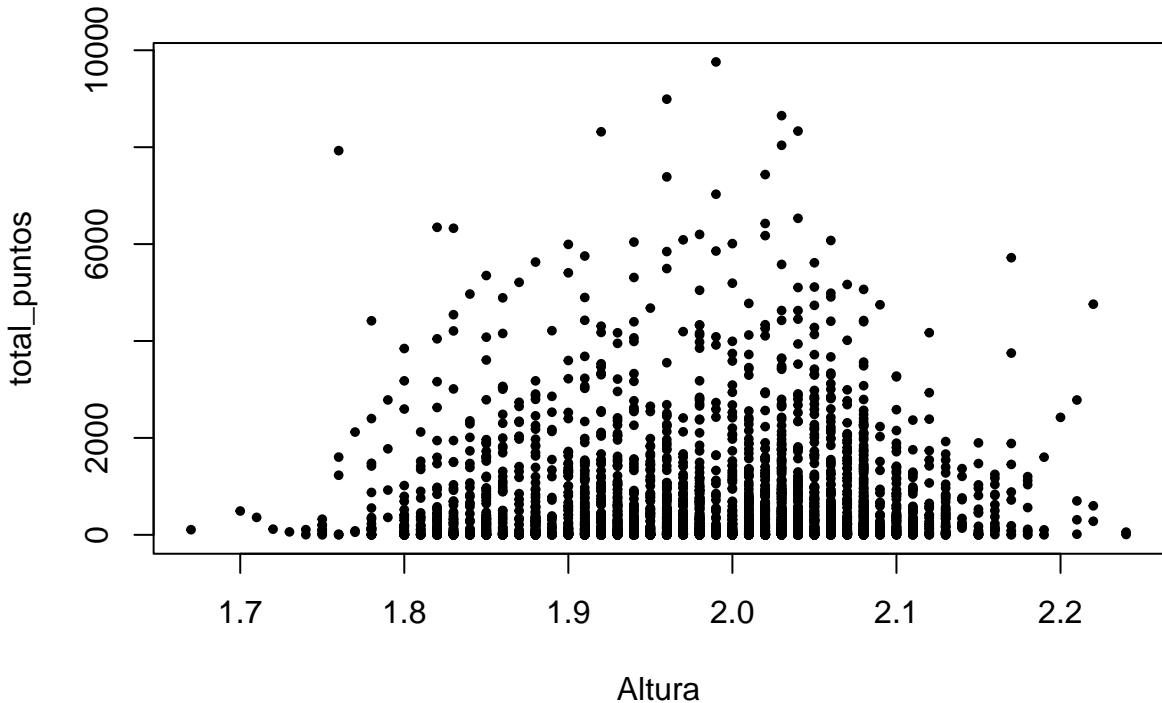


Observamos que aparecen gráficas que se asemejan a la forma de una flecha. Esto indica fuertes dependencias entre la variables. Estas dependencias van de menos a más si partimos desde la base y vamos a la punta.

Vemos también que cuando “max\_puntos” está en el eje X y “total” está en el eje Y obtenemos una gráfica que se asemeja a la distribución normal. Esto tiene sentido, pues hay muy pocos jugadores que tienen un mínimo de puntos muy bajo o un mínimo de puntos muy alto. Además ocurre algo curioso, los jugadores con un máximo de puntos alto, ha estado poco tiempo en la ACB, pues de haber pasado más tiempo sus totales estarían más altos. Esto puede ser señal de que al ser tan buenos, han sido fichados por equipos de mejores ligas y ya han permanecido en esas ligas el resto de su carrera.

Ahora podemos hacer un plot para ver mejor cómo se relaciona la altura con el número total de puntos en concreto:

```
plot(acb_jugador_hist[,c(3,10)],pch=20,cex=0.8)
```



Vemos que hay muchos jugadores que han metido pocos puntos. Conforme vamos subiendo el número de puntos el número de jugadores decrece hasta llegar a la punta de la normal, donde ya solo hay un único jugador.

Observamos que los jugadores más altos no son necesariamente los que más puntos meten, pues de ser así la curva estaría inclinada a la derecha. En cambio parece ser bastante simétrica, lo que indica que los jugadores con una altura “normal” (en el contexto del baloncesto) meten más puntos.

### 3.2 PCA (Análisis de las Componentes Principales)

Para hacer el análisis de las componentes principales usaremos la matriz de correlaciones, pues tenemos datos con rangos muy diferentes y unidades diferentes. Haciendo esto conseguimos estandarizar las variables y que las unidades de medida desaparezcan.

Ahora vamos a hacer un dataframe con solamente las variables que tengan datos numéricos para poder hacer el PCA:

Empezaremos haciendo el PCA al dataset de ‘acb\_jugador\_hist’

```
#Vemos los tipos de columnas por si hay que eliminar alguna
tipos_columnas <- lapply(acb_jugador_hist, typeof)
print(tipos_columnas)
```

```
#Eliminamos las que no son de tipo numérico
datos_pca_hist <- acb_jugador_hist[,-c(1,2,4:6)]
```

Ahora hacemos el PCA:

```

PCA_hist<-princomp(datos_pca_hist,cor=TRUE)
summary(PCA_hist,loadings=TRUE)

## Importance of components:
##                               Comp.1      Comp.2      Comp.3      Comp.4      Comp.5
## Standard deviation      4.3268057 1.68243885 1.4514963 0.94217565 0.8489981
## Proportion of Variance 0.6455603 0.09760691 0.0726497 0.03061017 0.0248551
## Cumulative Proportion  0.6455603 0.74316717 0.8158169 0.84642705 0.8712821
##                               Comp.6      Comp.7      Comp.8      Comp.9      Comp.10
## Standard deviation      0.8012847 0.74216273 0.69244458 0.66564298 0.57079320
## Proportion of Variance 0.0221399 0.01899329 0.01653378 0.01527864 0.01123465
## Cumulative Proportion  0.8934220 0.91241534 0.92894911 0.94422775 0.95546240
##                               Comp.11     Comp.12     Comp.13     Comp.14
## Standard deviation      0.54424993 0.471081309 0.400717662 0.359544993
## Proportion of Variance 0.01021407 0.007652331 0.005537057 0.004457676
## Cumulative Proportion  0.96567647 0.973328804 0.978865860 0.983323536
##                               Comp.15     Comp.16     Comp.17     Comp.18
## Standard deviation      0.33284414 0.322712383 0.28811488 0.242767749
## Proportion of Variance 0.00382018 0.003591148 0.00286242 0.002032282
## Cumulative Proportion  0.98714372 0.990734864 0.99359728 0.995629566
##                               Comp.19     Comp.20     Comp.21     Comp.22
## Standard deviation      0.201399588 0.17198671 0.1367648728 0.134708479
## Proportion of Variance 0.001398683 0.00101998 0.0006449873 0.000625737
## Cumulative Proportion  0.997028249 0.99804823 0.9986932165 0.999318953
##                               Comp.23     Comp.24     Comp.25     Comp.26
## Standard deviation      0.0784073545 0.0744383325 0.0573147219 0.0522822420
## Proportion of Variance 0.0002119901 0.0001910712 0.0001132751 0.0000942563
## Cumulative Proportion  0.9995309436 0.9997220148 0.9998352899 0.9999295462
##                               Comp.27     Comp.28     Comp.29
## Standard deviation      4.518742e-02 1.12118e-03 9.212110e-08
## Proportion of Variance 7.041044e-05 4.33464e-08 2.926309e-16
## Cumulative Proportion  1.000000e+00 1.000000e+00 1.000000e+00
##
## Loadings:
##                               Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8
## Altura                      0.374      0.614  0.126  0.443  0.419  0.113
## total_partidos                0.217      0.120  0.132 -0.157  0.110
## total_minutos                 0.226
## total_titular                  0.214      0.159  0.161 -0.222
## total_puntos                   0.224      -0.146
## max_puntos                     0.140      0.460  -0.223
## total_triple_anotados        0.166  -0.333      0.148      0.311
## total_triple_intentados      0.171  -0.334      0.152      0.276
## porcentaje_triple              -0.194  0.377  -0.462  0.505  0.129  0.460 -0.266
## total_dos_anotados            0.211  0.144      -0.124  -0.223      0.144
## total_dos_intentados          0.214  0.118      -0.114  -0.217      0.146
## porcentaje_dos                  0.123  0.433  0.441      -0.425 -0.184 -0.583
## total_t1_anotados              0.221      -0.151
## total_t1_intentados            0.222      -0.136
## porcentaje_t1                  0.194  0.264      0.516      0.106 -0.210 -0.138  0.701
## total_rbo

```



|                            |         |         |         |         |         |         |         |
|----------------------------|---------|---------|---------|---------|---------|---------|---------|
| ## porcentaje_triple       |         |         |         |         |         |         |         |
| ## total_dos_anotados      | 0.350   |         | -0.137  |         |         | 0.120   |         |
| ## total_dos_intentados    | 0.339   |         | -0.105  |         |         | 0.145   |         |
| ## porcentaje_dos          |         |         |         |         |         |         |         |
| ## total_tl_anotados       | -0.344  | 0.201   |         |         |         | -0.201  | -0.338  |
| ## total_tl_intentados     | -0.353  | 0.154   |         |         |         | -0.157  | -0.195  |
| ## porcentaje_tl           |         |         |         |         |         |         |         |
| ## total_rbo               | -0.108  |         | 0.244   | -0.129  | 0.685   | -0.129  |         |
| ## total_rbd               |         | 0.131   |         |         | -0.572  | 0.137   |         |
| ## total_rb                |         | 0.114   |         |         | -0.170  |         |         |
| ## total_asis              | 0.264   | 0.311   |         |         | 0.212   | 0.172   | -0.281  |
| ## total_bal_per           |         | 0.182   |         | -0.477  | -0.228  | -0.498  | 0.523   |
| ## total_bal_rec           | -0.306  | -0.471  | 0.503   | 0.316   |         |         | 0.209   |
| ## total_tapones_real      |         |         |         |         |         |         |         |
| ## total_tapones_rec       | 0.367   | -0.152  | 0.455   |         | -0.160  |         | -0.178  |
| ## total_mates             |         | 0.102   |         |         |         |         |         |
| ## total_faltas_real       |         | -0.234  |         | -0.695  |         | 0.436   | -0.267  |
| ## total_faltas_rec        | -0.259  |         | -0.261  |         |         | 0.488   | 0.522   |
| ## valoracion_pro          |         |         |         |         |         |         |         |
| ## total_ganados           | 0.222   | -0.165  | -0.149  | 0.153   |         | -0.141  |         |
| ## total_perdidos          |         | 0.336   | -0.115  | 0.228   |         |         |         |
| ##                         | Comp.23 | Comp.24 | Comp.25 | Comp.26 | Comp.27 | Comp.28 | Comp.29 |
| ## Altura                  |         |         |         |         |         |         |         |
| ## total_partidos          |         |         |         |         |         | 0.805   |         |
| ## total_minutos           | 0.858   | 0.263   |         | -0.144  |         |         |         |
| ## total_titular           | -0.181  |         |         |         |         |         |         |
| ## total_puntos            |         | -0.618  | 0.504   | -0.167  | -0.346  |         |         |
| ## max_puntos              |         |         |         |         |         |         |         |
| ## total_triple_anotados   | 0.117   | -0.292  | -0.559  | 0.279   |         |         |         |
| ## total_triple_intentados | -0.243  | 0.491   | 0.440   | -0.186  | 0.105   |         |         |
| ## porcentaje_triple       |         |         |         |         |         |         |         |
| ## total_dos_anotados      |         |         |         | 0.348   | 0.658   |         |         |
| ## total_dos_intentados    | -0.151  | 0.370   | -0.394  | -0.229  | -0.477  |         |         |
| ## porcentaje_dos          |         |         |         |         |         |         |         |
| ## total_tl_anotados       | -0.170  |         | -0.214  | -0.503  | 0.314   |         |         |
| ## total_tl_intentados     |         | 0.240   | 0.165   | 0.631   | -0.311  |         |         |
| ## porcentaje_tl           |         |         |         |         |         |         |         |
| ## total_rbo               |         |         |         |         |         | -0.259  |         |
| ## total_rbd               |         |         |         |         |         | -0.547  |         |
| ## total_rb                |         |         |         |         |         | 0.796   |         |
| ## total_asis              |         |         |         |         |         |         |         |
| ## total_bal_per           |         |         |         |         |         |         |         |
| ## total_bal_rec           |         | -0.137  |         |         |         |         |         |
| ## total_tapones_real      |         |         |         |         |         |         |         |
| ## total_tapones_rec       |         |         |         |         |         |         |         |
| ## total_mates             |         |         |         |         |         |         |         |
| ## total_faltas_real       |         |         |         |         |         |         |         |
| ## total_faltas_rec        | 0.203   |         |         |         |         |         |         |
| ## valoracion_pro          |         |         |         |         |         |         |         |
| ## total_ganados           |         |         |         |         |         | -0.456  |         |
| ## total_perdidos          | -0.115  |         |         |         |         | -0.379  |         |

Si nos fijamos en la fila que pone ‘Cumulative Proportion’ podemos ver que la primera componente principal tiene aproximadamente el 64.56% de la información. Y si cogemos las tres primeras componentes principales llegamos al 81.58%. Es decir, estamos resumiendo el 81.58% de la información de 29 variables en solamente 3 variables.

Ahora podemos pasar a analizar los loadings o cargas de la primera componente principal para ver qué es lo que esta representa. Podríamos observar los datos de arriba, pero para hacerlo más sencillo vamos a imprimir solamente los datos correspondientes a la primera componente principal:

```
PCA_hist$loadings->L_hist
L_hist[,1]
```

```
##          Altura      total_partidos      total_minutos
## 0.008234993 0.216616334 0.226265062
##      total_titular      total_puntos      max_puntos
## 0.213609871 0.224129527 0.139696079
##  total_triple_anotados total_triple_intentados porcentaje_triple
## 0.166299197 0.170586750 0.047606497
##  total_dos_anotados  total_dos_intentados porcentaje_dos
## 0.210830546 0.214277545 0.063273130
##  total_tl_anotados  total_tl_intentados porcentaje_tl
## 0.220779053 0.222314303 0.076122320
##      total_rbo          total_rbd      total_rb
## 0.194215701 0.214144866 0.210514873
##      total_asis          total_bal_per      total_bal_rec
## 0.178858992 0.222562543 0.211090724
##  total_tapones_real  total_tapones_rec      total_mates
## 0.145130900 0.198872556 0.123849710
##  total_faltas_real  total_faltas_rec  valoracion_pro
## 0.219960298 0.213973607 0.133736955
##  total_ganados      total_perdidos
## 0.213841874 0.203403344
```

Podemos observar que la altura y los porcentajes no se tienen prácticamente en cuenta, en cambio el total de cada estadística está más o menos al 20% cada una. Lo que esto quiere indicar es que la primera componente principal nos dice los jugadores que tienen todas sus estadísticas altas (sin tener en cuenta sus porcentajes). Por lo que si buscamos un jugador que haya jugado mucho y haya hecho muchos puntos, asistencias, rebotes... tendrá un valor alto en esta componente. Vamos a comprobarlo:

Vamos primero a buscar por ejemplo en el dataset a Sergio Llull:

```
library(dplyr)

sergio_llull <- acb_jugador_hist %>%
  filter(grep("Llull", Nombre))

print(sergio_llull)

## # A tibble: 1 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [1]
##   Nombre   Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
```

```

##   <chr>     <chr>     <dbl> <chr>           <chr>           <chr>
## 1 Sergio ~ Escolta     1.9 Mahón, España      15/11/1987 (35 año~ España
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...

```

```
which(acb_jugador_hist$Nombre == "Sergio Llull Meliá")
```

```
## [1] 2991
```

Así podemos saber el número que ocupa Sergio Llull en el dataset, ahora lo único que queda es ver el score que este tiene en la primera componente:

```

PCA_hist$scores->S_hist
S_hist[2991,1]

```

```

##   Comp.1
## 18.46979

```

Podemos comprobar que si ponemos a los 20 primeros jugadores ninguno de estos le supera:

```
S_hist[1:20,1]
```

```

## [1] -1.179263  6.940617 -2.549861  8.437873  1.313400 -0.879088  5.607658
## [8] -2.300303 -2.544072  1.684588  1.413093 -2.224880 -1.660611  0.139084
## [15] 11.936130 -1.281331 -3.002309 -2.722093 -2.712444 -2.272607

```

De hecho, podríamos buscar el jugador que tiene el score más alto en la primera componente. Antes de ejecutar ya podemos suponer que nos va a salir un jugador histórico de la competición con muchos partidos, puntos, minutos...

```
max(S_hist[,1])
```

```
## [1] 35.48703
```

```
acb_jugador_hist[which.max(S_hist[,1]),]
```

```

## # A tibble: 1 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [1]
##   Nombre   Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>     <chr>     <dbl> <chr>           <chr>           <chr>
## 1 Felipe R~ Ala-pív~    2.04 Córdoba, España      16/03/1980 (43 año~ España
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,

```

```
## #  porcentaje_triple <dbl>, total_dos_anotados <int>,
## #  total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #  total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #  total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...
```

Obtenemos que el score más alto es de 35.48703 y vemos que, como ya habíamos predicho, aparece Felipe Reyes.

También podríamos buscar el jugador con “menos intervención” de la competición:

```
min(S_hist[,1])
```

```
## [1] -3.341696
```

```
acb_jugador_hist[which.min(S_hist[,1]),]
```

```
## # A tibble: 1 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [1]
##   Nombre   Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>     <chr>    <dbl> <chr>           <chr>           <chr>
## 1 Pedro Ja~ Base      1.8 Córdoba, España  04/11/1972 (50 año~ España
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...
```

Tiene un score de -3.341696 y vemos que solamente hay registrado 1 partido y 1 minuto de juego.

Pasamos a analizar la segunda componente principal:

```
L_hist[,2]
```

|    | Altura                | total_partidos          | total_minutos     |
|----|-----------------------|-------------------------|-------------------|
| ## | 0.37360586            | -0.09900281             | -0.07141919       |
| ## | total_titular         | total_puntos            | max_puntos        |
| ## | -0.05347448           | -0.02016011             | 0.05482239        |
| ## | total_triple_anotados | total_triple_intentados | porcentaje_triple |
| ## | -0.33319453           | -0.33432192             | -0.19382493       |
| ## | total_dos_anotados    | total_dos_intentados    | porcentaje_dos    |
| ## | 0.14410249            | 0.11766172              | 0.12340781        |
| ## | total_tl_anotados     | total_tl_intentados     | porcentaje_tl     |
| ## | -0.01850951           | 0.02017938              | -0.02477096       |
| ## | total_rbo             | total_rbd               | total_rb          |
| ## | 0.26386206            | 0.16202799              | 0.19739246        |
| ## | total_asis            | total_bal_per           | total_bal_rec     |
| ## | -0.26637989           | -0.07666185             | -0.12212130       |
| ## | total_tapones_real    | total_tapones_rec       | total_mates       |
| ## | 0.36450528            | 0.01250342              | 0.34506826        |

```

##      total_faltas_real      total_faltas_rec      valoracion_pro
##      -0.03024503      -0.03795452      0.16700959
##      total_ganados      total_perdidos
##      -0.08737118      -0.10545945

```

Esta variable puede ser muy complicada de analizar para alguien que no sepa mucho de baloncesto, pero si te gusta el baloncesto se puede ver claramente lo que esta indica. La segunda componente principal indica los jugadores que juegan en la posición de pívot.

Los pívot tienen las siguientes características: son jugadores altos, generalmente son malos tirando triples pues su fuerte es el juego en la zona (cerca de la canasta), cogen muchos rebotes y hacen muchos tapones (debido a su físico). Además son los que más mates hacen (por su gran altura).

Vamos a hacer lo siguiente para comprobar que nuestras suposiciones son correctas: imprimiremos los 10 jugadores con el score más alto en la segunda componente y vamos a ver la posición en la que juegan:

```
acb_jugador_hist[order(S_hist[,2], decreasing = TRUE)[1:10],]
```

```

## # A tibble: 10 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [10]
##   Nombre  Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>    <chr>    <dbl> <chr>          <chr>          <chr>
## 1 "Franci~ Pívot      2.09 Chantada, España  01/05/1983 (39 año~ España
## 2 "Walter~ Pívot      2.2  Maio, Cabo Verde  22/03/1992 (31 año~ Cabo Verde
## 3 "Grange~ Pívot      2.03 Newark (New Jersey~ 18/06/1962 (60 año~ EE.UU.
## 4 "George~ Pívot      2.02 Kershaw (South Car~ 09/07/1961 (61 año~ EE.UU.
## 5 "Joseph~ Ala-pív~  2.04 Rochester (New Yor~ 20/07/1965 (57 año~ EE.UU.
## 6 "Arvyda~ Pívot      2.22 Kaunas, Lituania  19/12/1964 (58 año~ Lituania
## 7 "Derric~ Pívot      2.08 Bronx (New York), ~ 20/08/1972 (50 año~ EE.UU.
## 8 "Felipe~ Ala-pív~  2.04 Córdoba, España    16/03/1980 (43 año~ España
## 9 "Richar~ Pívot      2.04 Bangui, Repùblica ~ 04/07/1963 (59 año~ España
## 10 "Robert~ Pívot     2.21 Madrid, España     01/11/1975 (47 año~ España
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...

```

Podemos ver que 8 de los 10 jugadores juegan de pívot y 2 de ala-pívot. Podemos ver que en la segunda posición aparece Tavares, uno de los mejores pívots de europa actualmente. Y en la séptima posición aparece Felipe Reyes, uno de los mejores pívots que ha jugado para España.

De hecho podemos comprobar si es Tavares el que más score obtiene en la segunda componente en la temporada 22-23:

```

datos_pca_22_23 <- acb_2022_2023[,-c(1:3,5:8)]
PCA_22_23<-princomp(datos_pca_22_23,cor=TRUE)
PCA_22_23$scores->S_22_23
acb_2022_2023[order(S_22_23[,2], decreasing = TRUE)[1:10],c(1,3)]

```

```

##                                     Nombre Posición
## 6175          Tryggvi Snaer Hlinason     Pívote
## 78    Walter Samuel Tavares Da Veiga     Pívote
## 7737          Mamadou Ndiaye Niang     Pívote
## 7274          Khalifa Ababacar Diop     Pívote
## 119           Vincent Yann Poirier     Pívote
## 8618           Ethan Michael Happ     Pívote
## 1534           Artem Pustovyi     Pívote
## 9507           Michale Devonta Kyser     Pívote
## 7240 Aleksander Roman Balcerowski     Pívote
## 10734          Dusan Miletic     Pívote

```

Aparece segundo, pero ahora sí que todos los que aparecen juegan en la posición de pívot.

Pero primero deberíamos de comprobar qué es lo que nos dice la segunda componente principal del df “acb\_2022\_2023”, porque podríamos haber obtenido una cosa completamente diferente:

```

PCA_22_23$loadings->L_22_23
L_22_23[,2]

```

```

##                                     Altura
##                                     0.41221190
##                                     Número.de.partidos
##                                     -0.02933312
##                                     Total.minutos.jugados
##                                     -0.07892027
## Número.de.partidos.en.alineación.inicial
##                                     -0.04815239
##                                     Total.puntos
##                                     -0.07507167
##                                     Máximo.de.puntos
##                                     -0.08240650
##                                     Tiros.de.3.puntos.anotados
##                                     -0.33338027
##                                     Tiros.de.3.puntos.intentados
##                                     -0.33648528
## Porcentaje.de.acierto.tiros.de.3.puntos
##                                     -0.14599116
##                                     Tiros.de.2.puntos.anotados
##                                     0.14529989
##                                     Tiros.de.2.puntos.intentados
##                                     0.08439144
## Porcentaje.de.acierto.tiros.de.2.puntos
##                                     0.10996201
##                                     Tiros.libres.anotados
##                                     -0.01745132
##                                     Tiros.libres.intentados
##                                     0.03105408
## Porcentaje.de.acierto.tiros.libres
##                                     -0.08151165
## Número.de.rebotes.ofensivos
##                                     0.30419072

```

```

## Número.de.rebotes.defensivos
##                               0.15438144
## Total.rebotes
##                               0.21402960
## Total.asistencias
##                               -0.24052862
## Número.de.balones.recuperados
##                               -0.12407378
## Número.de.balones.perdidos
##                               -0.12143989
## Número.de.tapones.realizados
##                               0.34562010
## Numero.de.tapones.recibidos
##                               -0.01436962
## Número.de.mates
##                               0.37578372
## Número.de.faltas.realizadas
##                               0.03124244
## Número.de.faltas.recibidas
##                               -0.02504811
## Valoración
##                               0.02659578
## Partidos.ganados
##                               -0.01997160
## Partidos.perdidos
##                               -0.02402766
##
```

Podemos observar que la altura actualmente se prioriza más y que han mejorado un poco más en el triple aunque siguen siendo malos.

Finalmente vamos a ver qué nos dice la tercera componente:

```
L_hist[,3]
```

```

##          Altura      total_partidos      total_minutos
##      0.046140956      -0.074355916      -0.040755031
##      total_titular      total_puntos      max_puntos
##      -0.049698341      -0.005031169      0.459931042
##      total_triple_anotados      total_triple_intentados      porcentaje_triple
##      0.053941680      0.045510170      0.377199770
##      total_dos_anotados      total_dos_intentados      porcentaje_dos
##      -0.021403182      -0.021695075      0.432559772
##      total_tl_anotados      total_tl_intentados      porcentaje_tl
##      -0.027274370      -0.041378920      0.516419365
##      total_rbo          total_rbd          total_rb
##      -0.077719120      -0.054213924      -0.062601780
##      total_asis          total_bal_per      total_bal_rec
##      -0.044005928      -0.041347446      -0.050838768
##      total_tapones_real      total_tapones_rec      total_mates
##      -0.055357053      -0.054325650      -0.056184992
##      total_faltas_real      total_faltas_rec      valoracion_pro
##      -0.066755529      -0.065411417      0.353432933
##
```

```
##          total_ganados          total_perdidos
##          -0.081556750          -0.060035477
```

En la tercera componente nos indica los jugadores que tienen muy buenos porcentajes pero que en total no tienen una gran cantidad de puntos, ni asistencias, ni rebotes, ni partidos... Lo que quiere indicar que son jugadores muy buenos que han estado poco tiempo en la liga.

Vamos a ver a los 10 primeros:

```
acb_jugador_hist[order(S_hist[,3], decreasing = TRUE)[1:10],]
```

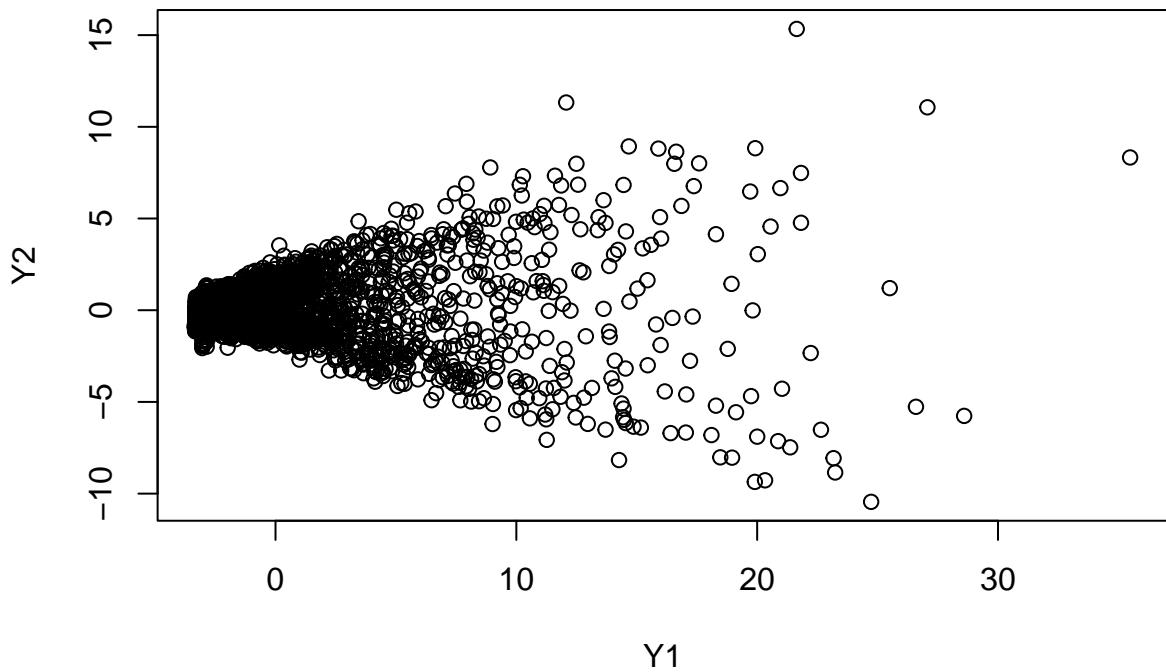
```
## # A tibble: 10 x 34
## # Groups:   Nombre, Posición, Altura, Lugar.de.nacimiento, Fecha.de.nacimiento
## #   [10]
##   Nombre  Posición Altura Lugar.de.nacimiento Fecha.de.nacimiento Nacionalidad
##   <chr>    <chr>    <dbl> <chr>           <chr>           <chr>
## 1 "Oscar ~ Alero" 2.04 Natal (Río Grande ~ 16/02/1958 (65 año~ Brasil
## 2 "Dale S~ Alero" 2.04 Annapolis, Marylan~ 26/08/1958 (64 año~ EE.UU.
## 3 "Darrel~ Base"  1.82 Gastonia (Carolina~ 22/06/1968 (54 año~ EE.UU.
## 4 "Kevin ~ Base"  1.91 Bloomington (India~ 17/07/1967 (55 año~ EE.UU.
## 5 "Charli~ Escolta" 1.91 , EE.UU.           12/03/1979 (44 año~ EE.UU.
## 6 "Franki~ Base"  1.85 Baxley (Georgia), ~ 06/06/1972 (50 año~ EE.UU.
## 7 "Norris~ Pívot"  2.04 Jacksonville (Flor~ 27/09/1961 (61 año~ EE.UU.
## 8 "Alfons~ Alero"  2.01 Birmingham (Alabam~ 03/01/1964 (59 año~ EE.UU.
## 9 "Gerald~ Pívot"  2.05 Nanaimo, Canadá 12/10/1960 (62 año~ Canadá
## 10 "Arvyda~ Escolta" 1.93 Klaipeda, Lituania 19/01/1980 (43 año~ Lituania
## # i 28 more variables: total_partidos <int>, total_minutos <dbl>,
## #   total_titular <int>, total_puntos <int>, max_puntos <int>,
## #   total_triple_anotados <int>, total_triple_intentados <int>,
## #   porcentaje_triple <dbl>, total_dos_anotados <int>,
## #   total_dos_intentados <int>, porcentaje_dos <dbl>, total_tl_anotados <int>,
## #   total_tl_intentados <int>, porcentaje_tl <dbl>, total_rbo <int>,
## #   total_rbd <int>, total_rb <int>, total_asis <int>, total_bal_per <int>, ...
```

Lo primero que podemos notar es que la mayoría de los jugadores son de estados unidos. Después podemos ver que ninguno de ellos superan los 100 partidos. Pero lo que sí tienen son buenos porcentajes y buena valoración promedio por temporada.

Nótese que el primer jugador que nos aparece es Oscar Schmidt, que era hasta hace poco el jugador con más anotaciones en la historia del baloncesto (49.737 puntos), siendo adelantado este 3 de abril de 2024 por Lebrón James. Vemos que a pesar de no haber pasado mucho tiempo en la ACB (71 partidos) aparece en esta lista por sus grandes porcentajes y valoración, además de una alta puntuación en el máximo de puntos.

Podemos representar gráficamente los scores de las dos primeras componentes principales sin estandarizar de la siguiente manera:

```
plot(S_hist[,1],S_hist[,2],xlab='Y1',ylab='Y2')
```

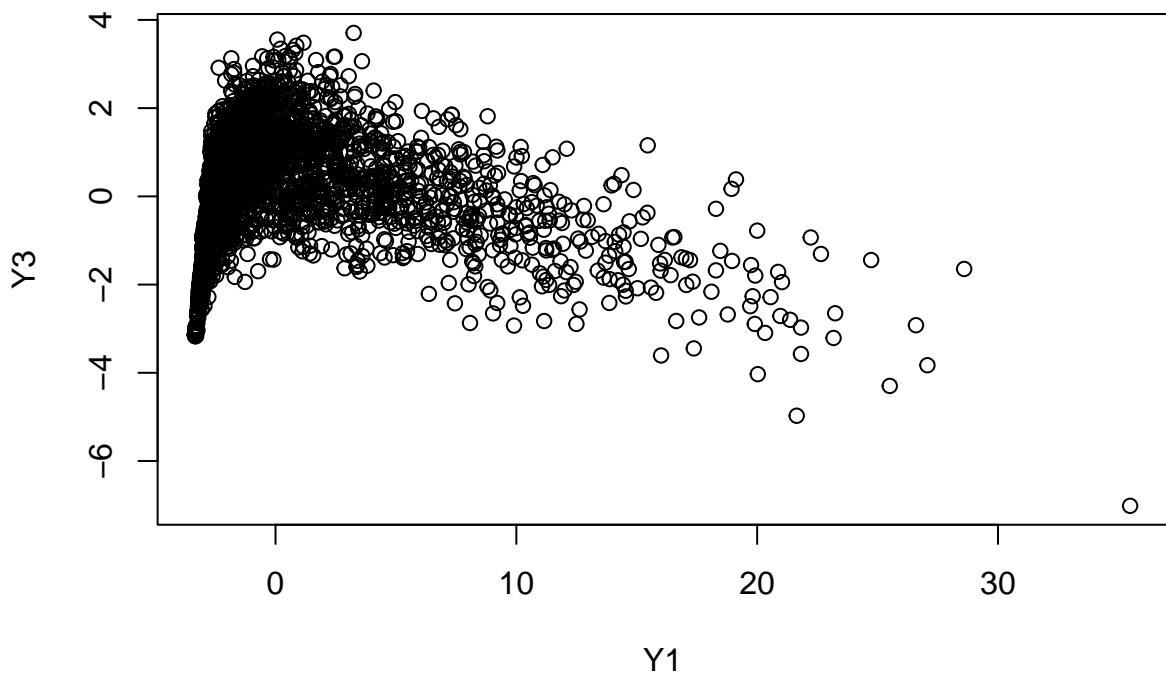


Notamos que cuando los jugadores no han jugado muchos partidos y no tienen muchas estadísticas (Y1), no pueden ser ni “buenos” ni “malos” pívots en términos de estadísticas (Y2) pues no han podido jugar suficientes minutos.

En cambio, conforme nos desplazamos para la derecha podemos ver que empiezan a aparecer “buenos” y “malos” pívots, ya que tienen suficientes estadísticas para ser calificados.

Ahora veamos lo que sucede con Y1 e Y3:

```
plot(S_hist[,1],S_hist[,3],xlab='Y1',ylab='Y3')
```



Es curioso, la gráfica parece indicar que los jugadores que pasan menos tiempo en la liga son los que mejores porcentaje tienen. Aunque esto es engañoso, porque las cargas de Y3, aunque no con mucha intensidad, perjudican a los jugadores con buenas estadísticas, por lo que a la larga tienen su efecto conforme incrementamos en el eje de Y1.

También podemos ver cuales son las variables mejor representadas de cada componente con las saturaciones:

```

SAT_hist<-cor(datos_pca_hist,S_hist)

cat("La variable mejor representada en Y1 es",
  rownames(SAT_hist)[which.max(SAT_hist[,1])], "\nManteniendo un", max(SAT_hist[,1])^2,

## La variable mejor representada en Y1 es total_minutos
## Manteniendo un 0.9584507 de su informacion

cat("La variable mejor representada en Y2 es", rownames(SAT_hist)[which.max(SAT_hist[,2])]

## La variable mejor representada en Y2 es Altura
## Manteniendo un 0.395099 de su informacion

cat("La variable mejor representada en Y3 es", rownames(SAT_hist)[which.max(SAT_hist[,3])]

## La variable mejor representada en Y3 es porcentaje_tl
## Manteniendo un 0.5618713 de su informacion

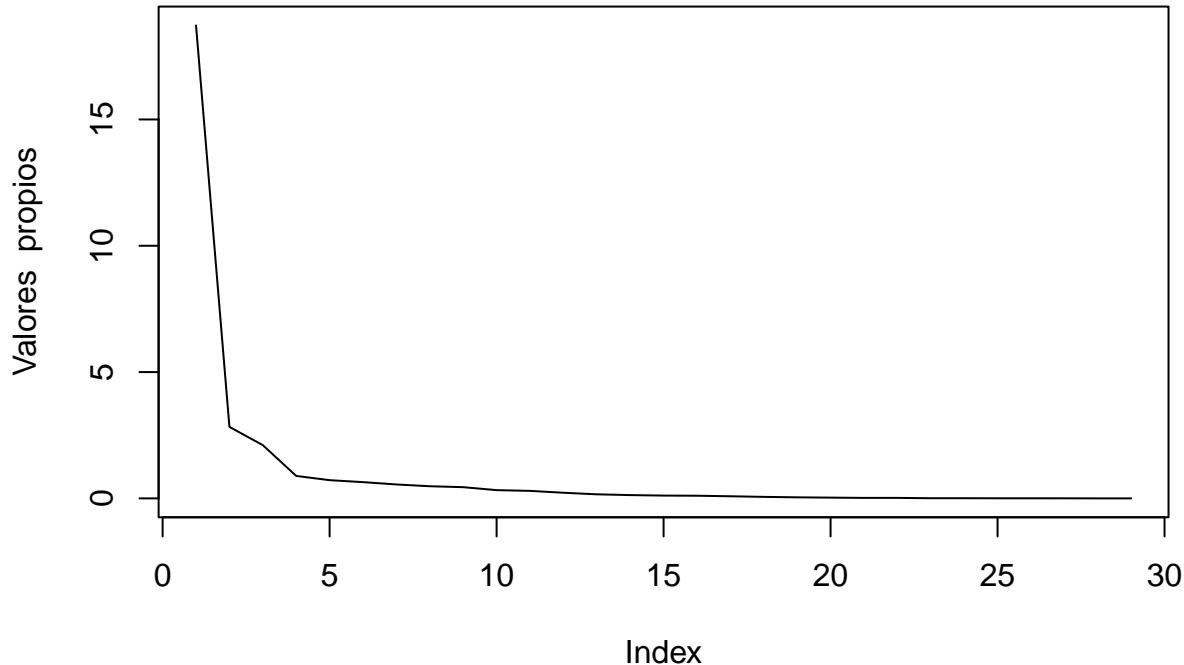
```

Vemos que todas ellas tienen sentido, pues Y1 nos indica los jugadores con estadísticas altas (y por tanto muchos minutos jugados), Y2 nos indica los jugadores que juegan de pivot (por lo que harán

muchos tapones) e Y3 nos dice los jugadores que son buenos pero no han estado mucho tiempo (buenos porcentajes).

Para terminar con esta sección vamos a pintar un gráfico de sedimentación, para emplear la regla del codo para escoger un número óptimo de componentes para algún análisis posterior:

```
plot(eigen(cor(datos_pca_hist))$values,type='l',ylab='Valores propios')
```



Podemos ver que el codo (los sedimentos) se encuentra en  $j = 4$ , por lo que cogeríamos las 3 primeras componentes.

### 3.3 Análisis Discriminante

En el análisis discriminante vamos a utilizar clasificar a los jugadores en la posición en la que deberían jugar según sus características/estadísticas.

Para ello, primero vamos a crear una columna nueva llamada ‘num\_posicion’ para poder realizar los cálculos:

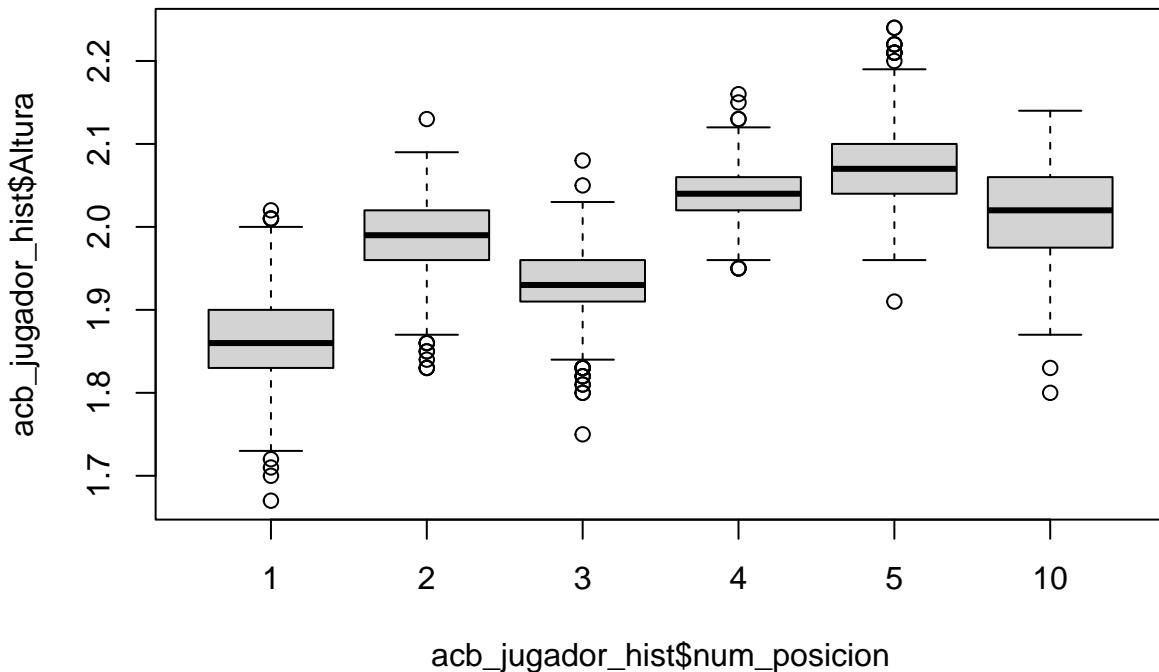
```
mapeo_posiciones <- c(  
  'Base' = 1,  
  'Alero' = 2,  
  'Escolta' = 3,  
  'Ala-pívot' = 4,  
  'Pívot' = 5,  
  'Desconocido' = 10  
)  
  
# Crear una nueva columna 'Position_Num' mapeando los valores
```

```
acb_jugador_hist$num_posicion <- mapeo_posiciones[acb_jugador_hist$Posición]

#Si no hay valor, es desconocido, por lo que le asignamos el 10
acb_jugador_hist$num_posicion[is.na(acb_jugador_hist$num_posicion)] <- 10
```

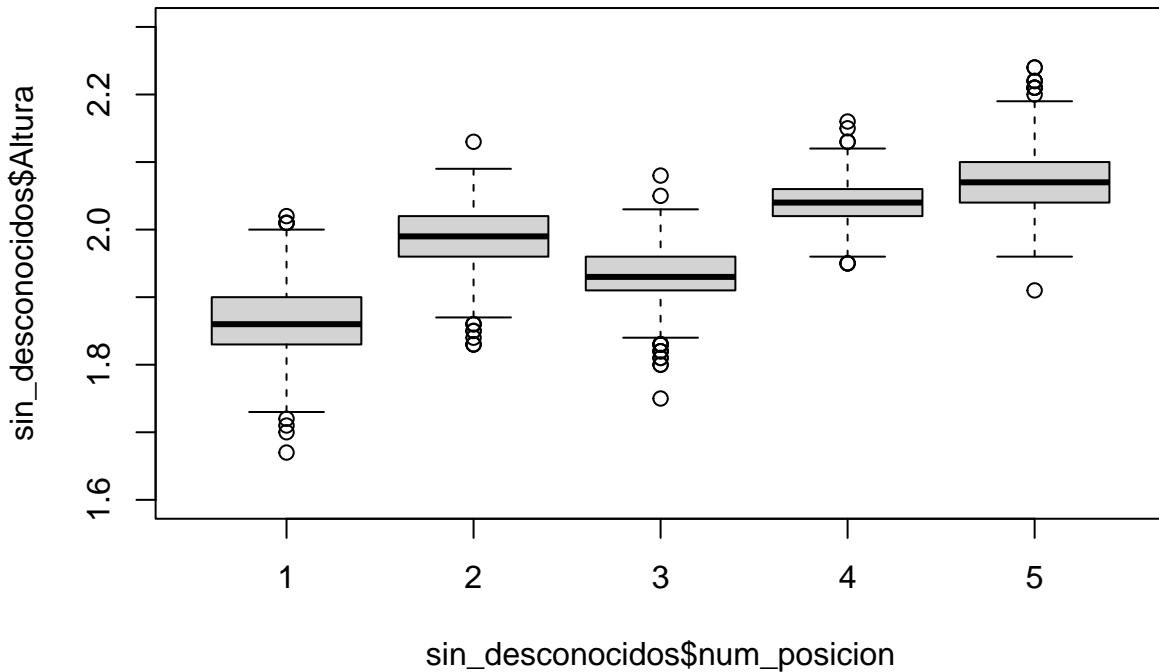
Ahora vamos a ver cuál de las variables del dataset es la que mejor separa las clases:

```
boxplot(acb_jugador_hist$Altura~acb_jugador_hist$num_posicion)
```



Podemos ver que nos ha graficado la clase con los jugadores de clase desconocida (ya que no la hemos quitado), así que vamos a volver a hacer el gráfico otra vez, pero sin los jugadores con posición desconocida:

```
sin_desconocidos <- acb_jugador_hist[acb_jugador_hist$num_posicion != 10, ]
boxplot(sin_desconocidos$Altura~sin_desconocidos$num_posicion, ylim=c(1.6,2.3))
```



Vemos que esta variable más o menos consigue diferenciar las clases, aunque ala-pívot y pívot están algo mezcladas.

Primero vamos a ver si las matrices de covarianzas de cada clase son parecidas o no, para así realizar un LDA o un QDA:

```
base<-sin_desconocidos[sin_desconocidos$num_posicion==1,c(3,7:34)]
pivot<-sin_desconocidos[sin_desconocidos$num_posicion==5,c(3,7:34)]

cov(base[1:5])
```

```
##                               Altura total_partidos total_minutos total_titular
## Altura          0.00279390 -2.472898e-01      -8.21518 -3.926489e-02
## total_partidos -0.24728977  1.567264e+04   360640.57893  7.450769e+03
## total_minutos   -8.21518042  3.606406e+05  8756147.65965  1.791448e+05
## total_titular   -0.03926489  7.450769e+03  179144.78888  4.369153e+03
## total_puntos   -2.31438275  1.157766e+05 2837930.48497  5.985079e+04
##                               total_puntos
## Altura          -2.314383e+00
## total_partidos  1.157766e+05
## total_minutos   2.837930e+06
## total_titular   5.985079e+04
## total_puntos   9.797025e+05
```

```
cov(pivot[1:5])
```

```
##                               Altura total_partidos total_minutos total_titular
## Altura          0.001928387  1.820339e-01 -1.294196e+00  2.236816e-01
```

```

## total_partidos 0.182033931 9.350554e+03 2.016739e+05 4.249358e+03
## total_minutos -1.294196471 2.016739e+05 5.070929e+06 1.058702e+05
## total_titular 0.223681595 4.249358e+03 1.058702e+05 2.967586e+03
## total_puntos -0.261229676 8.157124e+04 2.184570e+06 4.559323e+04
## total_puntos
## Altura -2.612297e-01
## total_partidos 8.157124e+04
## total_minutos 2.184570e+06
## total_titular 4.559323e+04
## total_puntos 1.004144e+06

```

Hemos visto solo las 5 primeras variables, porque para verlas todas es muy complicado.

Ya con estas se puede apreciar que las matrices no son iguales por lo que hacer LDA no nos daría una buena solución. Así que pasamos a hacer el QDA.

Pero antes de hacerlo tenemos que comprobar que todas las variables sean normales:

```

library(mvnormtest)

#base no devuelve valor porque la matriz es singular (no la puede resolver)
mshapiro.test(t(pivot))

##
## Shapiro-Wilk normality test
##
## data: Z
## W = 0.42679, p-value < 2.2e-16

```

Con la primera variable que analizamos ya vemos que esta no es normal, por lo que tampoco podríamos hacer el QDA.

Aún así vamos a realizar el LDA y el QDA y ver qué obtenemos:

\*Inciso: Ya he intentado hacer el LDA con las variables, pero como obtenemos un warning de que las variables están altamente correlacionadas entre sí vamos a emplear las 7 primeras componentes principales.

Para ello haremos lo siguiente:

```

#Guardamos el vector con las etiquetas de los grupos
num_posicion <- sin_desconocidos$num_posicion
#Construimos el PCA, pero ahora sin los desconocidos
PCA_sindesc <- princomp(sin_desconocidos[,c(3,7:34)],cor=TRUE)
S_sindesc <- PCA_sindesc$scores
#Construimos un tibble con las componentes principales y una columna para los grupos
comp_princ <- as_tibble(S_sindesc[, 1:7])
comp_princ$num_posicion <- num_posicion

comp_princ

## # A tibble: 3,370 x 8
##       Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7 num_posicion
##       <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>       <dbl>
## 1      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 2      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 3      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 4      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 5      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 6      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 7      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 8      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 9      0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 10     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 11     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 12     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 13     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 14     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 15     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 16     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 17     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 18     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 19     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 20     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 21     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 22     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 23     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 24     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 25     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 26     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 27     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 28     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 29     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 30     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 31     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 32     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 33     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 34     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 35     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 36     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 37     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 38     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 39     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 40     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 41     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 42     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 43     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 44     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 45     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 46     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 47     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 48     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 49     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 50     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 51     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 52     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 53     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 54     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 55     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 56     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 57     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 58     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 59     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 60     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 61     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 62     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 63     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 64     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 65     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 66     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 67     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 68     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 69     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 70     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 71     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 72     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 73     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 74     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 75     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 76     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 77     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 78     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 79     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 80     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 81     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 82     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 83     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 84     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 85     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 86     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 87     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 88     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 89     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 90     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 91     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 92     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 93     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 94     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 95     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 96     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 97     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 98     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 99     0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 100    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 101    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 102    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 103    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 104    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 105    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 106    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 107    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 108    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 109    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 110    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 111    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 112    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 113    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 114    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 115    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 116    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 117    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 118    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 119    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 120    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 121    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 122    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 123    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 124    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 125    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 126    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 127    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 128    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 129    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 130    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 131    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 132    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 133    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 134    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 135    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 136    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 137    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 138    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 139    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 140    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 141    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 142    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 143    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 144    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 145    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 146    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 147    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 148    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 149    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 150    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 151    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 152    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 153    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 154    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 155    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 156    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 157    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 158    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 159    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 160    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 161    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 162    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 163    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 164    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 165    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 166    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 167    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 168    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 169    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.100
## 170    0.182  -0.294  0.224 -0.261  -0.004  0.081  0.218      0.
```

```

##  1 -1.17 -1.38  2.06 -1.62 -0.272 -0.407 -0.844      1
##  2  6.95  4.14  1.13 -2.40  0.453  1.04 -2.94      2
##  3 -2.54  0.101  0.276  0.216  0.934  0.259  0.668      5
##  4  8.43  5.13 -1.22 -0.743  0.483  0.378 -1.76      5
##  5  1.32 -0.577  1.76 -0.804 -0.552  0.512 -0.559      1
##  6 -0.869 -0.560  1.60 -1.13 -0.597 -0.297 -0.879      1
##  7  5.61 -2.22  0.892  1.10  1.20  1.19  0.467      4
##  8 -2.29  0.279  1.28  0.581  0.917 -0.0855  0.311      4
##  9 -2.54  0.836  0.575  1.84 -0.161 -0.190 -0.183      5
## 10  1.70  1.90  3.11 -0.427 -2.01  0.385  1.14      5
## # i 3,360 more rows

library(MASS)

## 
## Attaching package: 'MASS'

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

LDA_sindesc<-lda(comp_princ[,1:7],comp_princ$num_posicion,prior=c(1/5,1/5,1/5,1/5,1/5), CV=TRUE)

t <- table(comp_princ$num_posicion,LDA_sindesc$class)
accuracy <- (t[1,1] + t[2,2] + t[3,3] + t[4,4] + t[5,5])/ sum(t)
t

## 
##      1   2   3   4   5
##  1 503  23 156   1   0
##  2  19 397 179 169  74
##  3  78  93 316   8   1
##  4   0  77   4 272  98
##  5   0  53   3 201 645

cat("El accuracy es de:", accuracy)

## El accuracy es de: 0.6329377

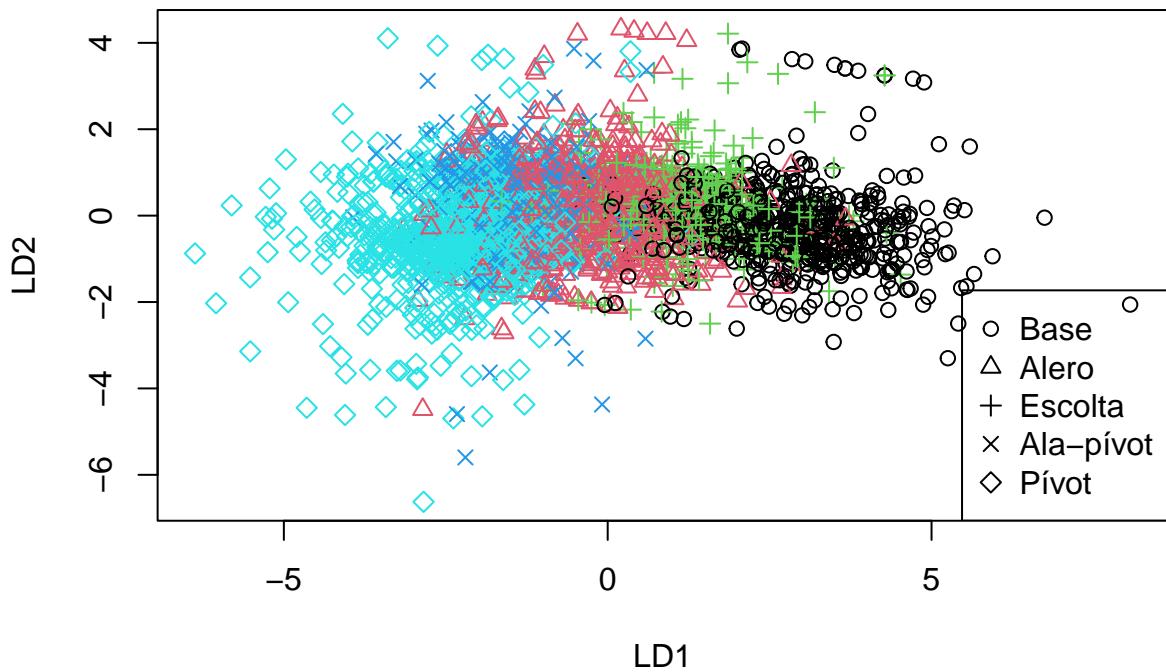
```

Podemos ver que hace una clasificación algo mala, de más o menos el 63%, aunque algo sí que predice. Lo podemos ver gráficamente con:

```

#Volvemos a poner LDA_sindesc pero ahora sin CV=TRUE para que funcione
LDA_sindesc<-lda(comp_princ[,1:7],comp_princ$num_posicion,prior=c(1/5,1/5,1/5,1/5,1/5))
predict(LDA_sindesc,comp_princ[,1:7])>P
plot(P$x, pch = as.integer(comp_princ$num_posicion), col = comp_princ$num_posicion)
legend('bottomright',legend=c('Base','Alero','Escolta', 'Ala-pivot','Pivot'),pch=1:5)

```



Vemos que la variable LD1 separa más o menos los datos, mientras que LD2 no parece hacer mucho. Se comprobamos que todos grupos se superponen unos con otros, aunque sí que se puede ver cierta separabilidad. También vemos que la clase que más parece diferenciarse es la de base.

Obtenemos el accuracy del QDA:

```
QDA_sindesc<-qda(comp_princ[,1:7],comp_princ$num_posicion,
                     prior=c(1/5,1/5,1/5,1/5,1/5), CV = TRUE)

t2 <- table(comp_princ$num_posicion,QDA_sindesc$class)
accuracy2 <- (t2[1,1] + t2[2,2] + t2[3,3] + t2[4,4] + t2[5,5])/ sum(t2)
t2
```

```
##
##      1   2   3   4   5
## 1 418 13 251 1 0
## 2 21 183 365 215 54
## 3 56 26 399 14 1
## 4 1 22 41 339 48
## 5 2 19 14 357 510
```

```
cat("El accuracy es de:", accuracy2)
```

```
## El accuracy es de: 0.5486647
```

Vemos que el QDA es todavía peor, por lo que no merece la pena analizarlo más en profundidad.

## 3.4 Análisis cluster

### 3.4.1 Cluster con K-means

Vamos a intentar agrupar los datos en 5 clusters para ver si consigue encontrar las posiciones en las que juegan los jugadores:

```
#Estandarizamos los datos
ds <- as.data.frame(scale(sin_desconocidos[,c(3,7:34)]))
#Inicializamos el número de centros a 5
CA1<-kmeans(ds,centers=5,nstar=10)
```

Si queremos ver el número de coincidencias entre la clasificación que ha hecho y la que nosotros ya teníamos podemos hacer:

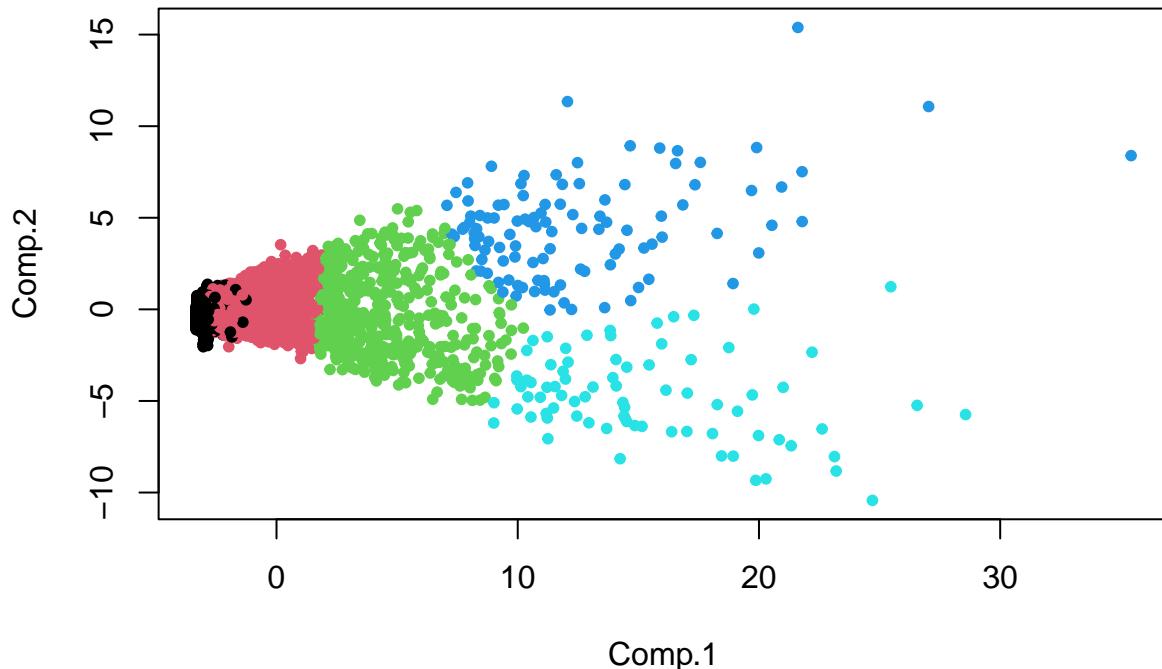
```
Y<-CA1$cluster
aciertos <- sum(Y==sin_desconocidos$num_posicion)
cat("Coinciden en un", 100*aciertos/nrow(sin_desconocidos), "%")

## Coinciden en un 21.18694 %
```

Así podemos ver que claramente no nos ha hecho el agrupamiento por las posiciones, ya que solamente acierta en el 16.84% de las veces.

Vamos a representar los grupos:

```
plot(comp_princ[,1:2],col=CA1$cluster,pch=20)
```



Vemos que parte la “flecha” primero en tres trozos, siendo un grupo la punta de esta y finalmente parte la “cola” en otros dos grupos.

```
CA1$withinss
```

```
## [1] 4802.686 10179.807 8659.707 6186.681 4158.187
```

Estas son las sumas de las distancias al cuadrado de todos los puntos a los centroides de cada grupo. Vemos que el tercer grupo (el azul celeste) tiene un centroide que es el que mejor actúa como centro de gravedad para los puntos.

Podemos calcular cuánto hemos reducido la variabilidad de los datos al hacer esta agrupación:

```
1-CA1$tot.withinss/CA1$totss
```

```
## [1] 0.6521318
```

Hemos reducido la variabilidad en un 65.21%.

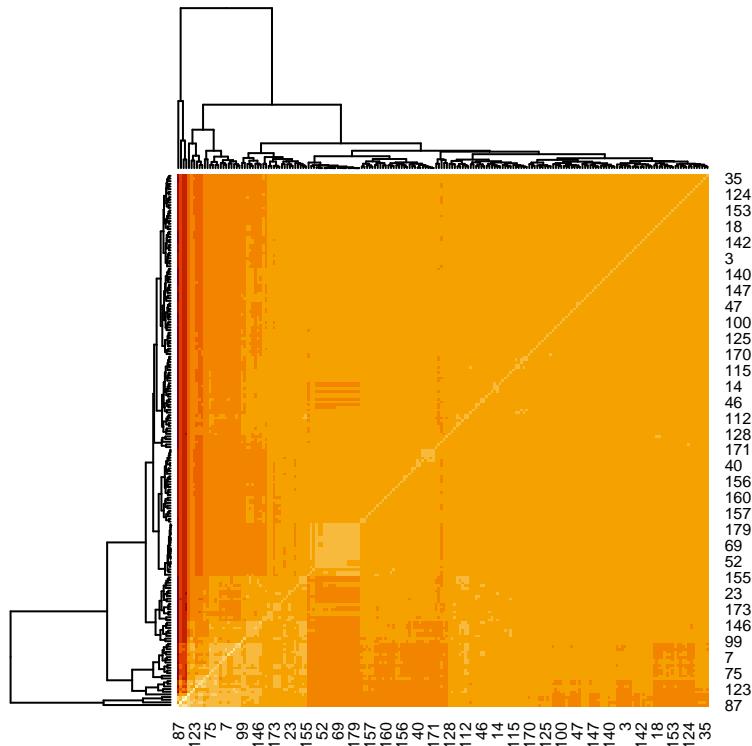
No obtenemos una muy buena agrupación de los datos. Esto es porque hemos fijado el número de grupos, pero con el agrupamiento jerárquico podemos encontrar el óptimo.

### 3.4.2 Cluster jerárquico

Volvemos a calcular un agrupamiento de 5 clusters, pero ahora con este algoritmo.

Antes de eso, vamos a hacer un mapa de calor para ver la “cercanía” de los datos.

```
D <- dist(ds[1:200,], method = 'euclidean')
M<-as.matrix(D)
heatmap(M)
```



Hacemos el mapa de calor de los primeros 200 datos, porque si lo hacemos de todo va a tardar demasiado.

Podemos ver en la zona de la izquierda que hay unos cuantos datos que se encuentran muy alejados y a medida que nos movemos hacia la derecha encontramos datos que están muy juntos.

Volvemos a hacer 5 clusters pero ahora con el método jerarquizado:

```
D <- dist(ds, method = 'euclidean')
M<-as.matrix(D)
CA2<-hclust(D,method='complete')
grupos<-cutree(CA2, k=5)
```

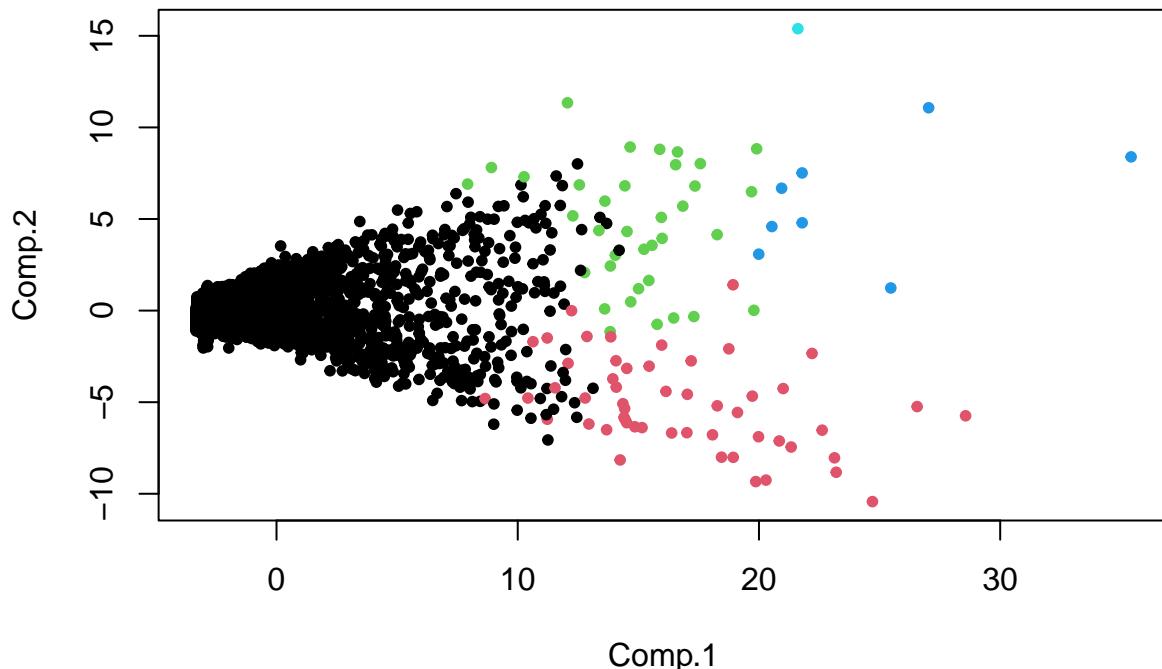
Vamos a comprobar si con este algoritmo obtenemos resultados parecidos al anterior:

```
aciertos <- sum(grupos==Y)
cat("Coinciden en un", 100*aciertos/length(Y), "%")

## Coinciden en un 27.21068 %
```

Se puede ver que ha formado grupos completamente diferente, vamos a verlo gráficamente:

```
plot(comp_princ[,1:2], col=grupos, pch=20)
```



Obtenemos ahora un grupo enorme que constituye la punta de la “flecha” y luego unos grupillos más pequeños de unos pocos individuos. No vamos a ver el accuracy con las etiquetas de posición porque ya gráficamente se puede ver que no va a funcionar.

Se puede apreciar claramente cómo funciona cada algoritmo: el cluster con K-means ha agrupado los datos intentando repartir equitativamente los datos para cada centroide, mientras que el cluster

jerárquico ha preferido hacer grupos donde tenemos una gran densidad de puntos. Esto es debido a que el cluster de K-means parte de unos centroides inicializados aleatoriamente y luego los va ajustando según la media, mientras que el jerárquico va agrupando los datos más cercanos, consiguiendo así formar grupos más compactos.

### 3.4.3 Número óptimo de clusters

Para intentar deducir el número óptimo de grupos a tomar podemos hacer lo siguiente:

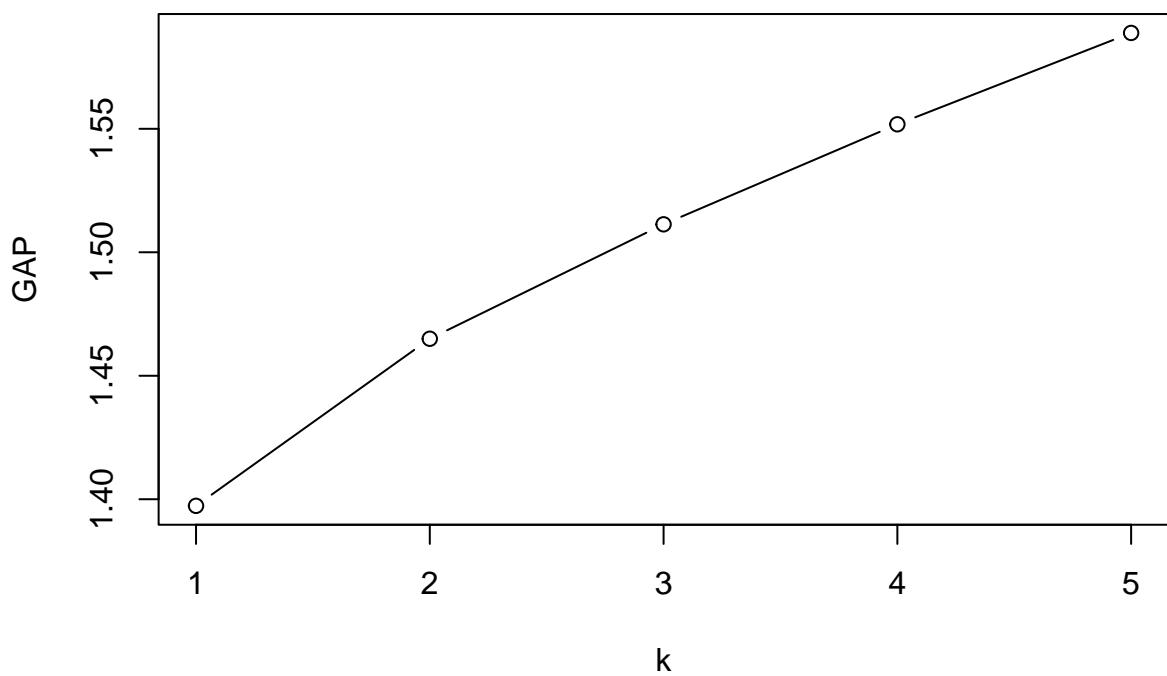
```
#library(NbClust)
#NbClust(ds,method='complete',index='all')$Best.nc
#Este no funciona porque la matriz TSS es indefinida.

library(cluster)
gap<-clusGap(ds,FUNcluster=kmeans,K.max=5) #Recomiendo no volver a ejecutar
                                                #esta celda porque tarda mucho
gap

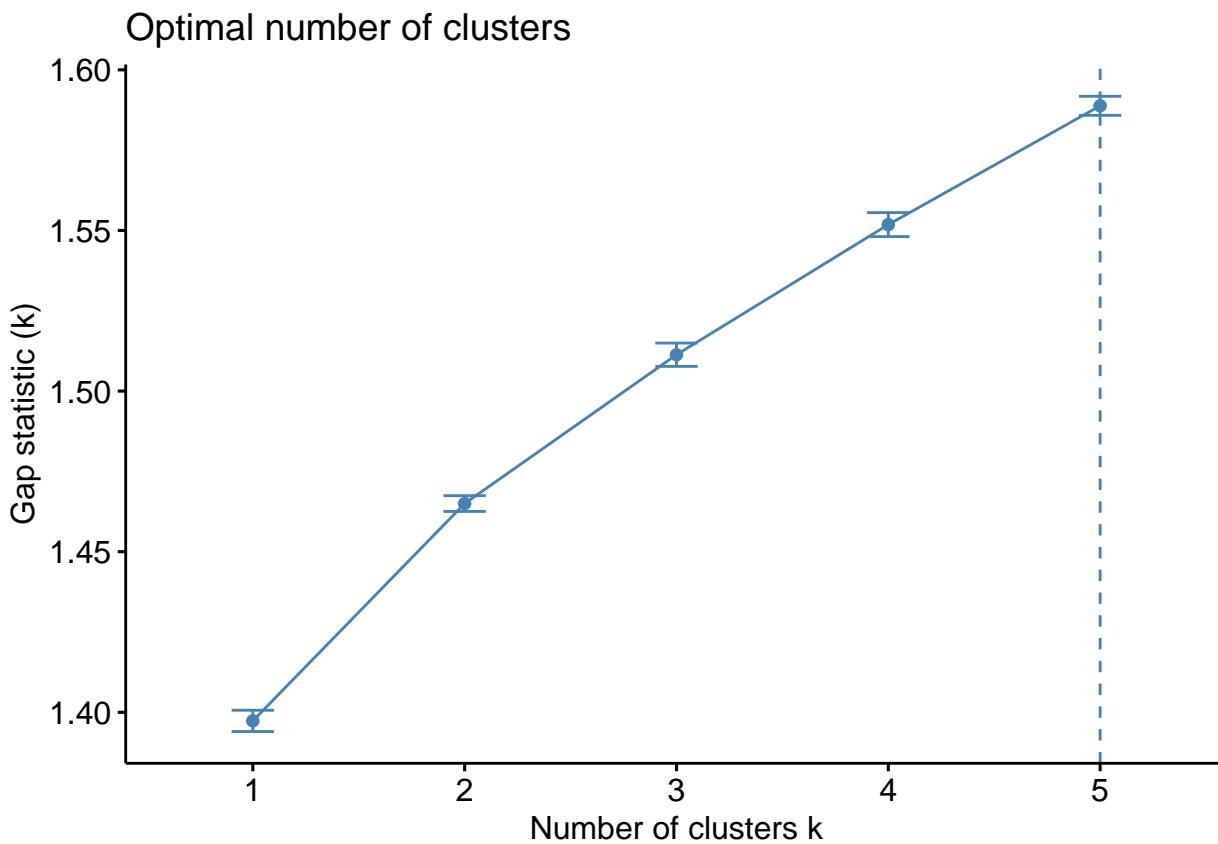
## Clustering Gap statistic ["clusGap"] from call:
## clusGap(x = ds, FUNcluster = kmeans, K.max = 5)
## B=100 simulated reference sets, k = 1..5; spaceH0="scaledPCA"
## --> Number of clusters (method 'firstSEmax', SE.factor=1): 5
##           logW   E.logW      gap      SE.sim
## [1,] 8.532235 9.929566 1.397331 0.003317519
## [2,] 8.302762 9.767748 1.464986 0.002453817
## [3,] 8.204332 9.715656 1.511324 0.003626839
## [4,] 8.117348 9.669177 1.551829 0.003735363
## [5,] 8.052829 9.641644 1.588814 0.002969627
```

El valor máximo del GAP es 1.584221 y se obtiene para  $k = 5$ . Para representarlo podemos hacer:

```
plot(gap$Tab[,3],type='b',xlab='k',ylab='GAP')
library(ggplot2)
```



```
library(factoextra)  
## Warning: package 'factoextra' was built under R version 4.3.3  
fviz_gap_stat(gap)
```



El estadístico GAP compara el total de las distancias intra-cluster para diferentes valores de k. Nótese que la técnica que hemos utilizado para calcular los grupos es k-means, por lo que si utilizamos otra podríamos obtener resultados diferentes. Además parece que el estadístico GAP tiene una tendencia a seguir creciendo, por lo que convendría aumentar el k máximo para poder encontrar el máximo. Pero no lo vamos a hacer porque tarda demasiado en calcular los resultados.

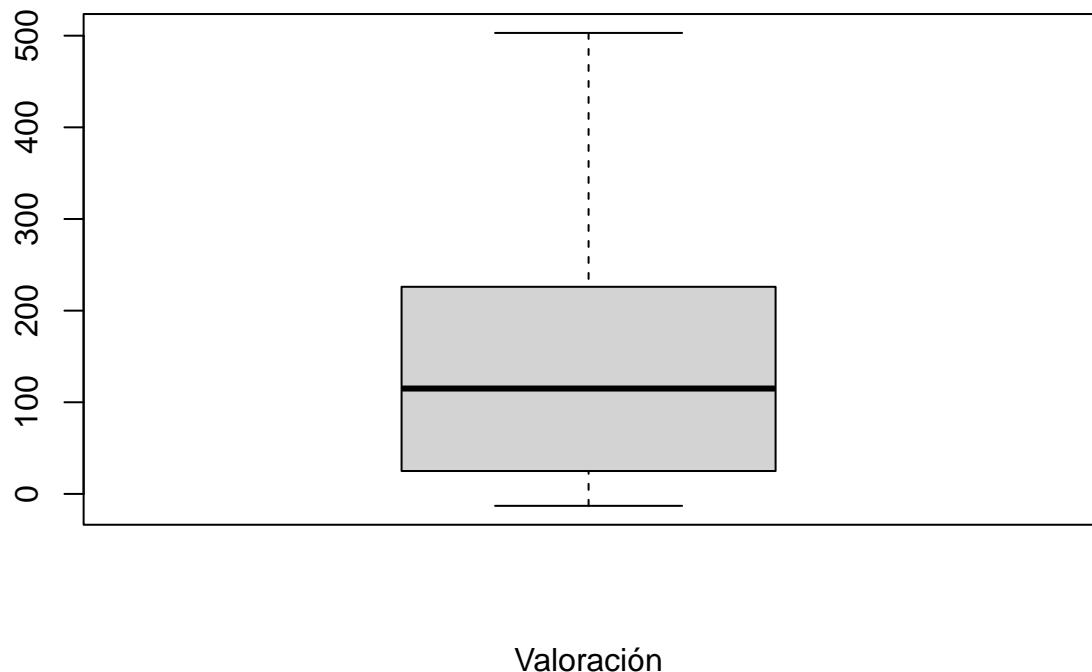
### 3.5 Regresión

#### 3.5.1 Regresión lineal múltiple

Vamos a intentar predecir la fórmula de la valoración de un jugador en la temporada 2022-2023 según sus estadísticas.

Para ello, primero de todo vamos a estudiar un poco cómo se comporta la variable a predecir:

```
boxplot(acb_2022_2023$Valoración, xlab = "Valoración")
```



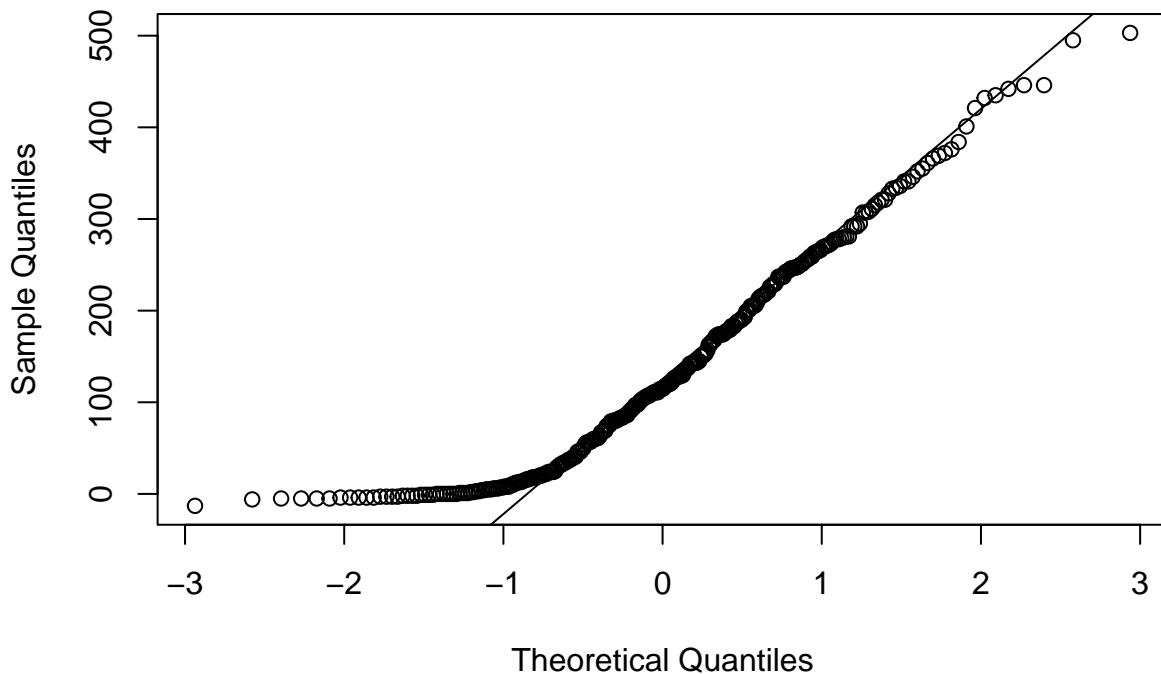
No parece tener ningún atípico, lo cuál es buena señal para hacer buenas predicciones. Continuamos el análisis estudiando la normalidad:

```
shapiro.test(acb_2022_2023$Valoración)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: acb_2022_2023$Valoración  
## W = 0.92749, p-value = 5.903e-11
```

```
qqnorm(acb_2022_2023$Valoración)  
qqline(acb_2022_2023$Valoración)
```

## Normal Q-Q Plot



Claramente no es normal, ya que obtenemos un p-valor de 5.903e-11 y se puede ver en los gráficos que el extremo izquierdo no se ajusta nada.

Primero vamos a crear el modelo con todas las variables, para luego disminuir el número mediante los métodos de selección de regresores backward, forward y stepwise.

```
modelo_completo <- lm(acb_2022_2023$Valoración ~ ., data = acb_2022_2023[, c(4, 9:36)])
```

Ahora aplicamos los métodos:

```
modelo_cte <- lm(acb_2022_2023$Valoración ~ 1, data = acb_2022_2023[, c(4, 9:36)])
modelo_backward <- step(modelo_completo, direction = "backward")
modelo_stepwise <- step(modelo_cte, direction = "both",
                         scope = formula(modelo_completo))
modelo_forward <- step(modelo_cte, direction = "forward",
                         scope = formula(modelo_completo))
```

Si nos ponemos a ver cada uno por separado vemos que el más simple es el forward y el stepwise, que obtienen ambos el siguiente modelo:

```
modelo_forward$coefficients
```

```
##                               (Intercept)          Total.puntos
##                               -3.663488e-13 1.333333e+00
##                               Total.rebotes      Total.asistencias
##                               1.000000e+00 1.000000e+00
## Número.de.faltas.realizadas  Tiros.de.3.puntos.intentados
```

```

## -1.000000e+00 -1.000000e+00
## Tiros.de.2.puntos.intentados Número.de.balones.perdidos
## -1.000000e+00 -1.000000e+00
## Número.de.faltas.recibidas Tiros.libres.intentados
## 1.000000e+00 -1.000000e+00
## Número.de.tapones.realizados Número.de.balones.recuperados
## 1.000000e+00 1.000000e+00
## Numero.de.tapones.recibidos Tiros.libres.anotados
## -1.000000e+00 6.666667e-01
## Tiros.de.2.puntos.anotados Partidos.ganados
## 3.333333e-01 -9.413083e-15

```

Aquí es donde me di cuenta del error sobre que las columnas “Número.de.balones.perdidos” y “Número.de.balones.recuperados” estaban intercambiadas. Pues por balones recuperados le quitaban puntuación al jugador, mientras que por balones perdidos le añadían puntuación (lo cuál no tiene sentido).

Vemos que tanto el intercepto como el número de partidos es prácticamente cero. Aunque ocurre una cosa un tanto extraña, pues mientras que vemos que todas las variables tienen un coeficiente de más y menos uno. En las variables “Total.puntos”, “Tiros.libres.anotados” y “Tiros.de.2.puntos.anotados” tenemos coeficientes de  $4/3$ ,  $2/3$  y  $1/3$  respectivamente.

Podemos buscar la fórmula en internet para ver que es:

Valoración=(Puntos+Rebotes+Asistencias+Robos+Tapones)-(Tiros\_de\_Campo\_Fallados+Tiros\_Libres\_Fallados)

Podemos hacer las siguientes cuentas para llegar a nuestra fórmula y ver que efectivamente, son la misma:

- (1)  $Tiros\_de\_Campo\_Fallados = Tiros.de.3.puntos.intentados + Tiros.de.2.puntos.intentados - Tiros.de.3.puntos.anotados - Tiros.de.2.puntos.anotados$
- (2)  $Total.puntos = 3*Tiros.de.3.puntos.anotados + 2*Tiros.de.2.puntos.anotados + Tiros.libres.anotados$   
 $\Leftrightarrow 1/3*Total.puntos = Tiros.de.3.puntos.anotados + 2/3*Tiros.de.2.puntos.anotados + 1/3*Tiros.libres.anotados \Leftrightarrow Tiros.de.3.puntos.anotados = 1/3*Total.puntos - (2/3*Tiros.de.2.puntos.anotados + 1/3*Tiros.libres.anotados)$

Si ahora sustituimos en (1) tenemos que:

- (1)  $Tiros\_de\_Campo\_Fallados = Tiros.de.3.puntos.intentados + Tiros.de.2.puntos.intentados - (1/3*Total.puntos - (2/3*Tiros.de.2.puntos.anotados + 1/3*Tiros.libres.anotados)) - Tiros.de.2.puntos.anotados \Leftrightarrow Tiros\_de\_Campo\_Fallados = -1/3*Total.puntos + Tiros.de.3.puntos.intentados + Tiros.de.2.puntos.intentados - 1/3*Tiros.de.2.puntos.anotados + 1/3*Tiros.libres.anotados$

También sabemos que:

- (3)  $Tiros.libres.intentados = Tiros.libres.anotados - Tiros\_Libres\_Fallados \Leftrightarrow Tiros\_Libres\_Fallados = Tiros.libres.intentados - Tiros.libres.anotados$

Si sustituimos “Tiros\_de\_Campo\_Fallados” y “Tiros\_Libres\_Fallados” en la fórmula de Valoración, obtendremos exactamente los coeficientes de los regresores.

Es curioso ver, que al no crear una variable llamada “Tiros.de.x.fallados” el regresor ha hecho los cálculos empleando combinaciones lineales de esta variable no creada.

Podemos ver los estadísticos del modelo creado:

```
summary(modelo_forward)
```

Vemos que obtiene la fórmula perfectamente, pues el R cuadrado es de 1. También podemos observar que todos los regresores son significativos (aunque realmente podríamos quitar tanto el intercepto como el número de partidos, ya que están muy cerca de cero).

Si intentamos predecir el valor de un nuevo individuo, lo que hará la recta de regresión será actuar como una calculadora de la valoración de un jugador en una temporada, por ejemplo:

```
nuevo <- data.frame(  
  Total.puntos = 100,  
  Total.rebotes = 20,  
  Total.asistencias = 30,  
  Número.de.faltas.realizadas = 10,  
  Tiros.de.3.puntos.intentados = 15,  
  Tiros.de.2.puntos.intentados = 50,  
  Número.de.balones.perdidos = 5,  
  Número.de.faltas.recibidas = 7,  
  Tiros.libres.intentados = 10,  
  Número.de.tapones.realizados = 2,  
  Número.de.balones.recuperados = 4,  
  Numero.de.tapones.recibidos = 6,  
  Tiros.libres.anotados = 7,  
  Tiros.de.2.puntos.anotados = 40,  
  Partidos.ganados = 20  
)  
  
resultado <- predict(modelo_forward, newdata = nuevo, interval = "prediction",  
                      level = 0.95)  
resultado  
  
##      fit      lwr      upr  
## 1 118.3333 118.3333 118.3333
```

Obtenemos una valoración de 118.3333, además, hemos puesto un intervalo de confianza al 95%, pero como tenemos una bondad de ajuste de 1, obtenemos que el límite superior y el inferior coinciden. Esto es por lo que he dicho más arriba: estamos seguros que el valor que nos vaya a salir va a ser el correcto.

Podríamos validar el modelo, pero tampoco lo vamos a hacer, ya que ya sabemos que esa es la fórmula que nos tiene que salir.

### 3.5.2 Regresión Multinomial

En esta sección vamos a tratar de hacer Regresión Multinomial como un problema de clasificación. De nuevo vamos a intentar predecir la posición en la que juegan los jugadores, ya que en casos anteriores no hemos conseguido obtener buenos resultados.

Para ello vamos a empezar pasando la variable “Posición” a tipo factor:

```
sin_desconocidos$Posición <- factor(sin_desconocidos$Posición)
```

Aunque primero, vamos a hacer una “poda” de las variables que tengan mucha correlación para tratar de reducir el número de variables. Esta reducción de variables la hacemos porque es mejor tener regresores que no estén muy correlados y además nos ayuda para que más tarde, cuando creemos el modelo, la red neuronal no tenga un número tan grande de pesos que calcular.

```
# Calcula la matriz de correlación
tmp <- cor(sin_desconocidos[, c(3, 7:34)])

# Establece la parte superior de la matriz a cero
tmp[upper.tri(tmp)] <- 0

# Establece la diagonal a cero (opcional)
diag(tmp) <- 0

# Crea un nuevo conjunto de datos sin las variables altamente correlacionadas
column_names <- colnames(tmp)[colSums(tmp > 0.95) == 0]
```

Como vamos a emplear el modelo como clasificador, vamos a partir el dataset en train y test. Para poder más tarde hacer una correcta valoración del modelo.

```
set.seed(31416)
indices_entrenamiento <- sample(1:nrow(sin_desconocidos), 0.8*nrow(sin_desconocidos))

datos_entrenamiento <- sin_desconocidos[indices_entrenamiento, c(column_names, "Posición")]
datos_test <- sin_desconocidos[-indices_entrenamiento, c(column_names, "Posición")]
```

Vamos a comenzar usando un modelo multinomial que tenga todos los predictores, para luego tratar de reducir este número de variables mediante los métodos de selección de regresores.

Tomaremos como clase de referencia la posición “Pívot”.

```
library("nnet")
datos_entrenamiento$Posición <- relevel(datos_entrenamiento$Posición, ref = "Pívot")
mymultinom <- multinom(Posición ~ ., data = datos_entrenamiento)
```

Podemos ver el modelo obtenido haciendo:

```
summary(mymultinom)
```

Podemos fijarnos en las medidas para la bondad del ajuste para ver si luego, cuando apliquemos los otros métodos, se reducen.

- Residual Deviance: 3668.151
- AIC: 3844.151

Continuamos ahora empleando los tres métodos de reducción de variables:

```
modelo_backward <- step(mymultinom, direction = "backward")
```

```
summary(modelo_backward)
```

```
## Call:  
## multinom(formula = Posición ~ Altura + total_titular + max_puntos +  
##           total_triple_intentados + porcentaje_triple + total_dos_intentados +  
##           total_tl_intentados + porcentaje_tl + total_rb + total_asis +  
##           total_bal_per + total_bal_rec + total_tapones_rec + total_mates +  
##           total_faltas_real + total_faltas_rec + total_ganados, data = datos_entrenamiento)  
##  
## Coefficients:  
## (Intercept) Altura total_titular max_puntos  
## Ala-pívot 40.22234 -20.00072 -0.0106620627 -0.039497080  
## Alero 124.69951 -61.62519 0.0185521613 0.006792734  
## Base 238.25944 -120.87067 -0.0165148883 0.060124326  
## Escolta 189.66350 -95.38344 0.0004169781 0.057150097  
##           total_triple_intentados porcentaje_triple total_dos_intentados  
## Ala-pívot 0.011186619 2.496937 0.002427665  
## Alero 0.011181431 1.367458 0.002222997  
## Base 0.001058936 2.409946 -0.006328460  
## Escolta 0.011815806 2.292358 -0.000770697  
##           total_tl_intentados porcentaje_tl total_rb total_asis  
## Ala-pívot -0.012084876 0.79161884 -0.007504498 0.007821854  
## Alero -0.002638626 -0.02058241 -0.015448886 0.004526254  
## Base -0.003082097 0.69916343 -0.019878557 0.033100041  
## Escolta -0.011684175 0.35749672 -0.020242360 0.015859688  
##           total_bal_per total_bal_rec total_tapones_rec total_mates  
## Ala-pívot -0.01002781 0.01982348 0.04221447 0.012260119  
## Alero 0.01693384 0.02565524 -0.02991422 0.003361422  
## Base 0.03131034 0.03217737 0.01904990 -0.075874606  
## Escolta 0.01586000 0.02141116 0.01418837 -0.023033694  
##           total_faltas_real total_faltas_rec total_ganados  
## Ala-pívot -0.007104013 0.015137859 0.007143119  
## Alero -0.011027545 0.005096891 0.032180300  
## Base -0.014394122 0.005823694 0.037433572  
## Escolta -0.010806272 0.015141521 0.034572177  
##  
## Std. Errors:  
## (Intercept) Altura total_titular max_puntos  
## Ala-pívot 0.03521854 0.07147075 0.008198331 0.01241508  
## Alero 0.03423314 0.07197325 0.010229662 0.01196772  
## Base 0.05242284 0.10073822 0.016236641 0.01902497  
## Escolta 0.04584349 0.09056251 0.013140006 0.01588503  
##           total_triple_intentados porcentaje_triple total_dos_intentados  
## Ala-pívot 0.001375705 0.4090684 0.0008459354  
## Alero 0.001471906 0.3862926 0.0009153415  
## Base 0.002290067 0.3243034 0.0022539324  
## Escolta 0.001702501 0.3135900 0.0015214080  
##           total_tl_intentados porcentaje_tl total_rb total_asis  
## Ala-pívot 0.003407837 0.3098535 0.001536781 0.003353027  
## Alero 0.002720222 0.2439273 0.001813564 0.004217320  
## Base 0.005325720 0.2220197 0.003439233 0.006057764
```

```

## Escolta          0.004612861    0.2035664  0.002497728 0.005388732
##               total_bal_per total_bal_rec total_tapones_rec total_mates
## Ala-pívot      0.005286568    0.005888499    0.01101568 0.003823121
## Alero          0.005738687    0.006794159    0.01376892 0.005640674
## Base           0.009559838    0.009398326    0.02572553 0.023663009
## Escolta        0.007917946    0.008568408    0.01962487 0.012069554
##               total_faltas_real total_faltas_rec total_ganados
## Ala-pívot      0.002325538    0.003515194    0.006696024
## Alero          0.002634147    0.003157475    0.007784627
## Base           0.004065708    0.005730382    0.013714475
## Escolta        0.003445295    0.005087711    0.010557682
##
## Residual Deviance: 3681.472
## AIC: 3825.472

```

En este caso reducimos en número de variables a 17, vamos a ver los siguientes. Y obtenemos las siguientes medidas:

- Residual Deviance: 3681.472
- AIC: 3825.472

Reduciendo un poco más el valor del criterio de Akaike, pero aumentando la devianza.

```

modelo_nulo <- multinom(Posición ~ 1, data = datos_entrenamiento)
modelo_forward <- step(modelo_nulo, scope = formula(mymultinom), direction = "forward")

summary(modelo_forward)

```

```

## Call:
## multinom(formula = Posición ~ Altura + total_triple_intentados +
##           total_asis + total_rb + total_bal_rec + porcentaje_triple +
##           total_tapones_rec + total_bal_per + total_dos_intentados +
##           max_puntos + total_mates + total_titular + total_tapones_real +
##           porcentaje_tl + total_perdidos, data = datos_entrenamiento)
##
## Coefficients:
## (Intercept) Altura total_triple_intentados total_asis
## Ala-pívot  41.05685 -20.39954      0.0101041769 0.009615089
## Alero      121.82365 -60.16526      0.0113572040 0.009636327
## Base       236.47287 -119.95086     0.0006943595 0.041454105
## Escolta    187.16581 -94.11756      0.0119273506 0.026129849
##           total_rb total_bal_rec porcentaje_triple total_tapones_rec
## Ala-pívot -0.005736539  0.01477367      2.611853  0.05002140
## Alero      -0.012941820  0.02058842      1.455128 -0.02318920
## Base       -0.016867113  0.02702954      2.484369  0.02907923
## Escolta   -0.017842243  0.01097645      2.388331  0.03403784
##           total_bal_per total_dos_intentados max_puntos total_mates
## Ala-pívot -0.012822848  0.0008664763 -0.049614693 0.01670842
## Alero      0.007599806  0.0014593752 -0.001567285 0.01315913
## Base       0.020789234  -0.0071857179 0.050290020 -0.06851103
## Escolta   0.005389213  -0.0031488089 0.045550362 -0.01421363

```

```

##          total_titular total_tapones_real porcentaje_tl total_perdidos
## Ala-pívot  -0.001889228      -0.006755302     0.86152785 -8.824763e-03
## Alero       0.019146107      -0.007829312    -0.01241791 -4.738605e-03
## Base        -0.015608297     -0.025579868     0.72437483 -1.602011e-02
## Escolta     0.006184814      0.009749357     0.37100457 -1.604663e-05
##
## Std. Errors:
##          (Intercept)      Altura total_triple_intentados  total_asis
## Ala-pívot  0.03525354  0.07134945      0.001335881 0.003092934
## Alero       0.03382640  0.07096091      0.001441302 0.003827367
## Base        0.05197955  0.09995480      0.002399981 0.005708713
## Escolta     0.04537781  0.08966711      0.001664207 0.005012135
##          total_rb total_bal_rec porcentaje_triple total_tapones_rec
## Ala-pívot 0.001262658   0.005365178     0.4035532 0.009034197
## Alero       0.001644336   0.006152017     0.3850716 0.012266284
## Base        0.003155170   0.008670904     0.3236364 0.022139377
## Escolta     0.002300652   0.007757838     0.3143444 0.017651428
##          total_bal_per total_dos_intentados max_puntos total_mates
## Ala-pívot 0.004201142   0.0006401413  0.01193013 0.004310104
## Alero       0.004166956   0.0007268538  0.01142599 0.006367380
## Base        0.008245997   0.0017926070  0.01836075 0.024321411
## Escolta     0.006430561   0.0012268284  0.01527114 0.012715401
##          total_titular total_tapones_real porcentaje_tl total_perdidos
## Ala-pívot 0.006521422   0.003832261   0.3076172 0.004586579
## Alero       0.008243690   0.005736151   0.2434511 0.005445940
## Base        0.013475570   0.015622055   0.2217141 0.009606245
## Escolta     0.011022647   0.012275467   0.2037773 0.007470292
##
## Residual Deviance: 3724.844
## AIC: 3852.844

```

Este modelo consigue reducir aún más el número de variables, en este caso a 15. Obtenemos las siguientes medidas:

- Residual Deviance: 3724.844
- AIC: 3852.844

Aunque aumenta el valor de ambas medidas para la bondad de ajuste.

```
modelo_stepwise <- step(modelo_nulo, scope = formula(mymultinom), direction = "both")
```

```
summary(modelo_stepwise)
```

```

## Call:
## multinom(formula = Posición ~ Altura + total_triple_intentados +
##           total_asis + total_rb + total_bal_rec + porcentaje_triple +
##           total_tapones_rec + total_bal_per + total_dos_intentados +
##           max_puntos + total_mates + total_titular + total_tapones_real +
##           porcentaje_tl + total_perdidos, data = datos_entrenamiento)
##
## Coefficients:

```

```

##          (Intercept)      Altura total_triple_intentados total_asis
## Ala-pívot  41.05685  -20.39954          0.0101041769  0.009615089
## Alero      121.82365 -60.16526          0.0113572040  0.009636327
## Base       236.47287 -119.95086          0.0006943595  0.041454105
## Escolta    187.16581 -94.11756          0.0119273506  0.026129849
##          total_rb total_bal_rec porcentaje_triple total_tapones_rec
## Ala-pívot -0.005736539  0.01477367      2.611853      0.05002140
## Alero      -0.012941820  0.02058842      1.455128     -0.02318920
## Base       -0.016867113  0.02702954      2.484369      0.02907923
## Escolta    -0.017842243  0.01097645      2.388331      0.03403784
##          total_bal_per total_dos_intentados max_puntos total_mates
## Ala-pívot -0.012822848  0.0008664763 -0.049614693  0.01670842
## Alero      0.007599806  0.0014593752 -0.001567285  0.01315913
## Base       0.020789234  -0.0071857179  0.050290020 -0.06851103
## Escolta    0.005389213  -0.0031488089  0.045550362 -0.01421363
##          total_titular total_tapones_real porcentaje_tl total_perdidos
## Ala-pívot -0.001889228  -0.006755302  0.86152785 -8.824763e-03
## Alero      0.019146107  -0.007829312 -0.01241791 -4.738605e-03
## Base       -0.015608297  -0.025579868  0.72437483 -1.602011e-02
## Escolta    0.006184814  0.009749357  0.37100457 -1.604663e-05
##
## Std. Errors:
##          (Intercept)      Altura total_triple_intentados total_asis
## Ala-pívot  0.03525354  0.07134945          0.001335881  0.003092934
## Alero      0.03382640  0.07096091          0.001441302  0.003827367
## Base       0.05197955  0.09995480          0.002399981  0.005708713
## Escolta    0.04537781  0.08966711          0.001664207  0.005012135
##          total_rb total_bal_rec porcentaje_triple total_tapones_rec
## Ala-pívot 0.001262658  0.005365178      0.4035532      0.009034197
## Alero      0.001644336  0.006152017      0.3850716      0.012266284
## Base       0.003155170  0.008670904      0.3236364      0.022139377
## Escolta    0.002300652  0.007757838      0.3143444      0.017651428
##          total_bal_per total_dos_intentados max_puntos total_mates
## Ala-pívot 0.004201142  0.0006401413  0.01193013  0.004310104
## Alero      0.004166956  0.0007268538  0.01142599  0.006367380
## Base       0.008245997  0.0017926070  0.01836075  0.024321411
## Escolta    0.006430561  0.0012268284  0.01527114  0.012715401
##          total_titular total_tapones_real porcentaje_tl total_perdidos
## Ala-pívot 0.006521422  0.003832261  0.3076172      0.004586579
## Alero      0.008243690  0.005736151  0.2434511      0.005445940
## Base       0.013475570  0.015622055  0.2217141      0.009606245
## Escolta    0.011022647  0.012275467  0.2037773      0.007470292
##
## Residual Deviance: 3724.844
## AIC: 3852.844

```

De nuevo obtenemos 15 variables, en este caso las medidas son las mismas que las del modelo forward.

De hecho, si nos fijamos, nos daremos cuenta que han devuelto el mismo modelo.

De esta manera, obtenemos dos modelos diferentes, uno que tiene 15 variables pero peores medidas de bondad del ajuste. Y otro con 17 y medidas algo parecidas de bondad del ajuste.

Yo voy a escoger el modelo backward, ya que es un modelo que es más o menos igual de bueno que

el multinomial y además consigue reducir el número de variables a 17.

Ahora vamos a comprobar si el modelo es significativo, para ello ejecutaremos los siguientes comandos:

```
diferencia_devianzas <- modelo_nulo$deviance - modelo_backward$deviance

grados_libertad <- nrow(datos_entrenamiento) - 1 - (nrow(datos_entrenamiento) - 17 - 1)

p_valor <- pchisq(diferencia_devianzas, df = grados_libertad, lower.tail = FALSE)
p_valor

## [1] 0
```

Recordemos que los grados de libertad para el modelo nulo se corresponden con  $n - 1$  (tamaño de la muestra menos 1) y los grados de libertad para el modelo completo se corresponden con  $n - k - 1$  (tamaño de la muestra menos número de coeficientes de regresión)

Obtenemos un p-valor de cero, por lo que podemos concluir que el modelo backward es significativo. Podemos ver los coeficientes de los parámetros de regresión sobre los odds:

```
exp(coef(mymultinom))
```

|                            | (Intercept)  | Altura              | total_titular        | max_puntos                   |
|----------------------------|--------------|---------------------|----------------------|------------------------------|
| ## Ala-pívot               | 5.468897e+15 | 1.304364e-08        | 0.9893639            | 0.9636132                    |
| ## Alero                   | 8.504770e+47 | 1.969057e-24        | 1.0219209            | 1.0096367                    |
| ## Base                    | 9.964081e+97 | 1.646388e-50        | 0.9783232            | 1.0535360                    |
| ## Escolta                 | 2.914112e+76 | 3.061566e-39        | 1.0010592            | 1.0652168                    |
| ## total_triple_intentados |              | porcentaje_triple   | total_dos_intentados |                              |
| ## Ala-pívot               |              | 1.0117411           | 11.625616            | 1.0024679                    |
| ## Alero                   |              | 1.0111205           | 3.970657             | 1.0026474                    |
| ## Base                    |              | 0.9991931           | 12.094137            | 0.9949664                    |
| ## Escolta                 |              | 1.0115323           | 10.096796            | 0.9997214                    |
| ## porcentaje_dos          |              | total_tl_intentados | porcentaje_tl        | total_rb total_asis          |
| ## Ala-pívot               | 1.6648658    |                     | 0.9891503            | 2.1353535 0.9931913 1.006792 |
| ## Alero                   | 0.8744616    |                     | 0.9981799            | 0.9656655 0.9827944 1.008723 |
| ## Base                    | 0.5064944    |                     | 0.9953141            | 1.9931614 0.9786008 1.041069 |
| ## Escolta                 | 0.5434336    |                     | 0.9884105            | 1.4717527 0.9766901 1.023173 |
| ## total_bal_per           |              | total_bal_rec       | total_tapones_real   | total_tapones_rec            |
| ## Ala-pívot               | 0.9913276    | 1.022283            | 0.9949011            | 1.042539                     |
| ## Alero                   | 1.0172731    | 1.028860            | 0.9957485            | 0.966378                     |
| ## Base                    | 1.0277472    | 1.035647            | 0.9723124            | 1.005497                     |
| ## Escolta                 | 1.0129207    | 1.023753            | 1.0222580            | 1.007371                     |
| ## total_mates             |              | total_faltas_real   | total_faltas_rec     | valoracion_pro               |
| ## Ala-pívot               | 1.0153304    | 0.9919532           | 1.013697             | 0.9991576                    |
| ## Alero                   | 1.0093104    | 0.9872056           | 1.004084             | 0.9994911                    |
| ## Base                    | 0.9170277    | 0.9893941           | 1.009325             | 1.0020102                    |
| ## Escolta                 | 0.9771330    | 0.9866623           | 1.016528             | 0.9997516                    |
| ## total_ganados           |              | total_perdidos      |                      |                              |
| ## Ala-pívot               | 1.005855     | 0.9998288           |                      |                              |
| ## Alero                   | 1.031005     | 1.0127900           |                      |                              |
| ## Base                    | 1.037590     | 0.9994644           |                      |                              |
| ## Escolta                 | 1.034573     | 1.0164431           |                      |                              |

Nótese que todos los coeficientes de las diferentes posiciones son muy parecidos, lo que parece indicar que no va a poder hacer regresión muy bien, aún así, vamos a comprobarlo.

```
pos_predict <- predict(modelo_backward, newdata = datos_test, "class")
test_predict <- cbind(datos_test, pos_predict)

## New names:
## * ' ' -> '...23'

#Cambiamos el nombre de la columna porque por defecto le da "...23"
column_index <- which(colnames(test_predict) == "...23")
colnames(test_predict)[column_index] <- "pos_predict"

matriz_confusion <- table(test_predict$Posición, test_predict$pos_predict,
                           dnn = c("real", "predicho"))
matriz_confusion

## predicho
## real      Píivot Ala-píivot Alero Base Escolta
## Ala-píivot 32      32    23    0    0
## Alero      15      18   108    7   25
## Base       0       0    7   106   14
## Escolta    0       0   31   22   55
## Píivot     157     10   12    0    0
```

Si queremos “ordenar” esta matriz, podemos hacer:

```
matriz_confusion <- matriz_confusion[order(rownames(matriz_confusion)),
                                         order(colnames(matriz_confusion))]

print(matriz_confusion)

## predicho
## real      Ala-píivot Alero Base Escolta Píivot
## Ala-píivot 32    23    0    0    32
## Alero      18   108    7   25   15
## Base       0     7   106   14    0
## Escolta    0    31   22   55    0
## Píivot     10   12    0    0   157
```

Me ha sorprendido bastante el acierto, pues como he dicho antes, los coeficientes en muchos de los regresores para las diferentes posiciones eran muy parecidos. Aunque si lo pensamos un poco son muchas variables, y puede ser que aunque los coeficientes sean parecidos, las pequeñas desviaciones hacen la diferencia entre la posición en la que juega un jugador y otra distinta.

Podemos ver ahora las métricas que se generan de la matriz de confusión:

```
accuracy <- sum(diag(matriz_confusion)) / sum(matriz_confusion)
accuracy
```

```
## [1] 0.6795252
```

Obtenemos un accuracy del 67.95%, que no es que sea muy bueno, pero es mejor que el que hemos obtenido con LDA (63.29%). Tampoco podemos pedir mucho más, porque una persona promedio viendo las estadísticas (sin conocer al jugador) difícilmente tendría ese acierto. Es complejo distinguir entre posiciones como escolta y alero. Y hay jugadores que a veces juegan por ejemplo de escolta y si el base está lesionado, pueden jugar de base.

Teniendo eso en cuenta creo que es un modelo bastante bueno.

Se puede apreciar que el más diferenciable es el base, mientras que el más complicado de acertar es el Ala-pívot.

Finalmente podríamos emplear el modelo para predecir alguno de los jugadores que teníamos marcado como “Desconocido” para ver qué predicción nos da:

```
nuevo_jugador <- acb_jugador_hist[head(which(acb_jugador_hist$num_posicion == 10), 1),  
, colnames(datos_test)]
```

```
predicción <- predict(modelo_stepwise, newdata = nuevo_jugador, "class")  
predicción
```

```
## [1] Píivot  
## Levels: Píivot Ala-pívot Alero Base Escolta
```

Vemos que lo clasifica como pívot. Podemos buscar el nombre del jugador en internet para ver si ha acertado:

```
acb_jugador_hist[head(which(acb_jugador_hist$num_posicion == 10), 1), "Nombre"]
```

```
## # A tibble: 1 x 1  
## # Groups:   Nombre [1]  
##   Nombre  
##   <chr>  
## 1 "Alexander Belostenny "
```

Si entramos en la siguiente página: [https://es.wikipedia.org/wiki/Aleksandr\\_Belostenny](https://es.wikipedia.org/wiki/Aleksandr_Belostenny) podremos ver que ha acertado, pero eso no es muy difícil, pues es un jugador de 2,14, por lo que no es difícil predecir que sea pívot. Podemos probar con otro:

```
nuevo_jugador2 <- acb_jugador_hist[which(acb_jugador_hist$num_posicion == 10)[2],  
, colnames(datos_test)]
```

```
predicción <- predict(modelo_stepwise, newdata = nuevo_jugador2, "class")  
predicción
```

```
## [1] Alero  
## Levels: Píivot Ala-pívot Alero Base Escolta
```

```
acb_jugador_hist[which(acb_jugador_hist$num_posicion == 10)[2], "Nombre"]
```

```
## # A tibble: 1 x 1
## # Groups:   Nombre [1]
##   Nombre
##   <chr>
## 1 "Anthony Duane Frederick "
```

Ha vuelto acertar, se puede mirar en la siguiente página web: [https://es.wikipedia.org/wiki/Anthony\\_Frederick](https://es.wikipedia.org/wiki/Anthony_Frederick)

Aunque esta vez podría haber fallado, pues si calculamos el número de veces que acierta que es alero cada vez que predice alero podemos ver que no es muy alta:

```
matriz_confusion["Alero", "Alero"] / sum(matriz_confusion[, "Alero"])
```

```
## [1] 0.5966851
```

Podríamos de hecho conseguir un modelo más sencillo, pues si recordamos, muchos de los coeficientes de los regresores eran prácticamente iguales. De esta manera podríamos crear un modelo con solamente 4 regresores: altura y porcentajes de 3 puntos, 2 puntos y tiro libre.

```
library("nnet")
datos_entrenamiento$Posición <- relevel(datos_entrenamiento$Posición, ref = "Píivot")
mymultinom2 <- multinom(Posición ~ Altura + porcentaje_triple + porcentaje_dos
                           + porcentaje_tl, data = datos_entrenamiento)
```

```
summary(mymultinom2)
```

```
## Call:
## multinom(formula = Posición ~ Altura + porcentaje_triple + porcentaje_dos +
##           porcentaje_tl, data = datos_entrenamiento)
##
## Coefficients:
## (Intercept) Altura porcentaje_triple porcentaje_dos porcentaje_tl
## Ala-píivot 40.25986 -20.06997 4.023676 -0.9210655 0.06811393
## Alero 125.22788 -61.90128 4.100182 -1.0692868 0.19357535
## Base 244.14669 -123.79493 5.271952 -1.8002012 1.14619980
## Escolta 191.81010 -96.40319 5.654784 -1.9178321 0.99291863
##
## Std. Errors:
## (Intercept) Altura porcentaje_triple porcentaje_dos porcentaje_tl
## Ala-píivot 4.262784 2.091183 0.4393561 0.4704489 0.3365478
## Alero 6.011186 2.977443 0.5169633 0.4908819 0.3659963
## Base 8.149578 4.122935 0.8288575 0.7225791 0.5229905
## Escolta 7.449393 3.736229 0.7165608 0.6464101 0.4687950
##
## Residual Deviance: 4517.845
## AIC: 4557.845
```

```
pos_predict2 <- predict(mymultinom2, newdata = datos_test, "class")
test_predict2 <- cbind(datos_test, pos_predict2)
```

```

## New names:
## * ' ' -> '...23'

#Cambiamos el nombre de la columna porque por defecto le da "...23"
column_index <- which(colnames(test_predict2) == "...23")
colnames(test_predict2)[column_index] <- "pos_predict"

matriz_confusion2 <- table(test_predict2$Posición, test_predict2$pos_predict
                           , dnn = c("real", "predicho"))
matriz_confusion2

## predicho
## real      Pívot Ala-pívot Alero Base Escolta
## Ala-pívot 40      12    35    0    0
## Alero      26      8    109   12   18
## Base       0      0    9    100   18
## Escolta    1      0    39   31   37
## Pívot     147     12   20    0    0

matriz_confusion2 <- matriz_confusion2[order(rownames(matriz_confusion2))
                                         ,order(colnames(matriz_confusion2))]

print(matriz_confusion2)

## predicho
## real      Ala-pívot Alero Base Escolta Pívot
## Ala-pívot 12    35    0    0    40
## Alero      8    109   12   18   26
## Base       0    9    100   18   0
## Escolta    0    39   31   37   1
## Pívot     12   20    0    0   147

accuracy2 <- sum(diag(matriz_confusion2)) / sum(matriz_confusion2)
accuracy2

## [1] 0.6008902

```

Ahora obtenemos un modelo un poco peor, pero que tiene solamente 4 regresores.

## 4 Conclusión

Tras un análisis exhaustivo de los datos hemos conseguido interpretar los datos y crear modelos capaces de predecir bastante bien (o al menos mejor que una persona promedio) como puede ser este último. Además hemos explorado todos los ámbitos vistos en la asignatura: desde PCA hasta RM.

También hemos sacado datos muy interesantes y progresado aún más en el análisis estadístico multivariante y las técnicas aprendidas durante las prácticas. Por lo que concluyendo, ha sido un trabajo constoso de hacer, pero muy interesante y beneficioso para nuestro aprendizaje.