A1 - Preproceso de datos

Propuesta de solución

Semestre 2020.2

Índice

Introducción

En esta actividad realizaremos el preprocesado de un fichero de datos que contiene el estilo de juego del videojuego de consola Fifa 2017, así como estadísticas reales de los jugadores de futbol. El conjunto de datos contiene más de 17500 registros y 53 variables.

Las principales variables que se usarán en esta actividad son:

- Name (Nombre del jugador)
- Nationality (Nacionalidad del jugador)
- Club_Joining (Fecha en la que empezó en el club)
- Contract_Expire (Año finalización del contrato)
- Rating (Valoración global del jugador, entre 0 y 100)
- Height (Altura)
- Weight (Peso)
- Preffered_Foot (Pie preferido)
- Birth_Date (Fecha de nacimiento)
- Age (Edad)
- Work_Rate (valoración cualitativa en términos de ataque-defensa)

La descripción de los atributos se puede consultar en https://www.fifplay.com/encyclopedia. La descripción de las abreviaturas de la posición del jugador en el campo se puede consultar en https://www.dtgre.com/2016/10/fifa-17-position-abbreviations-acronyms.html.

El objetivo de esta actividad es preparar el fichero para su posterior análisis. Para ello, se examinará el fichero para detectar y corregir posibles errores, inconsistencias y valores perdidos. Además, se presentará una breve estadística descriptiva y se hará un análisis de componentes principales (ACP) con algunas variables cuantitativas.

La actividad consta de tres partes diferenciades:

- En la primera parte (secciones 2, 3, 4 y 5), se realiza verificaciones y normalización de algunes variables, siguiendo los criterios que se especifican más adelante.
- En la segunda parte (secciones 6 y 7), se tratan los valores atípicos y los valores perdidos.
- En la tercera parte (secciones 8 y 9), se calculan algunas métricas de tendencia central y dispersión, que seria el primer paso del análisis descriptivo y, por último, se realiza un análisis de componentes principales (ACP) con algunas variables cuantitativas.

Criterios de verificación y de normalización de las variables:

A continuación se muestran los criterios con los que deben limpiarse los datos del conjunto:

1. En los datos numéricos, el símbolo de separador decimal es el punto y no la coma.
2. Verificar si hay registros duplicados con el valor ID. En caso de duplicación, seleccionar el registro con menor numero de NAs en las variables.
3. Las variables Name y Nationality no han de tener espacios en blanco antes o después de su valor. El valor para estas variables han de ser mayúsculas en la primera letra de cada palabra, tal como "Lionel Messi".
4. La variable Height se ha de expresar en cm con 3 dígitos sin decimales. Para facilitar la lectura y tratamiento del fichero deben ser numéricas, por lo tanto se debe quitar el símbolo de cm.

5. La variable Weight se ha de expresar en kg con 2 dígitos sin decimales. Si hay decimales, se ha de *truncar* el valor. Para facilitar la lectura y tratamiento del fichero deben ser numéricas, por lo tanto se debe quitar el símbolo de kg.
6. Verificar que la variable Club_Joining está en el rango de los años 1990 a 2017. En caso de haber algún registro que no compla la condición, indicar el número de registro, Name y Club_joining.
7. Verificar que el año de expiración del contracto (Contract_Expiry) no es inferior al año de inicio del contracto(Club_Joining). En caso de haber algún registro que no cumple la condición, indicar el número de registro, Name, Club_Joining y Contract_Expiry.
8. Verificar que la edad (Age) en la fecha 1/1/2017 corresponde a la calculada con la fecha de nacimiento (Birth_Date). En caso de haber algún registro que no cumpla la condición, modificar la edad en función del valor obtenido con la fecha de nacimiento.
9. Verificar que la variable Rating esté entre 0 y 100.
10. La variable Preffered_Foot ha de tener los valores Left y Right que corresponde a los valores actuales de 1 y 2, respectivamente.
11. La variable Work_Rate se basa en la combinación de dos de estas tres categorías: Low, Medium y High. Verificar que se cumple y en caso contrario, hay que corregirlo. Puedes encontrar los nombres de las categorías cortado con tres letras.

Para realizar el preproceso del fichero, seguir los pasos que se indican a continuación.

Aspectos importantes a tener en cuenta para entregar la actividad:

- Es necesario entregar el archivo Rmd y el fichero de salida (PDF o html). El archivo de salida debe incluir: el código y el resultado de la ejecución del código (paso a paso).
- Se debe respetar la misma numeración de los apartados que el enunciado.
- No se pueden realizar listados completos del conjunto de datos en la solución. Esto generaría un documento con cientos de páginas y dificulta la revisión del texto. Para comprobar las funcionalidades del código sobre los datos, se pueden usar las funciones **head** y **tail** que sólo muestran unas líneas del fichero de datos.
- Se valora la precisión de los términos. Es decir, se debe usar la terminología propia de la estadística.
- Se valora también la concisión en la respuesta. No se trata de hacer explicaciones muy largas o documentos muy extensos. El documento debe estar bien estructurado y ser conciso.

1. Carga del archivo

Se debe abrir el archivo de datos y examinar el tipo de datos con los que R ha interpretado cada variable. Examinar también los valores resumen de cada tipo de variable.

```
ds<-read.csv("fifa_raw.csv", header=TRUE,fileEncoding = "UTF-8")
str(ds)
```

```
## 'data.frame':    17588 obs. of  54 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Name              : chr  "Cristiano Ronaldo" "Lionel Messi" "Neymar" "Luis Suárez" ...
##  $ Nationality       : chr  "Portugal" "Argentina" "Brazil" "Uruguay" ...
##  $ National_Position : chr  "LS" "RW" "LW" "LS" ...
##  $ National_Kit      : int  7 10 10 9 1 1 9 11 NA 1 ...
##  $ Club              : chr  "Real Madrid" "FC Barcelona" "FC Barcelona" "FC Barcelona" ...
##  $ Club_Position     : chr  "LW" "RW" "LW" "ST" ...
##  $ Club_Kit          : int  7 10 11 9 1 1 9 11 9 13 ...
##  $ Club_Joining      : chr  "07/01/2009" "07/01/2004" "07/01/2013" "07/11/2014" ...
```

```
##  $ Contract_Expiry   : int  2021 2018 2021 2021 2021 2019 2021 2022 2017 2019 ...
##  $ Rating            : int  94 93 92 92 92 90 90 90 90 89 ...
##  $ Height            : chr  "1,85 m" NA "1,74 m" "1,82 m" ...
##  $ Weight            : chr  NA "72475 gr" "68884 gr" "85511 gr" ...
##  $ Preffered_Foot    : int  2 1 2 2 2 2 2 1 2 1 ...
##  $ Birth_Date        : chr  "02/05/1985" "06/24/1987" "02/05/1992" "01/24/1987" ...
##  $ Age               : int  32 29 25 30 31 26 28 27 35 24 ...
##  $ Preffered_Position: chr  "LW/ST" "RW" "LW" "ST" ...
##  $ Work_Rate         : chr  "High / Low" "Medium / Medium" "High / Medium" "High / Medium" ...
##  $ Weak_foot         : int  4 4 5 4 4 3 4 3 4 3 ...
##  $ Skill_Moves       : int  5 4 5 4 1 1 3 4 4 1 ...
##  $ Ball_Control      : int  93 95 95 91 48 31 87 88 90 23 ...
##  $ Dribbling         : int  92 97 96 86 30 13 85 89 87 13 ...
##  $ Marking           : int  22 13 21 30 10 13 25 51 15 11 ...
##  $ Sliding_Tackle    : int  23 26 33 38 11 13 19 52 27 16 ...
##  $ Standing_Tackle   : int  31 28 24 45 10 21 42 55 41 18 ...
##  $ Aggression        : int  63 48 56 78 29 38 80 65 84 23 ...
##  $ Reactions         : int  96 95 88 93 85 88 88 87 85 81 ...
##  $ Attacking_Position: int  94 93 90 92 12 12 89 86 86 13 ...
##  $ Interceptions     : int  29 22 36 41 30 30 39 59 20 15 ...
##  $ Vision            : int  85 90 80 84 70 68 78 79 83 44 ...
##  $ Composure         : int  86 94 80 83 70 60 87 85 91 52 ...
##  $ Crossing          : int  84 77 75 77 15 17 62 87 76 14 ...
##  $ Short_Pass        : int  83 88 81 83 55 31 83 86 84 32 ...
##  $ Long_Pass         : int  77 87 75 64 59 32 65 80 76 31 ...
##  $ Acceleration      : int  91 92 93 88 58 56 79 93 69 46 ...
##  $ Speed             : int  92 87 90 77 61 56 82 95 74 52 ...
##  $ Stamina           : int  92 74 79 89 44 25 79 78 75 38 ...
##  $ Strength          : int  80 59 49 76 83 64 84 80 93 70 ...
##  $ Balance           : int  63 95 82 60 35 43 79 65 41 45 ...
##  $ Agility           : int  90 90 96 86 52 57 78 77 86 61 ...
##  $ Jumping           : int  95 68 61 69 78 67 84 85 72 68 ...
##  $ Heading           : int  85 71 62 77 25 21 85 86 80 13 ...
##  $ Shot_Power        : int  92 85 78 87 25 31 86 91 93 36 ...
##  $ Finishing         : int  93 95 89 94 13 13 91 87 90 14 ...
##  $ Long_Shots        : int  90 88 77 86 16 12 82 90 88 17 ...
##  $ Curve             : int  81 89 79 86 14 21 77 86 82 19 ...
##  $ Freekick_Accuracy : int  76 90 84 84 11 19 76 85 82 11 ...
##  $ Penalties         : int  85 74 81 85 47 40 81 76 91 27 ...
##  $ Volleys           : int  88 85 83 88 11 13 86 76 93 12 ...
##  $ GK_Positioning    : int  14 14 15 33 91 86 8 5 9 86 ...
##  $ GK_Diving         : int  7 6 9 27 89 88 15 15 13 84 ...
##  $ GK_Kicking        : int  15 15 15 31 95 87 12 11 10 69 ...
##  $ GK_Handling       : int  11 11 9 25 90 85 6 15 15 91 ...
##  $ GK_Reflexes       : int  11 8 11 37 89 90 10 6 12 89 ...
```

```
#summary(ds)
```

2. Verificar duplicación de registros

```
# Obtener los registros duplicados
idx <- table(ds$ID)>1
# Hay  alguno?
```

```
count <-sum(idx)
```

Respuesta: El número de registros duplicados es 0.

3. Normalización de los datos cuantitativos

Inspeccionar los valores de los datos cuantitativos y realizar las normalizaciones oportunas siguiendo los criterios especificados anteriormente. Estas normalizaciones tienen como objetivo uniformizar los formatos. Si hay valores perdidos o valores extremos, se tratarán más adelante.

Al realizar estas normalizaciones, se debe demostrar que la normalización sobre cada variable ha dado el resultado esperado. Por lo tanto, se recomienda mostrar un fragmento del archivo de datos resultante. Para evitar mostrar todo el conjunto de datos, se puede mostrar una parte del mismo, con las funciones **head** y/o **tail**.

Seguid el orden de los apartados.

3.1. Rating

Respuesta: Revisar que el su valor está entre 0 y 100. No hay ningún registro fuera de este rango. No se aplica ninguna normalización.

```
class(ds$Rating)
```

```
## [1] "integer"
```

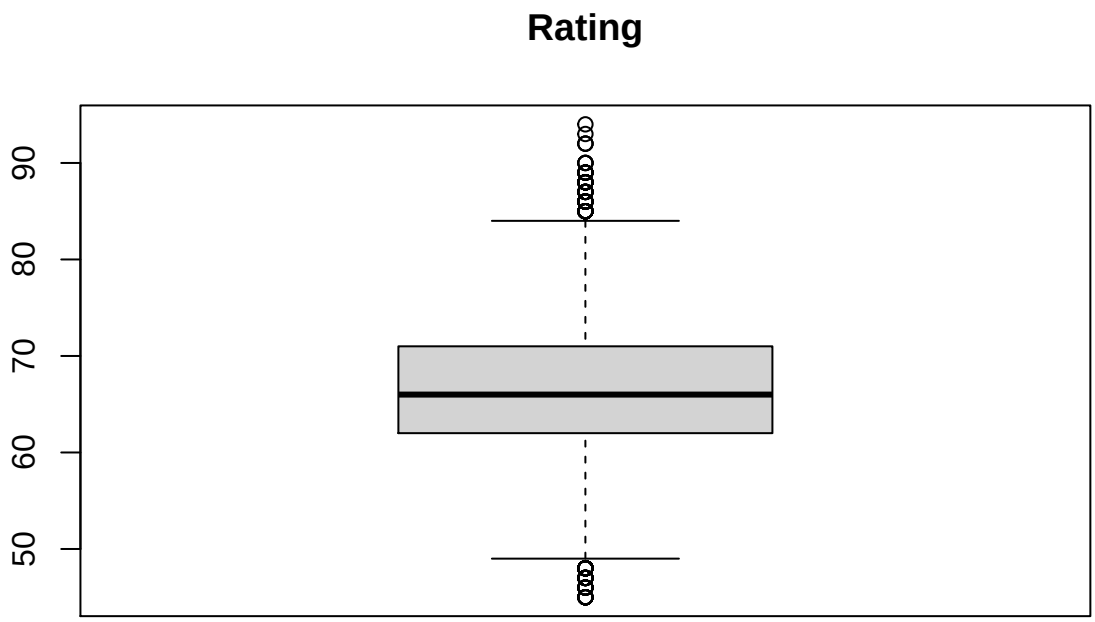
```
head(ds$Rating, 30)
```

```
##  [1] 94 93 92 92 92 90 90 90 90 89 89 89 89 89 89 89 89 89 88 88 88 88 88 88 88
## [26] 88 88 88 88 88
```

```
fivenum(ds$Rating)
```

```
## [1] 45 62 66 71 94
```

```
boxplot(ds$Rating, main="Rating")
```



3.2. Height

Respuesta:

Height se ha de expresar en cm con 3 dígitos sin decimales. En algunos casos hay decimales. Quitar las unidades. Pasar a formato numérico

```
dss<-ds  #guardar en dss el fichero original para posibles consultas


class(ds$Height)
```

```
## [1] "character"
```

```
head(ds$Height, 30)
```

```
##  [1] "1,85 m" NA       "1,74 m" "1,82 m" "1,93 m" NA       "1,85 m" "1,83 m"
##  [9] NA       "1,99 m" "1,92 m" "1,73 m" "1,74 m" "180 cm" "184 cm" "183 cm"
## [17] "1,83 m" "173 cm" "191 cm" "176 cm" "181 cm" "182 cm" "169 cm" "1,82 m"
## [25] "185 cm" "1,91 m" "188 cm" "187 cm" "1,7 m"  "1,88 m"
```

```
# Cambiamos , por . decimal
ds$Height<-gsub("\\,", "\\.", ds$Height)

# Verificación
idx.coma<-grep("\\,", ds$Height)
idx.coma
```

```
## integer(0)
```

```r
# función convertir metres a cms
m_to_cm <-function(height){
  height <- as.character(height)
  # indice para height en m
  idxm <- grep( "\\ m", height )
  # seleccionar los valores numéricos
  num_val <- word(height)
  # convertir a numerico
  num_val <- as.numeric(num_val)
  # pasar los valores en m a cm

  num_val[idxm] <- num_val[idxm]*100

  #sin decimales
  num_val <-round(num_val,0)

  return (num_val)
}


head(ds$Height, 30)
```

```
##  [1] "1.85 m" NA       "1.74 m" "1.82 m" "1.93 m" NA       "1.85 m" "1.83 m"
##  [9] NA       "1.99 m" "1.92 m" "1.73 m" "1.74 m" "180 cm" "184 cm" "183 cm"
## [17] "1.83 m" "173 cm" "191 cm" "176 cm" "181 cm" "182 cm" "169 cm" "1.82 m"
## [25] "185 cm" "1.91 m" "188 cm" "187 cm" "1.7 m"  "1.88 m"
```

```r
Height <- m_to_cm( ds$Height )
head( paste( Height, ds$Height), 30)
```

```
##  [1] "185 1.85 m" "NA NA"      "174 1.74 m" "182 1.82 m" "193 1.93 m"
##  [6] "NA NA"      "185 1.85 m" "183 1.83 m" "NA NA"      "199 1.99 m"
## [11] "192 1.92 m" "173 1.73 m" "174 1.74 m" "180 180 cm" "184 184 cm"
## [16] "183 183 cm" "183 1.83 m" "173 173 cm" "191 191 cm" "176 176 cm"
## [21] "181 181 cm" "182 182 cm" "169 169 cm" "182 1.82 m" "185 185 cm"
## [26] "191 1.91 m" "188 188 cm" "187 187 cm" "170 1.7 m"  "188 1.88 m"
```

```r
ds$Height <- Height
```

3.3. Weight

Respuesta:

Weight se ha de expresar en Kg sin decimales. Si hay decimales, se ha de truncar. Quitar las unitades. Formato numérico.

```r
  class(ds$Weight)
```

```
## [1] "character"
```

```r
  head(ds$Weight, 30)
```

```
##  [1] NA          "72475 gr"  "68884 gr"  "85511 gr"  NA          "82.671 kg"
##  [7] NA          "74683 gr"  "95.429 kg" "91394 gr"  "90865 gr"  "74579 gr"
## [13] "65474 gr"  "76,777 kg" "92,443 kg" "79,164 kg" "75432 gr"  "70.238 kg"
## [19] "84.588 kg" "67.393 kg" "68.275 kg" "76.384 kg" "62,52 kg"  "78729 gr"
## [25] "73.709 kg" "92828 gr"  "82,468 kg" "84,887 kg" "66268 gr"  "81269 gr"
```

```r
# Cambiamos , por . decimal
ds$Weight<-gsub("\\,", "\\.", ds$Weight)

# Verificación
idx.coma<-grep("\\,", ds$Weight)
idx.coma
```

```
## integer(0)
```

```r
# función para convertir grams a kg
gr_to_kg <-function(weight){
  height <- as.character(weight)
  # indice para weight en gr
  idxm <- grep( "\\ gr", weight )
  # seleccionar los valores numéricos
  num_val <- word(weight)
  # convertir a numerico
  num_val <- as.numeric(num_val)
  # pasar los valores en m a cm

  num_val[idxm] <- num_val[idxm]/1000

  #sin decimales
  num_val <-trunc(num_val,0)

  return (num_val)
}


head(ds$Weight, 30)
```

```
##  [1] NA          "72475 gr"  "68884 gr"  "85511 gr"  NA          "82.671 kg"
##  [7] NA          "74683 gr"  "95.429 kg" "91394 gr"  "90865 gr"  "74579 gr"
## [13] "65474 gr"  "76.777 kg" "92.443 kg" "79.164 kg" "75432 gr"  "70.238 kg"
## [19] "84.588 kg" "67.393 kg" "68.275 kg" "76.384 kg" "62.52 kg"  "78729 gr"
## [25] "73.709 kg" "92828 gr"  "82.468 kg" "84.887 kg" "66268 gr"  "81269 gr"
```

```r
Weight <- gr_to_kg( ds$Weight )
head( paste( Weight, ds$Weight), 30)
```

```
##  [1] "NA NA"        "72 72475 gr"  "68 68884 gr"  "85 85511 gr"  "NA NA"
##  [6] "82 82.671 kg" "NA NA"        "74 74683 gr"  "95 95.429 kg" "91 91394 gr"
## [11] "90 90865 gr"  "74 74579 gr"  "65 65474 gr"  "76 76.777 kg" "92 92.443 kg"
## [16] "79 79.164 kg" "75 75432 gr"  "70 70.238 kg" "84 84.588 kg" "67 67.393 kg"
## [21] "68 68.275 kg" "76 76.384 kg" "62 62.52 kg"  "78 78729 gr"  "73 73.709 kg"
## [26] "92 92828 gr"  "82 82.468 kg" "84 84.887 kg" "66 66268 gr"  "81 81269 gr"
```

```r
ds$Weight <- Weight
```

4. Normalización de los datos cualitativos

Inspeccionar los valores de los datos cualitativos y realizar las transformaciones oportunas, siguiendo los criterios especificados. Al igual que en el apartado anterior, mostrar el resultado sobre un fragmento del conjunto de datos.

Se debe seguir el orden especificado de los apartados.

4.1. Name y Nationality

Respuesta: Primer se eliminan los espacios en blanco antes y después del valor. Después se pasa a mayúsculas la primera letra de cada palabra.

```
# Name
head(ds$Name, 30)
```

```
##  [1] "Cristiano Ronaldo"   "Lionel Messi"        "Neymar"
##  [4] "Luis Suárez"         "Manuel Neuer"        "De Gea"
##  [7] "robert Lewandowski " "Gareth Bale"         "Zlatan Ibrahimovic"
## [10] "Thibaut Courtois"    "Jérôme Boateng"      "Eden Hazard"
## [13] "Luka Modric"         "mesut Özil   "       "Gonzalo Higuaín"
## [16] "Thiago Silva"        "Sergio Ramos"        "sergio Agüero   "
## [19] "Paul Pogba"          "Antoine Griezmann"   "Kevin De Bruyne"
## [22] "Marco Reus"          "Alexis Sánchez"      "Toni Kroos"
## [25] "Diego Godín"         "Mats Hummels"        "Hugo Lloris"
## [28] "Giorgio Chiellini"   "Philipp Lahm"        "Pepe"
```

```
# Elimina espacios en blanco antes y después
ds$Name <- trimws(ds$Name)

# Primera palabra mayúscula
ds$Name <- str_to_title(ds$Name)

head( paste(ds$Name, dss$Name, sep="-"), 30)
```

```
##  [1] "Cristiano Ronaldo-Cristiano Ronaldo"
##  [2] "Lionel Messi-Lionel Messi"
##  [3] "Neymar-Neymar"
##  [4] "Luis Suárez-Luis Suárez"
##  [5] "Manuel Neuer-Manuel Neuer"
##  [6] "De Gea-De Gea"
##  [7] "Robert Lewandowski-robert Lewandowski "
##  [8] "Gareth Bale-Gareth Bale"
##  [9] "Zlatan Ibrahimovic-Zlatan Ibrahimovic"
## [10] "Thibaut Courtois-Thibaut Courtois"
## [11] "Jérôme Boateng-Jérôme Boateng"
## [12] "Eden Hazard-Eden Hazard"
## [13] "Luka Modric-Luka Modric"
## [14] "Mesut Özil-mesut Özil   "
## [15] "Gonzalo Higuaín-Gonzalo Higuaín"
## [16] "Thiago Silva-Thiago Silva"
## [17] "Sergio Ramos-Sergio Ramos"
## [18] "Sergio Agüero-sergio Agüero   "
## [19] "Paul Pogba-Paul Pogba"
## [20] "Antoine Griezmann-Antoine Griezmann"
## [21] "Kevin De Bruyne-Kevin De Bruyne"
## [22] "Marco Reus-Marco Reus"
## [23] "Alexis Sánchez-Alexis Sánchez"
## [24] "Toni Kroos-Toni Kroos"
## [25] "Diego Godín-Diego Godín"
## [26] "Mats Hummels-Mats Hummels"
## [27] "Hugo Lloris-Hugo Lloris"
## [28] "Giorgio Chiellini-Giorgio Chiellini"
## [29] "Philipp Lahm-Philipp Lahm"
```

```
## [30] "Pepe-Pepe"
```

```
# Nationality
head(ds$Nationality, 30)
```

```
##  [1] "Portugal"  "Argentina" "Brazil"    "Uruguay"   "Germany"   "Spain"
##  [7] "Poland"    "Wales"     "Sweden"    "Belgium"   "Germany"   "Belgium"
## [13] "Croatia"   "Germany"   "Argentina" "Brazil"    "Spain"     "Argentina"
## [19] "France"    "France"    "Belgium"   "Germany"   "Chile"     "Germany"
## [25] "Uruguay"   "Germany"   "France"    "Italy"     "Germany"   "Portugal"
```

```
# Elimina espacios en blanco antes y después
ds$Nationality <- trimws(ds$Nationality)

# Primera palabra mayúscula
ds$Nationality <- str_to_title(ds$Nationality)
head( paste(ds$Nationality, dss$Nationality, sep="-"), 30)
```

```
##  [1] "Portugal-Portugal"   "Argentina-Argentina" "Brazil-Brazil"
##  [4] "Uruguay-Uruguay"     "Germany-Germany"     "Spain-Spain"
##  [7] "Poland-Poland"       "Wales-Wales"         "Sweden-Sweden"
## [10] "Belgium-Belgium"     "Germany-Germany"     "Belgium-Belgium"
## [13] "Croatia-Croatia"     "Germany-Germany"     "Argentina-Argentina"
## [16] "Brazil-Brazil"       "Spain-Spain"         "Argentina-Argentina"
## [19] "France-France"       "France-France"       "Belgium-Belgium"
## [22] "Germany-Germany"     "Chile-Chile"         "Germany-Germany"
## [25] "Uruguay-Uruguay"     "Germany-Germany"     "France-France"
## [28] "Italy-Italy"         "Germany-Germany"     "Portugal-Portugal"
```

4.2. Preffered_Foot

Respuesta: La variable Preffered_Foot tiene los valores 1 y 2. Se normalizan a Left y Right, respectivamente. Variable tipo factor.

```
 unique( ds$Preffered_Foot)
```

```
## [1] 2 1
```

```
 class( ds$Preffered_Foot )
```

```
## [1] "integer"
```

```
 ds$Preffered_Foot <- factor(ds$Preffered_Foot,
                         levels = c(1,2),
                         labels = c("Left","Right"))

 head( paste(ds$Preffered_Foot, dss$Preffered_Foot), 30)
```

```
##  [1] "Right 2" "Left 1"  "Right 2" "Right 2" "Right 2" "Right 2" "Right 2"
##  [8] "Left 1"  "Right 2" "Left 1"  "Right 2" "Right 2" "Right 2" "Left 1"
## [15] "Right 2" "Right 2" "Right 2" "Right 2" "Right 2" "Left 1"  "Right 2"
## [22] "Right 2" "Right 2" "Right 2" "Right 2" "Right 2" "Left 1"  "Left 1"
## [29] "Right 2" "Right 2"
```

4.3. Work_Rate

Respuesta: Se encuentran los registros con categorías incorrectas y se corrige.

```r
unique( ds$Work_Rate)
```

```
##  [1] "High / Low"      "Medium / Medium" "High / Medium"   "Medium / Low"
##  [5] "High / High"     "Med / Med"       "Medium / High"   "Low / High"
##  [9] "Low / Medium"    "Hig / Med"       "Low / Low"
```

```r
class( ds$Work_Rate )
```

```
## [1] "character"
```

```r
# cambiar "Med / Med" per "Medium / Medium"
idx <- which(ds$Work_Rate=="Med / Med")
ds$Work_Rate[idx] <- "Medium / Medium"
#check
head( paste(ds$Work_Rate[idx], dss$Work_Rate[idx]), 30)
```

```
##  [1] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
##  [3] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
##  [5] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
##  [7] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
##  [9] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [11] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [13] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [15] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [17] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [19] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [21] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [23] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [25] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [27] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
## [29] "Medium / Medium Med / Med" "Medium / Medium Med / Med"
```

```r
# cambiar "Hig / Med" per "High / Medium"
idx <- which(ds$Work_Rate=="Hig / Med")
ds$Work_Rate[idx] <- "High / Medium"
#check
head( paste(ds$Work_Rate[idx], dss$Work_Rate[idx]), 30)
```

```
##  [1] "High / Medium Hig / Med" "High / Medium Hig / Med"
##  [3] "High / Medium Hig / Med" "High / Medium Hig / Med"
##  [5] "High / Medium Hig / Med" "High / Medium Hig / Med"
##  [7] "High / Medium Hig / Med" "High / Medium Hig / Med"
##  [9] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [11] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [13] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [15] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [17] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [19] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [21] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [23] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [25] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [27] "High / Medium Hig / Med" "High / Medium Hig / Med"
## [29] "High / Medium Hig / Med" "High / Medium Hig / Med"
```

```r
ds$Work_Rate <- as.factor(ds$Work_Rate)
 # check
```

```
 levels(ds$Work_Rate)
```

```
## [1] "High / High"     "High / Low"      "High / Medium"   "Low / High"
## [5] "Low / Low"       "Low / Medium"    "Medium / High"   "Medium / Low"
## [9] "Medium / Medium"
```

5. Posibles inconsistencias y variables tipo fecha

Verificar si existen inconsistencias entre algunas variables. Por otra parte, algunas de las variables es necesario indicar que son de tipo fecha (en r, tipo `Date`) para luego hacer las transformaciones adecuadas. Observar que la configuración del tipo fecha es mes/día/año. Al igual que en el apartado anterior, muestre el resultado sobre un fragmento del conjunto de datos.

Se debe seguir el orden especificado de los apartados.

5.1. Club_Joining

Respuesta: Hay que pasar la variable a formato data para extraer los años.

```
class( ds$Club_Joining )
```

```
## [1] "character"
```

```
# convertir a Dates
dates <- mdy(ds$Club_Joining)
class(dates)
```

```
## [1] "Date"
```

```
# check: rango de los años 1990 a 2017
unique(year(dates))
```

```
##  [1] 2009 2004 2013 2014 2011 2016 2012 2005 2015 2010 2002 2007 2001 2008 2017
## [16] 2006 2003   NA 1998 1993 1991 1999 2000
```

5.2. Contract_Expiry >= Club_Joining?

Respuesta: Hay que extraer el año de la variable Club_Joining. Después comparar los años.

```
class(ds$Contract_Expiry)
```

```
## [1] "integer"
```

```
# Check Contract_Expiry > Club_Joining
dif.year <- ds$Contract_Expiry - year(dates)
unique(dif.year)
```

```
##  [1] 12 14  8  7 10  9  1  5 15  6  4 13 16 17 11  3  2  0 18 NA 19 24 20 30 22
## [26] 23 21
```

```
# Es correcto
```

5.3. Revisar si la edad corresponde a la fecha de nacimiento

Respuesta: Birth_date se convierte a tipo `Date`. Se calcula la edas desde la fecha 1/1/2017. Se compara la edad calculada con la variable Age. En cass de haber algun registro que no cumpla la condición, se modifica la edat en función del valor obtenido con la fecha de nacimiento.

```
class(ds$Birth_Date)
```

```
## [1] "character"
```

```
# convertir a Date
dates <- mdy(ds$Birth_Date)
class(dates)
```

```
## [1] "Date"
```

```
# Cálculo de la edad

years_comp <- trunc(time_length(difftime(dmy("01-01-2017"),
                                         dates), "years"))


# Age diferente que la edad calculada
idx <- which( years_comp != ds$Age)

# check
head(paste(years_comp[idx], dss$Age[idx]),30)
```

```
##  [1] "31 32" "24 25" "29 30" "30 31" "25 26" "30 31" "23 24" "25 26" "26 27"
## [10] "30 31" "33 34" "38 39" "23 24" "28 29" "30 31" "29 30" "32 33" "27 28"
## [19] "24 25" "27 28" "28 29" "29 30" "30 31" "24 25" "27 28" "29 30" "24 25"
## [28] "25 26" "23 24" "24 25"
```

```
ds$Age[idx] <- years_comp[idx]

# check
head(paste(ds$Age[idx], dss$Age[idx]),30)
```

```
##  [1] "31 32" "24 25" "29 30" "30 31" "25 26" "30 31" "23 24" "25 26" "26 27"
## [10] "30 31" "33 34" "38 39" "23 24" "28 29" "30 31" "29 30" "32 33" "27 28"
## [19] "24 25" "27 28" "28 29" "29 30" "30 31" "24 25" "27 28" "29 30" "24 25"
## [28] "25 26" "23 24" "24 25"
```

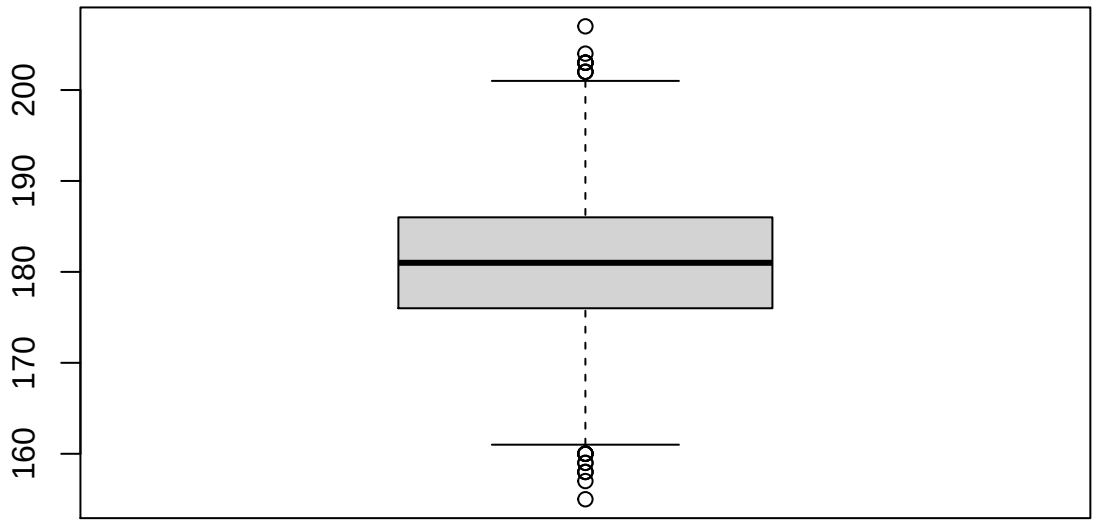
```
# Check: Age diferente que la edad calculada
sum( years_comp != ds$Age)
```

```
## [1] 0
```

6. Valores atípicos

Revisar si hay valores atípicos en la variable Height y Weight. Si es así, y se trata de un valor *anormalmente* alto o bajo, se recomienda sustituir el valor por "NA".

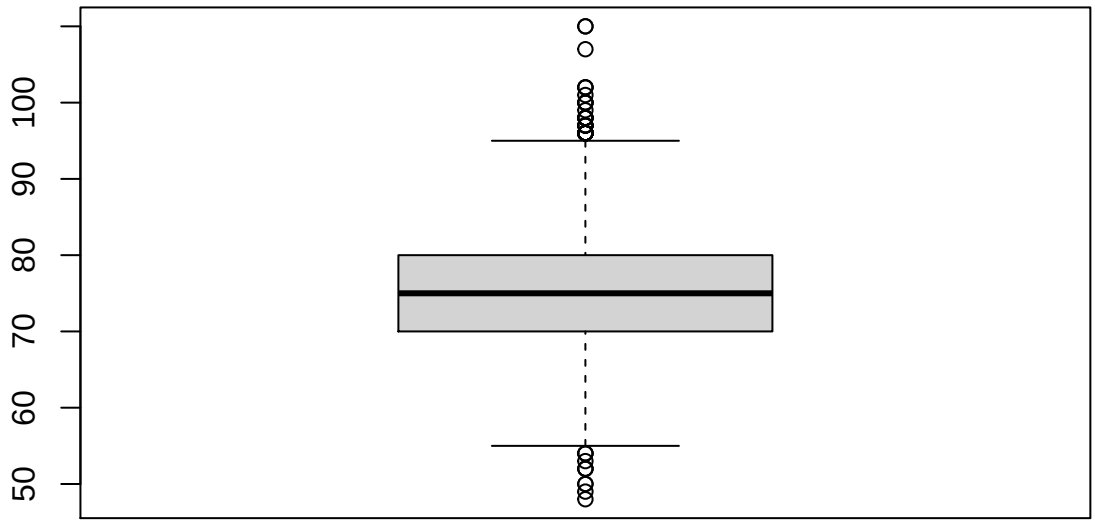
```
 boxplot(ds$Height)
```




```
x<-boxplot.stats(ds$Height)$out
idx <- which( ds$Height %in% x)
sort(ds$Height[idx])    # no se ha encontrado ningún valor anormalmente alto o bajo
```

```
##  [1] 155 157 158 158 158 159 159 160 160 160 160 160 160 202 202 202 203 203 203
## [20] 203 203 204 207
```

```
boxplot(ds$Weight)
```




```
x<-boxplot.stats(ds$Weight)$out
idx <- which( ds$Weight %in% x)
sort(ds$Weight[idx])    # no se ha encontrado ningún valor anormalmente alto o bajo
```

```
##  [1]  48  49  50  50  52  52  52  53  54  54  54  96  96  96  96  96  96  96  96
## [20]  96  96  96  96  96  96  96  96  96  96  96  96  96  97  97  97  97  97  98
## [39]  98  98  98  98  98  99 100 100 100 100 101 102 102 102 107 110 110
```

Respuesta: No se ha encontrado ningún valor anormalmente alto o bajo.

7. Imputación de valores

Buscar en las variables Weight y Height donde haya valores perdidos (NA) y realice una imputación de valores.

Para realizar una imputación de valores necesita hacer una regresión lineal que tenga como variable a predecir la variable con la NAs. Esto significa que hay que realizar dos modelos de regresión lineal, uno para predecir los valores NAs en Weight a partir de la variable Height. El otro es precisamente al revés, predecir los valores NAs en Height a partir de Weight.

La función para hacer regresión lineal es `lm()`.

Muestre el resultado de la imputación y de la variable explicativa en aquellos casos donde había un NA.

```
# Caso Height

# registros con valores NA
idx <- which(is.na(ds$Height))
```

```r
# modelo lineal
lm_H.W <- lm(Height ~ Weight, data=ds)

# predicción
ds$Height[idx] <- predict(lm_H.W, newdata=ds[idx,c(12,13)] )

# redondear a cm sin decimales

ds$Height[idx] <- round(ds$Height[idx],0)

# Mostrar Weight / Height de la predicción

paste(ds$Weight[idx], ds$Height[idx])
```

```
## [1] "72 179" "82 186" "95 196"
```

```r
# Caso Weight

# registros con valores NA
idx <- which(is.na(ds$Weight))

# modelo lineal
lm_W.H <- lm(Weight ~ Height, data=ds)

# predicción
ds$Weight[idx] <- predict(lm_W.H, newdata=ds[idx,c(12,13)] )

# redondear a cm sin decimales

ds$Weight[idx] <- round(ds$Weight[idx],0)

# Mostrar Weight / Height de la predicción

paste(ds$Weight[idx], ds$Height[idx])
```

```
## [1] "78 185" "85 193" "78 185"
```

8. Estudio descriptivo de las variables cuantitativas.

Realice un breve estudio descriptivo de las variables cuantitativas una vez depuradas. Hay que crear una tabla con medidas de tendencia central y de dispersión, tanto robustas como no robustas. Haga un breve comentario sobre los resultados obtenidos entre estos tipos de medidas en todas las variables.

```r
res <- c(11,12,13,16)
mean.n <- as.vector(sapply( ds[,res ],mean,na.rm=TRUE ) )
std.n <- as.vector(sapply(ds[,res ],sd, na.rm=TRUE))
median.n <- as.vector(sapply(ds[,res],median, na.rm=TRUE))
mean.trim.0.05 <- as.vector(sapply(ds[,res],mean, na.rm=TRUE, trim=0.05))
mean.winsor.0.05 <- as.vector(sapply(ds[,res],winsor.mean, na.rm=TRUE,trim=0.05))
IQR.n <- as.vector(sapply(ds[,res],IQR, na.rm=TRUE))
mad.n <- as.vector(sapply(ds[,res],mad, na.rm=TRUE))

kable(data.frame(variables= names(ds)[res],
                 Media = mean.n,
```

Cuadro 1: Estimaciones de Tendéncia Central

variables	Media	Mediana	Media.recort.0.05	Media.winsor.0.05
Rating	66.17	66	66.16	66.15
Height	181.11	181	181.11	181.10
Weight	75.25	75	75.16	75.19
Age	25.14	25	24.99	25.04

Cuadro 2: Estimaciones de Dispersión

variables	Desv.Standard	IQR	MAD
Rating	7.08	9	7.41
Height	6.67	10	7.41
Weight	6.90	10	7.41
Age	4.68	7	4.45

```
                Mediana = median.n,
                Media.recort.0.05= mean.trim.0.05,
                Media.winsor.0.05= mean.winsor.0.05
                ),
      digits=2, caption="Estimaciones de Tendéncia Central")
```

```
kable(data.frame(variables= names(ds)[res],
                Desv.Standard = std.n,
                IQR = IQR.n,
                MAD = mad.n
                ),
      digits=2, caption="Estimaciones de Dispersión")
```

Respuesta: 'Respecto a las estimaciones de tendencia central no se observa cambios importantes entre ellas en ninguna de las variables. Esto indica que no hay valores muy extremos. El jugador tipo tiene un Rating de 66, un Height de 181 cm, un Weight de 75 kg y un Age de 25 años.

Respecto a las estimaciones de dispersión se puede decir lo mismo. Los valores de desviación estandard y MAD son muy similares para las cuatro variables. En cambio, la IQR es más alta como pasa usualmente.

9. Análisis de Componentes Principales (ACP)

Realizar el Análisis de Componentes Principales sobre las variables "Rating", "Height", "Weight" y "Age". Representar el gráfico biplot de dos dimensiones. Hacer un breve comentario indicando el porcentaje de variabilidad explicada en cada componente principal, la variabilidad explicada en las dos primeras dimensiones y qué variable original está más asociada a cada una de las dos primeras componentes principales. Interpretar.

Por ultimo, ¿hay algún punto más fuera de la nube de puntos?, si es así puedes mostrar los valores de las variables originales e indicar que tiene de especial este punto.

Para responder a este apartado, puede usar la función `prcomp` y `ggbiplot`.

```
# Cálculo ACP caso M. correlaciones


ds.ACP <-ds[,c(11,12,13,16)]
```

```r
ACP.cor <- prcomp(x=ds.ACP, center = TRUE, scale.= TRUE)
summary(ACP.cor)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4
## Standard deviation     1.3773 1.1541 0.7374 0.47694
## Proportion of Variance 0.4742 0.3330 0.1359 0.05687
## Cumulative Proportion  0.4742 0.8072 0.9431 1.00000
```

```r
ACP.cor$sdev # raiz cuadrada de los valores propios
```

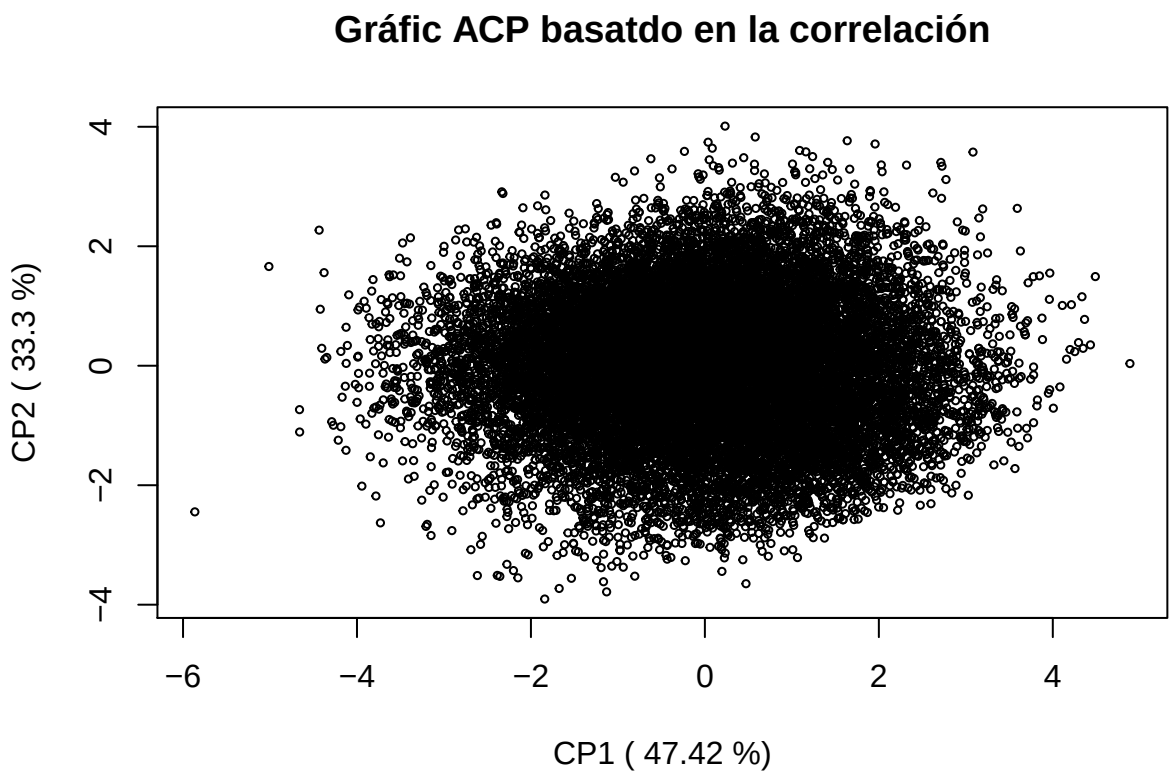
```
## [1] 1.3772799 1.1540780 0.7373825 0.4769393
```

```r
var.exp.cor <- (ACP.cor$sdev^2 / sum(ACP.cor$sdev^2))*100 # % de variánza explicada en cada CP
ACP.cor$rotation # vectors propis
```

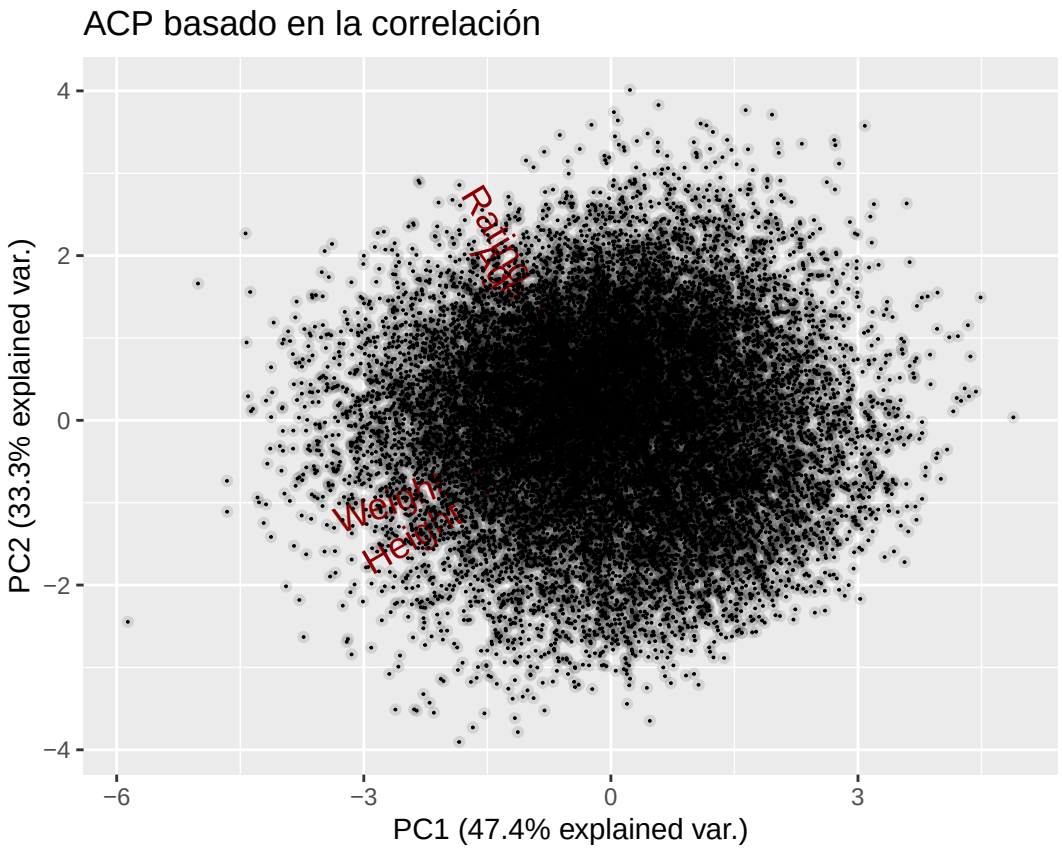
```
##               PC1        PC2         PC3          PC4
## Rating -0.3197977  0.6350967  0.70306433  0.009062855
## Height -0.5904440 -0.4122256  0.09494332  0.687336661
## Weight -0.6409909 -0.2795857 -0.02979932 -0.714194974
## Age    -0.3718009  0.5903847 -0.70412947  0.131953468
```

```r
#ACP.cor$x # Componentes principales: Valores proyectados

plot(ACP.cor$x[,1],ACP.cor$x[,2],cex=0.5,
xlab = paste("CP1 (", round(var.exp.cor[1],2),"%)"),
ylab = paste("CP2 (", round(var.exp.cor[2],2),"%)"),
main = "Gráfic ACP basatdo en la correlación")
```



```
biplot.ACP.cor <- ggbiplot(ACP.cor, scale = 1, obs.scale = 1,
                           varname.size=5, alpha=1/10) +
geom_point(colour = "black", fill = "white", size = 0.01 )+
ggtitle("ACP basado en la correlación") +
theme(legend.direction = 'horizontal',
legend.position = 'bottom',
legend.title = element_blank())
print(biplot.ACP.cor)
```




```
# Punto fuera de la nube de puntos.

#ACP.cor$x # Componentes principales: Valores proyectados
# Selección puntn
idx <- which(ACP.cor$x[,1] < -5.5)

# valores ACP
ACP.cor$x[idx,]
```

```
##        PC1        PC2        PC3        PC4
## -5.8642611 -2.4462872 -0.2796097 -0.8215292
```

```
# Valores originals
ds.ACP[idx,]
```

```
##      Rating Height Weight Age
## 8359     67    207    110  29
```

```
# comparación con el resuldo de todos los puntos
summary(ds.ACP)
```

```
##      Rating          Height          Weight            Age
##  Min.   :45.00   Min.   :155.0   Min.   : 48.00   Min.   :16.00
##  1st Qu.:62.00   1st Qu.:176.0   1st Qu.: 70.00   1st Qu.:21.00
##  Median :66.00   Median :181.0   Median : 75.00   Median :25.00
##  Mean   :66.17   Mean   :181.1   Mean   : 75.25   Mean   :25.14
##  3rd Qu.:71.00   3rd Qu.:186.0   3rd Qu.: 80.00   3rd Qu.:28.00
##  Max.   :94.00   Max.   :207.0   Max.   :110.00   Max.   :47.00
```

Respuesta: La variabilidad explicada (en %) de las componentes principales son: 47.42, 33.3, 13.59, 5.69, respectivamente. Así, las dos primeras componentes principales explican un 80.72 % de la variabilidad original.

Los valores propios indican la importancia de cada variable original en cada componente principal:

```
kable(ACP.cor$rot)
```

	PC1	PC2	PC3	PC4
Rating	-0.3197977	0.6350967	0.7030643	0.0090629
Height	-0.5904440	-0.4122256	0.0949433	0.6873367
Weight	-0.6409909	-0.2795857	-0.0297993	-0.7141950
Age	-0.3718009	0.5903847	-0.7041295	0.1319535

Aquellos valores propios con valor absoluto más altos son los que tienen más relevancia en la componente principal. Entonces, la primera componente está asociada a la variable Weight y Height, de tal manera que un valor menor de la primera componente principal corresponde a un valor mayor en Weight y Height ya que su valor es negativo. En cambio, la segunda componente principal está asociada a la variable Rating y Age, tal que a major valor de la segunda componente principal corresponde a un valor mayor en Rating y Age.

El valor más apartado de la nube de puntos es (-5.86, -2.45) y corresponde al registro 8359. Es el punto menor de la primera componente principal, como está asociada a Weight y Height, se espera que tenga un valor muy alto respecto al resto de valores. Los valores originales son:

```
kable(ds.ACP[idx,])
```

	Rating	Height	Weight	Age
8359	67	207	110	29

Observar que los valores de Weight y Height son los valores máximos de la matriz de datos.

```
kable(summary(ds.ACP))
```

	Rating	Height	Weight	Age
	Min. :45.00	Min. :155.0	Min. : 48.00	Min. :16.00
	1st Qu.:62.00	1st Qu.:176.0	1st Qu.: 70.00	1st Qu.:21.00
	Median :66.00	Median :181.0	Median : 75.00	Median :25.00
	Mean :66.17	Mean :181.1	Mean : 75.25	Mean :25.14
	3rd Qu.:71.00	3rd Qu.:186.0	3rd Qu.: 80.00	3rd Qu.:28.00
	Max. :94.00	Max. :207.0	Max. :110.00	Max. :47.00

10. Archivo final

Una vez realizado el preprocesamiento sobre el archivo, copie el resultado de los datos en un archivo llamado "fifa_clean.csv".

```
write.csv(ds, "fifa_clean.csv", row.names = FALSE)
```

11. Avaluación de la actividad

- Apartados 1, 2, 3 y 4 (30 %)
- Apartado 5 (10 %)
- Apartado 6 (10 %)
- Apartado 7 (20 %)
- Apartados 8 y 9 (20 %)
- Calidad del informe dinámico (calidad del código, formato y estructura del documento, concisión y precisión en las respuestas) (10 %)