Práctica 2: Análisis LFP

Fernando de Castilla

1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende resolver?

Siguiendo la línea de la práctica anterior, el dataset contiene estadísticas de todos los jugadores de la Primera División de la Liga de Nacional de Fútbol Profesional (LFP). En concreto, para cada jugador, se dispone de su nombre, equipo y una serie de variables numéricas: minutos jugados, tarjetas amarillas recibidas, goles anotados, fueras de juego cometidos, faltas recibidas, faltas cometidas, centros realizados, córners lanzados, entradas realizadas (existosas y fracasadas), duelos disputados, duelos cuerpo a cuerpo (exitosos, fracasados y totales), duelos aéreos (exitosos, fracasados y totales), pases (cortos, largos, al hueco y totales), tiros, tiros a puerta, asistencias de gol, regates (exitosos y fracasados) y recuperaciones de balón.

El dataset es importante porque contiene datos acumulados de los jugadores tras disputarse las 38 jornadas de la temporada 2017/2018. Estos datos (que bien pueden ser integrados con otras fuentes externas) son de gran utilidad para que los clubes de fútbol puedan valorar el rendimiento de sus jugadores, a la vez que puedan compararlo con el resto de jugadores de la misma categoría. Se pretende analizar y dar respuesta a las siguientes dos preguntas:

- ¿Es posible estimar los goles anotados por un futbolista a partir de otros parámetros medidos sobre el futbolista?
- ¿Qué parámetros clave debe cumplir un futbolista para ser importante dentro de cualquier equipo en Primera División?

2. Integración y selección de los datos de interés a analizar

La web de la LFP ofrece estadísticas de los jugadores agrupadas por tipo (generales, disciplinarias, ofensivas, defensivas y de eficiencia). Para responder a las preguntas objeto de la práctica con la mayor certidumbre posible, es necesario integrar todos los tipos diferentes de estadísticas disponibles. Gracias a que el script desarrollado en la práctica anterior (para hacer scraping de la web de la LFP) permite configurar ciertos parámetros para obtener diferentes descargas de datos, vamos a aprovechar para extraer los datos de todos los jugadores, incluyendo todas las jornadas, y de los 5 tipos de estadísticas disponibles (5 ejecuciones diferentes del script para obtener 5 ficheros csv).

Tras este hito, seleccionaremos los datos que consideramos relevantes para nuestro cometido, además de renombrarlos. Posteriormente, debemos verificar que todos los datos son correctos, consistentes y coherentes. Por ello, se verificarán una serie de condiciones para validar los datos.

Comenzamos con el dataset de estadísticas clásicas:

- Cabe destacar la corrección de la variable ‘Minutos’, que contiene el valor por duplicado en formato texto, separado por doble cero (por ejemplo, el valor ‘97900979’ se corregirá por 979; y el valor ‘1.234001.234’ se corregirá por 1234).
- Seleccionamos 5 variables (dos de ellas son identificadoras del jugador).

```
clasicas <- read.csv("csv/LFP_clasicas.csv", sep = ";", fileEncoding = "UTF-8", colClasses =
c("Min."="character"))

clasicas$Jug. <- NULL
clasicas$X. <- NULL
clasicas$Ent. <- NULL
clasicas$X..1 <- NULL
clasicas$Tit. <- NULL
clasicas$X..2 <- NULL
clasicas$Sust. <- NULL
clasicas$X..3 <- NULL
clasicas$Dob. <- NULL
clasicas$Roj. <- NULL
clasicas$Pen. <- NULL
clasicas$G.P.P. <- NULL
clasicas$Enc. <- NULL

names(clasicas)[names(clasicas) == "Min."] <- "Minutos"
names(clasicas)[names(clasicas) == "Am."] <- "T.Amarillas"
names(clasicas)[names(clasicas) == "Gol"] <- "Goles"

clasicas$Minutos <- gsub("\\.", "", clasicas$Minutos)
clasicas$Minutos <- substring(clasicas$Minutos, nchar(clasicas$Minutos) / 2 + 2)
clasicas$Minutos <- as.integer(clasicas$Minutos)

str(clasicas)
```

```
## 'data.frame':    487 obs. of  5 variables:
##  $ Nombre     : Factor w/ 482 levels "Aarón Martín",..: 1 2 4 5 6 6 7 8 3 9 ...
##  $ Equipo     : Factor w/ 20 levels "ALA","ATH","ATM",..: 9 16 11 17 7 15 20 2 5 17 ...
##  $ Minutos    : int  2815 757 1833 1579 2346 2341 233 2149 2700 223 ...
##  $ T.Amarillas: int  6 0 10 4 1 9 1 5 4 0 ...
##  $ Goles      : int  0 2 2 3 9 3 0 9 0 0 ...
```

En cuanto al dataset de estadísticas disciplinarias, seleccionamos 3 variables (más las dos identificadoras).

```
disciplina <- read.csv("csv/LFP_disciplina.csv", sep = ";", fileEncoding = "UTF-8")

disciplina[9:12] <- list(NULL)
disciplina[3:5] <- list(NULL)

names(disciplina)[names(disciplina) == "F..J."] <- "Fueras.De.Juego"
names(disciplina)[names(disciplina) == "Faltas.R."] <- "Faltas.Recibidas"
names(disciplina)[names(disciplina) == "Faltas.C."] <- "Faltas.Cometidas"

str(disciplina)
```

```
## 'data.frame':    487 obs. of  5 variables:
##  $ Nombre          : Factor w/ 482 levels "Aarón Martín",..: 1 2 4 5 6 6 7 8 3 9 ...
##  $ Equipo          : Factor w/ 20 levels "ALA","ATH","ATM",..: 9 16 11 17 7 15 20 2 5 17
...
##  $ Fueras.De.Juego : int  0 1 6 6 12 2 0 35 0 1 ...
##  $ Faltas.Recibidas: int  20 6 15 60 29 31 0 39 3 4 ...
##  $ Faltas.Cometidas: int  29 8 39 32 16 39 6 42 1 1 ...
```

En cuanto al dataset de estadísticas de eficiencia:

- Seleccionamos 10 variables (más las dos identificadoras).
- Transformamos en valores numéricos aquellas variables con valores superiores a 999, que incluyen el símbolo '.' como separador de las unidades de millar.
- Plantemos 2 validaciones (Duelos = Duelos.Cuerpo + Duelos.Aire; Pases = Pases.Cortos + Pases.Largos + Pases.Hueco), que se cumplen sin necesidad de aplicar corrección alguna.

```
eficiencia <- read.csv("csv/LFP_eficiencia.csv", sep = ";", fileEncoding = "UTF-8",
                       colClasses = c("Pas."="character", "P.C."="character"))

eficiencia[13:19] <- list(NULL)

names(eficiencia)[names(eficiencia) == "Cen."] <- "Centros"
names(eficiencia)[names(eficiencia) == "Cor."] <- "Corners"
names(eficiencia)[names(eficiencia) == "Ent."] <- "Entradas"
names(eficiencia)[names(eficiencia) == "Due."] <- "Duelos"
names(eficiencia)[names(eficiencia) == "D.C."] <- "Duelos.Cuerpo"
names(eficiencia)[names(eficiencia) == "D.A."] <- "Duelos.Aire"
names(eficiencia)[names(eficiencia) == "Pas."] <- "Pases"
names(eficiencia)[names(eficiencia) == "P.C."] <- "Pases.Cortos"
names(eficiencia)[names(eficiencia) == "P.L."] <- "Pases.Largos"
names(eficiencia)[names(eficiencia) == "P.H."] <- "Pases.Hueco"

eficiencia$Pases <- as.integer(gsub("\\.", "", eficiencia$Pases))
eficiencia$Pases.Cortos <- as.integer(gsub("\\.", "", eficiencia$Pases.Cortos))

eficiencia.val1 <- eficiencia$Duelos == eficiencia$Duelos.Cuerpo + eficiencia$Duelos.Aire
eficiencia.val2 <- eficiencia$Pases == eficiencia$Pases.Cortos + eficiencia$Pases.Largos + ef
iciencia$Pases.Hueco
summary(data.frame(eficiencia.val1, eficiencia.val2))
```

```
##  eficiencia.val1 eficiencia.val2
##  Mode:logical    Mode:logical
##  TRUE:487        TRUE:487
```

```
str(eficiencia)
```

```
## 'data.frame':    487 obs. of  12 variables:
##  $ Nombre       : Factor w/ 482 levels "Aarón Martín",..: 1 2 4 5 6 6 7 8 3 9 ...
##  $ Equipo       : Factor w/ 20 levels "ALA","ATH","ATM",..: 9 16 11 17 7 15 20 2 5 17 ...
##  $ Centros      : int  131 30 132 29 29 14 2 19 0 1 ...
##  $ Corners      : int  17 0 3 34 1 2 0 0 0 0 ...
##  $ Entradas     : int  56 21 17 16 24 40 6 11 1 0 ...
##  $ Duelos       : int  269 112 196 320 364 348 28 335 13 31 ...
##  $ Duelos.Cuerpo: int  208 93 140 301 207 188 25 137 8 13 ...
##  $ Duelos.Aire  : int  61 19 56 19 157 160 3 198 5 18 ...
##  $ Pases        : int  975 407 720 474 860 1048 89 490 1124 58 ...
##  $ Pases.Cortos : int  880 385 624 435 823 929 80 472 534 55 ...
##  $ Pases.Largos : int  93 22 95 37 31 117 9 16 590 2 ...
##  $ Pases.Hueco  : int  2 0 1 2 6 2 0 2 0 1 ...
```

En cuanto al dataset de estadísticas ofensivas:

- Seleccionamos 5 variables (más las dos identificadoras).
- Plantemos 1 validación (Tiros.Puerta <= Tiros), que se cumple sin necesidad de aplicar corrección alguna.

```
ofensivas <- read.csv("csv/LFP_ofensivas.csv", sep = ";", fileEncoding = "UTF-8")

ofensivas[8:16] <- list(NULL)

names(ofensivas)[names(ofensivas) == "Tiros.P."] <- "Tiros.Puerta"
names(ofensivas)[names(ofensivas) == "Asis."] <- "Asistencias"
names(ofensivas)[names(ofensivas) == "Reg..E."] <- "Regates.Exito"
names(ofensivas)[names(ofensivas) == "Reg..F."] <- "Regates.Fracaso"

ofensivas.val1 <- ofensivas$Tiros.Puerta <= ofensivas$Tiros
summary(data.frame(ofensivas.val1))
```

```
##  ofensivas.val1
##  Mode:logical
##  TRUE:487
```

```
str(ofensivas)
```

```
## 'data.frame':    487 obs. of  7 variables:
##  $ Nombre         : Factor w/ 482 levels "Aarón Martín",..: 1 2 4 5 6 6 7 8 3 9 ...
##  $ Equipo         : Factor w/ 20 levels "ALA","ATH","ATM",..: 9 16 11 17 7 15 20 2 5 17
...
##  $ Tiros          : int  4 3 25 40 47 26 0 52 0 5 ...
##  $ Tiros.Puerta   : int  2 3 14 14 20 12 0 28 0 2 ...
##  $ Asistencias    : int  0 0 0 5 6 0 0 1 2 0 ...
##  $ Regates.Exito  : int  35 22 27 67 38 8 2 19 5 3 ...
##  $ Regates.Fracaso: int  17 9 15 57 31 5 3 5 0 0 ...
```

En cuanto al dataset de estadísticas defensivas, seleccionamos 7 variables (más las dos identificadoras).

```
defensivas <- read.csv("csv/LFP_defensivas.csv", sep = ";", fileEncoding = "UTF-8")

defensivas[9] <- NULL
defensivas[6] <- NULL
defensivas[3:4] <- list(NULL)

names(defensivas)[names(defensivas) == "Rec."] <- "Recuperaciones"
names(defensivas)[names(defensivas) == "Ent..E."] <- "Entradas.Exito"
names(defensivas)[names(defensivas) == "Ent..F."] <- "Entradas.Fracaso"
names(defensivas)[names(defensivas) == "Due..E."] <- "Duelos.Cuerpo.Exito"
names(defensivas)[names(defensivas) == "Due..F."] <- "Duelos.Cuerpo.Fracaso"
names(defensivas)[names(defensivas) == "Aer..E."] <- "Duelos.Aire.Exito"
names(defensivas)[names(defensivas) == "Aer..F."] <- "Duelos.Aire.Fracaso"

str(defensivas)
```

```
## 'data.frame':    487 obs. of  9 variables:
##  $ Nombre              : Factor w/ 482 levels "Aarón Martín",..: 1 2 4 5 6 6 7 8 3 9 ...
##  $ Equipo              : Factor w/ 20 levels "ALA","ATH","ATM",..: 9 16 11 17 7 15 20 2 5
17 ...
##  $ Recuperaciones      : int  151 50 97 74 116 147 9 45 242 6 ...
##  $ Entradas.Exito      : int  36 15 15 11 14 25 4 8 0 0 ...
##  $ Entradas.Fracaso    : int  20 6 2 5 10 15 2 3 1 0 ...
##  $ Duelos.Cuerpo.Exito  : int  109 49 58 143 86 75 8 60 8 7 ...
##  $ Duelos.Cuerpo.Fracaso: int  99 44 82 158 121 113 17 77 0 6 ...
##  $ Duelos.Aire.Exito    : int  20 9 38 6 80 67 2 108 4 8 ...
##  $ Duelos.Aire.Fracaso  : int  41 10 18 13 77 93 1 90 1 10 ...
```

En este punto, procedemos con la integración definitiva de los 5 data frames anteriormente generados, identificando a cada jugador (registro) con los campos ‘Nombre’ y ‘Equipo’, comunes a todos ellos.

```
dataset_aux1 <- merge(clasicas, disciplina, by = c("Nombre", "Equipo"))
dataset_aux2 <- merge(dataset_aux1, eficiencia, by = c("Nombre", "Equipo"))
dataset_aux3 <- merge(dataset_aux2, ofensivas, by = c("Nombre", "Equipo"))
datasetLFP <- merge(dataset_aux3, defensivas, by = c("Nombre", "Equipo"))
```

Una vez consolidado el dataset definitivo ‘datasetLFP’, finalmente, validamos 4 condiciones generales:

- Goles <= Tiros.Puerta
- Entradas = Entradas.Exito + Entradas.Fracaso
- Duelos.Cuerpo = Duelos.Cuerpo.Exito + Duelos.Cuerpo.Fracaso
- Duelos.Aire = Duelos.Aire.Exito + Duelos.Aire.Fracaso

Tras descubrir una minoría de registros con errores, éstos quedan solventados.

```
datasetLFP.val1 <- datasetLFP$Goles <= datasetLFP$Tiros.Puerta
datasetLFP.val2 <- datasetLFP$Entradas == datasetLFP$Entradas.Exito + datasetLFP$Entradas.Fra
caso
datasetLFP.val3 <- datasetLFP$Duelos.Cuerpo == datasetLFP$Duelos.Cuerpo.Exito + datasetLFP$Du
elos.Cuerpo.Fracaso
datasetLFP.val4 <- datasetLFP$Duelos.Aire == datasetLFP$Duelos.Aire.Exito + datasetLFP$Duelo
s.Aire.Fracaso
summary(data.frame(datasetLFP.val1, datasetLFP.val2, datasetLFP.val3, datasetLFP.val4))
```

```
##  datasetLFP.val1 datasetLFP.val2 datasetLFP.val3 datasetLFP.val4
##  Mode:logical    Mode :logical   Mode:logical    Mode:logical
##  TRUE:487        FALSE:6         TRUE:487        TRUE:487
##                  TRUE :481
```

```
datasetLFP$Entradas <- ifelse(datasetLFP.val2, datasetLFP$Entradas,
                              datasetLFP$Entradas.Exito + datasetLFP$Entradas.Fracaso)
datasetLFP.val2 <- datasetLFP$Entradas == datasetLFP$Entradas.Exito + datasetLFP$Entradas.Fra
caso
summary(data.frame(datasetLFP.val2))
```

```
##  datasetLFP.val2
##  Mode:logical
##  TRUE:487
```

3. Limpieza de los datos. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

En primer lugar, confirmamos que los datos contienen ceros. En nuestro estudio, predominan las variables numéricas; por lo que el valor cero es totalmente válido y normal. Esto es, no debe ser tratado de ninguna manera.

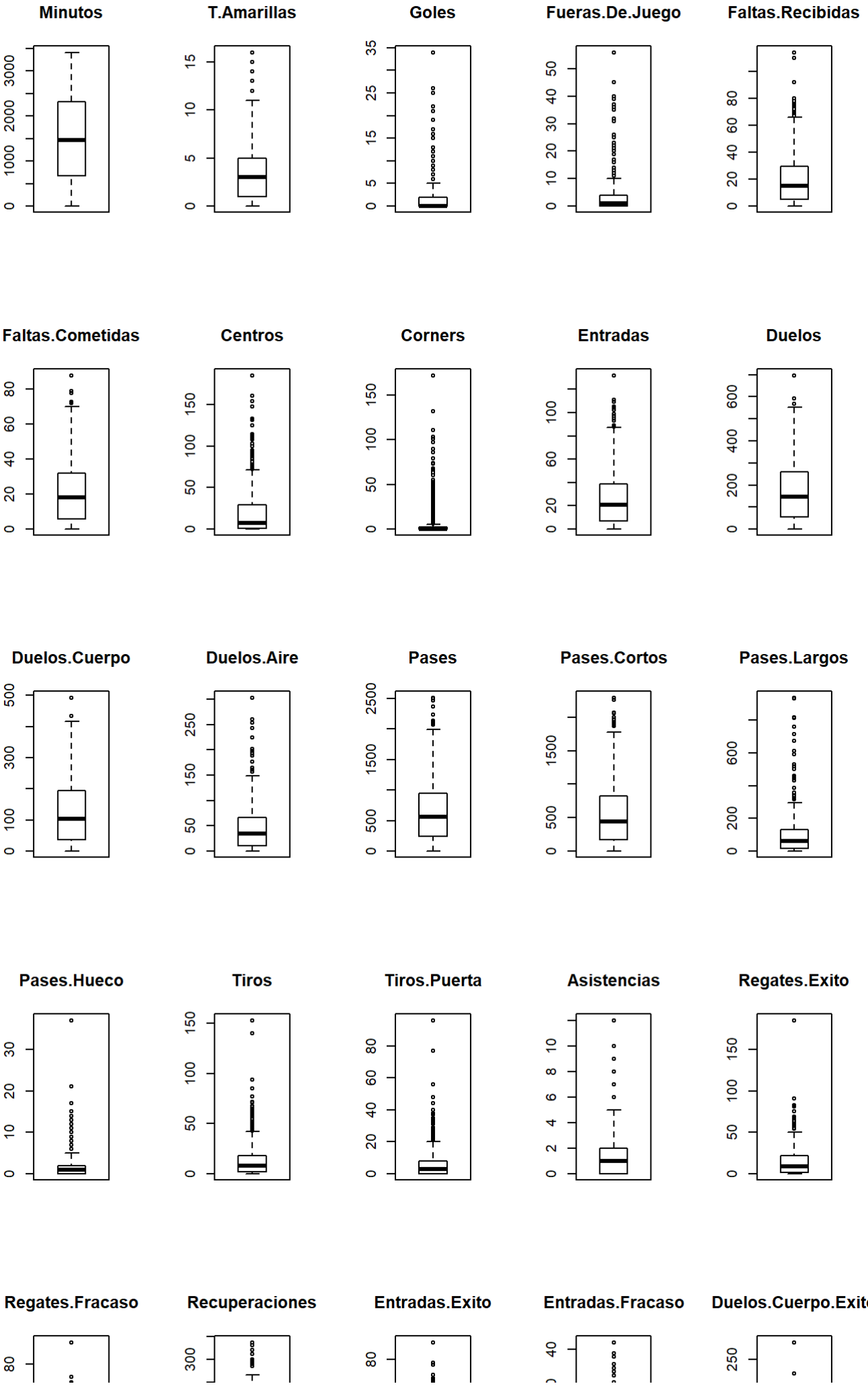
En cuanto a la detección de elementos vacíos, el siguiente código sirve para identificarlos:

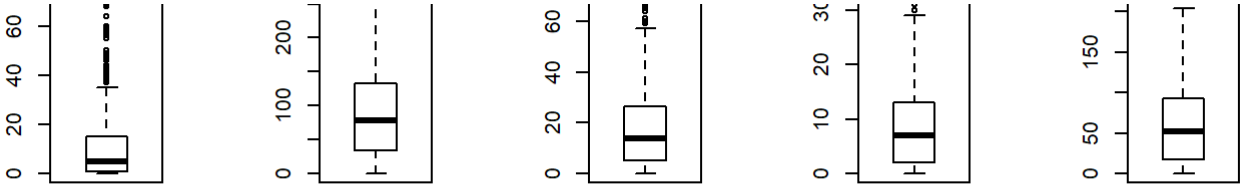
```
for(col in names(datasetLFP)){
  vectorNA <- table(is.na(datasetLFP[, col]))
  if(length(vectorNA) > 1)
    cat(col, ":", vectorNA[2], "\n")
}
```

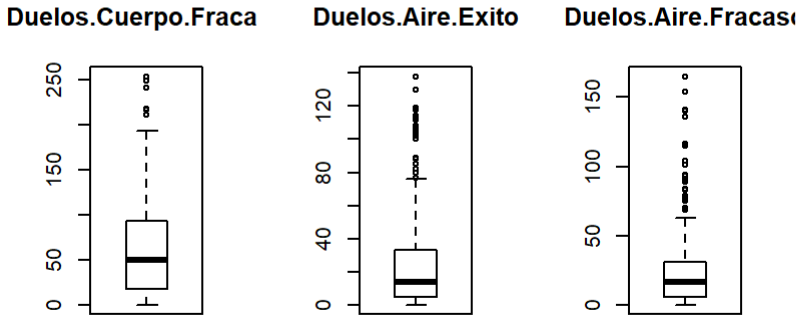
Puesto que el código anterior no imprime nada, concluimos que no existen elementos vacíos ni nulos.

Centrándonos en el análisis de valores extremos, proponemos una representación visual de diagramas de cajas por cada variable del dataset, para valorar si existen distribuciones con valores extremos. En tales casos, se valorará (mediante la consulta de datos históricos de otras temporadas) si dichos valores son posibles (reales) o no.

```
par(mfrow = c(2, 5))
for(col in names(datasetLFP))
  if(col != "Nombre" & col != "Equipo")
    boxplot(datasetLFP[, col], main = col)
```


Minutos
0 1000 2000 3000
T.Amarillas
0 5 10 15
Goles
0 5 15 25 35
Fueras.De.Juego
0 10 20 30 40 50
Faltas.Recibidas
0 20 40 60 80
Faltas.Cometidas
0 20 40 60 80
Centros
0 50 100 150
Corners
0 50 100 150
Entradas
0 20 60 100
Duelos
0 200 400 600
Duelos.Cuerpo
0 100 300 500
Duelos.Aire
0 50 150 250
Pases
0 500 1500 2500
Pases.Cortos
0 500 1500
Pases.Largos
0 200 600
Pases.Hueco
0 10 20 30
Tiros
0 50 100 150
Tiros.Puerta
0 20 40 60 80
Asistencias
0 2 4 6 8 10
Regates.Exito
0 50 100 150
Regates.Fracaso
80
Recuperaciones
300
Entradas.Exito
80
Entradas.Fracaso
40
Duelos.Cuerpo.Exit
250





En primer lugar, se aprecian ciertas similitudes entre distribuciones, lo que puede dar una idea de las posibles correlaciones existentes entre variables. Volviendo al tratamiento de valores extremos, tras apreciar que no se trata de valores numéricos disparatados, asumimos que son veraces y válidos para nuestros análisis posteriores. No obstante, dado el caso en que sea necesario plantear, por ejemplo, un modelo de regresión lineal (mínimos cuadrados), se valorará gráficamente si dichos valores extremos estuvieran perturbando la recta de regresión calculada; en cuyo caso, procederíamos a tratar dicha anomalía.

4. Análisis de los datos. Selección de los datos, planificación, comprobación de la normalidad, comprobación de la homogeneidad de la varianza, aplicación de pruebas estadísticas (contrastes de hipótesis, correlaciones, regresiones, etc.)

Puesto que el objetivo de esta práctica es reponder a dos preguntas diferentes, vamos a plantear dos análisis independientes para dar respuesta a cada una de ellas.

4.a. ¿Es posible estimar los goles anotados por un futbolista a partir de otros parámetros medidos sobre el futbolista?

Para dar respuesta a esta pregunta, vamos a exponer la planificación de los hitos que nos hemos marcado:

- Acometeremos el filtrado de aquellos jugadores que no acumulen una cantidad mínima de estadísticas que sea fiable para los posteriores estudios y análisis. En un paso posterior, mencionaremos la creación de variables ratio, por lo que este filtrado de jugadores es crucial para garantizar la fiabilidad de dichos indicadores.
- Comprobaremos si las variables estadísticas de las que disponemos siguen una distribución normal (tanto a través del test Shapiro-Wilk, como visualmente mediante gráficos Cuantil-Cuantil). El hecho de no poder asumir la normalidad limita la potencia en los test de hipótesis paramétricos (precisión de los

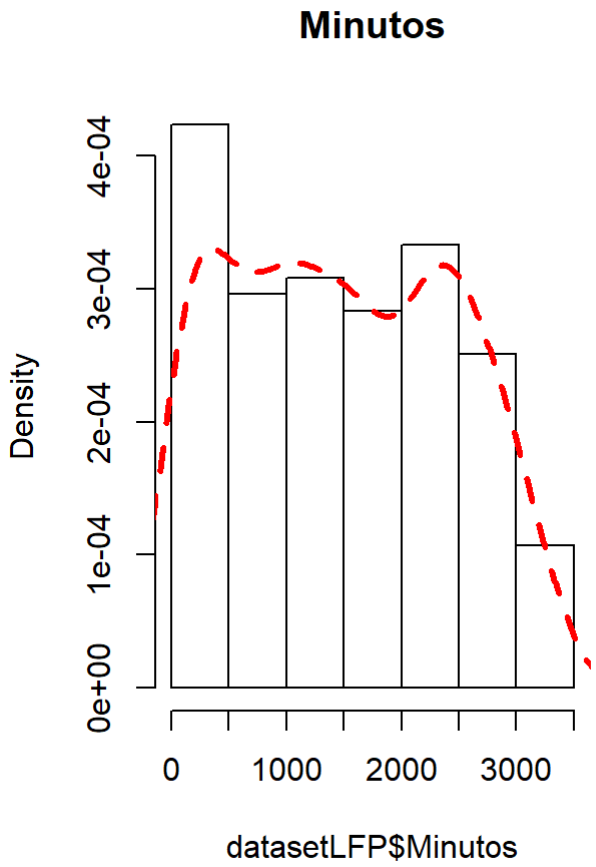
intervalos de confianza de los parámetros del modelo y los contrastes de significancia) y en los modelos de regresión (eficiencia de los estimadores mínimo-cuadráticos, al ser de mínima varianza), ademas de condicionar el coeficiente de correlación más idóneo a utilizar.

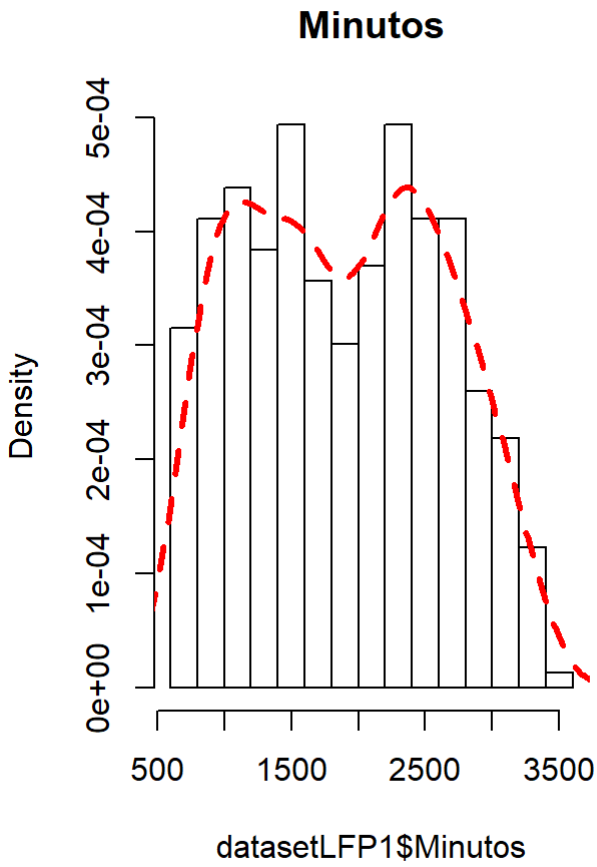
- Crearemos nuevas variables agregadas. La mayoría serán ratios entre variables originales, cuyo objetivo no es otro que el de optar a incrementar la precisión del futuro modelo de regresión lineal múltiple a construir.
- Una vez disponemos de todas las posibles variables candidatas para el modelo de regresión, acometeremos el estudio de la correlación de la variable a predecir (Goles) con cada una de las candidatas.
- A raíz de las conclusiones del estudio de correlación anterior, construiremos una serie de modelos de regresión permutando las variables con mayor grado de correlación, evitando tomar simultáneamente en un mismo modelo las que puedan guardar correlación entre ellas mismas.
- Finalmente, seleccionaremos el modelo que más se ajuste a los datos, graficaremos su recta de regresión dentro del gráfico de dispersión y plantearemos algunos ejemplos para que el modelo arroje una predicción.

Comenzamos decidiendo cómo acometer el filtrado de jugadores con estadísticas poco sólidas. Para ello, tomamos la variable ‘Minutos’ jugados como clave para determinarlo (a más minutos jugados en liga, más fiables deben ser las estadísticas de los jugadores). Si graficamos el histograma junto con la gráfica de densidad de dicha variable, podemos apreciar que, a partir del primer cuartil de jugadores, la tendencia se estabiliza prácticamente en el máximo de densidad. Parece razonable establecer el umbral de filtrado en este punto.

Comparando con el gráfico del nuevo dataset filtrado, se aprecia visualmente una mejora de la simetría en la distribución.

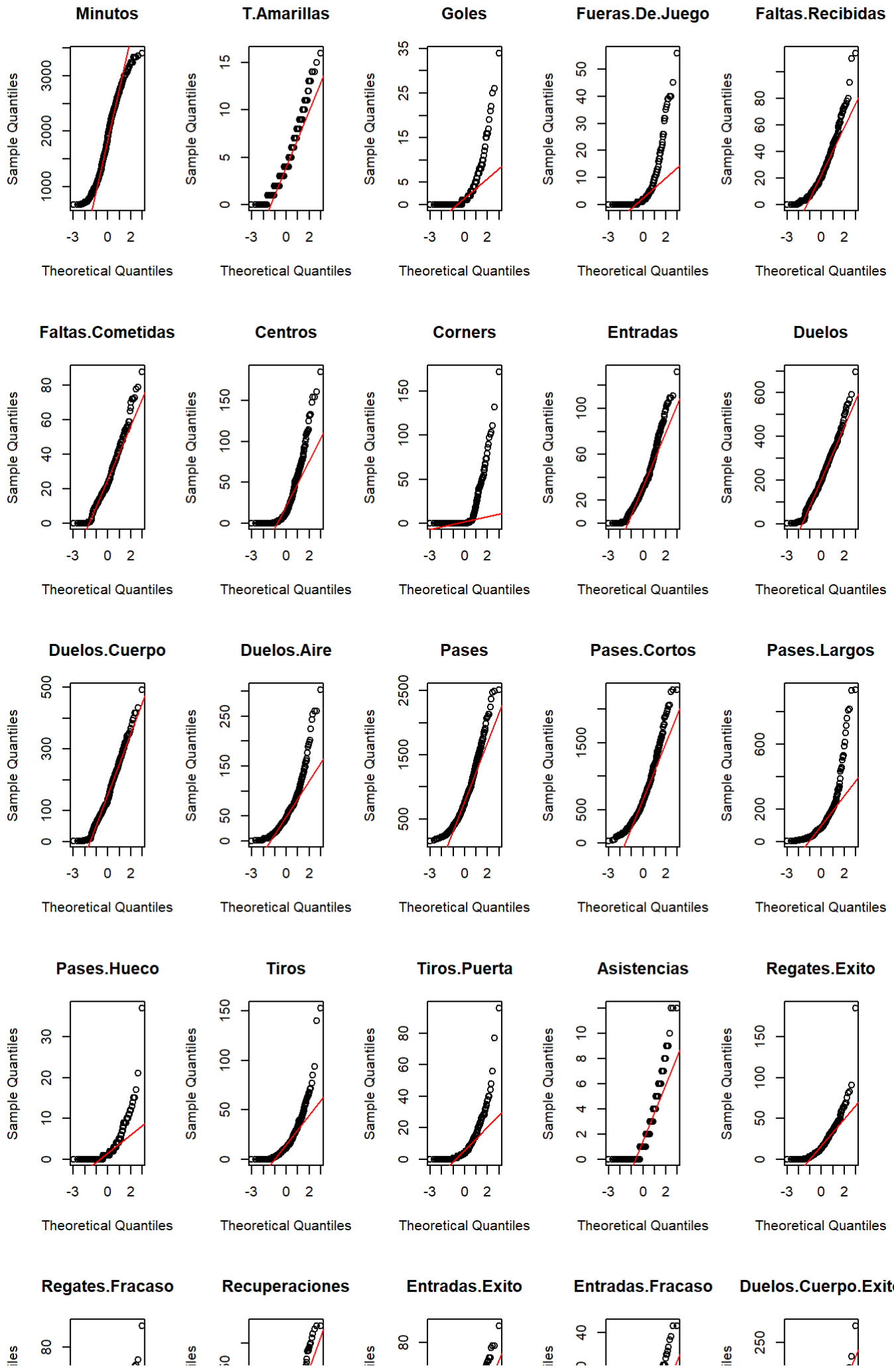
```
# Umbral establecido en el primer cuartil de la variable 'Minutos'
umbral1 <- 670
datasetLFP1 <- datasetLFP[datasetLFP$Minutos > umbral1, ]

par(mfrow = c(1, 2))
hist(datasetLFP$Minutos, freq = FALSE, main = "Minutos")
lines(density(datasetLFP$Minutos), col = "red", lty = 2, lwd = 3)
hist(datasetLFP1$Minutos, freq = FALSE, main = "Minutos")
lines(density(datasetLFP1$Minutos), col = "red", lty = 2, lwd = 3)
```

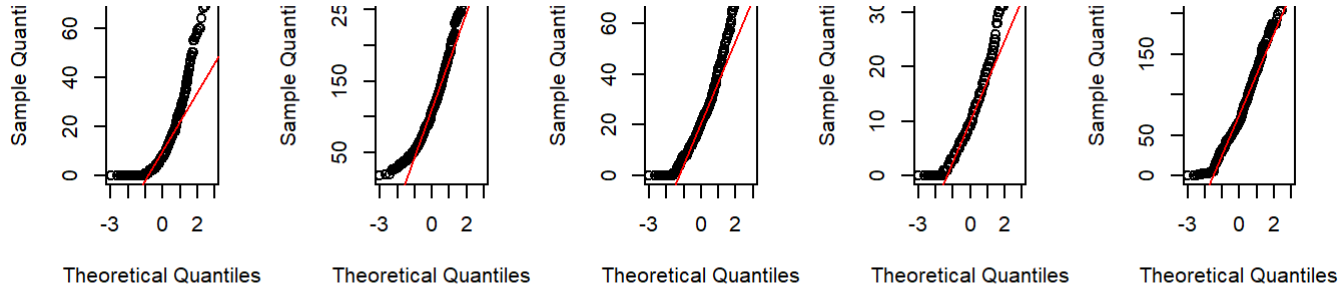


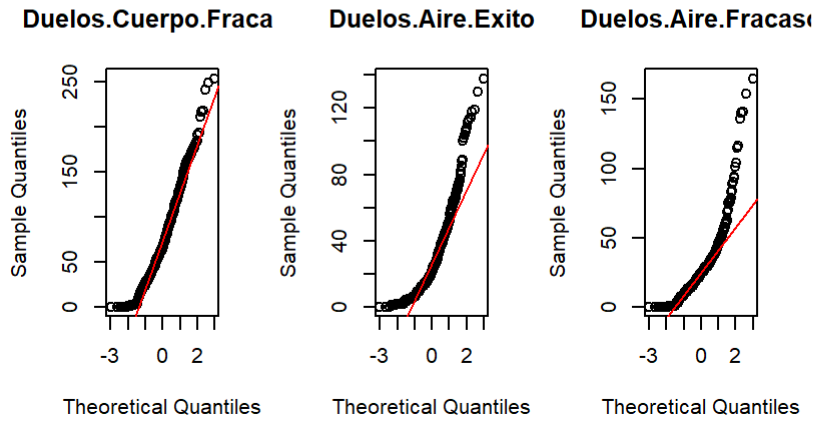


Abordemos el estudio de la normalidad de las variables del conjunto de datos. Aquellas variables de las que no podamos rechazar su normalidad podrán ser de mayor utilidad a la hora de construir muestro modelo. Por un lado, con los gráficos Cuantil-Cuantil se puede apreciar visualmente si las distribuciones de cada variable guardan normalidad (las muestras deben estar dispersas siguiendo una línea recta).

```
# QQ-plots
par(mfrow = c(2, 5))
for(col in names(datasetLFP1)){
  if(col != "Nombre" & col != "Equipo"){
    qqnorm(datasetLFP1[, col], main = col)
    qqline(datasetLFP1[, col], col = "red")
  }
}
```


Minutos
T.Amarillas
Goles
Fueras.De.Juego
Faltas.Recibidas
Faltas.Cometidas
Centros
Corners
Entradas
Duelos
Duelos.Cuerpo
Duelos.Aire
Pases
Pases.Cortos
Pases.Largos
Pases.Hueco
Tiros
Tiros.Puerta
Asistencias
Regates.Exito
Regates.Fracaso
Recuperaciones
Entradas.Exito
Entradas.Fracaso
Duelos.Cuerpo.Exit
Sample Quantiles
Theoretical Quantiles





Por otro lado, el test de Shapiro-Wilk confirmará aquellas variables que no siguen una distribución normal (valor p menor que un nivel de significación igual a 0,05; lo que supone un grado de confianza mayor a un 95%).

```
# Tests Shapiro-Wilk de normalidad
for(col in names(datasetLFP1)){
  if(col == "Nombre")
    cat ("Variables NO normales:\n")
  if(is.numeric(datasetLFP1[, col])){
    valor_p <- shapiro.test(datasetLFP1[, col])$p.value
    if(valor_p < 0.05)
      cat(col, "| ")
  }
}
```

```
## Variables NO normales:
## Minutos | T.Amarillas | Goles | Fueras.De.Juego | Faltas.Recibidas | Faltas.Cometidas | Ce
ntros | Corners | Entradas | Duelos | Duelos.Cuerpo | Duelos.Aire | Pases | Pases.Cortos | Pa
ses.Largos | Pases.Hueco | Tiros | Tiros.Puerta | Asistencias | Regates.Exito | Regates.Fraca
so | Recuperaciones | Entradas.Exito | Entradas.Fracaso | Duelos.Cuerpo.Exito | Duelos.Cuerp
o.Fracaso | Duelos.Aire.Exito | Duelos.Aire.Fracaso |
```

El estudio arroja por pantalla todas las variables del dataset, por lo que no podemos suponer normalidad en ninguna de ellas. A continuación, creamos 3 nuevas variables calculadas en el dataset, tal y como mencionamos anteriormente al enumerar los hitos de la planificación propuesta (2 ratios y una acumulada).

```
datasetLFP1$Precision.Tiros <- ifelse(datasetLFP1$Tiros == 0, 0, datasetLFP1$Tiros.Puerta / d
atasetLFP1$Tiros)
datasetLFP1$Regates <- datasetLFP1$Regates.Exito + datasetLFP1$Regates.Fracaso
datasetLFP1$Precision.Regates <- ifelse(datasetLFP1$Regates == 0, 0,
                                datasetLFP1$Regates.Exito / datasetLFP1$Regates)
```

Una vez cerrado un conjunto extenso y variado de variables, vamos a calcular el coeficiente de correlación de cada una de ellas con respecto a la variable 'Goles'. La tabla resultado estará ordenada ascendentemente por el valor p de cada correlación (de mayor a menor correlación con los goles anotados).

Dado que los datos no son normales, planteamos usar el coeficiente de Spearman o el de Kendall. Debemos recurrir, nuevamente, a la naturaleza de nuestros datos para decidir el coeficiente más adecuado. Debido a que nuestras variables, en general, poseen un rango de valores enteros pequeño, sucede que existen muchos valores idénticos. Por ello, el coeficiente de Kendall será la opción ideal. Además, en comparación con el coeficiente de Spearman, se trata de un coeficiente con un menor error estándar y una menor varianza asintótica. Su inconveniente a destacar es el coste computacional (n^2 frente a n*log n de Spearman), aunque nuestra muestra no es grande y no se apreciará una diferencia significativa.

```
valores_p <- vector()
cols <- vector()
for(col in names(datasetLFP1)){
  if(is.numeric(datasetLFP1[, col])){
    valor_p <- cor.test(datasetLFP1[, col], datasetLFP1$Goles, method = "kendall")$p.value
    valores_p = c(valores_p, valor_p)
    cols = c(cols, col)
  }
}

df1 <- data.frame(cols, valores_p)
df1 <- df1[order(valores_p),]
rownames(df1) <- NULL
head(df1, n = 10)
```

```
##                    cols     valores_p
## 1                 Goles 4.968091e-127
## 2          Tiros.Puerta  8.624094e-81
## 3                 Tiros  9.721667e-69
## 4        Fueras.De.Juego  1.703736e-39
## 5        Precision.Tiros  8.134880e-31
## 6  Duelos.Cuerpo.Fracaso  3.895389e-27
## 7        Regates.Fracaso  3.319718e-25
## 8                Duelos  5.395013e-25
## 9               Regates  1.796804e-24
## 10      Faltas.Recibidas  3.269077e-22
```

Una vez conocemos las variables más determinantes para estimar el número de goles anotados, vamos a plantear una batería de modelos de regresión lineal múltiple. Como adelantábamos antes, debemos aplicar un cierto criterio adicional a la hora de escoger las variables a utilizar. Por ejemplo, aquellas variables con un rango de valores más amplio dotan de mayor fiabilidad sobre la precisión de los modelos.

Para comparar la calidad de los modelos, tomaremos aquel con el mayor coeficiente R^2 ajustado (indicador útil para comparar varios modelos). El modelo ganador (el tercero) contiene las siguientes variables: 'Tiros.Puerta', 'Fueras.De.Juego' y 'Duelos.Cuerpo.Fracaso'. En el detalle del modelo, podemos constatar que la variable más determinante es 'Tiros.Puerta', dado que posee el mayor coeficiente dentro de la recta de regresión (0.38). Este número son los goles que debe anotar un jugador por cada tiro a puerta; es decir, aproximadamente 1 gol por cada 3 tiros a puerta.

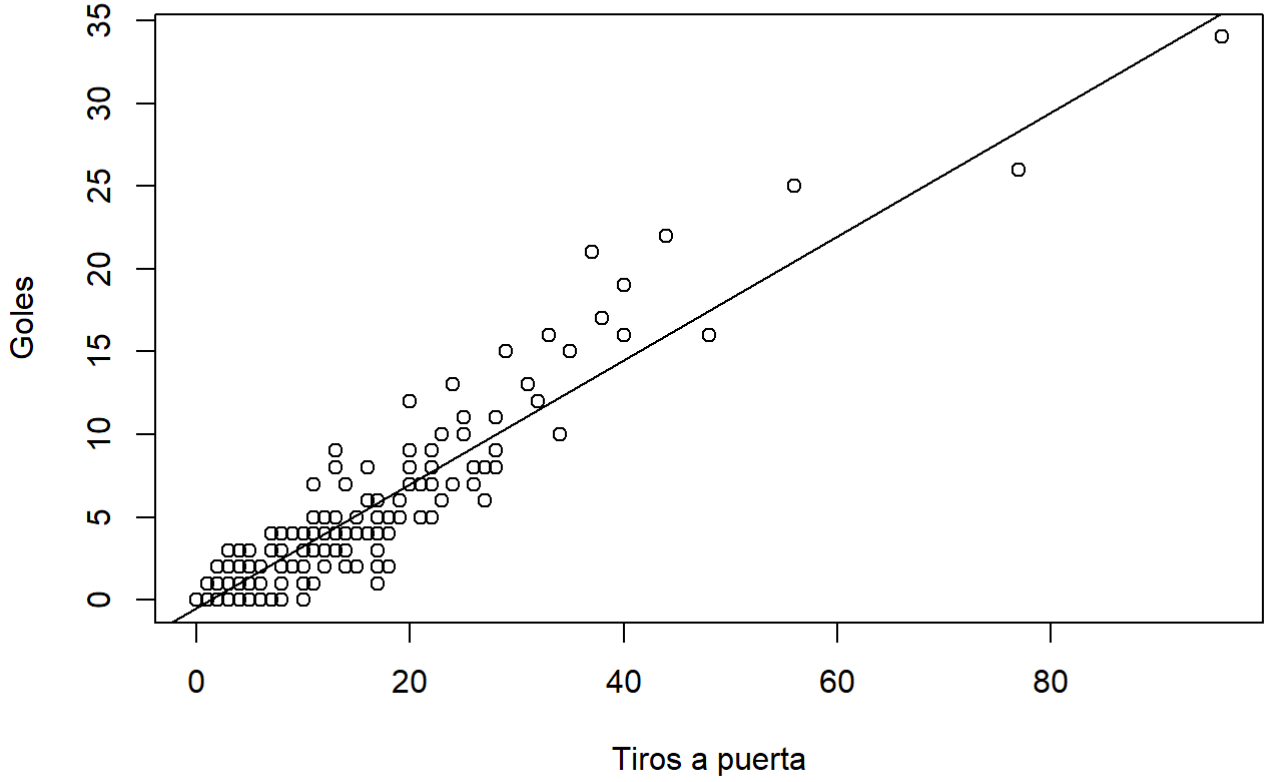
```
goles.modelo_1 <- lm(Goles ~ Tiros.Puerta, data = datasetLFP1)
goles.modelo_2 <- lm(Goles ~ Tiros.Puerta + Fueras.De.Juego, data = datasetLFP1)
goles.modelo_3 <- lm(Goles ~ Tiros.Puerta + Fueras.De.Juego + Duelos.Cuerpo.Fracaso, data = d
atasetLFP1)
goles.modelo_4 <- lm(Goles ~ Tiros.Puerta + Duelos.Cuerpo.Fracaso, data = datasetLFP1)

modelos <- c(summary(goles.modelo_1)$adj.r.squared, summary(goles.modelo_2)$adj.r.squared,
             summary(goles.modelo_3)$adj.r.squared, summary(goles.modelo_4)$adj.r.squared)

for (i in 1:length(modelos))
  cat("Coeficiente R^2 ajustado del modelo", i, ":", modelos[i], "\n")
```

```
## Coeficiente R^2 ajustado del modelo 1 : 0.8880122
## Coeficiente R^2 ajustado del modelo 2 : 0.8889666
## Coeficiente R^2 ajustado del modelo 3 : 0.8960512
## Coeficiente R^2 ajustado del modelo 4 : 0.8950505
```

```
summary(goles.modelo_3)
```

```
##
## Call:
## lm(formula = Goles ~ Tiros.Puerta + Fueras.De.Juego + Duelos.Cuerpo.Fracaso,
##     data = datasetLFP1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.1616 -0.6081  0.0413  0.5040  7.5244
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -0.032583   0.131110  -0.249   0.8039
## Tiros.Puerta           0.381440   0.011020  34.614  < 2e-16 ***
## Fueras.De.Juego        0.028497   0.013456   2.118   0.0349 *
## Duelos.Cuerpo.Fracaso -0.008676   0.001712  -5.067 6.47e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.398 on 361 degrees of freedom
## Multiple R-squared:  0.8969, Adjusted R-squared:  0.8961
## F-statistic:  1047 on 3 and 361 DF,  p-value: < 2.2e-16
```

```
# Gráfica
plot(datasetLFP1$Tiros.Puerta, datasetLFP1$Goles, xlab = "Tiros a puerta", ylab = "Goles")
abline(goles.modelo_1)
```




```
# Línea comentada para etiquetar cada muestra del gráfico de dispersión con el nombre del jug
ador
#text(datasetLFP1$Tiros.Puerta, datasetLFP1$Goles, labels = datasetLFP1$Nombre, cex =  0.4, p
os = 2)
```

La gráfica de dispersión anterior, junto con la recta de regresión obtenida, demuestran visualmente que los valores extremos no han afectado dramáticamente a su pendiente. Esto refuerza la validez de dichos valores extremos. Dado que los tiros a puerta deciden prácticamente la precisión (casi idéntica) de todos los modelos planteados, hemos tomado el modelo 1 (que sólo hace uso de esta variable) para que la gráfica pueda ser visualizada en dos dimensiones. así como fácilmente interpretable.

Los dos parámetros que tienen un peso bastante menor en el modelo ganador (‘Fueras.De.Juego’ y ‘Duelos.Cuerpo.Fracaso’) quedan reflejados en los siguientes ejemplos de predicción:

```
pred1 <- predict(goles.modelo_3, data.frame(Tiros.Puerta = 19, Fueras.De.Juego = 19, Duelos.C
uerpo.Fracaso = 76))
cat("Goles predicción 1:", pred1)
```

```
## Goles predicción 1: 7.096841
```

```
pred2 <- predict(goles.modelo_3, data.frame(Tiros.Puerta = 19, Fueras.De.Juego = 9, Duelos.Cu
erpo.Fracaso = 38))
cat("Goles predicción 2:", pred2)
```

```
## Goles predicción 2: 7.14156
```

```
pred3 <- predict(goles.modelo_3, data.frame(Tiros.Puerta = 38, Fueras.De.Juego = 9, Duelos.Cu
erpo.Fracaso = 38))
cat("Goles predicción 3:", pred3)
```

```
## Goles predicción 3: 14.38892
```

4.b. ¿Qué parámetros clave debe cumplir un futbolista para ser importante dentro de cualquier equipo en Primera División?

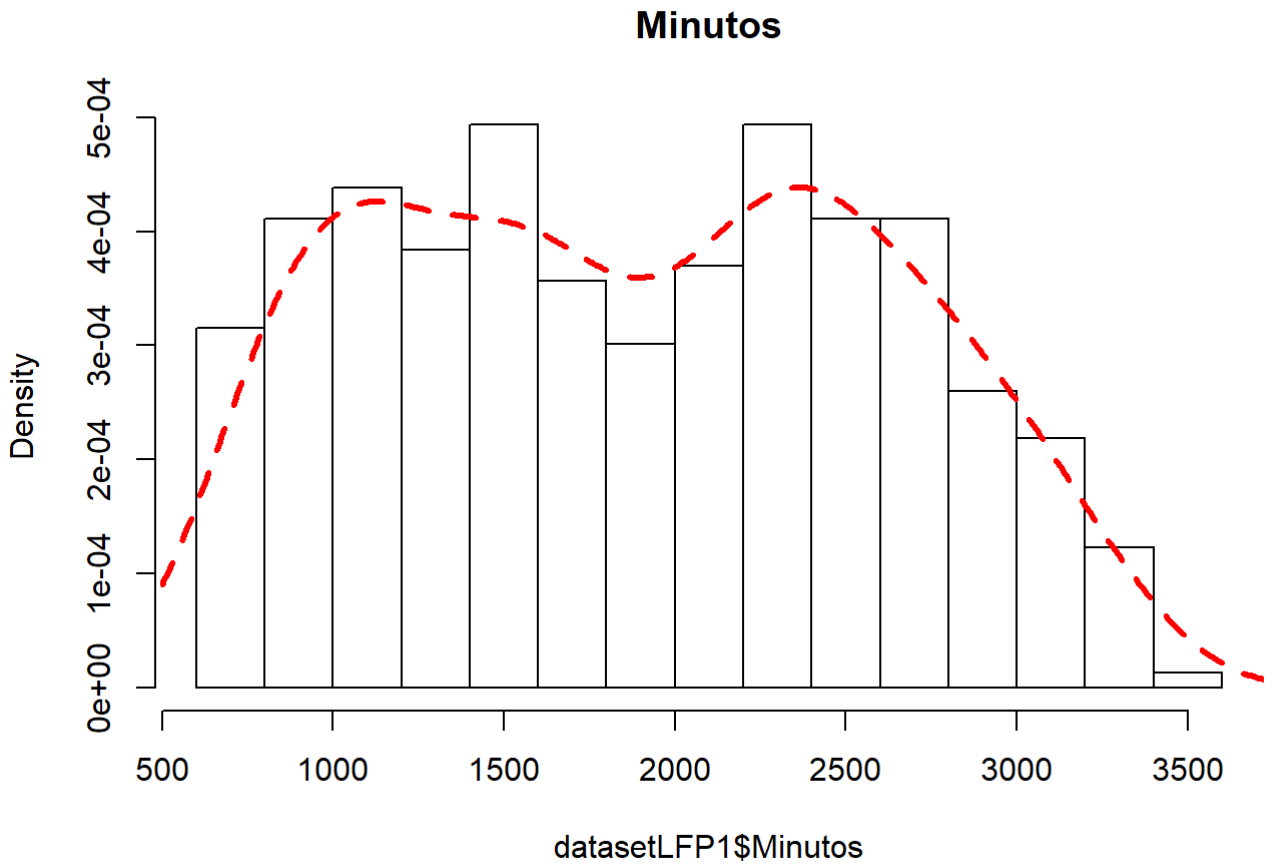
Para dar respuesta a esta segunda cuestión, vamos a exponer la planificación de los hitos que nos hemos marcado:

- Definiremos una nueva variable binaria para indicar si un jugador es importante o no. Para ello, haremos nuevamente uso de la variable 'Minutos', estableciendo un umbral adecuado para discernir entre ambos grupos de jugadores.
- Trataremos de verificar que, tanto el conjunto de jugadores importantes como el de no importantes, contiene una proporción similar de jugadores de todos los equipos de la liga. Para garantizar que haya una representación equitativa de todos los equipos, plantearemos un test para comprobar si las distribuciones parciales de 'Minutos' disfrutados por los jugadores de cada 'Equipo' por separado son similares; así como otro test para comprobar la homogeneidad de la varianza de cada una de estas distribuciones parciales por 'Equipo'.
- Acometeremos el estudio de la correlación de la variable a predecir (jugador 'Importante') con cada una de las demás variable del dataset.
- A raíz de las conclusiones del estudio de correlación anterior, construiremos una batería de modelos de regresión logística permutando las variables con mayor grado de correlación.
- Finalmente, seleccionaremos el modelo que mejor se ajuste a los datos, realizaremos la predicción de la variable binaria sobre todo el conjunto de datos de entrenamiento, graficaremos su curva ROC, obtendremos el área bajo dicha curva y fijaremos aquel valor umbral de decisión que minimice el número de jugadores clasificados incorrectamente en una matriz de confusión.

Comenzamos decidiendo qué umbral de minutos disputados vamos a establecer para discernir si un jugador debe considerarse importante o no. Si graficamos el histograma junto con la gráfica de densidad de la variable 'Minutos', podemos apreciar que, en torno a 1900 minutos, la gráfica de densidad alcanza un mínimo local para alcanzar, poco después, su máximo absoluto. Si atendemos a la propia naturaleza del fútbol, los equipos suelen alinear a 11 jugadores titulares para cada partido y, por norma general, sólo 3 de ellos serán sustituidos habiendo disputado además la mayoría de los 90 minutos del partido. Parece razonable establecer el umbral de filtrado en este punto, que separa inequívocamente los jugadores clave en cada equipo.

Una vez fijado el umbral, la variable 'Importante' tendrá el valor 0 en aquellos jugadores con menos de 1900 minutos disputados en liga. El resto de jugadores, los importantes, tendrán el valor 1.

```
umbral2 <- 1900
hist(datasetLFP1$Minutos, freq = FALSE, main = "Minutos")
lines(density(datasetLFP1$Minutos), col = "red", lty = 2, lwd = 3)
```




```
datasetLFP1$Importante <- ifelse(datasetLFP1$Minutos < umbral2, 0, 1)
```

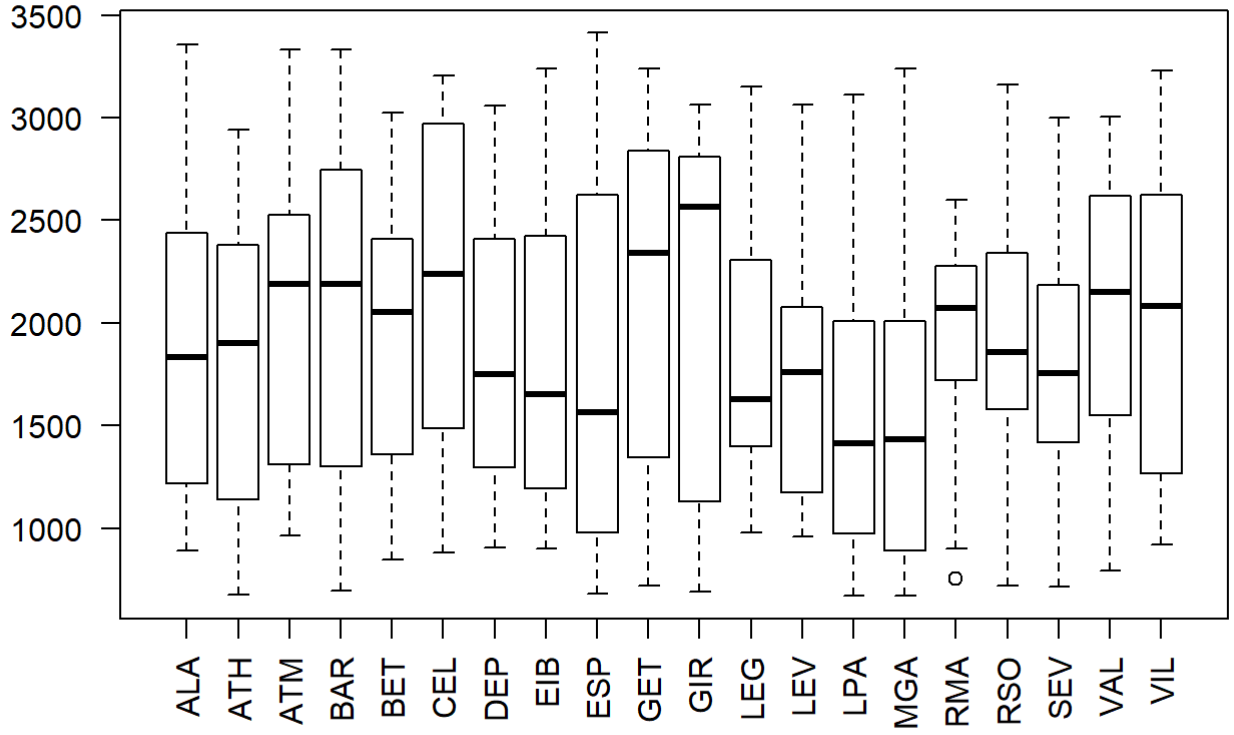
Umbral establecido: 1900 minutos.
Total de jugadores importantes: 181 de 365.
Media por equipo: 9.

Con el objeto de verificar que el grupo de jugadores importantes está representado equitativamente por todos los equipos de la liga, nos interesa comprobar que todos los grupos parciales de 'Minutos' por 'Equipo' siguen la misma distribución y sus varianzas son homogéneas.

Primeramente, graficamos un diagrama de cajas para cada grupo de minutos para disponer de una representación visual de las distribuciones.

En la pregunta anterior, rechazamos la normalidad de los datos, por lo que ejecutaremos pruebas no paramétricas. En la primera comprobación, aplicaremos el test de Kruskal-Wallis (para comparar la igualdad de distribuciones en más de dos muestras); en la segunda, el test de Fligner-Killeen.

```
boxplot(datasetLFP1$Minutos ~ datasetLFP1$Equipo, las = 2)
```




```
kruskal.test(Minutos ~ Equipo, data = datasetLFP1)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Minutos by Equipo
## Kruskal-Wallis chi-squared = 19.605, df = 19, p-value = 0.4187
```

```
fligner.test(Minutos ~ Equipo, data = datasetLFP1)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Minutos by Equipo
## Fligner-Killeen:med chi-squared = 18.028, df = 19, p-value =
## 0.5205
```

A raíz de los resultados de los tests no paramétricos se concluye, en términos generales, la imposibilidad de asumir que los jugadores importantes (con más minutos jugados) pertenecen a un subconjunto del total des equipos de la liga, sino que en cada equipo hay jugadores importantes que, por consiguiente, disfrutan de más minutos que el resto de sus compañeros.

Por tanto, podemos plantear un modelo de regresión logística fiable que, en función de ciertos datos, garantice cuándo un jugador será importante para cualquier equipo de la liga, en términos generales. Este modelo tendrá mayor validez para aquellos equipos cuyo objetivo sea mantener la categoría (no descender a Segunda División).

Vamos a calcular el coeficiente de correlación de cada una de ellas con respecto a la variable 'Importancia'. La tabla resultado estará ordenada ascendentemente por el valor p de cada correlación.

```
valores_p2 <- vector()
cols2 <- vector()
for(col in names(datasetLFP1)){
  if(is.numeric(datasetLFP1[, col])){
    valor_p <- cor.test(datasetLFP1[, col], datasetLFP1$Importante, method = "kendall")$p.val
ue
    valores_p2 = c(valores_p2, valor_p)
    cols2 = c(cols2, col)
  }
}

df2 <- data.frame(cols2, valores_p2)
df2 <- df2[order(valores_p2),]
rownames(df2) <- NULL
head(df2, n = 15)
```

```
##                    cols2   valores_p2
## 1             Importante 3.789733e-81
## 2                Minutos 2.537258e-61
## 3         Recuperaciones 5.770706e-45
## 4                  Pases 2.471055e-40
## 5           Pases.Cortos 1.006579e-27
## 6                 Duelos 7.570712e-24
## 7           Pases.Largos 1.972208e-22
## 8    Duelos.Cuerpo.Exito 1.484003e-19
## 9           Duelos.Cuerpo 1.307000e-17
## 10      Faltas.Cometidas 9.935131e-17
## 11           T.Amarillas 1.247048e-15
## 12              Entradas 3.194366e-14
## 13 Duelos.Cuerpo.Fracaso 1.811435e-13
## 14        Entradas.Exito 1.855809e-13
## 15      Entradas.Fracaso 6.920989e-13
```

Una vez conocemos las variables más determinantes para estimar la importancia de un jugador, vamos a plantear una batería de modelos de regresión logística mediante permutaciones de las variables con mayor correlación.

Para comparar la calidad de los modelos, tomaremos aquel con el menor valor AIC (Criterio de Información de Akaike). El modelo ganador (el octavo) contiene las siguientes variables: 'Recuperaciones', 'Pases', 'Duelos', 'Duelos.Cuerpo.Exito'.

```
importante.modelo_1 <- glm(Importante ~ Recuperaciones,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_2 <- glm(Importante ~ Pases,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_3 <- glm(Importante ~ Duelos,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_4 <- glm(Importante ~ Recuperaciones + Pases,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_5 <- glm(Importante ~ Recuperaciones + Duelos,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_6 <- glm(Importante ~ Pases + Duelos,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_7 <- glm(Importante ~ Recuperaciones + Pases + Duelos,
                           data = datasetLFP1, family = binomial(link = "logit"))
importante.modelo_8 <- glm(Importante ~ Recuperaciones + Pases + Duelos + Duelos.Cuerpo.Exito
,
                           data = datasetLFP1, family = binomial(link = "logit"))

modelos <- c(summary(importante.modelo_1)$aic, summary(importante.modelo_2)$aic,
             summary(importante.modelo_3)$aic, summary(importante.modelo_4)$aic,
             summary(importante.modelo_5)$aic, summary(importante.modelo_6)$aic,
             summary(importante.modelo_7)$aic, summary(importante.modelo_8)$aic)

for (i in 1:length(modelos))
  cat("AIC del modelo", i, ":", modelos[i], "\n")
```

```
## AIC del modelo 1 : 248.584
## AIC del modelo 2 : 299.6128
## AIC del modelo 3 : 406.8025
## AIC del modelo 4 : 232.1243
## AIC del modelo 5 : 197.2842
## AIC del modelo 6 : 279.044
## AIC del modelo 7 : 191.5976
## AIC del modelo 8 : 178.2139
```

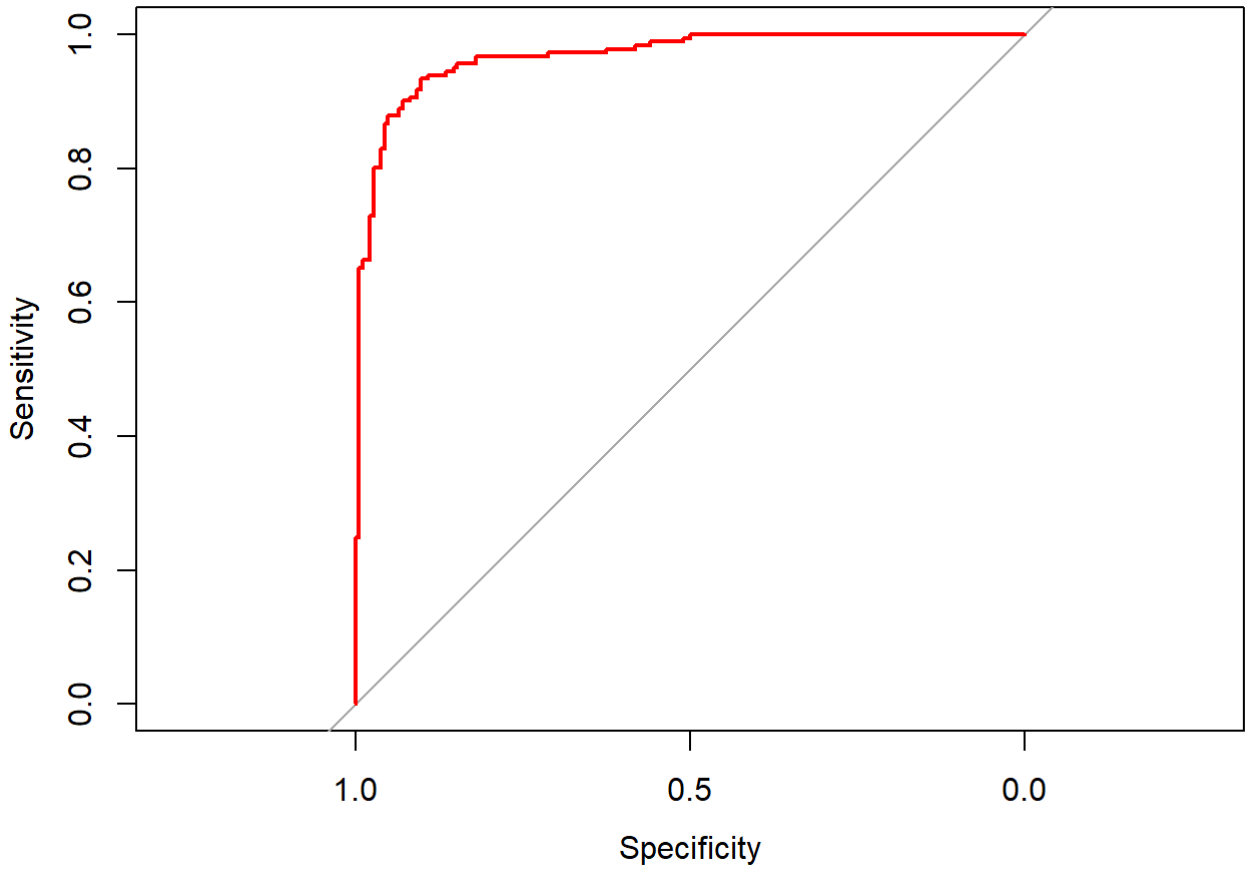
```
summary(importante.modelo_8)
```

```
##
## Call:
## glm(formula = Importante ~ Recuperaciones + Pases + Duelos +
##     Duelos.Cuerpo.Exito, family = binomial(link = "logit"), data = datasetLFP1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5034  -0.2842  -0.0474   0.2026   2.5460
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -9.6205782  1.0873714  -8.848  < 2e-16 ***
## Recuperaciones       0.0449475  0.0067015   6.707 1.99e-11 ***
## Pases                0.0027614  0.0007993   3.455 0.000551 ***
## Duelos               0.0249042  0.0043812   5.684 1.31e-08 ***
## Duelos.Cuerpo.Exito -0.0367164  0.0097323  -3.773 0.000162 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 505.97  on 364  degrees of freedom
## Residual deviance: 168.21  on 360  degrees of freedom
## AIC: 178.21
##
## Number of Fisher Scoring iterations: 7
```

Finalmente, queremos calcular la precisión del modelo ganador. La curva ROC muestra la precisión del modelo, en continuo, para los diferentes umbrales de decisión. Esta curva es clave para establecer el equilibrio deseado entre la tasa de falsos positivos y falsos negativos. La disminución de una, implica el aumento de la otra. Para nuestro problema, fichar a un jugador catalogado erróneamente como importante tiene un impacto similar a no fichar un jugador catalogado erróneamente como no importante. Estamos teniendo en cuenta tanto el aspecto deportivo como el aspecto económico.

El área bajo la curva ROC muestra un modelo cercano a la perfección: 0,9676 de 1. No obstante, vamos a obtener empíricamente el umbral óptimo que minimice el número de jugadores clasificados incorrectamente, para calcular la precisión del modelo sobre los propios datos de entrenamiento. Lo que obtenemos, consultando la matriz de confusión, es un nada desdeñable 91,2% de precisión en el modelo.

```
predicciones <- predict(importante.modelo_8, type = "response")
curvaROC <- roc(datasetLFP1$Importante, predicciones)
plot(curvaROC, col = "red")
```




```
auc(curvaROC)
```

```
## Area under the curve: 0.9676
```

```
# Obtención empírica del umbral óptimo que minimiza el número de jugadores clasificados incor
rectamente
umbral_importante <- 0.53
condicion_importante <- predicciones > umbral_importante
table(condicion_importante, datasetLFP1$Importante)
```

```
##
## condicion_importante   0   1
##                FALSE 171  19
##                TRUE   13 162
```

Para concluir, a continuación, generamos en formato CSV el data frame utilizado en la práctica.

```
write.csv(datasetLFP1, file = "datasetLFP1.csv")
```

5. Representación de los resultados a partir de tablas y gráficas

Durante todo el desarrollo de la práctica, para facilitar el seguimiento de las tareas que se iban acometiendo, los análisis y estudios realizados se han acompañado de las tablas y/o gráficas oportunas para complementar/justificar los resultados obtenidos.

6. Resolución del problema. ¿Cuáles son las conclusiones? ¿Los resultados permiten responder al problema?

En el primer problema planteado se ha planteado la construcción de un modelo de regresión lineal múltiple que estime el número de goles que anotará un jugador durante una temporada. Las variables del modelo con el mejor ajuste son los tiros a puerta, los fueras de juego y los duelos cuerpo a cuerpo fallidos. En particular, los tiros a puerta poseen el mayor peso en el modelo, otorgando 0,38 goles por cada tiro a puerta realizado. Los resultados del modelo permiten realizar estimaciones veraces sobre el número de goles de un jugador, tal y como se requería.

En el segundo problema planteado se ha planteado la construcción de un modelo de regresión logística que estime la importancia de un jugador dentro de la categoría, en función de si los minutos disputados superan un umbral fijado. Las variables del modelo con el mejor ajuste son las recuperaciones de balón, pases, duelos afrontados y duelos cuerpo a cuerpo ganados. El modelo devuelve un valor decimal entre 0 y 1, indicando la probabilidad de ser considerado importante. Hemos establecido un umbral de 0,53, que minimiza los errores del modelo sobre el conjunto de todos los jugadores de la liga. Los resultados del modelo permiten estimar verazmente la importancia de un jugador, tal y como se requería.

7. Código

El código utilizado en el desarrollo de la práctica se encuentra en formato .Rmd (fichero RMarkdown) en el fichero 'AnálisisLFP.Rmd'.

8. Bibliografía

- Dalgaard, P. (2008). Introductory statistics with R. Springer Science & Business Media.
- Test for normality – Shapiro-Wilks test (2016) [en línea]. bioSt@TS (mailto:bioSt@TS). [Consulta: 2 de junio de 2018] https://biostats.w.uib.no/test-for-normality-shapiro-wilks-test/ (https://biostats.w.uib.no/test-for-normality-shapiro-wilks-test/)
- Comparing two variances – Fisher's F test (2016) [en línea]. bioSt@TS (mailto:bioSt@TS). [Consulta: 2 de junio de 2018] https://biostats.w.uib.no/1-comparing-two-variances/ (https://biostats.w.uib.no/1-comparing-two-variances/)
- Análisis de Normalidad: gráficos y contrastes de hipótesis (2016) [en línea]. [Consulta: 2 de junio de 2018] https://rpubs.com/Joaquin_AR/218465 (https://rpubs.com/Joaquin_AR/218465)
- Análisis de la homogeneidad de varianza (homocedasticidad) (2016) [en línea]. [Consulta: 2 de junio de 2018] https://rpubs.com/Joaquin_AR/218466 (https://rpubs.com/Joaquin_AR/218466)
- No todo es normal - Manejo de datos no normales (2018) [en línea]. [Consulta: 2 de junio de 2018] https://anestesiar.org/2015/no-todo-es-normal-manejo-de-datos-no-normales/ (https://anestesiar.org/2015/no-todo-es-normal-manejo-de-datos-no-normales/)
- Test for homogeneity of variances - Lavene's test and the Fligner Killeen test (2016) [en línea]. bioSt@TS (mailto:bioSt@TS). [Consulta: 2 de junio de 2018] https://biostats.w.uib.no/test-for-homogeneity-of-variances-levenes-test/ (https://biostats.w.uib.no/test-for-homogeneity-of-variances-levenes-test/)