Premier League Simulación

Hans Walter

9/15/2020

Librerías

```
library(dplyr)
library(readr)
library(tidyverse)
library(readxl)
library(ggplot2)
library(devtools)
```

Archivos Principales

El **cronograma** nos va a servir mas adelante para poder simular los juegos y el de **premier** nos sirve para poder hacer los cálculos históricos. Después nos serviarn para poder simular los goles que va a anotar cada equipo. Antes de haber cargado el Excel se filtro el dataset para que solo estuvieran presente las últimas 2 temporadas (para evitar sesgo).

```
cronograma <- read_excel("plschedule.xlsx")
premier <- read_excel("dataset_premier.xlsx")
```

Nota

El siguiente vector nos demuestra el nombre de los equipos que van a estar presentes en la Premier League. Más adelante veremos si en el dataset de **cronogramas** tienen los mismos nombres. Si son diferentes, hay que hacer algunos cambios.

```
equipos <- c("Arsenal","Aston Villa", "Brighton", "Burnley",
             "Chelsea", "Crystal Palace", "Everton","Fulham",
             "Leeds United", "Leicester City", "Liverpool",
             "Man City", "Man United", "Newscastle", "Sheffield United",
             "Southampton", "Spurs", "West Brom", "West Ham", "Wolves")
```

Preparar Data

Modificamos la data de manera que 1. Agruparamos por equipos y se sumaran los goles que anotaron ambos estando de visitante y de local 2. Nos mostrara en teoría cuantos puntos tendrían si la temporada hubiera durado 2 años

```r
temp_combinado <- subset(premier, select = c(HomeTeam:FTR))

resultados_home <- temp_combinado %>% filter(., FTR == "H") %>% group_by(HomeTeam) %>%
  summarise(victorias_casa = n())
resultados_away <- temp_combinado %>% filter(., FTR == "A") %>% group_by(AwayTeam) %>%
  summarise(victorias_visitante = n())
resultados_empate1 <- temp_combinado %>% filter(., FTR == "D") %>% group_by(AwayTeam) %>%
  summarise(empates1 = n())
resultados_empate2 <- temp_combinado %>% filter(., FTR == "D") %>% group_by(HomeTeam) %>%
  summarise(empates2 = n())
goles_home <- temp_combinado %>% group_by(HomeTeam) %>%
  summarise(hg = sum(FTHG))
goles_away <- temp_combinado %>%  group_by(HomeTeam) %>%
  summarise(ag = sum(FTAG))
past_results <- cbind(resultados_home,resultados_away,resultados_empate1,
                      resultados_empate2,goles_home,goles_away)
past_results$puntos = (past_results$victorias_casa*3)+(past_results$victorias_visitante*3)
past_results$puntos = past_results$puntos + past_results$empates1 + past_results$empates2

df_temporadas <- past_results[,c(1,10,12,13)]
df_temporadas <- arrange(df_temporadas, -puntos)
colnames(df_temporadas) <- c("equipo", "goles_home_total","goles_away_total","puntos")
df_temporadas
```

```
##              equipo goles_home_total goles_away_total puntos
## 1         Liverpool              107               26    196
## 2          Man City              114               25    179
## 3           Chelsea               69               28    138
## 4         Man United               73               42    132
## 5         Tottenham               70               33    130
## 6           Arsenal               78               40    126
## 7            Wolves               55               40    116
## 8         Leicester               59               37    114
## 9           Everton               54               42    103
## 10          Burnley               48               55     94
## 11   Crystal Palace               34               43     92
## 12      Southampton               48               65     91
## 13         West Ham               62               60     91
## 14        Newcastle               44               46     89
## 15          Watford               48               55     84
## 16      Bournemouth               52               55     79
## 17         Brighton               39               55     77
## 18 Sheffield United               24               15     54
## 19      Aston Villa               22               30     35
## 20          Cardiff               21               38     34
## 21           Fulham               22               36     26
## 22          Norwich               19               37     21
## 23     Huddersfield               10               31     16
```

Vemos que los top 3 equipos de la liga son **Liverpool**, **Man City** y el **Chelsea**

Parámetros

La variable **Gamma** refleja la ventaja que tiene un equipo de anotar más goles al estar en casa. La diferencia no es significativa pero queremos tomarla en cuenta. Nuestro resultado nos dara la variable **Alfa** que representa la cantidad de goles en promedio que mete el equipo (tomando en cuenta ambos siendo visitante y local) y **Beta** que representa la cantidad de goles en promedio que concede (tomando en cuenta ambos cuando es visitante y local). La desviación estándar la utilizaremos en la 2nda simulación. Adicionalmente, nos dimos cuenta que los nombres de los equipos varía un poco en como se escriben en el dataset histórico a comparación del dataset de cronogramas, y queremos también poder filtrar solo los equipos que se encuentran presente en esta Premier League.

```
gamma <- sum(temp_combinado$FTHG)/sum(temp_combinado$FTAG)

modelo_basico1 <- temp_combinado %>%
  group_by(HomeTeam) %>%
  summarise(goles_a_favor = mean(FTHG), goles_concedidos = mean(FTAG),
            gaf = sd(FTHG), gc = sd(FTAG))

modelo_basico2 <- temp_combinado %>%
  group_by(AwayTeam) %>%
  summarise(goles_a_favor2 = mean(FTAG), goles_concedidos2 = mean(FTHG),
            gaf1 = sd(FTAG), gc1 = sd(FTHG))

modelo_basico <- cbind(modelo_basico1, modelo_basico2)
modelo_basico$alpha_prueba <- (modelo_basico$goles_a_favor +
                                 modelo_basico$goles_a_favor2)/2
modelo_basico$beta_prueba <- (modelo_basico$goles_concedidos +
                                modelo_basico$goles_concedidos2)/2
modelo_basico$alpha_prueba_sd <- (modelo_basico$gaf+modelo_basico$gaf1)/2
modelo_basico$beta_prueba_sd <- (modelo_basico$gc+modelo_basico$gc1)/2
modelo_basico <- subset(modelo_basico, select = c(HomeTeam, alpha_prueba,
                                                  beta_prueba, alpha_prueba_sd,
                                                  beta_prueba_sd))

nombres_equipos <- sort(as.vector(unique(cronograma$`Home Team`)))
nombres <- sort(as.vector(unique(cronograma$`Home Team`)))
nombres <- data.frame(HomeTeam=nombres)
modelo_basico$HomeTeam <- mgsub::mgsub(modelo_basico$HomeTeam, c("Man United",
                                                                 "Sheffield United",
                                                                 "Tottenham"),
                                       c("Man Utd", "Sheffield Utd", "Spurs"))

modelo_basico <- merge(x = modelo_basico, y = nombres, by.y = "HomeTeam")
modelo_basico
```

```
##          HomeTeam alpha_prueba beta_prueba alpha_prueba_sd beta_prueba_sd
## 1         Arsenal    1.6973684   1.3026316       1.1292892      1.0153274
## 2     Aston Villa    1.0789474   1.7631579       0.9832848      1.2942140
## 3        Brighton    0.9736842   1.5000000       0.9128584      1.2083032
## 4         Burnley    1.1578947   1.5526316       0.9808381      1.4872564
## 5         Chelsea    1.7368421   1.2236842       1.2941160      1.1035858
## 6  Crystal Palace    1.0789474   1.3552632       1.0696323      1.1200519
## 7         Everton    1.2894737   1.3421053       1.0789114      1.2561024
```

```
## 8          Fulham    0.8947368   2.1315789       0.9714361      1.2715092
## 9       Leicester    1.5526316   1.1710526       1.4591065      1.1106917
## 10      Liverpool    2.2894737   0.7236842       1.2269943      0.8775254
## 11       Man City    2.5921053   0.7631579       1.6499678      0.8858970
## 12        Man Utd    1.7236842   1.1842105       1.2696400      0.9528092
## 13      Newcastle    1.0526316   1.3947368       1.0187099      1.2444525
## 14  Sheffield Utd    1.0263158   1.0263158       0.9054305      0.9503598
## 15    Southampton    1.2631579   1.6447368       0.9202895      1.4900800
## 16          Spurs    1.6842105   1.1315789       1.2853233      0.8924005
## 17       West Ham    1.3289474   1.5394737       1.2487784      1.1306773
## 18         Wolves    1.2894737   1.1315789       1.0105678      0.9905445
```

Nos hace falta el equipo **West Brom** y **Leeds**.

Adición de West Brom

Después de un poco de investigación, notamos que **West Brom** no estaba presente en los años más recientes. Por lo tanto, cargamos un dataset donde se encuentra el mismo y filtramos que fueran solo 2 temporadas de datos para mantenernos bajo los parametros del anterior.

```
wb <- read_excel("westbrom.xlsx")
wb <- wb %>% filter(Temporada %in% c(1617, 1819))
wb_p1 <- wb %>% group_by(Local) %>%
  summarise(goles_favor = mean(`GOL L`), goles_concedidos = mean(`GOL V`),
            gaf = sd(`GOL L`), gc = sd(`GOL V`))
wb_p2 <- wb %>% group_by(Visitante) %>%
  summarise(goles_favor1 = mean(`GOL V`), goles_concedidos2 = mean(`GOL L`),
            gaf1 = sd(`GOL V`), gc1 = sd(`GOL L`))
wb_data <- cbind(wb_p1, wb_p2)
resultado_wb <- wb_data[26, ]
alpha_wb <- (resultado_wb$goles_favor+resultado_wb$goles_favor1)/2
beta_wb <- (resultado_wb$goles_concedidos+resultado_wb$goles_concedidos2)/2
alpha_wb_sd <- (resultado_wb$gaf + resultado_wb$gaf1)/2
beta_wb_sd <- (resultado_wb$gc + resultado_wb$gc1)/2

modelo_basico <- modelo_basico %>% add_row(HomeTeam = "West Brom",
                                           alpha_prueba = alpha_wb,
                                           beta_prueba = beta_wb,
                                           alpha_prueba_sd = alpha_wb_sd,
                                           beta_prueba_sd = beta_wb_sd)
modelo_basico
```

```
##          HomeTeam alpha_prueba beta_prueba alpha_prueba_sd beta_prueba_sd
## 1         Arsenal    1.6973684   1.3026316       1.1292892      1.0153274
## 2     Aston Villa    1.0789474   1.7631579       0.9832848      1.2942140
## 3        Brighton    0.9736842   1.5000000       0.9128584      1.2083032
## 4         Burnley    1.1578947   1.5526316       0.9808381      1.4872564
## 5         Chelsea    1.7368421   1.2236842       1.2941160      1.1035858
## 6  Crystal Palace    1.0789474   1.3552632       1.0696323      1.1200519
## 7         Everton    1.2894737   1.3421053       1.0789114      1.2561024
## 8          Fulham    0.8947368   2.1315789       0.9714361      1.2715092
## 9       Leicester    1.5526316   1.1710526       1.4591065      1.1106917
```

```
## 10      Liverpool    2.2894737   0.7236842       1.2269943      0.8775254
## 11       Man City    2.5921053   0.7631579       1.6499678      0.8858970
## 12        Man Utd    1.7236842   1.1842105       1.2696400      0.9528092
## 13      Newcastle    1.0526316   1.3947368       1.0187099      1.2444525
## 14  Sheffield Utd    1.0263158   1.0263158       0.9054305      0.9503598
## 15    Southampton    1.2631579   1.6447368       0.9202895      1.4900800
## 16          Spurs    1.6842105   1.1315789       1.2853233      0.8924005
## 17       West Ham    1.3289474   1.5394737       1.2487784      1.1306773
## 18         Wolves    1.2894737   1.1315789       1.0105678      0.9905445
## 19      West Brom    1.2368421   1.2105263       1.0575506      0.9343000
```

Adición de Leeds

Leeds es un equipo nuevo por lo cual no tenemos datos de donde extraer información. Sin embargo, decidimos promediar los parámetros de **Alpha** y **Beta**, y solo tomar en cuenta el 60% del dato obtenido dado a que el equipo Leeds no se considera un equipo de categoría top.

```
a_leeds <- mean(modelo_basico$alpha_prueba)*0.6
b_leeds <- mean(modelo_basico$beta_prueba)*0.6
a_leeds_sd <- mean(modelo_basico$alpha_prueba_sd)
b_leeds_sd <- mean(modelo_basico$beta_prueba_sd)
modelo_basico <- modelo_basico %>% add_row(HomeTeam = "Leeds",
                                           alpha_prueba = a_leeds, beta_prueba = b_leeds,
                                           alpha_prueba_sd = a_leeds_sd,
                                           beta_prueba_sd = b_leeds_sd)
modelo_basico <- arrange(modelo_basico, HomeTeam)
colnames(modelo_basico)[1] <- "Equipos"
modelo_limpio <- modelo_basico
modelo_limpio <- subset(modelo_limpio, select = c(1:3))
colnames(modelo_limpio) <- c("Equipos", "alpha","beta")
modelo_limpio
```

```
##           Equipos     alpha      beta
## 1         Arsenal 1.6973684 1.3026316
## 2     Aston Villa 1.0789474 1.7631579
## 3        Brighton 0.9736842 1.5000000
## 4         Burnley 1.1578947 1.5526316
## 5         Chelsea 1.7368421 1.2236842
## 6  Crystal Palace 1.0789474 1.3552632
## 7         Everton 1.2894737 1.3421053
## 8          Fulham 0.8947368 2.1315789
## 9           Leeds 0.8509695 0.7923823
## 10      Leicester 1.5526316 1.1710526
## 11      Liverpool 2.2894737 0.7236842
## 12       Man City 2.5921053 0.7631579
## 13        Man Utd 1.7236842 1.1842105
## 14      Newcastle 1.0526316 1.3947368
## 15  Sheffield Utd 1.0263158 1.0263158
## 16    Southampton 1.2631579 1.6447368
## 17          Spurs 1.6842105 1.1315789
## 18      West Brom 1.2368421 1.2105263
## 19       West Ham 1.3289474 1.5394737
## 20         Wolves 1.2894737 1.1315789
```

Función para Goles

La función toma en cuenta los promedios de goles que mete cada equipo junto con los goles que pueden conceder. Hay que notar que le damos ventaja a los que juegan en casa ya que le agregamos el **Gamma** calculado anteriormente.

```
prediccion <- function(home, away, parametros){
  expected_goles_home <- as.numeric(parametros$alpha[home] * parametros$beta[away] * gamma)
  expected_goles_away <- as.numeric(parametros$alpha[away] * parametros$beta[home])

  df <- data.frame(home = home, away = away, goles_home = expected_goles_home,
                   goles_away = expected_goles_away)
  return(df)
}
```

Lista de Nombres

Obtenemos la lista de nombres para poder usarla despues y limpiamos el dataset de cronograma para que trabaje solo con las columnas de Home y Away.

```
parametros_basicos <- modelo_limpio %>%
  select(-Equipos) %>%
  as.list() %>%
  lapply(., function(x){names(x) <- nombres_equipos;return(x)})

colnames(cronograma)[4] <- "home"
colnames(cronograma)[5] <- "away"
cronograma <- subset(cronograma, select = -c(Date, Location, Result))
```

Simulación de Goles

La función nos calculará los goles que tendran durante toda la temporada. Finalmente aplicamos la función *floor* ya que van a haber decimales y por criterio propio aunque tenga un decimal presente lo vamos a redondear para abajo.

```
goles_prediccion <- map2_df(cronograma$home, cronograma$away,
                            prediccion, parametros_basicos) %>%
  mutate_if(is.numeric, round, digits = 2)


goles_prediccion$goles_home <- floor(goles_prediccion$goles_home)
goles_prediccion$goles_away <- floor(goles_prediccion$goles_away)

head(goles_prediccion)
```

```
##             home        away goles_home goles_away
## 1         Fulham     Arsenal          1          3
## 2        Burnley     Man Utd          1          2
## 3       Man City Aston Villa          5          0
## 4 Crystal Palace Southampton          2          1
## 5      Liverpool       Leeds          2          0
## 6       West Ham   Newcastle          2          1
```

Etapa Final

La función de **datos_compilados** simplemente va a recopilar los goles que tuvieron en total los equipos estando en casa y de visitante, y los goles que concedieron durante el torneo. La función **scores** toma en cuenta lo que calculó **datos_compilados**. Va a calcular los puntos: victoria 3 puntos, empate 1 punto, si perdieron 0 puntos. Toma en cuenta que los equipos cambian de localidad y al final nos devolverá una tabla con la predicción final.

```
datos_compilados <- function(x){
  x %>%
    gather(localidad, equipo, -goles_home, -goles_away) %>%
    mutate(g_for = case_when(
      localidad == "home" ~ goles_home,
      localidad == "away" ~ goles_away
    )) %>%
    mutate(g_ag = case_when(
      localidad == "home" ~ goles_away,
      localidad == "away" ~ goles_home
    ))
}

scores <- function(x){
  x %>%
    datos_compilados(.) %>%
    mutate(puntos = case_when(
      g_for > g_ag ~ 3,
      g_ag > g_for~ 0,
      g_for == g_ag ~ 1
    )) %>%
    group_by(equipo) %>%
    summarise(juegos_jugados = n(),
              goles_total = sum(g_for),
              goles_concedidos_total = sum(g_ag),
              puntos = sum(puntos)) %>%
    arrange(-puntos)
}

final_scores <- goles_prediccion %>%
  scores(.)

top10 <- head(final_scores, 10)
top10
```

```
## # A tibble: 10 x 5
##    equipo    juegos_jugados goles_total goles_concedidos_total puntos
##    <chr>              <int>       <dbl>                  <dbl>  <dbl>
##  1 Liverpool             38         111                     19    111
##  2 Man City              38         127                     23    111
##  3 Spurs                 38          74                     47     86
##  4 Man Utd               38          77                     51     81
##  5 Chelsea               38          77                     53     77
##  6 Leicester             38          71                     51     77
##  7 Arsenal               38          74                     59     67
```

```
##  8 Wolves                38          51                      48      59
##  9 Leeds                 38          32                      31      51
## 10 West Brom             38          49                      55      48
```

Vemos que la predicción nos regresa que el **Liverpool* empata con *Man City**. Podríamos decir que el primer lugar se lo puede llevar el Liverpool ya que fue el que concedió menos goles, o podemos decir que Man City gana dado a que metió más goles.

Segundo Modelo de Predicción / Distribución Normal / Montecarlo

El modelo toma en cuenta las desviaciones estándar y trabaja con una distribución normal. Es exactamente lo mismo que se vio anteriormente, pero dentro de una función que puede iterar **n** veces. Los goles son calculados de manera aleatoria tomando en cuenta la media que tienen de meter goles, junto con la media que tienen de conceder goles, y usando la desviación estándar de los mismos. El resultado es un dataframe que cuenta quienes fueron los ganadores más frecuentes y su respectiva probabilidad que tienen de ganar Premier League.

```
pl_simulator <- function(modelo_basico, cronograma, n){

i = 0
ganadores = c()
for(i in 0:20) {
improved_model <- modelo_basico
improved_model$alpha <- rnorm(20, mean = improved_model$alpha_prueba,
                              sd = improved_model$alpha_prueba_sd)
improved_model$beta <- rnorm(20, mean = improved_model$beta_prueba,
                             sd = improved_model$beta_prueba_sd)
improved_model <- subset(improved_model, select = c(Equipos, alpha,beta))

prediccion <- function(home, away, parametros){
  expected_goles_home <- as.numeric(parametros$alpha[home] * parametros$beta[away] * gamma)
  expected_goles_away <- as.numeric(parametros$alpha[away] * parametros$beta[home])

  df <- data.frame(home = home, away = away,
                   goles_home = expected_goles_home, goles_away = expected_goles_away)
  return(df)
}

parametros_basicos <- improved_model %>%
  select(-Equipos) %>%
  as.list() %>%
  lapply(., function(x){names(x) <- nombres_equipos;return(x)})

goles_prediccion <- map2_df(cronograma$home, cronograma$away,
                            prediccion, parametros_basicos) %>%
  mutate_if(is.numeric, round, digits = 2)

goles_prediccion$goles_home <- floor(goles_prediccion$goles_home)
goles_prediccion$goles_away <- floor(goles_prediccion$goles_away)

datos_compilados <- function(x){
  x %>%
```

```
    gather(localidad, equipo, -goles_home, -goles_away) %>%
    mutate(g_for = case_when(
      localidad == "home" ~ goles_home,
      localidad == "away" ~ goles_away
    )) %>%
    mutate(g_ag = case_when(
      localidad == "home" ~ goles_away,
      localidad == "away" ~ goles_home
    ))
}

scores <- function(x){
  x %>%
    datos_compilados(.) %>%
    mutate(puntos = case_when(
      g_for > g_ag ~ 3,
      g_ag > g_for~ 0,
      g_for == g_ag ~ 1
    )) %>%
    group_by(equipo) %>%
    summarise(juegos_jugados = n(),
              goles__total = sum(g_for),
              goles_concedidos_total = sum(g_ag),
              puntos = sum(puntos)) %>%
    arrange(-puntos)
}

final_scores <- goles_prediccion %>%
  scores(.)

winner <- as.character(final_scores[1,1])
ganadores = append(ganadores, winner)
i = i + 1
}

outcomes <- as.data.frame(table(ganadores))
outcomes <- arrange(outcomes, -Freq)
outcomes$Probabilidad_Ganar <- (outcomes$Freq/n)
outcomes <- outcomes[c(1:3), ]

return(outcomes)
}

pl_simulator(modelo_basico, cronograma, 100)
```

```
##   ganadores Freq Probabilidad_Ganar
## 1   Everton    4              0.04
## 2     Leeds    2              0.02
## 3  Man City    2              0.02
```

Conclusión de 2ndo modelo

Si se corren varias veces, se puede observar que en algunos casos pueden ganar equipos que no son considerados los mejores en la liga. La justificación se puede dar dado a lo volátil que pueden ser los rankings de los equipos. A que nos referimos con esto: en el mundial pasado, vimos como Croacia logró llegar a la final del mundial cuando no era considerado un oponente significativamente bueno. Alemania también fue eliminado de manera rápida en el mundial. En general, el modelo le da el chance a que en algunas instancias logre sobresalir los equipos "chiquitos" o que alguno de los equipos formidables se queden un poco atrás.

Créditos y Bibliografía

Para tener una base e idea de como empezar con el código: Hickman, R. (2019, May 30). An Introduction to Modelling Soccer Matches in R (part 1). Retrieved September 16, 2020, from https://www.robert-hickman.eu/post/dixon_coles_1/

Los Excels con la información histórica: Sharma, S. (2020, August 19). English Premier League 10 Seasons. Retrieved September 16, 2020, from https://www.kaggle.com/somesh24/english-premier-league-football-10-seasons

El cronograma de partidos: Download Football/Soccer fixtures, schedules and results. (n.d.). Retrieved September 16, 2020, from https://fixturedownload.com/sport/football