The Guardian Knowledge June 2019

Robert Hickman

2019-06-20

```
library(tidyverse)
library(magrittr)
library(rvest)

set.seed(3459)
```

```
# get years of EPL seasons
years <- 1993:2019
# base url we'll scrape from
base_url <- "https://www.11v11.com"

# cat together
tables <- paste0(base_url, "/league-tables/premier-league/01-june-", years)
```

```
squads <- tables %>%
  # get a list of the links to every teams squad page
  map(., function(x) {
    x %>%
      read_html() %>%
      html_nodes("#table-league > tbody:nth-child(2) > tr > td:nth-child(2) > a:nth-child(1)") %>%
      html_attr("href") %>%
      # paste into working link for year and competition (EPL)
      paste0(base_url, ., "tab/players/season/", gsub(".*01-june-", "", x), "/comp/1/")
  }) %>%
  unlist() %>%
  # get the players/appearances/nationalities
  map_df(., function(y) {
    # read once to save server calls
    read <- y %>%
      read_html()

    # get the squad info
    squad <- read %>%
      html_nodes(".squad") %>%
      html_table(fill = TRUE) %>%
      as.data.frame() %>%
      # get rid of rows without player info
      filter(!is.na(Player))

    # get the listed nationalities
    flags <- read %>%
      html_nodes(".squad > tbody:nth-child(2) > tr > td:nth-child(3)")

    # from here get the actual nationalities per player
    nations <- flags %>%
      html_nodes("img") %>%
      html_attr("title")
```

```
    # these might mismatch in length
    # in which case append NA
    if(length(flags) != length(nations)) {
      missing <- which(!grepl("img", flags))

      nations <- c(nations[1:(missing-1)], NA, nations[missing:length(nations)])
    }

    # mutate nationality and team and season
    squad %>%
      mutate(nation = nations,
             year = gsub(".*season\\/", "", gsub("\\/comp.*", "", y)),
             team = gsub("\\/tab\\/players.*", "", gsub(".*teams\\/", "", y))) %>%
      # select useful appearance information
      select(player = Player, position = Position,
             appearances = A, sub_appearances = S,
             nation, year, team)
  }) %>%
  # manually add in some missing nationalities
  mutate(nation = case_when(
    grepl("Steffen Karl", player) ~ "Germany",
    grepl("Marc Muniesa", player) ~ "Spain",
    grepl("Oriol Romeu", player) ~ "Spain",
    grepl("Aleix García", player) ~ "Spain",
    grepl("Martín Montoya", player) ~ "Spain",
    TRUE ~ nation
  ))
```

```
# load a df of international results
# from https://www.kaggle.com/martj42/international-football-results-from-1872-to-2017/downloads/intern
international_results <- readRDS("../../static/files/international_results.rds")

head(international_results)
```

```
##         date home_team away_team home_score away_score tournament    city
## 1 1872-11-30  Scotland   England          0          0   Friendly Glasgow
## 2 1873-03-08   England  Scotland          4          2   Friendly  London
## 3 1874-03-07  Scotland   England          2          1   Friendly Glasgow
## 4 1875-03-06   England  Scotland          2          2   Friendly  London
## 5 1876-03-04  Scotland   England          3          0   Friendly Glasgow
## 6 1876-03-25  Scotland     Wales          4          0   Friendly Glasgow
##    country neutral
## 1 Scotland   FALSE
## 2  England   FALSE
## 3 Scotland   FALSE
## 4  England   FALSE
## 5 Scotland   FALSE
## 6 Scotland   FALSE
```

```
# prepare results df for ELO modelling
international_results %<>%
  # select relevant columns
  select(date, home = home_team, away = away_team, hgoal = home_score, agoal = away_score, neutral) %>%
  # convert date to date format
```

```
  mutate(date = as.Date(date)) %>%
  # K = match importance
  # don't have competition data in this dataset so just set to 40
  mutate(K =  40) %>%
  # G = goal difference factor
  # takes into account how much a team is beaten by
  mutate(G = case_when(
    abs(hgoal-agoal) < 2 ~ 1,
    abs(hgoal-agoal) < 3 ~ 1.5,
    abs(hgoal-agoal) >= 3 ~ 1.75 + (abs(hgoal-agoal)-3)/8
  )) %>%
  # results = 1 for win and 0.5 for a draw
  mutate(result = case_when(
    hgoal > agoal ~ 1,
    hgoal < agoal ~ 0,
    hgoal == agoal ~ 0.5
  )) %>%
  # arrange by date so ELO can be updated sequentially
  arrange(date)

head(international_results)
```

```
##         date     home     away hgoal agoal neutral  K     G result
## 1 1872-11-30 Scotland  England     0     0   FALSE 40 1.000    0.5
## 2 1873-03-08  England Scotland     4     2   FALSE 40 1.500    1.0
## 3 1874-03-07 Scotland  England     2     1   FALSE 40 1.000    1.0
## 4 1875-03-06  England Scotland     2     2   FALSE 40 1.000    0.5
## 5 1876-03-04 Scotland  England     3     0   FALSE 40 1.750    1.0
## 6 1876-03-25 Scotland    Wales     4     0   FALSE 40 1.875    1.0
```

```
# function to calculate updated ELO ratings
calc_ELO <- function(date, home, away, K, G, result) {
  #get the difference in ratings
  hr <- team_ratings$rating[which(team_ratings$team == home)]
  vr <- team_ratings$rating[which(team_ratings$team == away)]
  dr <- vr - (hr + 100)

  # calculate expected results
  e_result <- 1/ ((10^(dr/400))+1)

  # calculate new ratings
  new_hr <- hr + ((K*G) * (result - e_result))
  new_vr <- vr + ((K*G) * ((1-result) - (1-e_result)))

  # pipe these back into a df of team ratings to sample from
  team_ratings$rating[which(team_ratings$team == home)] <<- new_hr
  team_ratings$rating[which(team_ratings$team == away)] <<- new_vr

  # return new ratings
  return(list(h_rating = new_hr, v_rating = new_vr))
}


team_ratings <- international_results %>%
```

```
  # select date and teams
  select(date, home, away) %>%
  # melt
  gather(., "location", "team", home, away) %>%
  select(-location) %>%
  arrange(date) %>%
  # set out unique teams with a rating of 1200
  filter(!duplicated(team)) %>%
  mutate(rating = 1200) %>%
  select(-date)

head(team_ratings)
```

```
##                team rating
## 1          Scotland   1200
## 2           England   1200
## 3             Wales   1200
## 4 Northern Ireland   1200
## 5     United States   1200
## 6            Canada   1200
```

```
elo_data <- international_results %>%
  # select relevant variable
  # keep date so we know a teams ELO at specific date
  select(date, home, away, K, G, result) %>%
  # pmap doesn't play with dates so convert to character
  mutate(date = as.character(date)) %>%
  # apply the calc_ELO function over rows of df
  pmap_df(~c(..., calc_ELO(...))) %>%
  # reconvert to dates
  mutate(date = as.Date(date)) %>%
  # get rid of ELO parameters
  select(date, home, away, h_rating, v_rating) %>%
  # gather twice to get a long df of teams ratings after matches
  gather("location", "team", -date, -h_rating, -v_rating) %>%
  gather("rating", "value", -date, -location, -team) %>%
  # filter for home rating for teams at home and vice versa
  filter((location == "home" & rating == "h_rating") |
           (location == "away" & rating == "v_rating")) %>%
  select(date, team, rating = value) %>%
  # we only care about ratings from August 1992
  filter(date > "1992-07-31")
```

```
teams <- elo_data %>%
  filter(rating > 1600) %>%
  .$team %>%
  unique() %>%
  .[sample(length(.), 5)]

teams
```

```
## [1] "Ivory Coast" "Greece"      "Mexico"      "Peru"        "Argentina"
```

```
p <- elo_data %>%
  filter(team %in% teams) %>%
```

```
  ggplot(aes(x = date, y = rating, colour = team, group = team)) +
  geom_point() +
  geom_line() +
  scale_colour_manual(values = c("skyblue", "darkblue", "darkorange", "darkgreen", "red")) +
  theme_minimal()

plot(p)
```