Could an Independent Yorkshire Win the World Cup - Finding British Player's Birthplaces

Robert Hickman

2018-06-07

Recently, a Yorkshire national football team appeared in a league of national teams for stateless people. This got me wondering how the historic counties of the UK would do at the world cup. Could any of them compete with full international teams?

This is the complete script for an short article I wrote for CityMetric on the topic. It's split over 6 separate parts and is pretty hefty but contains pretty much everything you need to clone the article. In the last post, we located the place and county of birth for British players, which we'll use to pick teams for counties now.

```
library(dplyr)
library(magrittr)
library(data.table)
library(ggplot2)
library(rvest)
```

Putting Together Teams

To work out how good each nation/county is, we need to select the best team that can be picked from the available pool of players. In theory we could just select the best 11 players, but this isn't how football works in real life. Instead, we want to pick the optimal 11 players for a set of realistic formations.

First, we need a list of plausible formations, and the positions they contain. There's a handy list of the default FIFA18 formations online which we'll scrape.

```
#grab a link to all the default FIFA18 formations
link <- "https://www.fifauteam.com/fifa-18-formations-guide/#4222"

#get all the formations
formations <- read_html(link) %>%
  html_nodes("h2") %>%
  html_text() %>%
  .[2:length(.)]

#get all the positions per formation
positions <- read_html(link) %>%
  html_nodes("td:nth-child(1)") %>%
  html_text() %>%
  gsub(" .", "", .) %>%
  #make positions symmetric
  gsub("RF|LF", "CF", .) %>%
  gsub("CMR|CML", "CM", .) %>%
  gsub("^R|^L", "W", .)

#df of each formation and the positions it contains
formations_df <- data.frame(formation = rep(formations, each = 10),
                            position = positions)
```

Then, for each nation/county, we need to work out which of these formations (and the selection of players for it), gives the highest total ability (using the ability for each position that we worked out earlier).

To do this, I have two functions: - The first (find_optimal_team) selects the available players for that nation/county. It then wraps in a second function (pick_players) that takes a formation and tries to find the optimal team for that formation. Finally, we select the team that has the highest total_ability out of all the possibilities that pick_plyaers returns

-pick_players itself iterates through each formation that we scraped. It then shuffles the positions each trial and pseudo-randomly picks the best* players for each position until an entire team is picked.It does this a specified (replicates) times per formation I find that doing it 100x per formation almost always gives an answer == 10000x per formation so I limit it to 100 to save on time.

*it doesn't always necessarily pick the very best player, as we can imagine that picking the best (e.g.) centre forward, might mean that player can't be picked as a striker where they would be better. Instead it is biased towards picking the best player, though sometimes opting for the 2nd or 3rd best.

```
find_optimal_team <- function(nation, players, replicates) {
  #find only players available to play for that nation
  players_pool <- players %>%
    filter(nationality == nation)

  #find the best team that can be played using these players for each default formation
  best_team <- rbindlist(lapply(rep(unique(formations_df$formation), replicates), pick_players, players
    #select only the formation/team with the highest total ability
    filter(total_ability == max(total_ability)) %>%
    #in case there are multiple best teams, take the first
    .[1:11,] %>%
    #add the nation as an id
    mutate(nation = nation)

  return(best_team)
}

pick_players <- function(players, formation) {
  #get all the positions for he formation being tested
  formation_positions <- formations_df$position[formations_df$formation == formation]
  #randomise the order of positions to pick
  positions <- sample(as.character(formation_positions))
  #add the goalkeeper as the first to be picked
  positions <- append("GK", positions)

  #for each position that needs a player
  for(position in positions) {
    if(position != "GK") {
      #generate a random number to determine if picking the best, second best, or third best player for
      #might not always be optimal to pick the best player if they are even better in another position
      randomiser <- runif(1)
      #pick the corresponding player
      if(randomiser < 0.6 | nrow(players) < 3) {
        id <- players$id[which.max(players[[position]])]
      } else if(randomiser < 0.9) {
        id <- players$id[order(-players[[position]])][2]
      } else {
        id <- players$id[order(-players[[position]])][3]
      }
```

```
    } else {
      #always pick the best goalkeeper available
      id <- players$id[which.max(players[[position]])]
    }

    #get the ability of that player in the position sampled
    ability <- players[[position]][which(players$id == id)]

    #create a df of all the players picked for this formation
    if(position == "GK") {
      team <- data.frame(id = id, position = position, ability = ability)
    } else {
      team <- rbind(team, data.frame(id = id, position = position, ability = ability))
    }
    #for each player picked, remove it from further consideration for other positions
    players <- players[-which(players$id == id),]
  }

  #get the total ability of the team by averaging their position abilities
  team$total_ability <- sum(team$ability) / 11
  team$formation <- formation
  return(team)
}
```

Not every nation has enough players in FIFA18 to pick a whole side so first we need to select only those who have at leat 10 outfield players and at least one goalkeeper. This leaves us with 84 nations in total (most of the top nations and few random stragglers).

```
#find the number of FIFA players for each nation
national_teams <- data.frame(table(all_players_data$nationality)) %>%
  merge(., data.frame(table(all_players_data$nationality[which(all_players_data$symmetric_position == "
names(national_teams) <- c("nation", "players", "gks")

#select only nations that can field a team
#at least 1 goalkeeper and 10 outfield players
national_teams <- national_teams %>%
  mutate(players = players - gks) %>%
  filter(players >= 10) %>%
  filter(gks >= 1)
```

We can then running the picking functions for each of these nations, giving us a df of each nations best possible team in FIFA18.

This function takes a while to run (~1 hour total).

```
#find the optimal team for each nation
optimal_national_teams <- rbindlist(lapply(national_teams$nation, find_optimal_team,
                                           select(all_players_data, id, nationality, 49:60), replicates
```

We can then plot the national teams to take a look at the selections and check they make sense. I've only included the best 4 teams (Brazil, Germany, Spain, and Belgium) below to save space.

```
#get the names of each player to merge in
players <- all_players_data %>%
  select(id, name)
```

```r
#select the best 4 county teams by total ability
best_national_elevens <- optimal_national_teams %>%
  setDT() %>%
  .[, unique_position := make.unique(as.character(position)), by = "nation"] %>%
  merge(., formation_coords, by = c("formation", "unique_position")) %>%
  merge(players, by = "id") %>%
 .[total_ability >= abs(sort(unique(-.$total_ability)))[4]]

#plot the data
p <- ggplot(data = best_national_elevens)
p <- p %>%
  #custom pitch aesthetic function
  draw_pitch()
p <- p +
  geom_text(aes(x = player_x, y = player_y, label = gsub(".* ", "", name)), colour = "black") +
  facet_wrap(~nation)

plot(p)
```

We then need to do the same thing, but for the counties.

First the player position ability for all the british players needs to be merged in.

Then we select only those counties that can field a whole team, as we did before for nations. This leaves us with 20 counties overall which are plotted below.

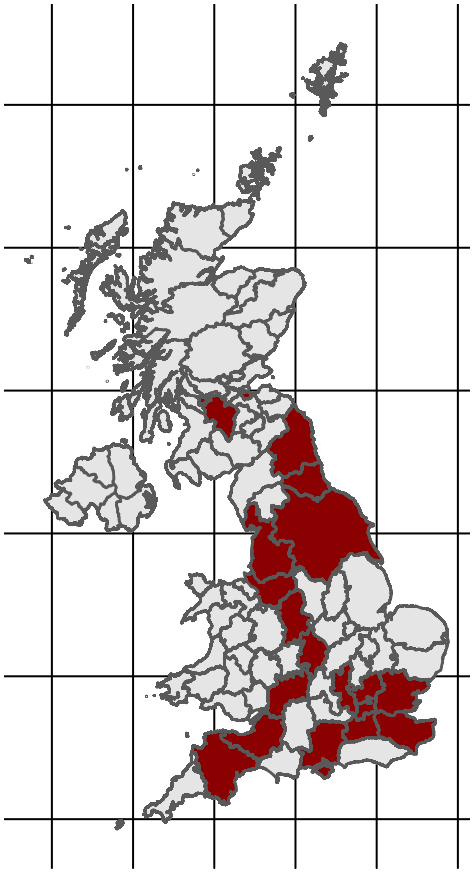
```r
#merge the birthplace data with the playing ability data
british_player_data <- merge(british_player_birthplaces, select(all_players_data, id, 49:60))

#find the number of FIFA players for each county
county_teams <- data.frame(table(british_player_data$county)) %>%
  merge(.,
        data.frame(table(british_player_data$county[which(british_player_data$symmetric_position == "GK"
        by = "Var1")
names(county_teams) <- c("county", "players", "gks")

#select only counties that can field a team
#at least 1 goalkeeper and 10 outfield players
county_teams <- county_teams %>%
  mutate(players = players - gks) %>%
  filter(players >= 10) %>%
  filter(gks >= 1)

#plot the counties which can field a whole team
p <- ggplot(data = uk_counties) +
  geom_sf() +
  geom_sf(data = uk_counties[which(uk_counties$county %in%
                                     county_teams$county),], fill = "darkred") +
  theme_void()

plot(p)
```



and, as before, use these to pick the optimal teams for these counties

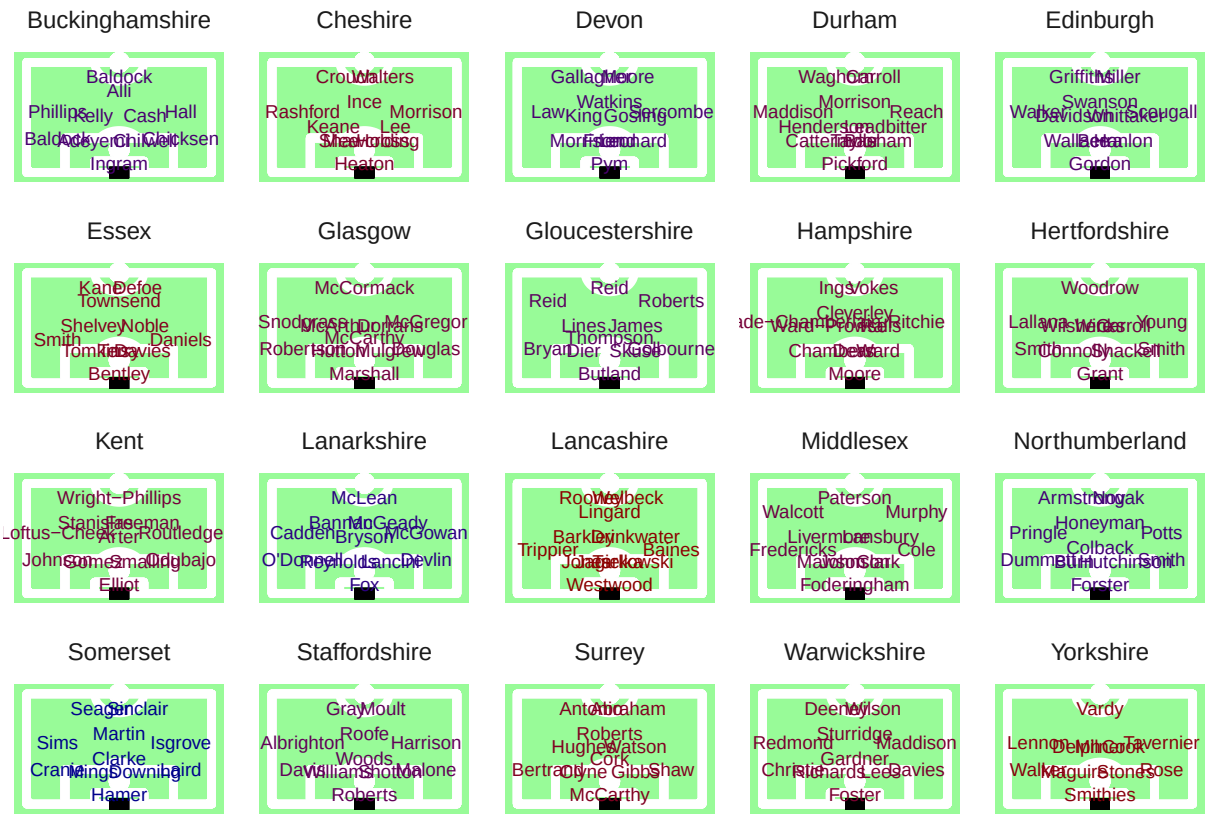
```
#find the optimal team for each county
optimal_county_teams <- rbindlist(lapply(county_teams$county, find_optimal_team,
                                         select(british_player_data, id, nationality = county, 49:60),
                                         replicates = 100))
```

This yields some interesting potential teams...

```
#merge in player names and position coordinates
county_elevens <- optimal_county_teams %>%
  setDT() %>%
  .[, unique_position := make.unique(as.character(position)), by = "nation"] %>%
  merge(., formation_coords, by = c("formation", "unique_position")) %>%
  merge(players, by = "id")

#plot the data
p <- ggplot(data = county_elevens)
p <- p %>%
  draw_pitch()
p <- p +
  geom_text(aes(x = player_x, y = player_y, label = gsub(".* ", "", name), colour = total_ability), size
  scale_colour_gradient(high = "darkred", low = "darkblue", guide = FALSE) +
  facet_wrap(~nation)

plot(p)
```



Which is a bit too small to make out. If we just plot the 4 best teams (Yorkshire, Lancashire, Essex, and

Surrey)

```
#select the best 4 county teams by total ability
county_elevens %<>%  .[total_ability >= abs(sort(unique(-county_elevens$total_ability)))[4]]

#plot the data
p <- ggplot(data = county_elevens)
p <- p %>%
  draw_pitch()
p <- p +
  geom_text(aes(x = player_x, y = player_y, label = gsub(" ", "\n", name)), colour = "black", size = 2.5
  facet_wrap(~nation)

plot(p)
```