Comprehensive Examination Responses

Jake Thompson

2017-03-30

Introduction

This document is a compressed version of a more detailed document available online. Specifically, this document aims to answers the specific questions for my written comprehensive examination. As such, this document may reference parts of the more complete document, but will also be comprehensive in containing all information needed to adequately address the questions.

Gather data

The first step to this analysis is to gather the data for the English Premier League. For this analysis, I will scrape the data from ESPN. I will first define a function that can scrape the team names and webpages from the league page. This functions uses the **purrr** (Wickham, 2016a) and **rvest** (Wickham, 2016b) packages to scrape the information, and **dplyr** (Wickham & Francois, 2016) to format the output.

```
library(dplyr)
library(purrr)
library(rvest)

scrape_league <- function(x) {
  cont <- TRUE
  while(cont) {
    url_data <- safe_read_html(x)

    if(is.null(url_data[[1]])) {
      closeAllConnections()
      Sys.sleep(5)
    } else {
      url_data <- url_data[[1]]
      cont <- FALSE
    }
  }

  league_table <- url_data %>%
    html_nodes(css = "table") %>%
    html_table()
  league_table <- league_table[[1]]
  colnames(league_table) <- as.character(league_table[1,])
  colnames(league_table) <- make.names(colnames(league_table), unique = TRUE)
  league_table <- league_table[-1,]

  league_table <- league_table %>%
    select(club = TEAM, goals_for = `F`, goals_against = A, points = PTS) %>%
    mutate(club = trimws(club, which = "both"))

  teams <- url_data %>%
    html_nodes("td a") %>%
```

```
    html_text() %>%
    as.character() %>%
    trimws(which = "both")
  team_urls <- url_data %>%
    html_nodes("td a") %>%
    html_attr("href") %>%
    as.character()

  league_table <- league_table %>%
    left_join(data_frame(club = teams, club_url = team_urls), by = "club") %>%
    as_data_frame()

  return(league_table)
}
```

We can then scrape use the `scrape_league` function to get the team names and URLs.

```
safe_read_html <- safely(read_html)
epl <- scrape_league("http://www.espnfc.us/english-premier-league/23/table")
epl
#> # A tibble: 20 × 5
#>                     club goals_for goals_against points
#>                    <chr>     <chr>         <chr>  <chr>
#> 1                Chelsea        59            21     69
#> 2     Tottenham Hotspur        55            21     59
#> 3       Manchester City        54            30     57
#> 4             Liverpool        61            36     56
#> 5     Manchester United        42            23     52
#> 6               Arsenal        56            34     50
#> 7               Everton        51            30     50
#> 8  West Bromwich Albion        39            38     43
#> 9            Stoke City        33            42     36
#> 10         Southampton        33            36     33
#> 11      AFC Bournemouth        42            54     33
#> 12      West Ham United        40            52     33
#> 13              Burnley        31            42     32
#> 14              Watford        33            48     31
#> 15       Leicester City        33            47     30
#> 16       Crystal Palace        36            46     28
#> 17         Swansea City        36            63     27
#> 18            Hull City        26            58     24
#> 19        Middlesbrough        20            33     22
#> 20           Sunderland        24            50     20
#> # ... with 1 more variables: club_url <chr>
```

I then define a function for scraping the game information from each team's webpage.

```
scrape_team <- function(x, y) {
    x <- gsub("/index", "/fixtures", x, fixed = TRUE)

    cont <- TRUE
    while(cont) {
      url_data <- safe_read_html(x)

      if(is.null(url_data[[1]])) {
```

```
      closeAllConnections()
      Sys.sleep(5)
    } else {
      url_data <- url_data[[1]]
      cont <- FALSE
    }
  }
  date <- url_data %>%
    html_nodes(".headline") %>%
    html_text() %>%
    as.character()
  if ("LIVE" %in% date) {
    date[which(date == "LIVE")] <- format(Sys.Date(), "%b %d, %Y")
  }
  date <- mdy(date)
  home_team <- url_data %>%
    html_nodes(".score-home-team .team-name") %>%
    html_text() %>%
    as.character()
  away_team <- url_data %>%
    html_nodes(".score-away-team .team-name") %>%
    html_text() %>%
    as.character()
  home_score <- url_data %>%
    html_nodes(".home-score") %>%
    html_text() %>%
    as.character() %>%
    gsub(" ", "", x = .) %>%
    gsub( " *\\(.*?\\) *", "", x = .) %>%
    as.numeric()
  away_score <- url_data %>%
    html_nodes(".away-score") %>%
    html_text() %>%
    as.character() %>%
    gsub(" ", "", x = .) %>%
    gsub( " *\\(.*?\\) *", "", x = .) %>%
    as.numeric()
  competition <- url_data %>%
    html_nodes(".score-column.score-competition") %>%
    html_text() %>%
    as.character()

  team_data <- data_frame(
    date = date,
    home = home_team,
    away = away_team,
    home_goals = home_score,
    away_goals = away_score,
    competition = competition
  ) %>%
    arrange(date) %>%
    unique()
```

```
    abbrev <- as_data_frame(table(c(team_data$home, team_data$away))) %>%
      top_n(n = 1, wt = n) %>%
      select(Var1) %>%
      flatten_chr()

    if (nrow(team_data) < 3) {
      ret_data <- data_frame(
        club = y,
        abbrev = y,
        team_data = NA
      )
    } else {
      if (abbrev == "Sporting") {
        team_data$home[which(team_data$home == "Sporting")] <- y
        team_data$away[which(team_data$away == "Sporting")] <- y
        ret_data <- data_frame(
          club = y,
          abbrev = y,
          team_data = list(team_data)
        )
      } else {
        team_data <- filter(team_data, home != "Sporting", away != "Sporting")
        ret_data <- data_frame(
          club = y,
          abbrev = abbrev,
          team_data = list(team_data)
        )
      }
    }

    return(ret_data)
}
```

And then I use that function to scrape game data. After the game data is scraped, I filter to only include games within the Premier League, and do some cleaning (e.g., replace ESPN abbreviations with the real club name).

```
library(lubridate)
epl_games <- map2_df(.x = epl$club_url, .y = epl$club, .f = scrape_team)

team_lookup <- select(epl_games, -team_data)

epl_games <- bind_rows(epl_games$team_data) %>%
  unique() %>%
  arrange(date, home) %>%
  left_join(select(epl_games, -team_data), by = c("home" = "abbrev")) %>%
  rename(home_club = club) %>%
  left_join(select(epl_games, -team_data), by = c("away" = "abbrev")) %>%
  rename(away_club = club) %>%
  mutate(
    real_home = ifelse(is.na(home_club), home, home_club),
    real_away = ifelse(is.na(away_club), away, away_club),
    home = real_home,
    away = real_away
```

```
  ) %>%
  select(-(home_club:real_away)) %>%
  filter(!(date < Sys.Date() & is.na(home_goals))) %>%
  filter(date > ymd("2016-03-01")) %>%
  rename(h_goals = home_goals, a_goals = away_goals) %>%
  filter(home %in% epl$club, away %in% epl$club, competition == "Prem",
    !is.na(h_goals))

knitr::kable(head(epl_games), caption = "English Premier League Games")
```

Table 1: English Premier League Games

date	home	away	h_goals	a_goals	competition
2016-08-13	Middlesbrough	Stoke City	1	1	Prem
2016-08-13	Burnley	Swansea City	0	1	Prem
2016-08-13	Crystal Palace	West Bromwich Albion	0	1	Prem
2016-08-13	Everton	Tottenham Hotspur	1	1	Prem
2016-08-13	Hull City	Leicester City	2	1	Prem
2016-08-13	Manchester City	Sunderland	2	1	Prem

Estimate the models

References

Wickham, H. (2016a). *Purrr: Functional programming tools.*

Wickham, H. (2016b). *Rvest: Easily harvest (scrape) web pages.* Retrieved from https://CRAN.R-project.org/package=rvest

Wickham, H., & Francois, R. (2016). *Dplyr: A grammar of data manipulation.* Retrieved from https://CRAN.R-project.org/package=dplyr