NHL Stanley Cup Playoffs Sports Reference Scraper

Adam Kiehl

7/8/20

This is a script that scrapes NHL Stanley Cup Playoff data from Hockey Reference (https://www.hockey-reference.com). Specifically, it extracts team-aggregated playoff statistics (https://www.hockey-reference.com/playoffs/) and individual skater/goalie playoff statistics (https://www.hockey-reference.com/teams/) for every playoff between 1999-2019. This code was originally sourced from Lyft Research Science Manager, Sean Taylor (https://github.com/seanjtaylor/learning-the-draft) and was modified with assistance from Colorado State University Ph.D student Connor Gibbs (https://github.com/ConGibbs10) and Philip Bulsink's Hockey and Chemistry Blog (https://pbulsink.github.io/blog/2016-12-26/scraping_player_data.html).

Variable Setup

Vectors and a mapping were compiled for later use in URL indexing. A `years` vector purposefully doesn't include the 2005 season due to a season-cancelling labor lockout. A `franchises` vector denotes teams by their standard abbreviations according to Sports Reference. A mapping was created between the two vectors in the form of `franchise/year`. Finally, corrections were made for the following:
1. Columbus Blue Jackets and Minnesota Wild being added as expansion teams in 2000
2. The Mighty Ducks of Anaheim being renamed to the Anaheim Ducks in 2006
3. The Phoenix Coyotes being renamed to the Arizona Coyotes in 2014
4. The Atlanta Thrashers becoming the Winnipeg Jets in 2011
5. The Vegas Golden Knights being added as an expansion team in 2017

```
years <- c('1999', '2000', '2001', '2002', '2003', '2004', '2006', '2007', '2008', '2009', '2010', '201
           '2015', '2016', '2017', '2018', '2019')
franchises <- c('ANA', 'PHX', 'BOS', 'BUF', 'CGY', 'CAR', 'CHI', 'COL', 'CBJ', 'DAL', 'DET', 'EDM', 'FL
                'NSH', 'NJD', 'NYI', 'NYR', 'OTT', 'PHI', 'PIT', 'SJS', 'STL', 'TBL', 'TOR', 'VAN', 'WS
map <- c()
for (i in franchises) {
  for (j in years) {
    map <- c(map, str_c(i, '/', j))
  }
}
map <- map[map %notin% c('CBJ/1999', 'CBJ/2000', 'MIN/1999', 'MIN/2000')]
map <- map[map %notin% c('ANA/1999', 'ANA/2000', 'ANA/2001', 'ANA/2002', 'ANA/2003', 'ANA/2004', 'ANA/2
map <- c(map, c('MDA/1999', 'MDA/2000', 'MDA/2001', 'MDA/2002', 'MDA/2003', 'MDA/2004', 'MDA/2006'))
map <- map[map %notin% c('PHX/2015', 'PHX/2016', 'PHX/2017', 'PHX/2018', 'PHX/2019')]
map <- c(map, c('ARI/2015', 'ARI/2016', 'ARI/2017', 'ARI/2018', 'ARI/2019'))
map <- map[map %notin% c('WPG/1999', 'WPG/2000', 'WPG/2001', 'WPG/2002', 'WPG/2003', 'WPG/2004', 'WPG/2
                         'WPG/2009', 'WPG/2010', 'WPG/2011')]
map <- c(map, c('ATL/2000', 'ATL/2001', 'ATL/2002', 'ATL/2003', 'ATL/2004', 'ATL/2006', 'ATL/2007', 'AT
                'ATL/2011'))
map <- c(map, c('VEG/2018', 'VEG/2019'))
```

A `headers` list was created containing headers for each table of interest. These headers replace the ones scraped from Sports Reference for consistency and naming ease.

```
headers <- list()
headers[['skaters_playoffs']] <- c('rank', 'player', 'age', 'pos', 'games', 'goals', 'assists', 'points
                                   'penalty.mins', 'even.goals','pp.goals', 'pk.goals', 'gw.goals', 'ev
                                   'pk.assists', 'shots', 'shot.percentage', 'toi', 'avg.toi')
headers[['goalies_playoffs']] <- c('rank', 'player', 'age', 'games', 'starts', 'wins', 'losses', 'ties'
                                   'save.percentage', 'avg.goals', 'shutouts', 'mins', 'quality.starts'
                                   'quality.start.percentage', 'really.bad.starts', 'goals.allowed.perc
                                   'goals.saved.above.average')
headers[['teams']] <- c('rank', 'team', 'games', 'wins', 'losses', 'ties', 'ot.wins', 'ot.losses', 'win
                        'goals.against', 'differential')
```

A `franchise.map` data frame was created mapping team names to their respective team codes. These codes will replace team names later for a standardized vocabulary between tables.

```
franchise.map <- list()
franchise.map[['team.code']] <- c('ANA', 'ARI', 'ATL', 'PHX', 'BOS', 'BUF', 'CGY', 'CAR', 'CHI', 'COL',
                                  'FLA', 'LAK', 'MDA', 'MIN', 'MTL', 'NSH', 'NJD', 'NYI', 'NYR', 'OTT',
                                  'TBL', 'TOR', 'VAN', 'WSH', 'WPG', 'VEG')
franchise.map[['team.name']] <- c('Anaheim Ducks', 'Arizona Coyotes', 'Atlanta Thrashers', 'Phoenix Coy
                                  'Buffalo Sabres', 'Calgary Flames', 'Carolina Hurricanes', 'Chicago Bl
                                  'Columbus Blue Jackets', 'Dallas Stars', 'Detroit Red Wings', 'Edmonto
                                  'Los Angeles Kings', 'Mighty Ducks of Anaheim', 'Minnesota Wild', 'Mon
                                  'Nashville Predators', 'New Jersey Devils', 'New York Islanders', 'New
                                  'Ottawa Senators', 'Philadelphia Flyers', 'Pittsburgh Penguins', 'San
                                  'Tampa Bay Lightning', 'Toronto Maple Leafs', 'Vancouver Canucks', 'Wa
                                  'Winnipeg Jets', 'Vegas Golden Knights')
franchise.map <- data.frame(franchise.map)
```

Function Setup

The `parse_tables` function is used to discriminate between all the tables found at a given URL. All the tables of a page and a table ID of interest are passed to the function and the table is returned with a revised header and a removed footer. This will be used specifically in the scraping of team playoff data.

```
parse_tables <- function(tables, tbl.id) {
  results = list()
  for (tbl in tables) {
    id <- html_attr(tbl, 'id')
    if (id %in% tbl.id) {
      df <- html_table(tbl) %>%
        as_tibble(.name_repair = 'universal') %>%
        slice(., -n())
      if(ncol(df) == length(headers[[tbl.id]])) {
        colnames(df) <- headers[[tbl.id]]
      } else {
        next;
      }
      results[[tbl.id]] <- df
    }
```

```
  }
  return(bind_rows(results))
}
```

Team Playoff Data

Here, scraping for the team-aggregated data is performed. https://www.hockey-reference.com/playoffs.html is indexed by year and the `teams` table is extracted from each year's page using the `parse_tables` function. A `team_playoffs.rds` file is written to document the data's structure and contents.

```
if(!file.exists('./data/team_playoffs.rds')){
  nodes <- list()
  for(year in years) {
    url <- paste('https://www.hockey-reference.com/playoffs/NHL_', year, '.html', sep ='')
    doc <- read_html(url)
    html.page <- doc %>%
        html_nodes('table') %>%
        parse_tables('teams')
    my.table <- html.page %>%
        mutate(., url = url)
    nodes[[year]] <- my.table
  }
  teams.table <- bind_rows(nodes, .id = 'year')
  write_rds(teams.table, './data/team_playoffs.rds', compress = 'xz')
}
```

Here, team-aggregated data is read from the `team_playoffs.rds` file, cleaned to include only fields of interest, and written to a `team_playoffs.csv` file. Additionally, an inner join was performed to match team names to their team codes and the team names were discarded.

```
if(!file.exists('./data/team_playoffs.csv')){
  team_playoffs <- read_rds('./data/team_playoffs.rds') %>%
    mutate(., url = if_else(str_sub(url, 1, 5) == 'http:',
                            str_c('https:', str_sub(url, 6, -1)),
                            url),
           key = ifelse(is.na(url),
                        year,
                        url)) %>%
    arrange(., year, wins, goals)
  team_playoffs <- inner_join(team_playoffs, franchise.map, by = c('team' = 'team.name')) %>%
    select(., year, team = team.code, games, wins, ot.wins, ot.losses, goals, goals.against) %>%
    mutate_if(., is.character, str_trim) %>%
    write.csv('./data/team_playoffs.csv')
}
```

The result of team playoff data scraping is shown below.

```
team_playoffs <- read_csv('./data/team_playoffs.csv')
head(team_playoffs)
```

```
## # A tibble: 6 x 22
```

```
##      X1  year team  games  wins goals goals.against points team.dps team.ops
##   <dbl> <dbl> <chr> <dbl> <dbl> <dbl>         <dbl>  <dbl>    <dbl>    <dbl>
## 1     1  1999 MDA       4     0     6            17      0   -0.201  -0.132
## 2     2  1999 OTT       4     0     6            12      1    1.24    0.257
## 3     3  1999 EDM       4     0     7            11      1    1.39   -0.0468
## 4     4  1999 CAR       6     2    10            16      4    2.64    0.889
## 5     5  1999 PHI       6     2    11             9      5    4.55    0.644
## 6     6  1999 SJS       6     2    17            19      6    1.38    5.63
## # ... with 12 more variables: team.gps <dbl>, dps.avg <dbl>, ops.avg <dbl>,
## #   gps.avg <dbl>, team.ps <dbl>, dps.prop <dbl>, dps.prop.adj <dbl>,
## #   ops.prop <dbl>, ops.prop.adj <dbl>, dps.full.prop <dbl>,
## #   dps.full.prop.adj <dbl>, champ <dbl>
```

Skater Playoff Data

Here, scraping for the individual skater data is performed. https://www.hockey-reference.com/teams.html is indexed by team and then by year according to the map defined above. The `skaters_playoffs` table is extracted from each year's page using the `readHTMLTable` function. HTML comment brackets are removed to unmask hidden tables. A `skaters_playoffs.rds` file is written to document the data's structure and contents.

```
if(!file.exists('./data/skaters_playoffs.rds')){
  nodes <- list()
  for (mapping in map) {
    url <- paste('https://www.hockey-reference.com/teams/', mapping, '.html', sep ='')
    doc <- read_html(url)
    html.page <- doc %>%
      gsub(pattern='<!--', replacement='') %>%
      gsub(pattern='-->', replacement='') %>%
      readHTMLTable()
    my.table <- html.page$skaters_playoffs
    if(length(my.table) != 0){
      names(my.table) = headers$skaters_playoffs
      my.table <- my.table %>%
        mutate(., url = url)
    }
    nodes[[mapping]] <- my.table
  }
  skaters.table <- bind_rows(nodes, .id = 'mapping')
  write_rds(skaters.table, './data/skaters_playoffs.rds', compress = 'xz')
}
```

Here, individual skater data is read from the `skaters_playoffs.rds` file, cleaned to include only fields of interest, and written to a `skaters_playoffs.csv` file.

```
if(!file.exists('./data/skaters_playoffs.csv')){
  skaters_playoffs <- read_rds('./data/skaters_playoffs.rds') %>%
  mutate(., url = if_else(str_sub(url, 1, 5) == 'http:',
                          str_c('https:', str_sub(url, 6, -1)),
                          url),
         key = ifelse(is.na(url),
                      year,
                      url),
```

```
        team = str_sub(mapping, end = 3),
        year = str_sub(mapping, start=5)) %>%
  arrange(., year, team, player) %>%
  select(., year, team, player, pos, goals, assists, plus.minus, toi) %>%
  mutate_if(., is.character, str_trim) %>%
  filter(pos %in% c('RW', 'C', 'LW', 'D')) %>%
  write.csv('./data/skaters_playoffs.csv')
}
```

The result of skater playoff data scraping is shown below.

```
skaters_playoffs <- read_csv('./data/skaters_playoffs.csv')
head(skaters_playoffs)
```

```
## # A tibble: 6 x 10
##    year team  player       pos   goals assists plus.minus   toi     dps      ops
##   <dbl> <chr> <chr>        <chr> <dbl>   <dbl>      <dbl> <dbl>   <dbl>    <dbl>
## 1  1999 BOS   Anson Carter F         4       3         -3   258 0.183    1.01
## 2  1999 BOS   Cameron Mann F         0       0          0     2 0.00234 -0.00886
## 3  1999 BOS   Darren Van ~ D         1       2         -3   196 0.191    0.428
## 4  1999 BOS   Dave Ellett  D         0       0          0    85 0.187   -0.153
## 5  1999 BOS   Dmitri Khri~ F         3       4          1   239 0.320    0.896
## 6  1999 BOS   Don Sweeney  D         3       0          2   240 0.687    0.740
```

Goalie Playoff Data

Here, scraping for the individual goalie data is performed. https://www.hockey-reference.com/teams.html is indexed by team and then by year according to the map defined above. The `goalies_playoffs` table is extracted from each year's page using the `readHTMLTable` function. HTML comment brackets are removed to unmask hidden tables. A `goalies_playoffs.rds` file is written to document the data's structure and contents.

```
if(!file.exists('./data/goalies_playoffs.rds')){
  nodes <- list()
  for (mapping in map) {
    url <- paste('https://www.hockey-reference.com/teams/', mapping, '.html', sep ='')
    doc <- read_html(url)
    html.page <- doc %>%
      gsub(pattern='<!--', replacement='') %>%
      gsub(pattern='-->', replacement='') %>%
      readHTMLTable()
    my.table <- html.page$goalies_playoffs
    if(length(my.table) != 0){
      names(my.table) = headers$goalies_playoffs
      my.table <- my.table %>%
        mutate(., url = url)
    }
    nodes[[mapping]] <- my.table
  }
  goalies.table <- bind_rows(nodes, .id = 'mapping')
  write_rds(goalies.table, './data/goalies_playoffs.rds', compress = 'xz')
}
```

Here, individual goalie data is read from the `goalies_playoffs.rds` file, cleaned to include only fields of interest, and written to a `goalies_playoffs.csv` file.

```
if(!file.exists('./data/goalies_playoffs.csv')){
  goalies_playoffs <- read_rds('./data/goalies_playoffs.rds') %>%
  mutate(., url = if_else(str_sub(url, 1, 5) == 'http:',
                          str_c('https:', str_sub(url, 6, -1)),
                          url),
         key = ifelse(is.na(url),
                      year,
                      url),
         team = str_sub(mapping, end = 3),
         year = str_sub(mapping, start=5)) %>%
  arrange(., year, team, player) %>%
  select(., year, team, player, goals, shots, mins) %>%
  mutate_if(., is.character, str_trim) %>%
  write.csv('./data/goalies_playoffs.csv')
}
```

The result of goalie playoff data scraping is shown below.

```
goalies_playoffs <- read_csv('./data/goalies_playoffs.csv')
head(goalies_playoffs)
```

```
## # A tibble: 6 x 7
##    year team  player           goals shots  mins     gps
##   <dbl> <chr> <chr>            <dbl> <dbl> <dbl>   <dbl>
## 1  1999 BOS   Byron Dafoe         26   330   768  2.07
## 2  1999 BUF   Dominik Hasek*      36   587  1217  4.83
## 3  1999 BUF   Dwayne Roloson      10    67   139 -0.107
## 4  1999 CAR   Arturs Irbe         15   181   408  1.05
## 5  1999 COL   Craig Billington     1     6     9 -0.0213
## 6  1999 COL   Patrick Roy*        52   650  1173  3.99
```

Note: This script only writes .rds and .csv files if they do not already exist in the `data` subdirectory. To attempt a fresh scrape, first delete the .rds and .csv files of interest and then run this file. After, the data can be easily drawn into other scripts using the `read_csv` function.