Project 1: NHL API Access and Analysis

Lee Pixton

6/20/2021

Contents

ST558 - NHL API Vignette

This is the first project for the course ST558 (Data Science for Statisticians) at NC State University. In this vignette, we will discuss reading data from the National Hockey League (NHL) API, and then providing summaries of the data pulled.

Required Project Packages

In order to run the code for the project, the required packages, as well as code to install them, are below.

- knitr
- httr
- RSQlite
- jsonlite
- tidyverse
- rmarkdown

Installing required packages

To install the packages required for this project, use the code chunk below, removing the beginning **#**

```
#install.packages("knitr", "httr", "RSQlite", "jsonlite", "rmarkdown", "tidyverse")
```

Function to Contact Franchise API

First thing to do is create a function that can access the franchise API. This will allow us to pull any number of information about each franchise. The `extension` option in the function takes in one of the following:

- `franchise` - Returns id, firstSeasonId and lastSeasonId and name of every team in the history of the nhl
- `franchise-team-totals` - Returns total stats for every franchise (ex roadTies, roadWins, etc)
- `franchise-season-records` - Returns drill-down into season records for a specific franchise
- `franchise-goalie-records` - Returns goalie records for the specified franchise
- `franchise-skater-records` - Returns skater records, same interaction as goalie endpoint
- `franchise-detail` - Returns captainHistory, coachingHistory, generalManagerHistory and a summary of retired numbers

And the `ID` option is available for specifying a team.

```
# This function is used to take input from the user and return appropriate data from the NHL franchise
franchiseAPI <- function(extension, ID = NULL){
  base <- "https://records.nhl.com/site/api/"

  if(is.null(extension)){
    return("Must include a valid franchise call")
  } else{
    if(!is.null(ID)){
      if(extension == "franchise" | extension == "franchise-team-totals"){
        warning("ID is not allowed for these calls, will return ", extension, "without ID")
        URL <- paste0(base, extension)
      } else if(extension == "franchise-detail"){
        URL <- paste0(base, extension, "?cayenneExp=mostRecentTeamId=", ID)
      } else{
        URL <- paste0(base, extension, "?cayenneExp=franchiseId=", ID)
      }
    } else{
      URL <- paste0(base, extension)
    }
  }

  get_nhl <- GET(URL)
  nhl_text <- content(get_nhl, "text")
  nhl_json <- fromJSON(nhl_text, flatten=T)
  return(tbl_df(nhl_json$data))
}
```

Once we have the function, the next step is to put together a mapping from team name to ID number. This allows the user to input the team name without having to know the specific ID. We will use our API call to retrieve this information into a dataframe, that we can then use to complete a mapping in the wrapper function. The user will be able to use the full name, the abbreviation, or just the mascot to pull the ID.

```
team_mapping <- franchiseAPI("franchise")
team_mapping <- team_mapping %>% select(!(ends_with("SeasonId") | id | ends_with("PlaceName")))
kable(team_mapping)
```

fullName	mostRecentTeamId	teamAbbrev	teamCommonName
Montréal Canadiens	8	MTL	Canadiens
Montreal Wanderers	41	MWN	Wanderers
St. Louis Eagles	45	SLE	Eagles
Hamilton Tigers	37	HAM	Tigers
Toronto Maple Leafs	10	TOR	Maple Leafs
Boston Bruins	6	BOS	Bruins
Montreal Maroons	43	MMR	Maroons
Brooklyn Americans	51	BRK	Americans
Philadelphia Quakers	39	QUA	Quakers
New York Rangers	3	NYR	Rangers
Chicago Blackhawks	16	CHI	Blackhawks
Detroit Red Wings	17	DET	Red Wings
Cleveland Barons	49	CLE	Barons
Los Angeles Kings	26	LAK	Kings
Dallas Stars	25	DAL	Stars
Philadelphia Flyers	4	PHI	Flyers
Pittsburgh Penguins	5	PIT	Penguins
St. Louis Blues	19	STL	Blues
Buffalo Sabres	7	BUF	Sabres
Vancouver Canucks	23	VAN	Canucks
Calgary Flames	20	CGY	Flames
New York Islanders	2	NYI	Islanders
New Jersey Devils	1	NJD	Devils
Washington Capitals	15	WSH	Capitals
Edmonton Oilers	22	EDM	Oilers
Carolina Hurricanes	12	CAR	Hurricanes
Colorado Avalanche	21	COL	Avalanche
Arizona Coyotes	53	ARI	Coyotes
San Jose Sharks	28	SJS	Sharks
Ottawa Senators	9	OTT	Senators
Tampa Bay Lightning	14	TBL	Lightning
Anaheim Ducks	24	ANA	Ducks
Florida Panthers	13	FLA	Panthers
Nashville Predators	18	NSH	Predators
Winnipeg Jets	52	WPG	Jets
Columbus Blue Jackets	29	CBJ	Blue Jackets
Minnesota Wild	30	MIN	Wild
Vegas Golden Knights	54	VGK	Golden Knights
Seattle Kraken	55	SEA	Kraken

Function to Contact Stats API

Next we need a function to access the stats API. Though there are many modifiers available to access multiple data sets, for our purposes we will only provide access to the `?expand=team.stats` modifier. By providing an `ID`, it will pull specific team data, but the function alone will pull data on all the teams.

```
# This function is used to take user input and output a tibble with data from the NHL Stats API
statsAPI <- function(ID = NULL){
  base <- "https://statsapi.web.nhl.com/api/v1/teams?expand=team.stats"
  if(!is.null(ID)){
    URL <- paste0(base,"=",ID)
  } else{
    URL <- paste0(base)
  }

  get_stats <- GET(URL)
  stats_text <- content(get_stats, "text")
  stats_json <- tbl_df(fromJSON(stats_text, flatten=T))
  stats_data <- stats_json$teams$teamStats

  if(!is.null(ID)){
    stats_data <- stats_json$teams$teamStats[stats_json$teams$id == ID]
    end_data <- stats_data[[1]]$splits[[1]]
  } else{
    end_data <- stats_data[[1]]$splits[[1]]

    for(i in c(2:length(stats_data))){
      stats_data <- stats_json$teams$teamStats
      end_data <- rbind(end_data,stats_data[[i]]$splits[[1]])
    }
  }
  return(tbl_df(end_data))
}
```

Wrapper Function for Both API Calls

Here we create a function that takes in the user input specific to either API.

```
# Allows for one function call to query either API
access_NHL_API <- function(extension, ID = NULL, ...){
  if(!is.numeric(ID) & !is.null(ID)){
    if(length(which(team_mapping$fullName == ID)) == 0){
      i <- which(team_mapping$fullName == ID)
    } else if(length(which(team_mapping$teamAbbrev == ID)) == 0){
      i <- which(team_mapping$teamAbbrev == ID)
    } else if(length(which(team_mapping$teamCommonName == ID)) == 0){
      i <- which(team_mapping$teamAbbrev == ID)
    } else{
      stop("Team ID not found!")
    }
  }

  if (extension == 'stats'){
    if(!is.null(ID)){
      ID <- team_mapping$teamAbbrev[i]
    }
    return(statsAPI(ID))
  } else {
    if(!is.null(ID)){
```

```
      ID = i
    }
    return(franchiseAPI(extension,ID))
  }
}
```

NHL Data Analysis Using Our API

First we need to pull the data. We will pull franchise data, team totals, and current season stats. We will conduct some filtering here to make the join simpler.

```
franchise <- access_NHL_API("franchise") %>% select(!(lastSeasonId | id))
team_totals <- access_NHL_API("franchise-team-totals") %>% filter(gameTypeId == 2 & activeFranchise == 
stats <- access_NHL_API("stats") %>% filter(!is.na(stat.gamesPlayed))
```

Combining Data

Next we need to combine these data together.

```
final_data <- right_join(franchise, team_totals, by = c("mostRecentTeamId" = "teamId")) %>% inner_join(
```