Package 'nhlscrapr'

July 2, 2014

Type Package

Title Compiling the NHL Real Time Scoring System Database for easy use in R

Version 1.5.1

Date 2014-06-10

Author A.C. Thomas, Samuel L. Ventura

Maintainer A.C. Thomas <act@acthomas.ca>

Depends R (>= 2.15.0)

Imports RCurl, rjson, biglm, bitops

Description Compiling the NHL Real Time Scoring System Database for easy use in R

License GPL (>= 3)

LazyData yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-06-10 22:12:58

R topics documented:

nhlscrapr-package *nhlscrapr: Retrieve and process play-by-play game data from the NHL Real Time Scoring System*

Description

This package contains routines for extracting play-by-play game data for regular-season and playoff NHL games, particularly for analyses that depend on which players are on the ice.

Details

This package processes data files of the type expanded summary (ES) and play-by-play (PL) for all possible games between 2002 and 2013, visual shift charts (SCH and SCV) for games between 2002 and 2007, and shot locations from 2008 until present. The purpose of this routine is to produce tables for players, games and on-ice events specifically to look at the impact of a player's contribution to how events unfold. The examples below illustrate how these tools can be used to produce these tables.

The command process.games() will download the appropriate game files and, for each game selected produce a table of events with all players on the ice and relevant information for each particular event, such as shot distance and type, goals and assists, and the zone in which the event took place (relative to the home team).

The command augment.game() takes a single game file and makes it more useful for the purpose of statistical modelling. It replaces player names with a unique player ID number, separates goaltenders from skaters, and obtains the event interval time. This routine is called for each game in the table by merge.to.mega.file() as it combines all games into one single R object, saving this object to disk along with the game table and unique player roster.

A user only need run one command, compile.all.games(), in order to get everything from the site, though this may take hours to download and compile.

Author(s)

A.C. Thomas <act@acthomas.ca>

Examples

```
## Not run:


  #What are all games that can/should be downloaded?
  #valid=FALSE implies previously screened problems.
  #Just a subset for testing.
  game.table <- full.game.database()[301:303,]

  #Get the details for these 20 games.
  print(game.table)
```

```
#Takes HTML files and (possibly) GIF images and produces event and player tables for each game.
process.games (game.table)

#Give me a single game record.
sample.game <- retrieve.game (season="20022003", gcode="20301")

#Augment all games and put them all in one big database.
compile.all.games (output.file="mynhlscrapes.RData")


#####################################################################
# Extras:


#Process games in parallel!
library(doMC)
registerDoMC(4)

res <- foreach (kk=1:dim(game.table)[1]) %dopar%
  {
    message (paste(kk, game.table[kk,1], game.table$gcode[kk]))
    item <- process.single.game(
              game.table[kk,1],
              game.table$gcode[kk],
              save.to.file=TRUE)
  }

## End(Not run)
```

---

capgeek *Download and process salary information from capgeek.com*

Description

Produces salary information from capgeek.com.

Usage

```
download.capgeek.files (max.player.num=2552,
                        files=1:max.player.num,
                        rdata.folder="nhlr-data",
                        just.process=FALSE,
                        verbose=TRUE,
                        retrieve.tables=TRUE)
```

Arguments

max.player.num	The highest capgeek.com database player number.
files	Which capgeek.com database entries should be retrieved.
rdata.folder	The location within the current directory to which to save the downloaded files. Will be created if it does not exist.
just.process	If TRUE, don't download the files if they are present in the rdata.folder.
verbose	Display additional information.
retrieve.tables	Brings

Value

download.capgeek.files() will not only download and save files to disk, but will process these results into tables.

Author(s)

A.C. Thomas <act@acthomas.ca>

---

fold.frames *fold.frames*

Description

Collapse a series of data frames with the same columns into one data frame.

Usage

```
fold.frames(frame.list)
```

Arguments

frame.list	A list of data frames.

Details

Most useful with compile.all.games().

Value

A single data frame with the same columns as the originals.

Author(s)

A.C. Thomas <act@acthomas.ca>

`full.game.database` *Create a shell database for NHL games from 2002 to the present, then download them*

Description

Creates a shell database for NHL games from 2002 to present, unfilled with game information. Uses this to collect and scrape NHL data, but not process them immediately.

Usage

```
full.game.database(extra.seasons=0)
download.single.game (season="20122013",
                      gcode="20001",
                      rdata.folder="nhlr-data",
                      verbose=TRUE)
download.games       (games=full.game.database(),
                      rdata.folder="nhlr-data")
```

Arguments

`extra.seasons`	Beyond 20132014, adds data for additional seasons (assuming nhlscrapr is not supported and updated before the season begins.
`season`	A character string for the two years specifying an NHL season.
`gcode`	The five-digit ID number for a particular NHL game.
`rdata.folder`	The location within the current directory to which to save the downloaded files. Will be created if it does not exist.
`verbose`	Report additional messages.
`games`	A game database, such as the one produced by full.game.database().

Details

full.game.database() gives ID numbers for all regular-season and playoff games played between 2002 and 2013, with indicators for whether any particular game is known to be unavailable.

download.single.game() retrieves the relevant files for a single game from NHL.com.

download.games() retrieves the files for all games in the table.

Value

full.game.database: a data frame with columns including season, session (Regular or Playoffs), game number/gcode (which game in the season?). Placeholders for teams, score and date of game are included to be filled in later. "valid" indicates whether a full record of the game is available for download.

download.single.game: returns a single Boolean value indicating if no errors were recorded during the download. Saves the game to disk, particularly the PL, ES, SCH and SCV files, along with JSON data for x-y events.

download.games: returns the input game database with "valid" changed to FALSE for any failed downloads.

Author(s)

A.C. Thomas <act@acthomas.ca>

Examples

```
  #Select a part of the history.
  game.table <- full.game.database()[201:220,]

  #Download one game.
  download.single.game(game.table$season[1], game.table$gcode[1])

  #Download all games.

## Not run:
  game.table.updated <- download.games (games=game.table)

## End(Not run)
```

---

nhlscrapr-data *nhlscrapr: Included Data Sets*

Description

Data sets included with the nhlscrapr package.

Usage

```
quadsarray
```

Format

quadsarray: a series of quadrilaterals that define distinct areas of the offensive zone.

Author(s)

A.C. Thomas <act@acthomas.ca>

NP.score *Calculate NP-tau score*

Description

Calculates the Net Probability score from a game table.

Usage

```
NP.score(grand.data, seconds=20)
```

Arguments

grand.data	A single data frame of all events to be considered.
seconds	The amount of time in seconds after the event with which we measure net goals.

Value

A table containing the goal counts for each team following the particular events and zones, as well as the calculated net probability.

Author(s)

A.C. Thomas <act@acthomas.ca>

References

Michael Schuckers and James Curro (2013) Total Hockey Rating (THoR): A comprehensive statistical rating of National Hockey League forwards and defensemen based upon all on-ice events. Sloan Sports Conference.

player.summary *Player summaries*

Description

Summarizes the event count for each player in the data frame.

Usage

```
player.summary (grand.data, roster.unique)
```

Arguments

grand.data	A single data frame of all events to be considered.
roster.unique	A list of all player IDs to consider.

Value

An array with event counts for each player in the data frame.

Author(s)

A.C. Thomas <act@acthomas.ca>

`process.games` *Given downloaded NHL games, produce event tables with players on ice.*

Description

Produces game tables from the NHL source files.

Usage

```
process.single.game (season="20122013",
                     gcode="20001",
                     rdata.folder="nhlr-data",
                     override.download=FALSE,
                     save.to.file=TRUE)
process.games       (games=full.game.database(),
                     rdata.folder="nhlr-data",
                     override.download=FALSE)
retrieve.game       (season="20122013",
                     gcode="20001",
                     rdata.folder="nhlr-data",
                     force=TRUE)
compile.all.games   (mega.file="nhlscrapr-probs.RData",
                     output.file=mega.file,
                     rdata.folder="nhlr-data",
                     new.game.table=NULL)
```

Arguments

`season`	A character string for the two years specifying an NHL season.
`gcode`	The five-digit ID number for a particular NHL game.
`rdata.folder`	The location within the current directory to which to save the downloaded files. Will be created if it does not exist.
`games, new.game.table`	A game database, such as the one produced by full.game.database().
`override.download`	Re-download the game files whether or not they are currently locally available.

save.to.file	Save the game object to file.
force	If TRUE, reprocesses the single game file.
output.file	The name of a file into which the game event list will be saved.
mega.file	Input file for storing all information.

Details

This group of functions takes the downloaded HTML, image and JSON files and produces an event table for each game, particularly focusing on the players on the ice during each event.

Value

process.single.game() produces an object called game.info that contains several pieces of information, primarily the data.frame playbyplay, with each column representing an event. Columns are as follows: event: The event number as recorded in the game. period, seconds: The period of the game and the elapsed time in seconds. a1,...,a6: The numbers and names of the away team players on the ice. h1,...,h6: The numbers and names of the home team players on the ice. ev.team: The team that registered the event in question. ev.player.1, ev.player.2, ev.player.3: The players involved in the event. distance: The distance in feet from the relevant goal for a shot on goal, missed shot or goal. type: Further information on the event, either shot type or penalty type. homezone: The zone in which the event took place, from the perspective of the home team. xcoord, ycoord: If available, the (x,y) location of a shot on goal or hit.

process.games() runs this routine and saves all game files to disk.

retrieve.game() retrieves the processed file from disk if it exists, and if not, it runs process.single.game() first.

compile.all.games() does everything that's needed to construct the entire record. This will be fastest if all games have already been downloaded and processed, possibly in parallel.

Author(s)

A.C. Thomas <act@acthomas.ca>

Index