Package 'ffanalytics'

April 9, 2016

Type Package

Title Scrape Data and Calculate Fantasy Football Projections

Version 0.1.1

Author Dennis Andersen

Maintainer Dennis Andersen <andersen.dennis@live.com>

Description Scrape data from multiple sources and calculate projected fantasy points for your drafts.

License GPL2

LazyData TRUE

Depends shiny,
miniUI

Imports data.table, stringr, DT, XML, httr

RoxygenNote 5.0.1

Collate 'calcStdDev.R'
'calculatePoints.R'
'calculateRisk.R'
'calculationDataPrep.R'
'columnVariables.R'
'confidenceInterval.R'
'createObject.R'
'data.R'
'dataGadget.R'
'dataPeriod.R'
'dataResult.R'
'externalConstants.R'
'ffanalytics.R'
'getAdpAav.R'
'getAdpData.R'
'getPlayerData.R'
'getPlayerName.R'
'getProjections.R'
'getRanks.R'
'sourceSite.R'
'sourceAnalyst.R'
'sourceTable.R'
'getUrls.R'
'helperFunctions.R'

'internalConstants.R'
'leagueScoring.R'
'mflPlayers.R'
'nflPlayerData.R'
'projectPoints.R'
'projectionGadget.R'
'readUrl.R'
'retriveData.R'
'runScrape.R'
'scoringUI.R'
'scrapeGadget.R'

R topics documented:

`analystOptions` *Analyst options for a period*

Description

Find the analysts that are projecting stats for the provided period

Usage

```
analystOptions(period)
```

Arguments

`scrapePeriod`	A dataPeriod object

`analystPositions` *Analyst positions*

Description

Data table identifying which positions the different analysts are projecting for

Usage

```
analystPositions
```

Format

A data.table with 151 rows and 4 columns

analystId An integer identifying the analyst the row is referring to. Refers to the linkanalyst table.
position Name of the position
season Indicates whether the analysts projects seasonal data for the position.
weekly Indicates whether the analysts projects weekly data for the position.

---

analysts *Analyst data*

Description

Data table with information on analysts that are bing used for data scrapes.

Usage

```
analysts
```

Format

A data.table with 27 rows and 7 columns

analystId Unique integer identifier for the analyst
analystName Name of analyst
siteId An integer identifying the site that the analyst is projecting for. Refers to sites table.
season Indicator of whether the analyst projects seasonal data or not
weekly Indicator of whether the analyst projects weekly data or not
sourceId The identifier that the site uses for the analyst if any
weight The weight used for the analyst in the weighted calculations

---

avgValue *Calculate average based on selected method*

Description

Will calculate average of provided data and weights based on the selected method

Usage

```
avgValue(calcMethod = "weighted", dataValue = as.numeric(),
  dataWeights = as.numeric(), na.rm = FALSE)
```

Arguments

calcMethod	One of c("average", "weighted", "robust")
dataValue	A numeric vector of values to average base on chosen calculation method.
dataWeights	A numeric vector of weight values associated with the dataValue parameter.
na.rm	A logical value determining if NA values should be removed.

calcStdDev *Calculate Standard Deviation*

Description

Standard Deviation is calculated based on method passed to the function.

Usage

```
calcStdDev(calcMethod = "weighted", dataValue = as.numeric(),
  dataWeights = as.numeric(), na.rm = FALSE)
```

Arguments

calcMethod	Calculation method. Can be one of "weighted", "robust" or "average"
dataValue	Vector of values to calculate standard deviation for
dataWeights	Vector of weights for weighted calculation

calculatePoints *Caluclate fantasy points*

Description

Based on list of scoring rules calculate fantasy points using projection data passed in a table

Usage

```
calculatePoints(projectionData = data.table(), scoringRules = list())
```

Arguments

projectionData	A data.table with projected stats. Should from the getMeltedData function
scoringRules	A list of scoring rules with one element per position

Value

A data.table with up to 4 cloumns

position The position the player is playing

playerId The ID of the player. Get player names by merging results from getPlayerData

analyst The ID of the analyst which projections are used as the basis for the points

points The calculated number of fantasy points

calculateRisk *Risk calculation based on to variables*

Description

Calculation of risk is done by scaling the standard deviation variables passed and averaging them before returning a measure with mean 5 and standard deviation of 2

Usage

```
calculateRisk(var1, var2)
```

calculateVor *Calculate Value Over Replacement*

Description

Based on provided ranks and points calculate the value over replacement. Function uses vorBaseline and vorAdjustment in the calculation. Please adjust these to match your league before running calculation.

Usage

```
calculateVor(ranks, points, position)
```

Arguments

ranks	Player ranks
points	Player Points
position	Player Position. Used to extract value from vorBaseline and vorAdjustment

clusterTier *Set tiers based on clusters*

Description

Set tiers based on clusters

Usage

```
clusterTier(points, position)
```

confidenceInterval *Calculate confidence interval*

Description

Confidence intervals are determined as percentiles (default 10 and 90). If the correcpsonding average value is less than the low point then we set the lower point to the minimum value. Conversely if the average value is higher than the upper limit then we set the upper limit to the max value.

Usage

```
confidenceInterval(calcMethod = "weighted", dataValue = as.numeric(),
  dataWeights = as.numeric(), pValues = c(0.1, 0.9), na.rm = FALSE)
```

Arguments

calcMethod	Calculation method. Can be one of "weighted", "robust" or "average"
dataValue	Vector of values to calculate confidence interval for
dataWeights	Vector of weights for weighted calculation
pValues	Vector of percentiles for the confidence interval. Defaults to `c(0.1, 0.9)` for 10 and 90 percentiles.

Constants *Package Constants*

Description

Constants build into the package

Usage

```
position.name

position.Id

yahooLeague

ffnAPI
```

Details

A number of constants are included in the package

createObject *Create objects from list*

Description

Allows creation of class objects based on a list.

Usage

```
createObject(objName = as.character(), targetValues = list())
```

Arguments

objName	Name of object to be created
targetValues	List of values representing slots in the object

dataGadget *Gadget used to display results of scrape calculations*

Description

Gadget used to display results of scrape calculations

Usage

```
dataGadget(inputData)
```

dataPeriod-class *Class to represent the period for a datascrape*

Description

The datascrapes need a week and season designation to work.

Slots

weekNo An integer representing the week number. 0 for preseason, 1-17 for regular season and 18-21 for post season

season A year representing the season. Should be 2008 or later but can't be higher than current year

Examples

```
dataPeriod(weekNo = 1, season = 2015) # Week 1 of the 2015 season
dataPeriod(season = 2015)             # 2015 season
dataPeriod(weekNo = 3)                # Week 3 of the current year
dataPeriod()                          # Current season
```

dataResult-class *Class to represent scrape results*

Description

Class to represent scrape results

Slots

`position` Position that the scrape data isfor

`resultData` A data.table holding the data. The columns in the table are determined by the staticColumns and resultColumns.

dropoffValue *Calculate Dropoff value*

Description

Calculate Dropoff value

Usage

```
dropoffValue(dataValue)
```

ffanalytics *Scraping and calculating data to use for fantasy football projections.*

Description

The ffanalytics package provides three categories of important functions: scrape, calculation and analysis.

Scrape functions

The scrape flow works like this:

1. User initiates the script and specifies the data period that needs to be scraped
2. The scripts displays available analysts to scrape and the user selects which to use
3. The script then displays available positions and asks the user to select positions to scrape.
4. Data scrape is executed and returns a list with a data table for each position

User can next specify which aggregate method to apply and execute the calculation scripts on this list to get a data table with projected points, confidence intervals, rankings, risk etc.

firstLast *Reverse Last and First Name*

Description

Takes an input string in the form of "Last Name, First Name" and converts it to "First Name Last Name".

Usage

```
firstLast(lastFirst)
```

Arguments

lastFist	A string with the name to be converted

Examples

```
firstLast("Smith, John") # Will return John Smith
```

getADPdata *Get ADP data*

Description

Retrieve ADP data from multiple sources and combine into average.

Usage

```
getADPdata(ADPsources = c("CBS", "ESPN", "FFC", "MFL", "NFL"),
  season = as.POSIXlt(Sys.Date())$year + 1900, teams = 12,
  format = "standard", mflMocks = NULL, mflLeagues = NULL)
```

Arguments

season	The season the ADP data is from
teams	Number of teams in the league
format	The format of the league, i.e. standard, ppr
mflMocks	Include mock drafts from MFL. Set to 1 if only mock drafts should be used, 0 if only real drafts should be used. If not speficied all types of drafts will be used.
mflLeagues	What type of leagues to include for MFL. Set to 0 to use redraft leagues only; 1 to only use keeper leagues, 2 for rookie drafts, and 3 for MFL Public Leagues. If not speficied all types of drafts will be used.
ADPsource	Character vector with one or more of "CBS", "ESPN", "FFC", "MFL", "NFL"

getCBSValues *ADP data from CBS*

Description

Retrieve ADP data from CBS

Usage

```
getCBSValues()
```

Value

data.table with 3 columns:

cbsId Player ID from CBS. Merge with results from getPlayerData to get player names

adp Average ADP

leagueType Assuming standard league type

getESPNValues *ADP and auction value data from ESPN*

Description

Retrieve ADP and auction value from ESPN

Usage

```
getESPNValues()
```

Value

data.table with 6 columns:

player Name of player

position Player position

team Team the player is playing for

adp Average ADP

aav Average auction value

leagueType Assuming standard league type

getFFCValues *ADP data from FantasyFootballCalculator.com*

Description

Retrieve ADP data from fantasyfootballcalculator.com.

Usage

```
getFFCValues(format = "standard", teams = 12)
```

Arguments

format	Format of league. Can be one of "standard", "ppr", "2qb", "dynasty", "rookie".
teams	Numer of teams in the league. One of 8, 10, 12, 14

Value

data.table with 5 columns:

player Name of player

position Player position

team Team the player is playing for

adp Average ADP

leagueType Chosen format for the league

getMeltedData *Melt data into long form*

Description

Takes the data result with a projected stat in each column and melts the data into one row per player per stat column.

Usage

```
getMeltedData(dataResult)
```

Arguments

dataResult	A dataResult object from the data scrape

`getMFLValues` *ADP and auction value data from MyFantasyLeague.com*

Description

Retrieve ADP and auction value data from MyFantasyLeague.com

Usage

```
getMFLValues(season = as.POSIXlt(Sys.Date())$year + 1900, type = "adp",
  teams = -1, ppr = -1, mock = -1, keeper = -1)
```

Arguments

`season`	Year the data is retrieved for
`type`	One of "adp" or "aav" to indicate whether ADP or auction values should be retrieved.
`teams`	Number of teams. If specified only drafts with that number of teams will be include
`ppr`	Specify if only ppr or non-ppr drafts should be considered. Set to 1 if only ppr drafts should be used, 0 if only standard drafts should be used. If not specified all types of draft will be used.
`mock`	Specify if only mock or real drafts should be used. Set to 1 if only mock drafts should be used, 0 if only real drafts should be used. If not speficied all types of drafts will be used.
`keeper`	Specify to select what types of leagues should be used. Set to 0 to use redraft leagues only; 1 to only use keeper leagues, 2 for rookie drafts, and 3 for MFL Public Leagues. If not speficied all types of drafts will be used.

Value

data.table wih up to 5 columns:

mflId Player ID from MFL Merge with results from getPlayerData to get player names

selectedIn Number of drafts player has been selected in

aav Average auction value

adp ADP

minPick Earliest pick

maxPick Latest pick

getNFLValues *ADP data from FantasyFootballCalculator.com*

Description

Retrieve ADP data from fantasyfootballcalculator.com.

Usage

```
getNFLValues()
```

Value

data.table with 5 columns:

esbid Player ID for the player. Merge with results from getPlayerData to get player names

player Name of player

position Player position

adp Average ADP

aav Average auction value

getPlayerData *Combine MFL and NFL player data*

Description

Retrieve data from MFL and NFL and combine

Usage

```
getPlayerData(season, weekNo, pos = position.name)
```

Arguments

season	The year that player data is to be retrieved from
weekNo	The weekNo that the player data is to be retrieved from
pos	A character vector with position names to be retrieved

Value

A data.table with 13 columns

playerId NFL's ID for the player

player Name of player

yahooId Yahoo's ID for the player

cbsId CBS's ID for the player

mflId MFL's ID for the player

position The position the player is playing

team The team the player is playing for

draftYear The year the player was drafted

birthData The players birth date

rookie A logical value indicating whether the player is a rookie

opponent Team the player is facing next

depthChart Number on the depth chart for the player 1 = starter

esbid Alternate ID for player. Used for ADP/AAV data from NFL

`getPlayerName` *Clean Player Data For Projections*

Description

For many of the the data soruces the player column contains more data than needed to identify the player. With the help of regular expression data such as position, team and injury information is cleaned from the player names.

Usage

```
getPlayerName(playerCol)
```

Arguments

`playerCol`	The vector of player data taken from the data table that is returned from the data scrape

Value

The updated vector of player data

`getProjections` *Projected points*

Description

Calculate projected points, confidence intervals, risk, tiers etc.

Usage

```
getProjections(scrapeData = NULL, avgMethod = "average",
  leagueScoring = scoringRules, teams = 12, format = "standard",
  mflMocks = NULL, mflLeagues = NULL, adpSources = c("CBS", "ESPN", "FFC",
  "MFL", "NFL"))
```

Arguments

scrapeData	The scraped data from runScrape
avgMethod	Which average method should be used for calculating averages
leagueScoring	List of scoring rules for the league see scoringRules for and example
teams	Number of teams in the league
format	League format
mflMocks	Include mock drafts from MFL. Set to 1 if only mock drafts should be used, 0 if only real drafts should be used. If not speficied all types of drafts will be used.
mflLeagues	What type of leagues to include for MFL. Set to 0 to use redraft leagues only; 1 to only use keeper leagues, 2 for rookie drafts, and 3 for MFL Public Leagues. If not speficied all types of drafts will be used.
ADPsource	Character vector with one or more of "CBS", "ESPN", "FFC", "MFL", "NFL"

---

getRanks *Expert Consensus Rankings*

Description

Reterieve expert consensus rankings from fantasypros.com

Usage

```
getRanks(rank.position = "consensus", leagueType = "std", weekNo = 0)
```

Arguments

rank.position	Position to retrieve ranks for. Use "consensus" to get overall rankings
leagueType	Indicate whether to get ppr rankings or standard rankings (std)
weekNo	Week number to retrieve ranks for. Use 0 for season ranks.

Value

data.table with up to 11 columns:

player Name of player

position Player's position

team Team the player is playing for

ecrRank Consensus Rank

bestRank Highest rank from experts

worstRank Lowest rank from experts

avgRank Average rank from experts

sdRank Standard deviation of ranks

adp Average Draft Position

vsAdp Difference between overall rank and ADP

rankType Will have value of "overall" or "position"

getUrls *URLs to scrape*

Description

Based on selected analysts, a given period and selected analysts generate URLs that will be scraped. URLs are generated as merges of the analysts, sites, siteUrls, and siteTables datasets.

Usage

```
getUrls(selectAnalysts = analysts$analystId,
  period = periodType(dataPeriod()), positions = position.name)
```

Arguments

selectAnalysts	An integer vector of selected analystIds. See analysts for possible values
period	A string indicating wether the URLs to produce are for weekly or seasonal data scrape
positions	A character vector of positions to scrape.

getYahooValues *ADP and auction value data from Yahoo*

Description

Retrieve ADP and auction value data from Yahoo

Usage

```
getYahooValues(type = "SD")
```

Arguments

type	Draft type: AD for auction draft, SD for standard draft.

Value

data.table with 3 columns:

cbsId Player ID from CBS. Merge with results from getPlayerData to get player names

adp/aav Average ADP if type = "SD"; Average auction value if type = "AD"

leagueType Assuming standard league type

mflPlayers *Read MFL Player Data*

Description

Function to read player data from MFL using the MFL API

Usage

```
mflPlayers(season = 2016, weekNo = 0, pos = position.name)
```

Arguments

season	The year that player data is to be retrieved from
weekNo	The weekNo that the player data is to be retrieved from
pos	A character vector with position names to be retrieved

Value

A data.table with 10 columns

playerId NFL's ID for the player
player Name of player
yahooId Yahoo's ID for the player
cbsId CBS's ID for the player
mflId MFL's ID for the player
position The position the player is playing
team The team the player is playing for
draftYear The year the player was drafted
birthData The players birth date
rookie A logical value indicating whether the player is a rookie

nflPlayerData *Player Data from NFL.com*

Description

Retrieve player data from NFL.com

Usage

```
nflPlayerData(season = 2016, weekNo = 0, positions = position.name)
```

Arguments

season	The year data is to be retrieved from
weekNo	The week that data is to be retrieved from
positions	A character vector of positions to be retrieved

Value

A data.table with 7 columns:

playerId NFL's ID For the player

player Name of player

position NFL's position designation for the player

team NFL Team that the player is playing for

opponent Team the player is facing next

depthChart Number on the depth chart for the player 1 = starter

esbid Alternate ID for player. Used for ADP/AAV data from NFL

---

periodType,dataPeriod-method *Determine if period is a week or season*

Description

Determine if period is a week or season

Usage

```
## S4 method for signature 'dataPeriod'
periodType(object)
```

Arguments

x	A dataPeriod object

---

projectPoints *Calculate points*

Description

For the scraped data, projected points, confidence interval, standard deviation and position ranks are calculated

Usage

```
projectPoints(projectionData, scoringRules, avgType = "average")
```

Arguments

projectionData	A data.table with projected stats
scoringRules	A list with tables for league scoring rules. See scoringRules for reference on format
avgType	Which average to use. Should be one of `average, robust, weighted`

readUrl *Read data from a URL*

Description

The task for this function is to read the data from the URL location and assign appropriate column names. The function will throw a warning if there are more columns in the data table than expected. If there are fewer columns than expected the function will retry up to 10 times to get the number of columns correct. If it fails after 10 tries then an error will be thrown.

Usage

```
readUrl(inpUrl, columnTypes, columnNames, whichTable, removeRow, dataType,
  idVar, playerLinkString)
```

Arguments

inpUrl	The URL to get data from
columnTypes	A character vector describing the types of columns in the data. Note: length(columnTypes) == leng
columnNames	A character vector describing the names of the columns in the data. Note: length(columnTypes) == length(columnNames).
whichTable	A number or character describing the table to get. This can be leveraged for HTML tables and spreadsheet files.
removeRow	A numeric vector indicating rows to skip at the top of the data. For exampe c(1,2) will skip the first two rows of data.
dataType	A character indicating the type of data (HTML, XML, file, xls)

Value

Returns a data.table with data from URL.

redistributeValues *Redistribute values*

Description

Allows for the redistribution of values from **one** variable to a set of others based on the averages. For example, the function can be used to redistribute total field goals to field goals per distances based of what the average values are for each of the field goals per distance.

Usage

```
redistributeValues(valueTable = data.table(), calcType = "weighted",
  fromVar = "fg", toVars = c("fg0019", "fg2029", "fg3039", "fg4049",
  "fg50"), excludeAnalyst = 20)
```

Arguments

valueTable	A data.table. Assumes outout from the getMeltedData function.
calcType	A string specifying which calculation method to use for the average values
fromVar	A string specifying the name of the variable to distribute from
toVars	A character vector with the names of variables to distribute to
excludeAnalyst	An integer indicating an analyst to exclude. This will exclude the analyst from the averages

---

replaceMissingData *Find replacement data for missing values*

Description

For analysts that don't report on certain values the averages across other analysts are calculated so they can be imputed.

Usage

```
replaceMissingData(statData = data.table(), calcType = "weighted")
```

Arguments

statData	A data.table. Assumes outout from the getMeltedData function.
calcType	A string specifying which calculation method to use for the average values

---

retrieveData *Retrive data from a source table*

Description

Data can be retrieved from a source table when specified along with a source analyst and a data period. The data scrape will translate the data columns for each source table into a uniform format.

Usage

```
retrieveData(srcTbl, srcPeriod, fbgUser = NULL, fbgPwd = NULL)
```

Arguments

srcTbl	A sourceTable object representing the table to get data from
srcPeriod	A dataPeriod object representing the period to get
fbgUser	User Name for an active footballguys.com account. Needed if data scrape is requested from Footballguys
fbgPwd	Password for an active footballguys.com account. Needed if data scrape is requested from Footballguys
srcAnalyst	A sourceAnalyst object representing the analyst to scrape data from

Value

scrapeData a scrapeData object

runScrape *Run a data scrape*

Description

Executing a data scrape based on inputs of week, season, analysts and positions. If no inputs are specified the user will be prompted.

Usage

```
runScrape(week = NULL, season = NULL, analysts = NULL, positions = NULL)
```

Arguments

week	The week number that the scrape is going to be executed for
season	The season that the scrape will be executed for
analysts	An integer vector of analystIds specifying which analysts to execute the scrape for. See analysts data set for values.
positions	A character vector of position names specifying which positions to execute the scrape for

Value

list of dataResults. One entry per position scraped.

Note

Historical scrapes are nearly impossible to do as very few if any sites actually stores their historical projections. An attempt to scrape historical projections will likely produce current projectsions in most cases.

Run_Scrape *Scrape gadget. Will be used as addin.*

Description

Scrape gadget. Will be used as addin.

Usage

```
Run_Scrape()
```

scoreThreshold *Default Scoring threshold for tiers*

Description

Default Scoring threshold for tiers

Usage

```
scoreThreshold
```

Format

An object of class `numeric` of length 9.

scoringRules *Default scoring rules*

Description

Default scoring rules

Usage

```
scoringRules
```

Format

An object of class `list` of length 7.

setTier *Set tiers based on thresholds*

Description

Set tiers based on thresholds

Usage

```
setTier(points, position)
```

sites *Site data*

Description

Data table with information on sites that are being used for datascrapes.

Usage

```
sites
```

Format

A data.table with 17 rows and 5 columns

siteId Unique integer identifier for the site

siteName The name of the site

subscription Indicator whether the site requires subscription to get to the data

playerId Name of column in the player data table that holds the id for the players

siteTables *Table information*

Description

Data table with information on tables that data will be scraped from.

Usage

```
siteTables
```

Format

A data.table with 111 rows and 9 columns.

tableId An integer that uniquely identifies the table

position Name of the position that the table holds data for

positionAlias The position identifier used in the URL for the table

siteId Integer identfying the site that the table is found on. See sites for values

startPage If the table spans more pages then the start number for the pages. If all the data is on one page the value is 1.

endPage If the table spans more pages then the end number for the page sequence. If all the data is on one page the value is 1

stepPage If the table spans more page then the step number of the page sequence. If all the data is on one page the value is 1

season Indicates whether the table can be used for seasonal data scrapes.

weekly Indicates whether the table can be used for weekly data scrapes.

siteUrls *URL information*

Description

Data table with information on the URLs that data will be scraped from.

Usage

```
siteUrls
```

Format

A data.table with 31 rows and 7 columns.

siteId An integer identifying the site the URL is referring to. See sites for values.

siteUrl The URL that the data will be scraped from. Use placeholders for parameters that needs to be substituted, like position, analyst, page, season and week.

urlPeriod Specifies whether the URL is used for week or season data.

urlType Specifies the data type that is returned from the URL. Could be HTML, XML, csv or file

urlTable The table number that the data scrape should read from the URL. Mostly used for HTML data, but can also be used to designate sheets in a spreadsheet file.

playerLink A string to identify the links to player profile pages on an HTML page.

sourceAnalyst-class *Class to represent the source Analysts*

Description

Class to represent the source Analysts

Slots

`analystName` The name of the analysts

`sourceId` The id for the source. This is only used if there are multiple analysts for a site

`analystId` A character string specifying the id to be used for the analyst. If left blank, it will be set as a 4 letter abbreviation of the site + analyst names using abbreviate.

Examples

```
cbs.avg <- sourceAnalyst(analystName = "Average",
                         sourceId = "avg",
                         analystId = "cbav")
```

sourceSite-class *Class to represent source sites*

Description

Source sites are one of the foundations to the data scrapes. They are representing the web sites that provides data for the projections. The sources can be different for weekly and seasonal data and the type of data can be different. If the data is scraped from HTML then the playerId can be derived as well.

Slots

`siteId` The ID for the site as identfied in configuration data

`siteName` Name of source site

`siteUrl` String that represents the URL that the site uses for the seasonal data, if any.

`urlType` String that identifies the type of data in seasonURL (HTML, XML, CSV or file)

`urlTable` A character string identifying which table to grab from seasonURL

`playerLink` A string representing part of the URL to a player profile

`playerId` What to call the id number for player if present

Note

When specifying the URLs paramters can be used as place holders.

Examples

```
sourceSite(siteName = "CBS",
         siteUrl = "http://www.cbssports.com/fantasy/football/stats/weeklyprojections/{$Pos}/season/{$SrcID
           urlType = "html",
           urlTable = "1",
           playerLink = "/fantasyfootball/players/playerpage/[0-9]{3,6}",
           playerId = "cbsId")
```

sourceTable-class *Class to represent a data table from a source site*

Description

Source table class extends the sourceAnalyst class

Slots

`sourcePosition` The position designation that the table represents

`positionAlias` The designation that the source site uses for the position

`startPage` If the table covers multiple pages, the start numbering for the pages. Otherwise 1

`endPage` If the table covers multiple pages, the end numbering for the pages. Otherwise 1

`stepPage` If the table covers multiple pages, the step in the sequence of page numbering. Otherwise 1

`tableId` ID from configuration data that identifies the table uniquely

Examples

```
cbs.qb <- sourceTable(sourcePosition = "QB",
                      positionAlias = "QB",
                      startPage = 1,
                      endPage = 1,
                      stepPage = 1,
                      tableId = 1
                     )
```

`tableColumns` *Constants defining result columns*

Description

The results in the dataResult table are determined by the values in these constants. See definitiions in details.

staticColumns Names of columns in all result tables.

resultColums A list of character vectors (one per position), indicating the columns in the results table for that position.

Data table with information on the columns in tables identified in siteTables.

Usage

```
staticColumns

resultColumns

tableColumns
```

Format

A data.table with 2924 rows and 6 columns.

tableId An integer identifying the table that the column belongs to. See siteTables for values

columnName The name of the column

columnType The data type for the column

columnOrder The order in which the column appears in the table

columnPeriod Indicates if the column appears in tables used for seasonal or weekly data scrapes

removeColumn Indicates if the column can be removed before the table is returned.

tableRowRemove *Table rows to remove*

Description

Data table with information on rows that will need to be removed from siteTables for the data scrape to be succesfull

Usage

```
tableRowRemove
```

Format

A data.table with 28 rows and 2 columns.

tableId An integer identifying the table that the row can be removed from. See siteTables for values

rowRemove An integer identifying the row that can be removed

tierGroups *Default number of tiers for clusters*

Description

Default number of tiers for clusters

Usage

```
tierGroups
```

Format

An object of class `numeric` of length 9.

vorAdjustment *Default VOR Adjustments*

Description

Default VOR Adjustments

Usage

```
vorAdjustment
```

Format

An object of class `numeric` of length 6.

vorBaseline *Default VOR Baseline*

Description

Default VOR Baseline

Usage

```
vorBaseline
```

Format

An object of class `numeric` of length 6.

Index