Package 'aRbs'

March 31, 2021

Title Find Arbitrage Opportunities for Sports Matches

Version 0.1.2

Maintainer Andrew Little <andrewlittlebristol@gmail.com>

Description Money doesn't grow on trees. Arbitrage opportunities do. Find the best arbitrage opportunities (arbs) in sports matches through <https://www.oddschecker.com/>. This package allows the user to input the URLs of <https://www.oddschecker.com/> subdomains in order to extract any opportunities for arbitrage. The majority of the functionality is implemented using the function `get_arbs_shiny()`, which simply provides an easy-to-use interface wrapping the functionality of the `get_arbs()` function. This function first finds all subdomains of the URL entered, then filters to only include those that look like event betting pages. Next, odds are scraped from the various bookmakers listed for that event, before returning combinations of bookmakers, odds or outcomes that could be used for arbitrage. Bets placed subsequently are done so at bettor's risk. Please see package README for full details.

License GPL-2

Imports data.table, crayon, dplyr, progress, stringr, xml2, rvest, purrr, parallel, stats, shiny, shinydashboard, shinycssloaders, DT, shinyBS

Depends R (>= 3.5.0)

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Andrew Little [aut, cre]

Repository CRAN

Date/Publication 2021-03-31 16:30:08 UTC

R topics documented:

`aRbs` *aRbs: A package for finding arbitrage opportunities*

Description

This package finds arbitrage opportunities from www.oddschecker.com and provides useful information so that the user can exploit this arbs in the most effective way.

Author(s)

Andrew Little <andrewlittlebristol@gmail.com>

`clean_links` *Clean links given by the* `scrape_links()` *function.*

Description

Filter links to only those ending in "winner" and clean for use in the `get_arb_single()` function.

Usage

```
clean_links(links)
```

Arguments

`links`	A `data.frame` of links with a column "url" containing www.oddschecker.com URLs, usually the output of the `scrape_links()` function.

Examples

```
links <- scrape_links("https://www.oddschecker.com/football")
clean_links(links)
```

get_arbs *Find Arbitrage Opportunities*

Description

Input a sport homepage URL as a string and return a list of arbitrage information. This will also print dataframes with details of the arbitrage opportunities, including the stake required for equal returns, the value of said equal returns, along with the relevant bookie for each outcome.

Usage

```
get_arbs(
  event = "https://www.oddschecker.com/football",
  in_play = FALSE,
  print_urls = FALSE,
  parallel = TRUE,
  debug = FALSE
)
```

Arguments

event	A URL to a www.oddschecker.com sport homepage, given as a string.
in_play	Logical. Should in-play arbitrage opportunities (arbs) also be returned? These are not likely to be accurate arbs as some bookie's odds will not be up-to-date, therefore default is FALSE.
print_urls	Logical. Should the URL of the event(s) be printed to the console while searching for arbitrage opportunities? Passed to get_arb_single.
parallel	Logical. Should iterative calls to get_arb_single be made in parallel.
debug	Logical. If set to TRUE, print_urls will be turned on and parallel will be turned off (so that URLs can be printed continuously).

Details

The workhorse function, clearly, is get_arb_single, however get_arbs implements it iteratively, after scraping and cleaning the required inputs for each distinct call to get_arb_single. This includes finding all the sub-URLs of the homepage. A progress bar is also used as the runtime is significant.

Value

A list, with arbitrage information printed as series of dataframes.

Examples

```
get_arbs()
get_arbs("https://www.oddschecker.com/football") # equivalent to get_arbs()
```

`get_arbs_shiny` *Run {aRbs} shiny app*

Description

Shiny app displaying the functionality of `get_arbs()`.

Usage

```
get_arbs_shiny()
```

`get_arb_single` *Find arbitrage opportunity information for an www.oddschecker.com event*

Description

Find (among other information) the implied total probability of all listed outcomes, Pr(Omega), and if an arb is available, the best bookie combination to exploit the arb.

Usage

```
get_arb_single(event, full = FALSE, print_urls = FALSE, in_play = TRUE)
```

Arguments

`event`	A www.oddschecker.com event URL, given as a string. This will be of the form: "https://www.oddschecker.com/football/english/championship/bournemouth-v-reading/winner"
`full`	A logical indicating whether or not the best bookie choices should still be returned for the event, even if there is no arb available.
`print_urls`	Logical. Should the URL of the event(s) be printed to the console while searching for arbitrage opportunities?
`in_play`	Logical. Should in-play arbitrage opportunities (arbs) also be returned?

Details

This function assumes that the listed outcomes are the only ones that are possible. Sometimes bookies may not offer odds on a realistic outcome, often if they feel like it is extremely likely to happen and therefore would offer next to no returns for any punter. This may mean that an outcome isn't listed and thus the function assumes it isn't possible. Therefore, the results of this function may be incorrect and assume a huge arbs is possible, when it isn't. It is therefore required that the user check the given outcomes given in the results, to ensure they form a complete set. Often, this is obvious however, as they are simply win, draw or lose.

---

oddschecker2 *Find the odds of each outcome of an event from www.oddschecker.com*

Description

Find the odds for each of the listed outcomes of an event on www.oddschecker.com, for each of the listed bookies. This function is essentially a re-exported fix of the function `oddschecker` from the gambleR package.

Usage

```
oddschecker2(event, in_play = TRUE)
```

Arguments

`event`	A www.oddschecker.com event path, given as a string. This is essentially an event URL with the "www.oddschecker.com/" base removed.
`in_play`	Logical. Should in-play arbitrage opportunities (arbs) also be returned? If `FALSE` and event is in-play, then a NULL `data.frame` is returned.

---

print.arb *Print method for class arb*

Description

An S3 method for printing objects of class `arb`. This will print the `best_choice` elements from

Usage

```
## S3 method for class 'arb'
print(x, ...)
```

Arguments

`x`	An object of class `arb` - usually the output of a call to `get_arbs()`
`...`	Arguments passed to `base::print.default()`.

Details

Prints the the best choices of arbitrage opportunities.

`scrape_links` *Scrape all subdomain URLs of a domain*

Description

Scrape all subdomain URLs of a domain

Usage

```
scrape_links(url)
```

Arguments

`url`	A head domain URL, given as a string.

Details

Used in the call to `get_arbs()` to find subdomains that may offer events, from which we can find an arb.

Value

A `data.frame`.

Examples

```
scrape_links("https://www.oddschecker.com/football")
```

Index