Package ‘valection’

February 6, 2018

Type Package

Title Sampler for Verification Studies

Version 1.0.0

Date 2018-02-05

Author Chris Cooper [aut],
Dorota H. Sendorek [ctb],
Paul C. Boutros [cre, cph]

Maintainer Paul C. Boutros <Paul.Boutros@oicr.on.ca>

Description A binding for the 'valection' program which offers various ways to sample the outputs of competing algorithms or parameterizations, and fairly assess their performance against each other. The 'valection' C library is required to use this package and can be downloaded from: <http://labs.oicr.on.ca/boutros-lab/software/valection>. Cooper CI, et al; Valection: Design Optimization for Validation and Verification Studies; Biorxiv 2018; <doi:10.1101/254839>.

Depends R (>= 3.1.0)

SystemRequirements valection (>= 1.0.0)

URL http://labs.oicr.on.ca/boutros-lab/software/valection

License GPL-3

Imports testthat

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2018-02-06 09:29:33 UTC

R topics documented:

run.decreasing.with.overlap

Run the decreasing with overlap algorithm

Description

Runs the decreasing with overlap algorithm from the valection library.

Usage

```
run.decreasing.with.overlap(budget, infile, outfile, seed);
```

Arguments

`budget` An integer specifying the number of candidates to select.

`infile` Path to input file. It should be formatted with a tab separating the caller and call on each line.

caller1 name a call this caller made caller2 name a call this caller made

`outfile` Path to a filename where the calls should be outputted.

`seed` An integer specifying the random seed value. Optional.

Details

Sampling calls where the likelihood of a call getting selected is inversely proportional to the number of callers that made the call.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.decreasing.with.overlap(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_decreasingWithOverlap.txt"
);

## End(Not run)
```

run.directed.sampling *Run the directed sampling algorithm*

Description

Runs the directed sampling algorithm from the valection library.

Usage

```
run.directed.sampling(budget, infile, outfile, seed);
```

Arguments

budget	An integer specifying the number of candidates to select.
infile	Path to input file. It should be formatted with a tab separating the caller and call on each line. caller1 name a call this caller made caller2 name a call this caller made
outfile	Path to a filename where the calls should be outputted.
seed	An integer specifying the random seed value. Optional.

Details

Sampling calls where a) an equal number of calls is selected from each caller and b) the likelihood of a call getting selected is proportional to the number of callers that made it.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.directed.sampling(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_directedSampling.txt"
);

## End(Not run)
```

`run.equal.per.caller` *Run the equal per caller algorithm*

Description

Runs the equal per caller algorithm from the valection library.

Usage

```
run.equal.per.caller(budget, infile, outfile, seed);
```

Arguments

`budget` An integer specifying the number of candidates to select.

`infile` Path to input file. It should be formatted with a tab separating the caller and call on each line.

caller1 name a call this caller made caller2 name a call this caller made

`outfile` Path to a filename where the calls should be outputted.

`seed` An integer specifying the random seed value. Optional.

Details

Sampling calls where an equal number of calls is selected from each caller.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.equal.per.caller(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_runEqualPerCaller.txt"
);

## End(Not run)
```

run.equal.per.overlap *Run the equal per overlap algorithm*

Description

Runs the equal per overlap algorithm from the valection library.

Usage

```
run.equal.per.overlap(budget, infile, outfile, seed);
```

Arguments

budget	An integer specifying the number of candidates to select.
infile	Path to input file. It should be formatted with a tab separating the caller and call on each line. caller1 name a call this caller made caller2 name a call this caller made
outfile	Path to a filename where the calls should be outputted.
seed	An integer specifying the random seed value. Optional.

Details

Sampling calls by, first, grouping calls by number of callers making the call and, second, selecting an equal number of calls from each group.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.equal.per.overlap(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_equalPerOverlap.txt"
);

## End(Not run)
```

`run.increasing.with.overlap`

Run the increasing with overlap algorithm

Description

Runs the increasing with overlap algorithm from the valection library.

Usage

```
run.increasing.with.overlap(budget, infile, outfile, seed);
```

Arguments

`budget`	An integer specifying the number of candidates to select.
`infile`	Path to input file. It should be formatted with a tab separating the caller and call on each line. caller1 name a call this caller made caller2 name a call this caller made
`outfile`	Path to a filename where the calls should be outputted.
`seed`	An integer specifying the random seed value. Optional.

Details

Sampling calls where the likelihood of a call getting selected is proportional to the number of callers that made the call.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.increasing.with.overlap(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_increasingWithOverlap.txt"
);

## End(Not run)
```

run.random.sampling *Run the random sampling algorithm*

Description

Runs the random sampling algorithm from the valection library.

Usage

```
run.random.sampling(budget, infile, outfile, seed);
```

Arguments

`budget`	An integer specifying the number of candidates to select.
`infile`	Path to input file. It should be formatted with a tab separating the caller and call on each line. caller1 name a call this caller made caller2 name a call this caller made
`outfile`	Path to a filename where the calls should be outputted.
`seed`	An integer specifying the random seed value. Optional.

Details

Sampling calls randomly where each call has an equal probability of getting selected.

Author(s)

Chris Cooper

Examples

```
## Not run:
run.random.sampling(
budget  = 5,
infile  = system.file("extdata/infile_example.tsv", package = "valection"),
outfile = "outfile_randomSampling.txt"
);

## End(Not run)
```

Index