Package 'exSamp'

April 6, 2021

Title What the Package Does (One Line, Title Case)

Version 0.0.1.1000

Description What the package does (one paragraph).

License Apache License (¿= 2)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Depends R (¿= 4.0.3)

Suggests testthat,
dplyr,
ggplot2,
covr,
moments

Config/testthat/edition 3

R topics documented:

`analyticalThompsonSampling`

Analytical Thompson shares

Description

Get the analytical Thompson shares based on the characteristics of the distributions of treatments.

Usage

```
analyticalThompsonSampling(dist_cache)
```

Arguments

`dist_cache`	List containing the characteristics of the distributions of treatments.

Value

List that contains (i) the analytical Thompson shares and (ii) the absolute error associated to the numerical integration procedure. TODO: be clear about the distribution. Only implemented for bernoulli.

`bayesianUpdateBinomial`

Bayesian update for a Binomial outcome

Description

Obtain the posterior of the parameter theta for an outcome distributed as a Binomial with prior Beta.

Usage

```
bayesianUpdateBinomial(treatment, outcome, alpha0 = 1, beta0 = 1)
```

Arguments

`treatment`	A vector.
`outcome`	A vector.
`alpha0`	A vector.
`beta0`	A vector.

Value

A list with the distribution and parameters of the posterior distribution of theta.

bayesianUpdatePoisson *Bayesian update for a poisson outcome*

Description

Obtain the posterior of the parameter theta for an outcome distributed as poisson with prior Gamma.

Usage

```
bayesianUpdatePoisson(treatment, outcome, alpha0 = 1, beta0 = 0)
```

Arguments

treatment	A vector.
outcome	A vector.
alpha0	A vector.
beta0	A vector.

Value

A list with the distribution and parameters of the posterior distribution of theta.

BinomialExplorationSampling *Full exploration sampling method*

Description

Get exploration sampling shares from a sample of outcomes and treatments.

Usage

```
BinomialExplorationSampling(
  treatment,
  outcome,
  alpha0 = 1,
  beta0 = 1,
  sample_resolution = 1e+05,
  treatment_cost = NA
)
```

Arguments

treatment	A vector.
outcome	A vector.
alpha0	A vector.
beta0	A vector.
sample_resolution	A number.
treatment_cost	A vector.

Value

A vector with the proportion shares, obtained through exploration sampling, for each treatment.

`cdfProductBeta` *Product of 1 Beta pdf and k-1 Beta cdfs*

Description

Calculate the product of 1 Beta pdf and k-1 Beta cdfs, where k is the number of treatments. The selected treatment that characterizes the Beta pdf is declared with the argument index. The k-1 beta cdfs are calculated using the distributions of the remaining treatments.

Usage

```
cdfProductBeta(x, dist_cache, index)
```

Arguments

`x`	Value at which the pdfs and cdfs are evaluated.
`dist_cache`	List containing the characteristics of the distributions of treatments.
`index`	Scalar denoting the treatment associated to the beta pdf.

Value

product of the beta pdf and k-1 beta cdfs.

`evaluateDiffThompsonShares` *Difference between Thompson shares and analytical Thompson shares*

Description

Get the difference between Thompson shares and analytical Thompson shares.

Usage

```
evaluateDiffThompsonShares(thompson_shares, analytical_thompson_shares)
```

Arguments

`thompson_shares`	Vector containing Thompson shares.
`analytical_thompson_shares`	List containing the analytical Thompson shares and their associated absolute error.

Value

Minimum difference between Thomson shares and analytical Thompson shares.

expectedRegret *Expected regret*

Description

Generate the expected policy regret of the sample allocation based on p-shares and true treatment effects.

Usage

```
expectedRegret(shares, true_treatment_effects)
```

Arguments

shares	A vector.
true_treatment_effects	A vector.

Value

A number equal to the expected policy regret.

explorationSampling *Exploration sampling*

Description

Get exploration sampling shares by modifying the proportion shares obtained through Thompson sampling.

Usage

```
explorationSampling(prop_shares)
```

Arguments

prop_shares	A vector.

Value

A vector with the proportion shares, obtained through exploration sampling, for each treatment.

`generateBinomialOutcome` *Simulate a binomial outcome*

Description

Simulate a single binomial outcome based on a vector of true treatment effects.

Usage

```
generateBinomialOutcome(single_treatment, true_theta)
```

Arguments

`single_treatment`	A number between 1 and the number of treatments.
`true_theta`	An ordered vector with the probabilities of success of each treatment. Its length must be equal to the number of treatments.

Value

A simulated binomial outcome.

`inSampleRegret` *In-sample regret*

Description

Generate the in-sample regret of the sample allocation based on p-shares and true treatment effects.

Usage

```
inSampleRegret(treatment, true_treatment_effects, sample_size)
```

Arguments

`true_treatment_effects`	A vector.
`shares`	A vector.

Value

A number equal to the expected policy regret.

`optimalSelected` *Optimal treatment selected*

Description

Find whether the optimal treatment has the highest proportion share.

Usage

```
optimalSelected(shares, true_treatment_effects)
```

Arguments

`shares`	A vector.
`true_treatment_effects`	A vector.

Value

1 if the highest proportion treatment is the optimal treatment, 0 otherwise.

`proportionalAssignment` *Sample based on proportions*

Description

Generate a sample of size sample_size based on the proportions in the vector shares.

Usage

```
proportionalAssignment(shares, sample_size)
```

Arguments

`shares`	A vector.
`sample_size`	A number.

Value

A vector with the new sample based on the proportions in the vector shares.

sampleDistribution *Sample from a posterior distribution.*

Description

Sample from a posterior distribution.

Usage

```
sampleDistribution(dist_cache, sample_resolution = 50000)
```

Arguments

dist_cache	A list with the structure of a bayesianUpdateBinomial output.
sample_resolution	A number.

Value

A matrix with the samples drawn from the posterior distribution.

thompsonSampling *Thompson sampling*

Description

Use Thompson sampling to find the proportion shares of each treatment.

Usage

```
thompsonSampling(theta_matrix, treatment_cost = NA)
```

Arguments

theta_matrix	A matrix.
treatment_cost	A vector.

Value

A vector with the proportion shares, obtained through Thompson sampling, for each treatment.

Index