Package 'elections.dtree'

January 20, 2023

Title Ranked Voting Election Audits with Dirichlet-Trees

Description Perform ballot-polling Bayesian audits for ranked voting elections using Dirichlet-tree prior distributions. Everest et al. (2022) <arXiv:2206.14605>, <arXiv:2209.03881>.

Version 1.1.2

Depends R (>= 4.0.0)

SystemRequirements C++17

License GPL-3

RoxygenNote 7.2.2

LinkingTo Rcpp, RcppThread, testthat

Imports Rcpp, methods, Rdpack, R6, parallel

RdMacros Rdpack

Suggests covr, testthat (>= 3.0.0), xml2

Config/testthat/edition 3

Encoding UTF-8

URL https://fleverest.github.io/elections.dtree/, https://github.com/fleverest/elections.dtree/

BugReports https://github.com/fleverest/elections.dtree/issues/

NeedsCompilation yes

Author Floyd Everest [aut, cre, cph] (<https://orcid.org/0000-0002-2726-6736>), Damjan Vukcevic [aut] (<https://orcid.org/0000-0001-7780-9586>)

Maintainer Floyd Everest <me@floydeverest.com>

Repository CRAN

Date/Publication 2023-01-20 07:30:02 UTC

R topics documented:

`dirichlet_tree` *Create a Dirichlet-tree for modelling ranked ballots*

Description

A `dirichlet_tree` object represents a Dirichlet-tree distribution on ballots. By specifying the tree structure for the ranked ballots, the Dirichlet-tree is initialized with the same prior structure described by Everest et al. (2022). There are methods provided for observing data (to obtain a posterior distribution) along with methods to sample election outcomes and sets of ballots from the posterior predictive distribution.

Format

An `R6Class` generator object.

Active bindings

`a0` Gets or sets the `a0` parameter for the Dirichlet-tree.

`min_depth` Gets or sets the `min_depth` parameter for the Dirichlet-tree.

`max_depth` Gets or sets the `max_depth` parameter for the Dirichlet-tree.

`vd` Gets or sets the `vd` parameter for the Dirichlet-tree.

Methods

Public methods:

- `dirichlet_tree$new()`
- `dirichlet_tree$print()`
- `dirichlet_tree$update()`
- `dirichlet_tree$reset()`
- `dirichlet_tree$sample_posterior()`

- dirichlet_tree$sample_predictive()

Method `new()`**:** Create a new `dirichlet_tree` prior distribution with the specified tree structure. See Everest et al. (2022) for further details.

Usage:

```
dirichlet_tree$new(
  candidates,
  min_depth = 0,
  max_depth = length(candidates) - 1,
  a0 = 1,
  vd = FALSE
)
```

Arguments:

`candidates` A character vector, with each element (must be unique) representing a single candidate.

`min_depth` The minimum number of candidates which must be specified for a valid ballot in the election.

`max_depth` The maximum number of candidates which can be specified for a valid ballot in the election.

`a0` The prior parameter for the distribution.

`vd` A flag which, when `TRUE`, employs a parameter structure which reduces to a regular Dirichlet distribution as described by Everest et al. (2022).

Returns: A new `dirichlet_tree` prior.

Examples:

```
dtree <- dirichlet_tree$new(candidates = LETTERS, a0 = 1., min_depth = 1)
```

Method `print()`**:** `print` shows some details of the distribution and its parameters.

Usage:

```
dirichlet_tree$print()
```

Returns: The `dirichlet_tree` object.

Method `update()`**:** Updates the `dirichlet_tree` object with observations of ballots. This updates the parameter structure of the tree to yield the posterior Dirichlet-tree, as described in Everest et al. (2022).

Usage:

```
dirichlet_tree$update(ballots)
```

Arguments:

`ballots` A set of ballots to observe - must be of class `ranked_ballots`.

Returns: The `dirichlet_tree` object.

Examples:

```
dirichlet_tree$new(
  candidates = LETTERS
)$update(
  ranked_ballots(c("A", "B", "C"))
)
```

Method `reset()`**:** Resets the `dirichlet_tree` observations to revert the parameter structure back to the originally specified prior.

Usage:

```
dirichlet_tree$reset()
```

Returns: The `dirichlet_tree` object.

Examples:

```
dirichlet_tree$new(
  candidates = LETTERS
)$update(
  ranked_ballots(c("A", "B", "C"))
)$reset()
```

Method `sample_posterior()`**:** Draws sets of ballots from independent realizations of the Dirichlet-tree posterior, then determines the probability for each candidate being elected by aggregating the results of the social choice function. See Everest et al. (2022) for details.

Usage:

```
dirichlet_tree$sample_posterior(
  n_elections,
  n_ballots,
  n_winners = 1,
  replace = FALSE,
  n_threads = NULL
)
```

Arguments:

`n_elections` An integer representing the number of elections to generate. A higher number yields higher precision in the output probabilities.

`n_ballots` An integer representing the total number of ballots cast in the election.

`n_winners` The number of candidates elected in each election.

`replace` A boolean indicating whether or not we should replace our sample in the monte-carlo step, drawing the full set of election ballots from the posterior

`n_threads` The maximum number of threads for the process. The default value of `NULL` will default to 2 threads. `Inf` will default to the maximum available, and any value greater than or equal to the maximum available will result in the maximum available.

Returns: A numeric vector containing the probabilities for each candidate being elected.

Examples:

```
dirichlet_tree$new(
  candidates = LETTERS,
  a0 = 1.,
  min_depth = 3,
  max_depth = 6,
  vd = FALSE
)$update(
  ranked_ballots(c("A","B","C"))
)$sample_posterior(
  n_elections = 10,
  n_ballots = 10
)
```

Method `sample_predictive()`**:** `sample_predictive` draws ballots from a multinomial distribution with ballot probabilities obtained from a single realization of the Dirichlet-tree posterior on the ranked ballots. See Everest et al. (2022) for details.

Usage:

```
dirichlet_tree$sample_predictive(n_ballots)
```

Arguments:

`n_ballots` An integer representing the total number of ballots cast in the election.

Returns: A `ranked_ballots` object containing `n_ballots` ballots drawn from a single realisation of the posterior Dirichlet-tree.

Examples:

```
dirichlet_tree$new(
  candidates = LETTERS,
  a0 = 1.,
  min_depth = 3,
  max_depth = 6,
  vd = FALSE
)$update(
  ranked_ballots(c("A","B","C"))
)$sample_predictive(
  n_ballots = 10
)
```

References

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Ballot-Polling Audits of Instant-Runoff Voting Elections with a Dirichlet-Tree Model." doi:10.48550/ARXIV.2209.03881..

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Auditing Ranked Voting Elections with Dirichlet-Tree Models: First Steps." doi:10.48550/ARXIV.2206.14605..

Examples

```
## ------------------------------------------------
## Method `dirichlet_tree$new`
## ------------------------------------------------

dtree <- dirichlet_tree$new(candidates = LETTERS, a0 = 1., min_depth = 1)


## ------------------------------------------------
## Method `dirichlet_tree$update`
## ------------------------------------------------

dirichlet_tree$new(
  candidates = LETTERS
)$update(
  ranked_ballots(c("A", "B", "C"))
)


## ------------------------------------------------
## Method `dirichlet_tree$reset`
## ------------------------------------------------

dirichlet_tree$new(
  candidates = LETTERS
)$update(
  ranked_ballots(c("A", "B", "C"))
)$reset()


## ------------------------------------------------
## Method `dirichlet_tree$sample_posterior`
## ------------------------------------------------

dirichlet_tree$new(
  candidates = LETTERS,
  a0 = 1.,
  min_depth = 3,
  max_depth = 6,
  vd = FALSE
)$update(
  ranked_ballots(c("A","B","C"))
)$sample_posterior(
  n_elections = 10,
  n_ballots = 10
)


## ------------------------------------------------
## Method `dirichlet_tree$sample_predictive`
## ------------------------------------------------

dirichlet_tree$new(
  candidates = LETTERS,
```

```
  a0 = 1.,
  min_depth = 3,
  max_depth = 6,
  vd = FALSE
)$update(
  ranked_ballots(c("A","B","C"))
)$sample_predictive(
  n_ballots = 10
)
```

---

dirtree *Create a Dirichlet-tree object*

Description

dirtree is used to create a Dirichlet-tree for modelling ballots, as described by Everest et al. (2022).

Usage

```
dirtree(
  candidates,
  min_depth = 0,
  max_depth = length(candidates),
  a0 = 1,
  vd = FALSE
)
```

Arguments

candidates	A character vector, with each element (must be unique) representing a single candidate.
min_depth	The minimum number of candidates which must be specified for a valid ballot.
max_depth	The maximum number of candidates which can be specified for a valid ballot.
a0	The prior parameter for the distribution.
vd	A flag which, when TRUE, employs a parameter structure which reduces to a regular Dirichlet distribution as described by Everest et al. (2022).

Value

A Dirichlet-tree representing ranked ballots, as an object of class dirichlet_tree.

References

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Ballot-Polling Audits of Instant-Runoff Voting Elections with a Dirichlet-Tree Model." doi:10.48550/ARXIV.2209.03881..

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Auditing Ranked Voting Elections with Dirichlet-Tree Models: First Steps." doi:10.48550/ARXIV.2206.14605..

`ranked_ballots` *Construct a set of ranked ballots.*

Description

`ranked_ballots` is used to easily construct a set of ranked ballots.

Usage

```
ranked_ballots(x, candidates = NULL, ...)
```

Arguments

`x`	A character vector representing a single ballot, or a list of character vectors representing multiple ballots.
`candidates`	A character vector of names corresponding to the candidates running in the election.
`\dots`	Additional parameters to pass to `ranked_ballots`.

Value

A `ranked_ballots` object representing the ballot(s).

Examples

```
ranked_ballots(LETTERS[1:5])
ranked_ballots(list(LETTERS[1:5], LETTERS[6:1]))
```

read_ballots *Read* ranked_ballots *from a file.*

Description

Reads a set of partial IRV ballots from a file. The file is expected to follow the ballot:count standard, with a header describing all participating candidates.

Usage

```
read_ballots(file)
```

Arguments

file	The name of the file to read from, or a character vector of file lines.

reset *Clear the internal state of a* dirichlet_tree *object.*

Description

Destroy the Tree's internal state and revert back to the prior.

Usage

```
reset(dtree)
```

Arguments

dtree	A dirichlet_tree object.

Value

The dirichlet_tree object.

`sample_posterior` *Draw election outcomes from the posterior distribution.*

Description

`sample_posterior` draws sets of ballots from independent realizations of the Dirichlet-tree posterior, then determines the probability for each candidate being elected by aggregating the results of the social choice function. See Everest et al. (2022) for details.

Usage

```
sample_posterior(
  dtree,
  n_elections,
  n_ballots,
  n_winners = 1,
  replace = FALSE,
  n_threads = NULL
)
```

Arguments

`dtree`	A `dirichlet_tree` object.
`n_elections`	An integer representing the number of elections to generate. A higher number yields higher precision in the output probabilities.
`n_ballots`	An integer representing the total number of ballots cast in the election.
`n_winners`	The number of candidates elected in each election.
`replace`	A boolean indicating whether or not we should re-use the observed ballots in the monte-carlo integration step to determine the posterior probabilities.
`n_threads`	The maximum number of threads for the process. The default value of `NULL` will default to 2 threads. `Inf` will default to the maximum available, and any value greater than or equal to the maximum available will result in the maximum available.

Value

A numeric vector containing the probabilities for each candidate being elected.

References

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Ballot-Polling Audits of Instant-Runoff Voting Elections with a Dirichlet-Tree Model." doi:10.48550/ARXIV.2209.03881..

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Auditing Ranked Voting Elections with Dirichlet-Tree Models: First Steps." doi:10.48550/ARXIV.2206.14605..

sample_predictive *Draw ballots from the posterior predictive distribution.*

Description

sample_predictive draws ballots from a multinomial distribution with probabilities obtained from a single realization of the Dirichlet-tree posterior on the ranked ballots. See Everest et al. (2022) for details.

Usage

```
sample_predictive(dtree, n_ballots)
```

Arguments

dtree	A dirichlet_tree object.
n_ballots	An integer representing the number of ballots to draw.

Value

A ranked_ballots object containing n_ballots ballots drawn from a single realisation of the posterior Dirichlet-tree.

References

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Ballot-Polling Audits of Instant-Runoff Voting Elections with a Dirichlet-Tree Model." doi:10.48550/ARXIV.2209.03881..

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Auditing Ranked Voting Elections with Dirichlet-Tree Models: First Steps." doi:10.48550/ARXIV.2206.14605..

social_choice *Compute the outcome of an election.*

Description

social_choice reads a set of ballots, and computes the outcome of the election. The outcome is described by a vector of winning candidates, along with the elimination order of the losing candidates.

Usage

```
social_choice(x, ...)
```

Arguments

x	The set of ballots for which to compute the outcome of the social choice function.
\dots	Additional parameters to pass to social_choice.

Examples

```
social_choice(ranked_ballots(list(LETTERS[1], LETTERS[1], LETTERS[2])))
```

---

social_choice.ranked_ballots

Evaluate a social choice function on a set of ranked_ballots.

Description

social_choice.ranked_ballots reads a set of ranked ballots and computes an election outcome. The outcome depends on the chosen social choice function, along with any its associated parameters.

Usage

```
## S3 method for class 'ranked_ballots'
social_choice(x, n_winners = 1, fn = "irv", ...)
```

Arguments

x	The set of ballots for which to compute the outcome of the IRV social choice function.
n_winners	The number of candidates to elect.
fn	The social choice function to use.
	"irv" The IRV social choice function.
\dots	Unused.

update *Update a* dirichlet_tree *model by observing some ranked ballots.*

Description

update updates a Dirichlet-tree model with observations to obtain a posterior distribution on the ranked ballots. See Everest et al. (2022) for implementation details.

Usage

```
## S3 method for class 'dirichlet_tree'
update(object, ballots, ...)
```

Arguments

object	A dirichlet_tree object.
ballots	A set of ballots - must be of type ranked_ballots.
\dots	Unused.

Value

The dirichlet_tree object.

References

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Ballot-Polling Audits of Instant-Runoff Voting Elections with a Dirichlet-Tree Model." doi:10.48550/ARXIV.2209.03881..

Everest F, Blom M, Stark PB, Stuckey PJ, Teague V, Vukcevic D (2022). "Auditing Ranked Voting Elections with Dirichlet-Tree Models: First Steps." doi:10.48550/ARXIV.2206.14605..

write_ballots *Write* ranked_ballots *to a file.*

Description

Writes a set of ballots to a new file. This follows the ballot:count standard, with a header describing the candidates.

Usage

```
write_ballots(ballots, filename = "", return_lines = FALSE, suppress = FALSE)
```

Arguments

ballots	The ranked_ballots to write to a file.
filename	The name of the file to write to, or "" to write output to stdout.
return_lines	A flag which determines whether or not the output should be returned as a character vector
suppress	A flag which, when True, suppresses any output to stdout.

Examples

```
write_ballots(ranked_ballots(c(LETTERS)), tempfile("test.txt"))
write_ballots(ranked_ballots(c(LETTERS)))
```

'[.ranked_ballots' *Access Subsets of Ballots.*

Description

Extract subsets of ballots by index.

Usage

```
## S3 method for class 'ranked_ballots'
x[i = NULL]
```

Arguments

x	Some ranked_ballots.
i	The index, or vector of indices corresponding to each ballot to be extracted.

Index