Package: STV (via r-universe)

September 8, 2024

Type Package

Title Single Transferable Vote Counting

Version 1.0.2

Maintainer John Emerson <john.emerson@yale.edu>

URL https://github.com/jayemerson/STV

BugReports https://github.com/jayemerson/STV/issues

Description Implementations of the Single Transferable Vote counting system. By default, it uses the Cambridge method for surplus allocation and Droop method for quota calculation. Fractional surplus allocation and the Hare quota are available as options.

License LGPL-3

Encoding UTF-8

LazyData true

Suggests testthat, crayon

RoxygenNote 7.1.0

Repository https://jayemerson.r-universe.dev

RemoteUrl https://github.com/jayemerson/stv

RemoteRef HEAD

RemoteSha 32940b4e746ded9e69f5f9d35233555337f76e63

Contents

`STV-package` *STV Vote Counting*

Description

Functions in this package are designed to clean, validate, and count single transferable votes.

Author(s)

Jay Emerson (expand this)

`ballots` *ballots data contains rankings of 44 candidates by 489 people. 10 rows are black, which will prompt an error. This data follows the structure of a 2016 Yale faculty senate election, but vote choices have been permuted and candidate names have been removed.*

Description

ballots data contains rankings of 44 candidates by 489 people. 10 rows are black, which will prompt an error. This data follows the structure of a 2016 Yale faculty senate election, but vote choices have been permuted and candidate names have been removed.

Author(s)

John W. Emerson <`john.emerson@yale.edu`>

`cleanBallots` *Tries to Address* `validateBallots()` *Warning(s).*

Description

The `cleanBallots()` function tries to clean data for use in `stv()`. Some warnings from `validateBallots()` can only be addressed by the user.

Usage

```
cleanBallots(x, cand.names = NULL)
```

Arguments

`x`	a `data.frame` with rows as ballots and columns as candidates.
`cand.names`	character vector of length equal to number of candidates (needed when column names of `x` are missing or not unique).

Details

The `cleanBallots()` function assumes `x` contains rows and columns corresponding to ballots and candidates respectively. Formatting corrections are attempted in the following order:

1. If `x` is a `matrix` it is convered to a `data.frame`. Otherwise, user must provide a `data.frame`.
2. If `x` has non-numeric entries, user must convert data to numeric type.
3. If `cand.names` is specified, it is assigned as column names of `x`. All collumn names must be uniquely specified. `x` already has valid column names, no need to specify `cand.names`. If column names of `x` missing and `cand.names` not specified, returns error message.
4. Removes blank columns.
5. Removes blank and/or non-sequentially ranked rows.

Value

a `data.frame` compatible for `stv()` function.

Examples

```
data(ballots)
cballots <- cleanBallots(ballots)
validateBallots(cballots)
```

`stv` *Implement STV Counting Systems*

Description

Analyze a data frame of STV election ballot rankings and return the elected voters and details of the steps in the election counting.

Usage

```
stv(
  x,
  seats = 1,
  file = "",
  surplusMethod = "Cambridge",
  quotaMethod = "Droop",
  quotaFloor = TRUE
)
```

Arguments

`x`	a data frame with rows of ballots and columns for each candidate. `x` must pass all checks from `validateBallots`.
`seats`	an integer (default = 1) indicating candidates to elect.
`file`	a character string naming a file; `""` (default) indicates output to the console only (default), otherwise a CSV file is created (and thus `file` should end with `".csv"`).
`surplusMethod`	a character string indicating which method to use for surplus allocation. Currently supports `"Cambridge"` (default) and `"Fractional"`.
`quotaMethod`	a character string indicating which method to use for calculation of quota. Currently supports `"Droop"` (default) and `"Hare"`.
`quotaFloor`	a logical value indicating that candidates must reach the integer part of the quota in order to be elected if `"TRUE"` (default) or that they must reach the exact quota which may be fractional if `"FALSE"`.

Details

Revise the details: `stv()` first validates `x` by running the `validateBallots()` function. Once validation is complete, it implements the selected single transferable vote counting method. Each round of counting starts with idetification of active ballots. Then a quota is calculated (currently only supports Droop method `floor(votes/(seats + 1)) + 1` and Hare method `floor(votes/seats))` with or without the `floor()` rounding. A tally of each candidate's vote share is obtained using top choices of active ballots, where a ballot stays active until it runs out of marked choices or gets removed during surplus reallocation. If a candidate reaches the quota, she/he is elected and associated surplus ballots are reallocated (currently only supports Cambridge and Fractional methods). If multiple candidates reach the quota, all of them are elected and their surpluses are all reallocated. If no candidate reaches the quota, then the candidate with the minimum number of votes is eliminated. If multiple candidates tie for minimum number of votes, one of them is selected at random and eliminated. The process is repeated until all of the seats are filled or the number of candidates still in race equals the number of unfilled seats. In the later case, all of the active candidates are elected.

Value

The object returned is a list consisting of two components: a vector of the elected candidates, and a data frame with rows containing detailed results from each round of STV counting.

For every round of vote counting, a row of the detailed information contains: number of active ballots, seats to remaining to be filled, the current quota, the maximum and minimum votes obtained by each candidate, who was eliminated (if applicable), whether there was a tie for elimination (indicated by how many tied), who was elected (if applicable), the surplus if elected (or multiple surpluses if multiple candidates were elected), and each candidate's votes tally for that round.

Examples

```
data(ballots)
cballots <- cleanBallots(ballots)
```

```
set.seed(1)
result1 <- stv(cballots, seats = 4)
names(result1)
result1$elected

set.seed(4)
result2 <- stv(cballots, seats = 4)
result2$elected

## Not run:
result3 <- stv(cballots, seats = 4, surplusMethod = "Fractional")
result3$elected

result4 <- stv(cballots, seats = 4, surplusMethod = "Fractional",
               quotaMethod = "Hare")
result4$elected

result5 <- stv(cballots, seats = 4, surplusMethod = "Cambridge",
               quotaMethod = "Hare")
result5$elected

## End(Not run)
```

`validateBallots` *Validates Input Data Format for* `stv()`.

Description

The function `validateBallots` validates data format for `stv()` input. It assumes that each row corresponds to a ballot and each column corresponds to a candidate. If input data is in the correct format, `validateBallots` returns string: "All tests passed. Please feel free to run `stv()` function." Otherwise, it prints a message corresponding to the formatting error.

Usage

```
validateBallots(x)
```

Arguments

`x`	a `data.frame` with rows as ballots and columns as candidates.

Details

The `validateBallots` function should be run before `stv()` is called. Before any calculation, `stv()` calls this function and proceeds only if `x` passes all of the following tests.

1. If `x` is a `data.frame`.
2. If `x` has unique column names.

3. If x has numeric entries.
4. If x doesn't have any blank columns.
5. If x doesn't have any blank rows.
6. If x doesn't have any rows with non-sequential ranks.

Value

string: "All tests passed. Please feel free to run stv() function." if x passes all tests. Otherwise, a message corresponding to the problem is retunred. NOTE: Some of the warnings can be fixed using `cleanBallots()` function. Other issues must be fixed by the user.

Examples

```
data(ballots)
result <- try(validateBallots(ballots), silent=TRUE)
print(result)
```

Index