Package ‘coalitions’

May 2, 2022

Type Package

Title Bayesian ```Now-Cast`` Estimation of Event Probabilities in Multi-Party Democracies

Version 0.6.23

Date 2022-05-01

Maintainer Andreas Bender <bender.at.R@gmail.com>

Description An implementation of a Bayesian framework for the opinion poll based estimation of event probabilities in multi-party electoral systems (Bender and Bauer (2018) <doi:10.21105/joss.00606>).

Depends R (>= 3.2.1)

Imports checkmate, gtools, rvest, xml2, jsonlite, RCurl, rlang, magrittr, lubridate, stringr, tidyr (>= 1.0.0), purrr (> 0.2.2), dplyr (> 0.5.0), ggplot2, tibble (>= 3.0.0)

Suggests testthat, covr,

Encoding UTF-8

License MIT + file LICENSE

URL https://adibender.github.io/coalitions/

BugReports https://github.com/adibender/coalitions/issues

RoxygenNote 7.1.1

LazyData true

NeedsCompilation no

Author Andreas Bender [aut, cre] (<https://orcid.org/0000-0001-5628-8611>), Alexander Bauer [aut] (<https://orcid.org/0000-0003-3495-5131>), Rebekka Schade [ctb]

Repository CRAN

Date/Publication 2022-05-02 10:10:02 UTC
**R topics documented:**

- calculate_prob
- calculate_probs
- collapse_parties
- dHondt
- draw_from_posterior
- get_probabilities
- get_seats
- get_surveys
- gg_survey
- hare_niemeyer
- have_majority
- party_colors_de
- party_labels_de
- pool_surveys
- redistribute
- scrape_austria
- scrape_wahlrecht
- sls
- surveys_sample
- try_readHTML

---

**calculate_prob**

*Calculate coalition probability from majority table*

**Description**

Given a table with simulations in the rows and coalitions in the columns, this function returns the coalition probabilities for a specified coalition, by default excluding superior coalitions first.

**Usage**

```r
calculate_prob(majority_df, coalition, exclude_superior = TRUE, ...)
```

**Arguments**

- `majority_df` A data frame containing logical values indicating if the coalitions (columns) have a majority (rows).
- `coalition` The coalition of interest for which superior coalitions will be obtained by `get_superior`.
- `exclude_superior` Logical. If TRUE, superior coalitions will be excluded, otherwise total coalition probabilities will be returned. Usually it makes sense to exclude superior coalitions.
- `...` Further arguments passed to `get_superior`
calculate_probs

Examples

test_df <- data.frame(
cdu = c(rep(FALSE, 9), TRUE),
cdu_fdp = c(rep(FALSE, 8), TRUE, TRUE),
cdu_fdp_greens = c(TRUE, TRUE, rep(FALSE, 6), TRUE, TRUE))
calculate_prob(test_df, "cdu_fdp_greens") # exclude_superior defaults to TRUE
calculate_prob(test_df, "cdu_fdp_greens", exclude_superior=FALSE)

calculate_probs

Calculate coalition probabilities for multiple coalitions

Description

Given a table with simulations in the rows and coalitions in the columns, this function returns the coalition probabilities for a specified coalition, by default excluding superior coalitions first

Usage

calculate_probs(majority_df, coalitions, exclude_superior = TRUE, ...)

Arguments

majority_df A data frame containing logical values indicating if the coalitions (columns) have a majority (rows).
coalitions A list of coalitions for which coalition probabilities should be calculated. Each list entry must be a vector of party names. Those names need to correspond to the names in majority_df.
exclude_superior Logical. If TRUE, superior coalitions will be excluded, otherwise total coalition probabilities will be returned. Usually it makes sense to exclude superior coalitions.
...

Further arguments passed to get_superior

See Also

calculate_prob

Examples

test_df <- data.frame(
cdu = c(rep(FALSE, 9), TRUE),
cdu_fdp = c(rep(FALSE, 8), TRUE, TRUE),
cdu_fdp_greens = c(TRUE, TRUE, rep(FALSE, 6), TRUE, TRUE))
calculate_probs(test_df, list("cdu", "cdu_fdp", "cdu_fdp_greens"))
calculate_probs(test_df, list("cdu", "cdu_fdp", "cdu_fdp_greens"), exclude_superior=FALSE)
collapse_parties  
*Transform surveys in long format*

**Description**

Given a data frame containing multiple surveys (one row per survey), transforms the data into long format with one row per party.

**Usage**

```r
collapse_parties(
  surveys,
  parties = c("cdu", "spd", "greens", "fdp", "left", "pirates", "fw", "afd", "others")
)
```

**Arguments**

- **surveys**: A data frame with one survey per row.
- **parties**: A character vector containing names of parties to collapse.

**Value**

Data frame in long format

**Examples**

```r
## Not run:
emnid <- scrape_wahlrecht()
emnid.long <- collapse_parties(emnid)
## End(Not run)
```

dHondt  
*Seat Distribution by D’Hondt*

**Description**

Calculates number of seats for the respective parties according to the method of d’Hondt.

**Usage**

```r
dHondt(votes, parties, n_seats = 183)
```
Arguments

votes Number of votes per party.
parties Names of parties (must be same length as votes).
n_seats Number of seats in parliament. Defaults to 183 (seats in Austrian parliament).

Value

A numeric vector containing the seats of all parties after redistribution via D’Hondt

See Also

sls

Examples

library(coalitions)
library(dplyr)
# get the latest survey for a sample of German federal election polls
surveys <- get_latest(surveys_sample) %>% tidy::unnest("survey")
# calculate the seat distribution based on D'Hondt for a parliament with 300 seats
dHondt(surveys$votes, surveys$party, n_seats = 300)

Description

Draw random numbers from posterior distribution

Usage

draw_from_posterior(
  survey,
  nsim = 10000,
  seed = as.numeric(now()),
  prior = NULL,
  correction = NULL
)

Arguments

survey survey object as returned by as_survey or getSurveys
nsim number of simulations
seed sets seed
prior optional prior information. Defaults to 1/2 (Jeffrey’s prior).
correction  A positive number. If not NULL, each sample from the Dirichlet distribution will be additionally "corrected" by a random number from U(-1*correction, 1*correction). This can be used to introduce extra variation which might be useful due to rounding errors from reported survey results (or add an additional source of variation in general).

Value  

data.frame containing random draws from Dirichlet distribution which can be interpreted as election results.

See Also  

as_survey

get_probabilities Wrapper for calculation of coalition probabilities from survey

Description  

Given a table with simulations in the rows and coalitions in the columns, this function returns the coalition probabilities for a specified coalition, by default excluding superior coalitions first

Usage

get_probabilities(
  x,
  coalitions = list(c("cdu"), c("cdu", "fdp"), c("cdu", "fdp", "greens"), c("spd"),
                   c("spd", "left"), c("spd", "left", "greens"),
    nsim = 1e+05,
    distrib.fun = sls,
    seats_majority = 300L,
    seed = as.numeric(now()),
    correction = NULL
)

Arguments

x  A table containing one row per survey and survey information in long format in a separate column named survey.

coalitions  A list of coalitions for which coalition probabilities should be calculated. Each list entry must be a vector of party names. Those names need to correspond to the names in majority_df.

nsim  number of simulations

distrib.fun  Function to calculate seat distribution. Defaults to sls (Sainte-Lague/Schepers).

seats_majority  The number of seats needed to obtain majority.
get_seats

Description

Calculate seat distribution from draws from posterior

Usage

get_seats(
  dirichlet.draws,
  survey,
  distrib.fun = sls,
  samplesize = NULL,
  hurdle = 0.05,
  others = "others",
  ...
)

Arguments

dirichlet.draws
  Matrix containing random draws from posterior.
survey
  The actual survey results on which dirichlet.draws were based on.
get_surveys

get_surveys

Scrape surveys from all pollsters

Description

Given a specific date, extract the survey from this date or the last one before this date.

Usage

get_surveys(country = c("DE", "AT"))

get_surveys_by()

get_surveys_rp()

get_surveys_nds()

get_surveys_saxony()

get_surveys_brbr()
get_surveys

get_surveys_thuringen()

get_latest(surveys = NULL, max_date = Sys.Date())

Arguments

country Choose country from which surveys should be scraped. Currently "DE" (Germany) and "AT" (Austria) are supported.
surveys If provided, latest survey will be obtained from this object, otherwise calls get_surveys.
max_date Specifies the date, relative to which latest survey will be searched for. Defaults to Sys.Date.

Value

Nested tibble. When fully unnested, the dataset contains the following columns:

pollster Character name of the polling institute.
date Publication date of the poll.
start, end Start and end date of the field period, i.e. the dates during which the poll was conducted.
respondents Number of respondents in the poll.
party Character name of an individual party.
percent Percentage of respondents that chose the party. Given in percentage points, i.e. 38% is given as 38.
votes Number of respondents that chose the party.

Examples

## Not run:
library(coalitions)
# scrape data for the German federal election
# get_surveys()

## End(Not run)
library(coalitions)
### Scrape the newest poll for the German federal election
# Possibility 1: Calling get_latest without arguments scrapes surveys from the web
# Possibility 2: Use get_latest() on an already scraped dataset
surveys <- get_latest(surveys_sample)
**gg_survey**  
*Plot voter shares observed in one survey*

**Description**
Bar chart of the raw voter shares observed in one survey. Additionally to plotting positive voter shares, the function can be used to plot party-specific differences (e.g. between a survey and the election result), including negative numbers.

**Usage**
gg_survey(data, colors = NULL, labels = NULL, annotate_bars = TRUE, hurdle = 5)

**Arguments**
- **data** Scraped dataset containing one row per party in the column party and the observed voter share in the column percent
- **colors** Named vector containing party colors. If NULL (default) tries to guess color based on party names, gray otherwise.
- **labels** Named vector containing party labels. If NULL (default) tries to guess party names from data.
- **annotate_bars** If TRUE (default) bars are annotated by the respective vote share (percentage).
- **hurdle** Hurdle for single parties to get into the parliament, e.g. '5' for '5%'. If set to NULL no horizontal line is plotted. The horizontal line can be suppressed using NULL.

**Examples**
```
library(tidyr)
library(dplyr)
library(coalitions)

survey <- surveys_sample$surveys[[1]]$survey[[1]]

gg_survey(survey)
```

**hare_niemeyer**  
*Seat Distribution by Hare/Niemeyer*

**Description**
Calculates number of seats for the respective parties that have received more than hurdle percent of votes (according to the method of Hare/Niemeyer)
have_majority

Usage

have_majority(votes, parties, n_seats = 183)

Arguments

votes Number of votes per party.
parties Names of parties (must be same length as votes).
n_seats Number of seats in parliament. Defaults to 183 (seats in Austrian parliament).

Value

A data.frame containing parties above the hurdle and the respective seats/percentages after redistribution via Hare/Niemeyer

See Also

sls

Examples

library(coalitions)
library(dplyr)
# get the latest survey for a sample of German federal election polls
surveys <- get_latest(surveys_sample) %>% tidyr::unnest("survey")
# calculate the seat distribution based on Hare/Niemeyer for a parliament with 300 seats
hare_niemeyer(surveys$votes, surveys$party, n_seats = 300)

have_majority Do coalitions have a majority

Description

Do coalitions have a majority

Usage

have_majority(
  seats_tab,
  coalitions = list(c("cdu"), c("cdu", "fdp"), c("cdu", "fdp", "greens"), c("spd"),
  c("spd", "left"), c("spd", "left", "greens")),
  seats_majority = 300L,
  collapse = "."
)


party_colors_de

Arguments

seats_tab A data frame containing number of seats obtained by a party. Must have columns party and seats.

coalitions A list of coalitions for which coalition probabilities should be calculated. Each list entry must be a vector of party names. Those names need to correspond to the names in majority_df.

seats_majority The number of seats needed to obtain majority.

collapse Character string passed to `base::paste`.

Examples

library(coalitions)
library(dplyr)
library(purrr)

# get the latest survey for a sample of German federal election polls
surveys <- get_latest(surveys_sample)

# check for majorities of two coalitions
coals <- list(c("cdu", "fdp"),
               c("spd", "left", "greens"))

# only use 100 simulations for a fast runtime
surveys <- surveys %>%
  mutate(draws = map(survey, draw_from_posterior, nsim = 100),
         seats = map2(draws, survey, get_seats),
         majorities = map(seats, have_majority, coalitions = coals))

surveys$majorities

party_colors_de

Colors for German parties

Description

A vector of colors associated with German parties.

Usage

party_colors_de

Format

A named character vector. Names indicate parties. Values contain color strings for the respective parties.
party_labels_de  

Labels for German parties

Description
A vector of labels associated with German parties.

Usage

party_labels_de

Format
A named character vector. Names indicate parties. Values contain party names suitable for plot labels.

pool_surveys  

Obtain pooled survey during specified period

Description
Per default, pools surveys starting from current date and going 14 days back. For each pollster within the defined time-frame, only the most recent survey is used.

Usage

pool_surveys(
  surveys,
  last_date = Sys.Date(),
  pollsters = c("allensbach", "emnid", "forsa", "fgw", "gms", "infratest", "dimap",
               "infratestdimap", "insa"),
  period = 14,
  period_extended = NA,
  corr = 0.5,
  weights = NULL
)

Arguments

surveys A tibble containing survey results for multiple pollsters as returned by get_surveys.
last_date Only surveys in the time-window from last_date to last_date - period will be considered for each pollster. Defaults to current date.
pollsters Character vector of pollsters that should be considered for pooling.
period See last_date argument.
period_extended
   Optional. If specified, all surveys in the time-window from last_date - period_extended to last_date - period will also be considered for each pollster, but only after down-weighting them by halving their true sample size.

corr
   Assumed correlation between surveys (of different pollsters). Defaults to 0.5.

weights
   Additional weights for individual surveys.

Examples

library(coalitions)
library(dplyr)
latest <- get_latest(surveys_sample)
pool_surveys(surveys_sample, last_date=as.Date("2017-09-02"))

redistribute CALCULATE PERCENTAGE OF VOTES/SEATS AFTER EXCLUDING PARTIES WITH VOTES < HURDLE

Description

Calculate percentage of votes/seats after excluding parties with votes < hurdle

Usage

redistribute(survey, hurdle = 0.05, others = "others", epsilon = 1e-05)

Arguments

survey
   The actual survey results on which dirichlet.draws were based on.

hurdle
   The percentage threshold which has to be reached by a party to enter the parliament.

others
   A string indicating the name under which parties not listed explicitly are subsumed.

epsilon
   Percentages should add up to 1. If they do not, within accuracy of epsilon, an error is thrown.

See Also

get_seats, sls
**Examples**

```r
library(coalitions)
library(dplyr)
# get the latest survey for a sample of German federal election polls
surveys <- get_latest(surveys_sample)
# redistribute the shares of 'others' parties and parties with a share of under 5%
surveys <- surveys %>% mutate(survey_redist = purrr::map(survey, redistribute))
surveys$survey # results before redistribution
surveys$survey_redist # results after redistribution
```

---

**scrape_austria**

*Import Austrian survey results*

**Description**

Reads JSON file from neuwal.com and performs some preprocessing to bring data into standardized format. Returns a nested tibble.

**Usage**

```r
scrape_austria(
  address = "https://neuwal.com/wahlumfragen/data/neuwal-wahlumfragen-user.json"
)
```

**Arguments**

- **address** URL of the JSON file.

---

**scrape_wahlrecht**

*Scrape surveys for German general election*

**Description**

Scrapes survey tables and performs sanitation to output tidy data

**Usage**

```r
scrape_wahlrecht(
  address = "https://www.wahlrecht.de/umfragen/emnid.htm",
  parties = c("CDU", "SPD", "GRUENE", "FDP", "LINKE", "PIRATEN", "FW", "AFD",
              "SONSTIGE")
)
```

```r
scrape_by(
  address = "https://www.wahlrecht.de/umfragen/landtage/bayern.htm",
  parties = c("CSU", "SPD", "GRUENE", "FDP", "LINKE", "PIRATEN", "FW", "AFD",
```

...
scrape_rp(
    address = "https://www.wahlrecht.de/umfragen/landtage/rheinland-pfalz.htm",
    parties = c("CDU", "SPD", "GRUENE", "FDP", "LINKE", "AFD", "FW", "SONSTIGE"),
    ind_row_remove = -c(1:3)
)

scrape_ltw(
    address = "https://www.wahlrecht.de/umfragen/landtage/niedersachsen.htm",
    parties = c("CDU", "SPD", "GRUENE", "FDP", "LINKE", "PIRATEN", "FW", "AFD",
               "SONSTIGE"),
    ind_row_remove = -c(1:2)
)

Arguments

address http-address from which tables should be scraped.

parties A character vector containing names of parties to collapse.

ind_row_remove Negative vector of rows that will be skipped at the beginning.

Examples

## Not run:
library(coalitions)
library(dplyr)
# select a polling agency from .pollster_df that should be scraped ...
coalitions:::.pollster_df
# ... here we choose Forsa
address <- coalitions:::.pollster_df %>% filter(pollster == "forsa") %>% pull(address)
scape_wahlrecht(address = address) %>% slice(1:5)

## End(Not run)
## Not run:
# Niedersachsen
scrape_ltw() %>% slice(1:5)

# Hessen
scrape_ltw("https://www.wahlrecht.de/umfragen/landtage/hessen.htm", ind_row_remove=-c(1)) %>%
  slice(1:5)

## End(Not run)
surveys_sample

Description

Calculates number of seats for the respective parties that have received more than 5% of votes (according to the method of Sainte-Lague/Schepers, see https://www.wahlrecht.de/verfahren/rangmasszahlen.html).

Usage

sls(votes, parties, n_seats = 598L)

Arguments

  votes       A numeric vector giving the redistributes votes
  parties     A character vector indicating the names of parties with respective votes.
  n_seats     The total number of seats that can be assigned to the different parties.

Value

  A numeric vector giving the number of seats each party obtained.

See Also

dHondt

Examples

library(coalitions)
library(dplyr)
# get the latest survey for a sample of German federal election polls
surveys <- get_latest(surveys_sample) %>% tidyr::unnest("survey")
# calculate the seat distribution based on Sainte-Lague/Schepers for a parliament with 300 seats
sls(surveys$votes, surveys$party, n_seats = 300)

surveys_sample

Sample of selected surveys

Description

A data set with surveys from seven different pollsters, three surveys per pollster. Surveys report support for different parties in the running for the German Bundestag prior to the 2017 election.

Usage

surveys_sample

Format

A nested data frame with 7 rows and 2 columns:

  institute  name of the pollster
  surveys    a list of data frames, each containing one survey
try_readHTML

Source

https://www.wahlrecht.de/

---

try_readHTML

Try call of read_html that throws an error if the url cannot be resolved

---

Description

Try call of read_html that throws an error if the url cannot be resolved

Usage

try_readHTML(url)

Arguments

url http-address that should be scraped.
## Index

<table>
<thead>
<tr>
<th><em>datasets</em></th>
<th>scrape_by (scrape_wahlrecht), 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>party_colors_de, 12</td>
<td>scrape_ltw (scrape_wahlrecht), 15</td>
</tr>
<tr>
<td>party_labels_de, 13</td>
<td>scrape_rp (scrape_wahlrecht), 15</td>
</tr>
<tr>
<td>surveys_sample, 17</td>
<td>scrape_wahlrecht, 15</td>
</tr>
<tr>
<td><em>distribution</em></td>
<td>sls, 5, 6, 8, 11, 14, 16</td>
</tr>
<tr>
<td>get_seats, 7</td>
<td>surveys_sample, 17</td>
</tr>
<tr>
<td><em>seat</em></td>
<td>try_readHTML, 18</td>
</tr>
<tr>
<td>get_seats, 7</td>
<td></td>
</tr>
</tbody>
</table>

| as_survey, 6 |
| calculate_prob, 2, 3, 7 |
| calculate_probs, 3 |
| collapse_parties, 4 |

| dHondt, 4, 8, 17 |
| draw_from_posterior, 5, 8 |

| get_latest (get_surveys), 8 |
| get_probabilities, 6 |
| get_seats, 7, 14 |
| get_superior, 2, 3 |
| get_surveys, 8, 9, 13 |

| get_surveys_brbr (get_surveys), 8 |
| get_surveys_by (get_surveys), 8 |
| get_surveys_nds (get_surveys), 8 |
| get_surveys_rp (get_surveys), 8 |
| get_surveys_saxony (get_surveys), 8 |
| get_surveys_thuringen (get_surveys), 8 |
| gg_survey, 10 |

| hare_niemeyer, 10 |
| have_majority, 11 |

| party_colors_de, 12 |
| party_labels_de, 13 |
| pool_surveys, 13 |

| redistribute, 14 |
| scrape_austria, 15 |