Package 'MetaculR'

October 12, 2022

Title Analyze Metaculus Predictions and Questions

Version 0.4.1

URL https://ntrlshrp.gitlab.io/metaculr,
https://gitlab.com/ntrlshrp/metaculr

BugReports https://gitlab.com/ntrlshrp/metaculr/-/issues

Description Login, download, and analyze questions predicted by you and/or the Metaculus community by interacting with the Metaculus API, currently located at <https://www.metaculus.com/api2/>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

Imports magrittr, dplyr, ggplot2, httr, jsonlite, progress, tidyr, verification, stats, clipr, spatstat.geom, ggrepel, assertthat, cowplot

NeedsCompilation no

Author Joseph de la Torre Dwyer [aut, cre] (<https://orcid.org/0000-0002-2717-9077>)

Maintainer Joseph de la Torre Dwyer <JosephD@BRdata.com>

Repository CRAN

Date/Publication 2022-04-25 19:30:02 UTC

R topics documented:

`MetaculR_aggregated_forecasts`
Aggregate Community Forecasts for MetaculR

Description

Provides different results of aggregating current community forecasts to help you make your next forecast.

Usage

```
MetaculR_aggregated_forecasts(MetaculR_questions, Metaculus_id, baseline = 0.5)
```

Arguments

`MetaculR_questions`	A MetaculR_questions object
`Metaculus_id`	The ID of the question to plot
`baseline`	Climatological baseline for binary questions

Details

Sevilla (2021) found a Metaculus baseline of 0.36 looking at ~900 questions. While Sevilla has at times referred to the geometric mean of odds, this function uses the equivalent mean of logodds. Also note that mu + (d - 1)(mu + b) (Neyman & Roughgarden) is equivalent to b + d(mu + b), this function uses the former.

Value

A dataframe of forecast aggregations.

`id`	Question ID.
`community_q2`	Community median.
`community_ave`	Community mean.
`community_q2_unweighted`	Community median, unweighted by recency.
`community_ave_unweighted`	Community mean, unweighted by recency.

community_mean_logodds
: Community mean of logodds.

community_mean_logodds_extremized_baseline
: Community mean of logodds, extremized with reference to a baseline. If the baseline is 0.5, this is "classical extremizing."

References

Neyman, E., & Roughgarden, T. (2022). Are You Smarter Than a Random Expert? The Robust Aggregation of Substitutable Signals. ArXiv:2111.03153 [Cs]. https://arxiv.org/abs/2111.03153

Sevilla, J. (2021, December 29). Principled extremizing of aggregated forecasts. https://forum.effectivealtruism.org/posts/biL94PKfeHmgHY6qe/principled-extremizing-of-aggregated-forecasts

Examples

```
## Not run:
MetaculR_aggregate_forecasts(
  MetaculR_questions = questions_myPredictions,
  Metaculus_id = 10004)

## End(Not run)
```

MetaculR_brier *Calculate Brier statistics on MetaculR_questions object*

Description

Calculate Brier statistics on MetaculR_questions object

Usage

```
MetaculR_brier(MetaculR_questions, me = TRUE, thresholds = seq(0, 1, 0.1))
```

Arguments

MetaculR_questions
: A MetaculR_questions object

me
: Show my scores alongside Metaculus scores

thresholds
: Thresholds to bin questions

Value

A list of Brier statistics for you and Metaculus.

`brier_me`, `brier_Metaculus`, `brier_community`

`baseline.tf`	Logical indicator of whether climatology was provided.
`bs`	Brier score
`bs.baseline`	Brier Score for climatology
`ss`	Skill score
`bs.reliability`	Reliability portion of Brier score.
`bs.resolution`	Resolution component of Brier score.
`bs.uncert`	Uncertainty component of Brier score.
`y.i`	Forecast bins – described as the center value of the bins.
`obar.i`	Observation bins – described as the center value of the bins.
`prob.y`	Proportion of time using each forecast.
`obar`	Forecast based on climatology or average sample observations.
`thresholds`	The thresholds for the forecast bins.
`check`	Reliability - resolution + uncertainty should equal brier score.

`Other`

`ss_me_Metaculus`, `ss_me_community`, `ss_Metaculus_community`
: Skill score, me vs. Metaculus, etc.

`count_questions`
: Number of total questions included.

`brier_df`: Used for plotting Brier score statistics

`ID`	Predictor.
`name`	Name of value, see above.
`value`	Value.

`brier_bins_df`: Used for plotting histogram and calibration plots.

`ID`	Predictor.
`centers`	y.i, see above.
`freqs`	prob.y, see above.
`obars`	obar.i, see above.
`ideal`	Ideal calibration where centers equals obars.
`ci_low`	Low end of 95% confidence interval for obar.i.
`ci_high`	High end of 95% confidence interval for obar.i.

Examples

```
## Not run:
brier_me <-
  MetaculR_brier(
    questions_myPredictions_resolved)

## End(Not run)
```

`MetaculR_excitement` *Find exciting questions*

Description

Find exciting questions

Usage

```
MetaculR_excitement(MetaculR_questions, days = 30)
```

Arguments

`MetaculR_questions`	A MetaculR_questions object
`days`	The time period used for the excitement calculations starts this number of days ago, prior to today. E.g., if your clock says it is day 12 and your `days` argument is 10, the time period is day 2 until the present.

Value

A dataframe of questions with excitement measures.

`id`	Question ID.
`title`	Question title.
`Total_Change`	Cumulative delta in time period, by probability.
`Total_logodds_Change`	Cumulative delta in time period, by logodds.
`Total_Change_Even`	Cumulative delta toward even odds in time period, by probability.
`Total_logodds_Change_Even`	Cumulative delta toward even odds in time period, by logodds.

Examples

```
## Not run:
questions_myPredictions_byExcitement <-
  MetaculR_excitement(
    questions_myPredictions)

## End(Not run)
```

`MetaculR_login` *Login to Metaculus*

Description

Login to Metaculus

Usage

```
MetaculR_login(api_domain = "www")
```

Arguments

`api_domain`	Use "www" unless you have a custom Metaculus domain

Value

Your Metaculus_user_ID.

Examples

```
## Not run:
Metaculus_user_id <-
  MetaculR_login()

## End(Not run)
```

`MetaculR_markdown_table` *Easily translate R dataframes to Metaculus Markdown*

Description

Easily translate R dataframes to Metaculus Markdown

Usage

```
MetaculR_markdown_table(df)
```

Arguments

`df`	A dataframe.

Value

A Markdown table.

Examples

```
## Not run:
my_data <- data.frame(Year = c(2020,2021), Value = c(6, 7.2))

MetaculR_markdown_table(my_data)

## End(Not run)
```

`MetaculR_myDiff` *Find important changes within MetaculR_questions object*

Description

Find important changes within MetaculR_questions object

Usage

```
MetaculR_myDiff(MetaculR_questions)
```

Arguments

`MetaculR_questions`	A MetaculR_questions object

Value

A dataframe of questions with difference measures (your most recent prediction vs. community's most recent prediction, etc.).

`id`	Question ID.
`title`	Question title.
`my_prediction`	My most recent prediction.
`community_q2`	Community median.
`community_ave`	Community average.
`community_q2_pre_me`	Community median immediately prior to my_prediction.
`community_ave_pre_me`	Community average immediately prior to my_prediction.
`diff_me_q2`	Difference between me and the community median, by logodds.
`diff_me_ave`	Difference between me and the community average, by logodds.
`diff_comm_q2_pre_me`	Difference between community_q2_pre_me and the community average, by logodds.
`diff_comm_ave_pre_me`	Difference between community_ave_pre_me and the community average, by logodds.

`diff_me_q2_abs` Absolute difference between me and the community median, by logodds.

`diff_me_ave_abs`
: Absolute difference between me and the community average, by logodds.

`diff_comm_q2_pre_me_abs`
: Absolute difference between community_q2_pre_me and the community average, by logodds.

`diff_comm_ave_pre_me_abs`
: Absolute difference between community_ave_pre_me and the community average, by logodds.

`diff_me_q2_abs_odds`
: Absolute difference between me and the community median, by odds.

`diff_me_ave_abs_odds`
: Absolute difference between me and the community average, by odds.

`diff_comm_q2_pre_me_abs_odds`
: Absolute difference between community_q2_pre_me and the community average, by odds.

`diff_comm_ave_pre_me_abs_odds`
: Absolute difference between community_ave_pre_me and the community average, by odds.

Examples

```
## Not run:
questions_myPredictions_byDiff <-
  MetaculR_myDiff(
    questions_myPredictions)

## End(Not run)
```

`MetaculR_myPredictions`
: *Retrieve questions from Metaculus API (A wrapper for MetaculR_questions())*

Description

Retrieve questions from Metaculus API (A wrapper for MetaculR_questions())

Usage

```
MetaculR_myPredictions(
  api_domain = "www",
  order_by = "last_prediction_time",
  status = "all",
  search = "",
  guessed_by = "",
```

```
  offset = 0,
  pages = 10
)
```

Arguments

api_domain	Use "www" unless you have a custom Metaculus domain
order_by	Default is "last_prediction_time"
status	Choose "all", "upcoming", "open", "closed", "resolved"
search	Search term(s)
guessed_by	Generally your Metaculus_user_id
offset	Question offset
pages	Number of pages to request

Value

A list of questions that I've predicted, ordered by last prediction time.

See Also

Other Question Retrieval functions: MetaculR_myPredictions_Resolved(), MetaculR_questions()

Examples

```
## Not run:
questions_myPredictions <-
  MetaculR_myPredictions(
    guessed_by = Metaculus_user_id)

## End(Not run)
```

MetaculR_myPredictions_Resolved

Retrieve questions from Metaculus API (A wrapper for MetaculR_questions())

Description

Retrieve questions from Metaculus API (A wrapper for MetaculR_questions())

Usage

```
MetaculR_myPredictions_Resolved(
  api_domain = "www",
  order_by = "-resolve_time",
  status = "resolved",
  search = "",
  guessed_by = "",
  offset = 0,
  pages = 10
)
```

Arguments

api_domain	Use "www" unless you have a custom Metaculus domain
order_by	Default is "-resolve_time"
status	Default is "resolved"
search	Search term(s)
guessed_by	Generally your Metaculus_user_id
offset	Question offset
pages	Number of pages to request

Value

A list of questions that I've predicted, ordered by last prediction time, and resolved.

See Also

Other Question Retrieval functions: MetaculR_myPredictions(), MetaculR_questions()

Examples

```
## Not run:
questions_myPredictions_resolved <-
  MetaculR_myPredictions_Resolved(
    guessed_by = Metaculus_user_id)

## End(Not run)
```

MetaculR_plot *Plot the history of a single question*

Description

Plot the history of a single question

Usage

```
MetaculR_plot(
  MetaculR_questions,
  Metaculus_id,
  scale_binary = "prob",
  tournament = FALSE
)
```

Arguments

MetaculR_questions	A MetaculR_questions object
Metaculus_id	The ID of the question to plot
scale_binary	Choose "prob", "odds", or "logodds"
tournament	Plot relative log score below main plot

Value

A ggplot.

Examples

```
## Not run:
MetaculR_plot(
  MetaculR_questions = questions_myPredictions,
  Metaculus_id = 10004)

## End(Not run)
```

MetaculR_probabilistic_consensus

Generate probabilistic consensus from multiple parameterized forecasts

---

Description

Generate probabilistic consensus from multiple parameterized forecasts

Usage

```
MetaculR_probabilistic_consensus(f)
```

Arguments

f	A list of forecasts (see example for necessary structure).

Value

A list of forecasts.

pdf	A dataframe of probability density functions corresponding to original forecasts and consensus forecast.
cdf	A dataframe of cumulative distribution functions corresponding to original forecasts and consensus forecast.
summary	A dataframe of summary statistics corresponding to original forecasts and consensus forecast, i.e., 10th, 25th, 50th, 75th, 90th centiles and mean.

References

McAndrew, T., & Reich, N. G. (2020). An expert judgment model to predict early stages of the COVID-19 outbreak in the United States [Preprint]. Infectious Diseases (except HIV/AIDS). https://doi.org/10.1101/2020.09.21.20196725

Examples

```
## Not run:
forecasts <- list(list(range = c(0, 250), resolution = 1),
  list(source = "Pishkalo",
    dist = "Norm",
    params = c("mu", "sd"),
    values = c(116, 12),
    weight = 0.2),
  list(source = "Miao",
    dist = "Norm",
    params = c("mu", "sd"),
    values = c(121.5, 32.9)),
  list(source = "Labonville",
    dist = "TPD",
    params = c("min", "mode", "max"),
    values = c(89-14, 89, 89+29)),
  list(source = "NOAA",
    dist = "PCT",
    params = c(0.2, 0.8),
    values = c(95, 130)),
  list(source = "Han",
    dist = "Norm",
    params = c("mu", "sd"),
    values = c(228, 40.5)),
  list(source = "Dani",
    dist = "Norm",
    params = c("mu", "sd"),
    values = c(159, 22.3)),
  list(source = "Li",
    dist = "Norm",
    params = c("mu", "sd"),
    values = c(168, 6.3)),
  list(source = "Singh",
    dist = "Norm",
```

```
    params = c("mu", "sd"),
    values = c(89, 9)))

MetaculR_probabilistic_consensus(
  f = forecasts)

## End(Not run)
```

`MetaculR_questions` *Retrieve questions from Metaculus API*

Description

Retrieve questions from Metaculus API

Usage

```
MetaculR_questions(
  api_domain = "www",
  order_by = "last_prediction_time",
  status = "all",
  search = "",
  guessed_by = "",
  offset = 0,
  pages = 10
)
```

Arguments

`api_domain`	Use "www" unless you have a custom Metaculus domain
`order_by`	Choose "last_prediction_time", "-activity", "-votes", "-publish_time", "close_time", "resolve_time", "last_prediction_time"
`status`	Choose "all", "upcoming", "open", "closed", "resolved"
`search`	Search term(s)
`guessed_by`	Generally your Metaculus_user_id
`offset`	Question offset
`pages`	Number of pages to request

Value

A list of questions, ordered by last prediction time.

See Also

Other Question Retrieval functions: `MetaculR_myPredictions_Resolved()`, `MetaculR_myPredictions()`

Examples

```
## Not run:
questions_recent_open <-
  MetaculR_questions(
    order_by = "close_time",
    status = "open",
    guessed_by = "")

## End(Not run)
```

Index