Package 'evaluator'

July 6, 2021

Title Quantified Risk Assessment Toolkit

Version 0.4.3

Description An open source risk analysis toolkit based on the OpenFAIR ontology <https://publications.opengroup.org/c20b> and risk analysis standard <https://publications.opengroup.org/c20a>. Empowers an organization to perform a quantifiable, repeatable, and data-driven risk review.

Depends R (>= 3.3.0)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports cli, crayon, dplyr, extrafont, ggplot2, mc2d, magrittr, purrr, readr, readxl, rlang, rstudioapi, scales, stringi, tibble, tidyr, vctrs, viridis

RoxygenNote 7.1.1

Suggests DT, EnvStats, covr, ggalt (>= 0.4.0), knitr, flexdashboard (>= 0.4), forcats, furrr, markdown, mockery, pander (>= 0.6.1), psych, rmarkdown (>= 1.9), shiny, shinytest, spelling, statip, testthat

SystemRequirements pandoc

VignetteBuilder knitr

URL https://evaluator.tidyrisk.org

BugReports https://github.com/davidski/evaluator/issues

Language en-US

NeedsCompilation no

Author David Severski [aut, cre] (<https://orcid.org/0000-0001-7867-0459>)

Maintainer David Severski <davidski@deadheaven.com>

Repository CRAN

Date/Publication 2021-07-06 04:30:02 UTC

R topics documented:

as_tibble.tidyrisk_scenario . . . 3
calculate_max_losses . . . 4
compare_tef_vuln . . . 4
create_templates . . . 5
create_tidyrisk_scenario_skeleton . . . 6
derive_controls . . . 6
derive_control_key . . . 7
dollar_millions . . . 7
encode_scenarios . . . 8
explore_scenarios . . . 9
exposure_histogram . . . 10
generate_event_outcomes_plot . . . 10
generate_heatmap . . . 11
generate_report . . . 12
get_base_fontfamily . . . 13
get_mean_control_strength . . . 14
identify_outliers . . . 14
import_capabilities . . . 15
import_scenarios . . . 16
import_spreadsheet . . . 16
is_tidyrisk_scenario . . . 17
loss_exceedance_curve . . . 17
loss_scatterplot . . . 18
mc_capabilities . . . 19
mc_domains . . . 19
mc_domain_summary . . . 20
mc_mappings . . . 21
mc_qualitative_scenarios . . . 21
mc_quantitative_scenarios . . . 22
mc_scenario_summary . . . 23
mc_simulation_results . . . 24
new_tidyrisk_scenario . . . 24
openfair_example . . . 25
openfair_tef_tc_diff_lm . . . 25
openfair_tef_tc_diff_plm_sr . . . 26
print.tidyrisk_scenario . . . 27
read_qualitative_inputs . . . 27
read_quantitative_inputs . . . 28
risk_dashboard . . . 29
risk_factory . . . 30
run_simulation . . . 30
run_simulations . . . 31
sample_diff . . . 32
sample_lef . . . 32
sample_lm . . . 33
sample_tc . . . 34

sample_tef 34
sample_vuln 35
select_loss_opportunities 35
split_sheet 36
summarize_domains 37
summarize_iterations 38
summarize_scenario 38
summarize_to_disk 39
theme_evaluator 40
tidyrisk_factor 41
validate_scenarios 41
validate_tidyrisk_scenario 42
vec_cast.tidyrisk_factor 43
vec_ptype_abbr.tidyrisk_scenario 43

Index **44**

`as_tibble.tidyrisk_scenario` *Coerce the parameters of a tidyrisk_scenario to a tibble*

Description

Coerce the parameters of a tidyrisk_scenario to a tibble

Usage

```
## S3 method for class 'tidyrisk_scenario'
as_tibble(x, ...)

## S3 method for class 'tidyrisk_scenario'
as.data.frame(x, ...)
```

Arguments

`x`	A tidyrisk_scenario
`...`	Currently not used

calculate_max_losses *Calculate maximum losses*

Description

Calculate the biggest single annual loss for each scenario, as well as the minimum and maximum ALE across all iterations. Calculations both with and without outliers (if passed) are returned.

Usage

```
calculate_max_losses(simulation_results, scenario_outliers = NULL)
```

Arguments

simulation_results	Simulation results dataframe.
scenario_outliers	Optional vector of IDs of outlier scenarios.

Value

A dataframe with the following columns:

- `iteration` - index of the iteration
- `biggest_single_scenario_loss` - the biggest annual loss in that iteration,
- `min_loss` - the smallest annual loss in that iteration,
- `max_loss` - the total annual losses in that iteration
- `outliers` - logical of whether or not outliers are included

Examples

```
data(mc_simulation_results)
calculate_max_losses(mc_simulation_results)
```

compare_tef_vuln *Calculate number of loss events which occur in a simulated period*

Description

Composition function for use in `sample_lef`. Given a count of the number of threat events (TEF) and the level of vulnerability (as a percentage), calculate how many of those become loss events (LEF).

Usage

```
compare_tef_vuln(tef, vuln, n = NULL)
```

Arguments

tef	Threat event frequency (n).
vuln	Vulnerability (percentage).
n	Number of samples to generate.

Value

List containing samples (as a vector) and details (as a list).

See Also

Other OpenFAIR helpers: get_mean_control_strength(), openfair_tef_tc_diff_lm(), sample_diff(), sample_lef(), sample_lm(), sample_tc(), sample_vuln(), select_loss_opportunities()

Examples

```
compare_tef_vuln(tef = 500, vuln = .25)
```

---

create_templates *Create a directory structure for risk analysis, pre-populated with templates*

Description

Copy the sample files into an inputs subdirectory. This makes the starter files available for customizing and data collection. The inputs directory will be created if not already present. Preexisting files, if present, will not be overwritten. Also creates an empty results subdirectory as a default location for evaluator output.

Usage

```
create_templates(base_directory)
```

Arguments

base_directory	Parent directory under which to create starter files.

Value

A dataframe of the starter filenames, along with a flag on whether a file was copied.

Examples

```
## Not run:
create_templates("~/evaluator")

## End(Not run)
```

create_tidyrisk_scenario_skeleton

Create a skeleton tidyrisk scenario object in the current document

Description

Inserts a code block into the active document in RStudio for creating a tidyrisk_scenario object. This is an easy way of rapidly running a simulation.

Usage

```
create_tidyrisk_scenario_skeleton()
```

derive_controls

Derive control difficulty parameters for a given qualitative scenario

Description

Given a comma-separated list of control IDs in a scenario, identify the qualitative rankings associated with each scenario, convert to their quantitative parameters, and return a dataframe of the set of parameters.

Usage

```
derive_controls(capability_ids, capabilities, mappings)
```

Arguments

capability_ids	Comma-delimited list of capabilities in scope for a scenario.
capabilities	Dataframe of master list of all qualitative capabilities.
mappings	Qualitative mappings dataframe.

Value

A named list of quantitative estimate parameters for the capabilities applicable to a given scenario.

Examples

```
data(mc_capabilities)
capability_ids <- c("1, 3")
mappings <- data.frame(type = "diff", label = "1 - Immature", l = 0, ml = 2, h = 10,
                       conf = 3, stringsAsFactors = FALSE)
derive_controls(capability_ids, mc_capabilities, mappings)
```

derive_control_key *Derive control ID to control description mappings*

Description

Given a comma-separated list of control IDs, return a named list of descriptions for each control with the names set to the control IDs.

Usage

```
derive_control_key(capability_ids, capabilities)
```

Arguments

capability_ids	Comma-delimited list of capabilities in scope for a scenario.
capabilities	Dataframe of master list of all qualitative capabilities.

Value

A named list of control IDs and descriptions.

Examples

```
data(mc_capabilities)
capability_ids <- c("CAP-01", "CAP-03")
derive_control_key(capability_ids, mc_capabilities)
```

dollar_millions *Format dollar amounts in terms of millions of USD*

Description

Given a number, return a string formatted in terms of millions of dollars.

Usage

```
dollar_millions(x)
```

Arguments

x	A number.

Value

String in the format of $xM.

Examples

```
dollar_millions(1.523 * 10^6)
```

`encode_scenarios` *Encode qualitative data to quantitative parameters*

Description

Given an input of:

- qualitative risk scenarios
- qualitative capabilities
- translation table from qualitative labels to quantitative parameters

Usage

```
encode_scenarios(scenarios, capabilities, mappings)
```

Arguments

`scenarios`	Qualitative risk scenarios dataframe.
`capabilities`	Qualitative program capabilities dataframe.
`mappings`	Qualitative to quantitative mapping dataframe.

Details

Create a unified dataframe of quantitative scenarios ready for simulation.

Value

A dataframe of capabilities for the scenario and parameters for quantified simulation.

Examples

```
data(mc_qualitative_scenarios, mc_capabilities, mc_mappings)
encode_scenarios(mc_qualitative_scenarios, mc_capabilities, mc_mappings)
```

`explore_scenarios` *Launch the Scenario Explorer web application*

Description

Evaluator provides a simple Shiny-based web application for interactive exploration of simulation results. This allows a user to interactively review simulation output without generating an extensive report. For users comfortable with R, working directly with the result dataframes will usually be preferable, with the Explorer application provided as a bare-bones data exploration tool.

Usage

```
explore_scenarios(
  input_directory = "~/evaluator/inputs",
  results_directory = "~/evaluator/results",
  styles = NULL,
  intermediates_dir = tempdir(),
  quiet = TRUE,
  ...
)
```

Arguments

`input_directory`	Location of input files to be read by `read_quantitative_inputs`.
`results_directory`	Directory where the `simulations_results.rds` file is stored.
`styles`	Optional full path to CSS file to override default styles.
`intermediates_dir`	Location for intermediate knit files.
`quiet`	`TRUE` to suppress printing of pandoc output.
`...`	Any other parameters to pass to `rmarkdown::run`.

Value

Invisible NULL.

Examples

```
## Not run:
explore_scenarios("~/inputs", "~/results")

## End(Not run)
```

exposure_histogram *Display a histogram of losses for a scenario*

Description

Given a results dataframe for a specific scenario, create a histogram of the annualized loss exposure. This provides a detailed view on the results for a particular scenario.

Usage

```
exposure_histogram(simulation_result, bins = 30, show_var_95 = FALSE)
```

Arguments

simulation_result	Simulation result from run_simulation.
bins	Number of bins to use for the histogram.
show_var_95	Set to TRUE to show the 95 percentile value at risk line.

Value

A ggplot object.

See Also

Other result graphs: generate_event_outcomes_plot(), generate_heatmap(), generate_scatterplot-deprecated, loss_exceedance_curve(), loss_scatterplot()

Examples

```
data(mc_simulation_results)
result <- mc_simulation_results$results[[1]]
exposure_histogram(result)
```

generate_event_outcomes_plot *Display the distribution of threat events contained vs. realized across all domains*

Description

Creates a barbell plot showing the number and percentage of events contained (not resulting in loss) vs the number and percentage of loss events (threat events resulting in losses).

Usage

```
generate_event_outcomes_plot(domain_summary, domain_id = domain_id)
```

Arguments

domain_summary	Domain-level summary from domain_summary.
domain_id	Variable to group plot by.

Value

A ggplot object.

See Also

Other result graphs: exposure_histogram(), generate_heatmap(), generate_scatterplot-deprecated, loss_exceedance_curve(), loss_scatterplot()

Examples

```
data(mc_domain_summary)
generate_event_outcomes_plot(mc_domain_summary)
```

---

generate_heatmap *Display a heatmap of impact by domain*

Description

Given a domain_summary and a list of all domains, generate a heatmap colored by the 95% VaR. This plot displays the domains in which aggregate risk is greater than others.

Usage

```
generate_heatmap(domain_summary)
```

Arguments

domain_summary	Simulations summarized at a domain level via summarize_domains.

Value

A ggplot object.

See Also

Other result graphs: exposure_histogram(), generate_event_outcomes_plot(), generate_scatterplot-deprecated loss_exceedance_curve(), loss_scatterplot()

Examples

```
data(mc_domain_summary)
generate_heatmap(mc_domain_summary)
```

`generate_report` *Generate sample analysis report*

Description

Given a set of input files and summarized simulation results, create a skeleton risk analysis report. This report attempts to summarize the results of the analysis at a top level, using 95% Value at Risk (VaR) as the primary metric, while also providing more detailed analysis at both a per-domain and per-scenario level.

Usage

```
generate_report(
  input_directory = "~/evaluator/inputs",
  results_directory = "~/evaluator/results",
  output_file,
  styles = NULL,
  include_header = NULL,
  focus_scenario_ids = c("RS-51", "RS-12"),
  format = "html",
  intermediates_dir = tempdir(),
  quiet = TRUE,
  ...
)
```

Arguments

`input_directory`	Location of input files.
`results_directory`	Location of simulation results.
`output_file`	Full path to output file.
`styles`	Optional full path to CSS file to override default styles.
`include_header`	Optional full path to HTML to include in the HEAD section (HTML formats only).
`focus_scenario_ids`	IDs of scenarios of special interest.
`format`	Format to generate (html, pdf, word).
`intermediates_dir`	Location for intermediate knit files.
`quiet`	`TRUE` to suppress printing of pandoc output.
`...`	Any other parameters to pass straight to `rmarkdown::render`.

Details

This report includes several sections where an analyst will need to modify and fill in details for their specific organization. Of particular note is the Recommendations section, which will always need to be updated.

Value

Default return values of the `rmarkdown::render` function.

Examples

```
## Not run:
generate_report("~/inputs", "~/results", "~/risk_report.html")

## End(Not run)
```

`get_base_fontfamily` *Select a base graphics font family*

Description

The Benton Sans Regular font is preferred with a fallback of Arial Narrow. If neither font is available, use a default `sans` family font.

Usage

```
get_base_fontfamily()
```

Value

String of the preferred base font.

Examples

```
get_base_fontfamily()
```

get_mean_control_strength

Calculate difficulty strength across multiple controls by taking the mean

Description

Given a set of estimation parameters, calculate control strength as the arithmetic mean of sampled control effectiveness.

Usage

```
get_mean_control_strength(n, diff_parameters)
```

Arguments

n	Number of threat events to generate control effectiveness samples.
diff_parameters	Parameters to pass to sample_diff.

Value

Vector of control effectiveness.

See Also

Other OpenFAIR helpers: compare_tef_vuln(), openfair_tef_tc_diff_lm(), sample_diff(), sample_lef(), sample_lm(), sample_tc(), sample_vuln(), select_loss_opportunities()

identify_outliers

Unnest a summarized results dataframe, adding outlier information

Description

Given a summarized results dataframe, unnest the summary results column and use the value at risk (VaR) column to identify all the elements that are outliers (having a VaR >= two standard deviations)

Usage

```
identify_outliers(results)
```

Arguments

results	Scenario summary results

Value

The supplied dataframe with the following additional columns:

- `ale_var_zscore` - Annual loss z-score
- `outlier` - Logical flag when the z-score is greater than or equal to two

Examples

```
data(mc_scenario_summary)
identify_outliers(mc_scenario_summary)
```

`import_capabilities` *Import capabilities from survey spreadsheet*

Description

Import capabilities from survey spreadsheet

Usage

```
import_capabilities(survey_file = NULL, domains = NULL)
```

Arguments

`survey_file`	Path to survey Excel file. If not supplied, a default sample file is used.
`domains`	Dataframe of domains and domain IDs.

Value

Extracted capabilities as a dataframe.

Examples

```
data(mc_domains)
import_capabilities(domains = mc_domains)
```

import_scenarios *Import scenarios from survey spreadsheet*

Description

Import scenarios from survey spreadsheet

Usage

```
import_scenarios(survey_file = NULL, domains = NULL)
```

Arguments

survey_file	Path to survey Excel file. Defaults to a sample file if not supplied.
domains	Dataframe of domains and domain IDs.

Value

Extracted qualitative scenarios as a dataframe.

Examples

```
data(mc_domains)
import_scenarios(domains = mc_domains)
```

import_spreadsheet *Import the scenario spreadsheet*

Description

This is a convenience wrapper around the import_scenarios and import_capabilities functions. Writes cleaned comma-separated formatted files for the scenarios and capabilities to disk.

Usage

```
import_spreadsheet(
  survey_file = system.file("survey", "survey.xlsx", package = "evaluator"),
  domains = NULL,
  output_dir = "~/evaluator/results"
)
```

Arguments

survey_file	Path to survey Excel file. Defaults to an Evaluator-provided sample spreadsheet.
domains	Dataframe of domains and domain IDs. Defaults to built-in sample domains dataset.
output_dir	Output file directory.

Value

Dataframe of file information on the two newly created files.

---

is_tidyrisk_scenario *Test if the object is a tidyrisk_scenario*

Description

This function returns TRUE for tidyrisk_scenario (or subclasses) and FALSE for all other objects.

Usage

```
is_tidyrisk_scenario(x)
```

Arguments

x	An object

Value

TRUE if the object inherits from the tidyrisk_scenario class.

---

loss_exceedance_curve *Display the loss exceedance curve for a group of one or more scenarios*

Description

Display the loss exceedance curve for a group of one or more scenarios

Usage

```
loss_exceedance_curve(iteration_results)
```

Arguments

iteration_results	Iteration-level summary from summarize_iterations.

Value

A ggplot object.

See Also

Other result graphs: exposure_histogram(), generate_event_outcomes_plot(), generate_heatmap(), generate_scatterplot-deprecated, loss_scatterplot()

Examples

```
data(mc_simulation_results)
summarize_iterations(mc_simulation_results$results) %>% loss_exceedance_curve()
```

`loss_scatterplot` *Display a scatterplot of loss events for a scenario*

Description

Given a detailed results dataframe create a scatterplot of the number of loss events versus the total amount of expected annual loss for each simulation. This provides a detailed view on the results.

Usage

```
loss_scatterplot(simulation_result)
```

Arguments

`simulation_result`
: Simulation results from `run_simulation`.

Value

A ggplot object.

See Also

Other result graphs: `exposure_histogram()`, `generate_event_outcomes_plot()`, `generate_heatmap()`, `generate_scatterplot-deprecated`, `loss_exceedance_curve()`

Examples

```
data(mc_simulation_results)
loss_scatterplot(mc_simulation_results$results[[1]])
```

mc_capabilities *Capabilities*

Description

A sample set of capabilities for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_capabilities
```

Format

capability_id unique id of the capability

domain_id domain id to which the capability applies

capability full text summary of the capability

diff qualitative label of control effectiveness

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

mc_domains *Domain mappings*

Description

A sample set of the domains for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_domains
```

Format

domain_id abbreviated name of the domain

domain full title of the domain

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

`mc_domain_summary` *Domain-level risk summary*

Description

A sample set of quantified information security risk exposure, summarized at the domain level, for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_domain_summary
```

Format

domain_id abbreviated name of the domain

loss_events_mean mean number of loss events

loss_events_min minimum number of loss events

loss_events_max maximum number of loss events

loss_events_median median number of loss events

ale_max minimum annual loss expected

ale_median median annual loss expected

ale_mean mean annual loss expected

ale_max maximum annual loss expected

ale_sd standard deviation annual loss expected

ale_var value at risk, ale

mean_threat_events mean threat events

mean_avoided_events mean avoided events

mean_tc_exceedance mean threat capability exceedance

mean_diff_exceedance mean difficulty exceedance

mean_vuln mean vulnerability of the scenario

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

mc_mappings *Qualitative to quantitative mappings*

Description

A sample set of qualitative to quantitative mappings for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_mappings
```

Format

type The element in the OpenFAIR ontology to which this mapping applies

label Qualitative label

l BetaPERT low value

ml BetaPERT most likely value

h BetaPERT high value

conf BetaPERT confidence value

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

mc_qualitative_scenarios *Qualitative information security risk scenarios*

Description

A sample set of qualitative information security risk scenarios for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_qualitative_scenarios
```

Format

scenario_id id of the scenario, primary key

scenario full text description of the risk scenario

tcomm full text name of threat community

tef qualitative label of threat frequency

tc qualitative label of threat capability

lm qualitative label of loss magnitude

domain_id domain id

controls comma delimited list of controls ids

Details

No connection with any other similarly named entity is intended or implied.

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

`mc_quantitative_scenarios`

Quantified information risk scenarios

Description

A sample set of quantified information security risk scenarios for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_quantitative_scenarios
```

Format

A dataset of quantified risk scenarios, with parameters describing the distribution of each input.

scenario_id id of the scenario, primary key

scenario_description full text description of the risk scenario

tcomm description of the threat community

domain_id domain id

control_descriptons named list of the text description of controls involved

scenario `tidyrisk_scenario` objects

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

`mc_scenario_summary` *Scenario-level risk summary*

Description

A sample set of quantified information security risk exposure, summarized at the scenario level, for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_scenario_summary
```

Format

scenario_id ID of the scenario

domain_id domain id

control_description control description

results nested data frame of simulation results for the scenario

loss_events_mean mean number of loss events

loss_events_median median number of loss events

loss_events_min minimum number of loss events

loss_events_max maximum number of loss events

ale_median median annual loss expected

ale_max maximum annual loss expected

ale_var value at risk, ale

sle_min minimum single loss expectancy

sle_max maximum single loss expectancy

sle_mean mean single loss expectancy

sle_median median single loss expectancy

mean_tc_exceedance mean threat capability exceedance

mean_diff_exceedance mean difficulty exceedance

mean_vuln mean vulnerability of the scenario

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

mc_simulation_results *Simulation results*

Description

A sample set of information security risk scenario simulation results for the demonstration (and artificial) MetroCare information security program.

Usage

```
mc_simulation_results
```

Format

scenario_id id of the scenario

domain_id domain id

results nested data frame of simulation results for the scenario

Source

This is hypothetical information. Any similarity to any other entity is completely coincidental.

new_tidyrisk_scenario *Construct a quantitative scenario object*

Description

Supply one or more named lists in the format of `foo_params`, where each `foo` is an OpenFAIR factor name (e.g. tef, tc, diff, lm). Each factor should include a function name (`func`) to which the other named elements in the list are passed as parameters when sampling.

Usage

```
new_tidyrisk_scenario(..., model = "openfair_tef_tc_diff_lm")

tidyrisk_scenario(..., model = "openfair_tef_tc_diff_lm")
```

Arguments

`...`	One or more named OpenFAIR factor with parameters for sampling
`model`	Name of model to run

openfair_example *Launch OpenFAIR demonstration web application*

Description

A simple web application to demonstrate OpenFAIR modeling. This application allows a user to enter beta PERT parameters and run simulations to see the distribution of results, with high level summary statistics. As a demonstration application, only TEF, TC, DIFF, and LM parameters may be entered.

Usage

```
openfair_example(intermediates_dir = tempdir(), quiet = TRUE)
```

Arguments

intermediates_dir	Location for intermediate knit files.
quiet	TRUE to suppress printing of pandoc output.

Value

Invisible NULL

Examples

```
## Not run:
openfair_example()

## End(Not run)
```

openfair_tef_tc_diff_lm *Run an OpenFAIR simulation at the TEF/TC/DIFF/LM levels*

Description

Run an OpenFAIR model with parameters provided for TEF, TC, DIFF, and LM sampling. If there are multiple controls provided for the scenario, the arithmetic mean (average) is taken across samples for all controls to get the effective control strength for each threat event.

Usage

```
openfair_tef_tc_diff_lm(tef, tc, diff, lm, n = 10^4, verbose = FALSE)
```

Arguments

tef	Parameters for TEF simulation
tc	Parameters for TC simulation
diff	Parameters for DIFF simulation
lm	Parameters for LM simulation
n	Number of iterations to run.
verbose	Whether to print progress indicators.

Value

Dataframe of scenario name, threat_event count, loss_event count, mean TC and DIFF exceedance, and ALE samples.

See Also

Other OpenFAIR helpers: compare_tef_vuln(), get_mean_control_strength(), sample_diff(), sample_lef(), sample_lm(), sample_tc(), sample_vuln(), select_loss_opportunities()

Examples

```
data(mc_quantitative_scenarios)
params <- mc_quantitative_scenarios$scenario[[1]]$parameters
openfair_tef_tc_diff_lm(params$tef, params$tc, params$diff, params$lm, 10)
```

---

openfair_tef_tc_diff_plm_sr

Run an OpenFAIR simulation at the TEF/TC/DIFF/PLM/SR levels

Description

Run an OpenFAIR model with parameters provided for TEF, TC, DIFF, PLM, and SR sampling. If there are multiple controls provided for the scenario, the arithmetic mean (average) is taken across samples for all controls to get the effective control strength for each threat event.

Usage

```
openfair_tef_tc_diff_plm_sr(tef, tc, diff, plm, sr, n = 10^4, verbose = FALSE)
```

Arguments

tef	Parameters for TEF simulation.
tc	Parameters for TC simulation.
diff	Parameters for DIFF simulation.
plm	Parameters for PLM simulation.
sr	Parameters for SR simulation.
n	Number of iterations to run.
verbose	Whether to print progress indicators.

Value

Dataframe of scenario name, threat_event count, loss_event count, mean TC and DIFF exceedance, and ALE samples.

See Also

Other OpenFAIR models: `sample_tef()`

---

print.tidyrisk_scenario *Default printing of a tidyrisk_scenario*

Description

Basic printing of a tidyrisk scenario

Usage

```
## S3 method for class 'tidyrisk_scenario'
print(x, ...)
```

Arguments

x	A tidyrisk_scenario
...	Currently not used

---

read_qualitative_inputs *Load qualitative inputs*

Description

Given an input directory, load the key qualitative objects into memory.

Usage

```
read_qualitative_inputs(input_directory = "~/evaluator/inputs")
```

Arguments

input_directory	Location of input files.

Details

The key qualitative inputs for Evaluator processing include:

- `domains.csv`: domains and domain_ids
- `mappings.csv`: qualitative to quantitative mappings
- `capabilities.csv`: qualitative capabilities
- `qualitative_scenarios.csv`: qualitative risk scenarios

Value

List of domains, mappings, capabilities, and qualitative_scenarios

Examples

```
## Not run:
read_qualitative_inputs("~/evaluator/inputs")

## End(Not run)
```

`read_quantitative_inputs` *Load quantitative inputs*

Description

Given an input directory, load the quantitative objects into memory.

Usage

```
read_quantitative_inputs(input_directory = "~/evaluator/inputs")
```

Arguments

`input_directory` Location of input files.

Details

The key quantitative inputs for Evaluator processing include:

- `domains.csv` - domains and domain_ids
- `risk_tolerances.csv` - the risk tolerances of the organization
- `quantitative_scenarios.rds` - risk scenarios and quantified parameters

Value

List of domains, quantitative_scenarios, and risk_tolerances

Examples

```
## Not run:
read_quantitative_inputs("~/evaluator/inputs")

## End(Not run)
```

`risk_dashboard` *Launch a single page summary risk dashboard*

Description

Given the input files and the analysis summary file, create a basic one- page summary with an overview of the results per domain and scenario. Intended as a skeleton showing how the results could be displayed at an executive level.

Usage

```
risk_dashboard(
  input_directory = "~/evaluator/inputs",
  results_directory = "~/evaluator/results",
  output_file,
  intermediates_dir = tempdir(),
  quiet = TRUE,
  ...
)
```

Arguments

`input_directory`	Location of input files read by `read_quantitative_inputs`.
`results_directory`	Directory where the `simulation_results.rds` file is located.
`output_file`	Full path to the desired output file.
`intermediates_dir`	Location for intermediate knit files.
`quiet`	`TRUE` to suppress printing of pandoc output.
`...`	Any other parameters to pass to `rmarkdown::render`

Value

Default return values of the `rmarkdown::render` function.

Examples

```
## Not run:
risk_dashboard("~/inputs", "~/simulations")

## End(Not run)
```

risk_factory *Create a tidyrisk_factor sample function*

Description

Create a tidyrisk_factor sample function

Usage

```
risk_factory(factor_label = "TC")
```

Arguments

factor_label	abbreviation of the OpenFAIR element

run_simulation *Run simulations for a scenario*

Description

Given a quantitative scenario object of type `tidyrisk_scenario`, run an OpenFAIR Monte Carlo simulation.

Usage

```
run_simulation(
  scenario,
  iterations = 10000L,
  ale_maximum = NULL,
  verbose = FALSE,
  simulation_count = NULL
)
```

Arguments

scenario	A tidyrisk_scenario object.
iterations	Number of iterations to run on each scenario.
ale_maximum	Maximum practical annual losses.
verbose	Whether verbose console output is requested.
simulation_count	**DEPRECATED** Number of simulations to perform.

Value

Dataframe of results.

Examples

```
data(mc_quantitative_scenarios)
run_simulation(mc_quantitative_scenarios$scenario[[1]], 10)
```

`run_simulations` *Run simulations for a list of scenarios*

Description

Given a list of quantitative scenario objects of type `tidyrisk_scenario`, run a OpenFAIR Monte Carlo simulation for each scenario.

Usage

```
run_simulations(
  scenario,
  ...,
  iterations = 10000L,
  ale_maximum = NULL,
  verbose = FALSE,
  simulation_count = NULL
)
```

Arguments

`scenario`	A tidyrisk_scenario object.
`...`	Additional `tidyrisk_scenario` objects to simulate.
`iterations`	Number of iterations to run on each scenario.
`ale_maximum`	Maximum practical annual losses.
`verbose`	Whether verbose console output is requested.
`simulation_count`	**DEPRECATED** Number of simulations to perform.

Value

A list of one dataframe of results for each scenario.

Examples

```
# fetch three scenarios for this example
data(mc_quantitative_scenarios)
scenario_a <- mc_quantitative_scenarios$scenario[[1]]
scenario_b <- mc_quantitative_scenarios$scenario[[2]]
scenario_c <- mc_quantitative_scenarios$scenario[[3]]
run_simulations(scenario_a, scenario_b, scenario_c, iterations = 10)
```

sample_diff — *Calculate the difficulty presented by controls, given a function and parameters for that function*

Description

Calculate the difficulty presented by controls, given a function and parameters for that function

Usage

```
sample_diff(n, .func = NULL, params = NULL)
```

Arguments

n	Number of samples to generate.
.func	Function to use to simulate DIFF, defaults to rpert.
params	Optional parameters to pass to .func.

Value

List containing type ("diff"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR helpers: compare_tef_vuln(), get_mean_control_strength(), openfair_tef_tc_diff_lm(), sample_lef(), sample_lm(), sample_tc(), sample_vuln(), select_loss_opportunities()

sample_lef — *Sample loss event frequency*

Description

Sample loss event frequency

Usage

```
sample_lef(n, .func = NULL, params = NULL)
```

Arguments

n	Number of samples to generate.
.func	Function to use to simulate LEF, defaults to rnorm.
params	Optional parameters to pass to .func.

Value

List containing type ("lef"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR helpers: `compare_tef_vuln()`, `get_mean_control_strength()`, `openfair_tef_tc_diff_lm()`, `sample_diff()`, `sample_lm()`, `sample_tc()`, `sample_vuln()`, `select_loss_opportunities()`

`sample_lm` *Given a number of loss events and a loss distribution, calculate losses*

Description

Given a number of loss events and a loss distribution, calculate losses

Usage

```
sample_lm(n, .func = NULL, params = NULL)
```

Arguments

`n`	Number of samples to generate.
`.func`	Function to use to simulate TEF, defaults to `rpert`.
`params`	Optional parameters to pass to `.func`.

Value

List containing type ("lm"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR helpers: `compare_tef_vuln()`, `get_mean_control_strength()`, `openfair_tef_tc_diff_lm()`, `sample_diff()`, `sample_lef()`, `sample_tc()`, `sample_vuln()`, `select_loss_opportunities()`

sample_tc *Sample threat capabilities (TC) from a distribution function*

Description

Sample threat capabilities (TC) from a distribution function

Usage

```
sample_tc(n, params = NULL, .func = NULL)
```

Arguments

n	Number of samples to generate.
params	Optional parameters to pass to .func.
.func	Function to use to simulate TC, defaults to rpert.

Value

List containing type ("tc"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR helpers: compare_tef_vuln(), get_mean_control_strength(), openfair_tef_tc_diff_lm(), sample_diff(), sample_lef(), sample_lm(), sample_vuln(), select_loss_opportunities()

sample_tef *Calculate the number of simulated threat event frequencies (TEF)*

Description

Calculate the number of simulated threat event frequencies (TEF)

Usage

```
sample_tef(n, params = NULL, .func = NULL)
```

Arguments

n	Number of samples to generate.
params	Optional parameters to pass to .func.
.func	Function to use to simulate TEF, defaults to rpert.

Value

List containing type ("tef"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR models: `openfair_tef_tc_diff_plm_sr()`

`sample_vuln` *Calculate the vulnerability*

Description

Calculate the vulnerability

Usage

```
sample_vuln(n, .func = NULL, params = NULL)
```

Arguments

`n`	Number of samples to generate.
`.func`	Function to use to simulate VULN, defaults to `rbinom`.
`params`	Optional parameters to pass to `.func`.

Value

List containing type ("vuln"), samples (as a vector), and details (as a list).

See Also

Other OpenFAIR helpers: `compare_tef_vuln()`, `get_mean_control_strength()`, `openfair_tef_tc_diff_lm()`, `sample_diff()`, `sample_lef()`, `sample_lm()`, `sample_tc()`, `select_loss_opportunities()`

`select_loss_opportunities` *Determine which threat events result in loss opportunities*

Description

Composition function for use in `sample_vuln`, does a simple compare of all threat events where the threat capability (TC) is greater than the difficulty (DIFF).

Usage

```
select_loss_opportunities(tc, diff, n = NULL, ...)
```

Arguments

tc	Threat capability (as a percentage).
diff	Difficulty (as a percentage).
n	Number of samples to generate.
...	Optional parameters (currently ignored).

Value

List containing boolean values of length TC (as a vector) and details (as a list).

See Also

Other OpenFAIR helpers: compare_tef_vuln(), get_mean_control_strength(), openfair_tef_tc_diff_lm(), sample_diff(), sample_lef(), sample_lm(), sample_tc(), sample_vuln()

Examples

```
threat_capabilities <- c(.1, .5, .9)
difficulties <- c(.09, .6, .8)
select_loss_opportunities(threat_capabilities, difficulties)
```

---

split_sheet *Split a sheet of the survey spreadsheet into either capabilities or threats*

Description

The default data collection Excel spreadsheet solicits threat scenarios and applicable controls for each domain. This function takes a single sheet from the spreadsheet, as read by read_excel and pulls out either the capabilities or threats, as directed by the user.

Usage

```
split_sheet(dat, table_type = "capabilities")
```

Arguments

dat	Raw sheet input from read_excel.
table_type	Either capabilities or threats

Value

Extracted table as a dataframe

summarize_domains *Create domain-level summary of simulation results*

Description

Given a dataframe of raw results from run_simulations, summarize the individual results at a per-domain level. This domain-level summary is a useful data structure for aggregate reporting.

Usage

```
summarize_domains(simulation_results, domain_variable = "domain_id")
```

Arguments

simulation_results	Simulation results dataframe.
domain_variable	Variable by which individual simulations should be grouped.

Details

Summary stats created include:

- Mean/Min/Max/Median are calculated for loss events
- Median/Max/VaR are calculated for annual loss expected (ALE)
- Mean/Median/Max/Min are calculated for single loss expected (SLE)
- Mean percentage of threat capability exceeding difficulty on successful threat events
- Mean percentage of difficulty exceeding threat capability on defended events
- Vulnerability percentage

Value

Simulation results summarized across domains.

Examples

```
## Not run:
data(mc_simulation_results)
summarize_domains(mc_simulation_results)

## End(Not run)
```

summarize_iterations *Create a summary of outcomes across all scenarios*

Description

Given a dataframe of raw results from run_simulations, summarize the individual results at a per-iteration level.

Usage

```
summarize_iterations(simulation_result, ..., .key = "iteration")
```

Arguments

simulation_result	Results object for a single scenario.
...	Additional simulation result objects to summarize.
.key	Iteration ID field

Details

Summary stats created include: * Mean/Min/Max/Median are calculated for loss events * Median/Max/VaR are calculated for annual loss expected (ALE) * Mean/Median/Max/Min are calculated for single loss expected (SLE) * Mean percentage of threat capability exceeding difficulty on successful threat events * Mean percentage of difficulty exceeding threat capability on defended events * Vulnerability percentage * Z-score of ALE (outliers flagged as 2 >= z-score)

Value

Dataframe.

Examples

```
data(mc_simulation_results)
summarize_iterations(mc_simulation_results$results)
```

summarize_scenario *Create a summary of the simulation results for a single scenario*

Description

Given a dataframe of raw results from run_simulations, create summary statistics for the scenario. This is generally the most granular level of useful data for reporting and analysis (full simulation results are rarely directly helpful).

Usage

```
summarize_scenario(simulation_result)

summarize_scenarios(simulation_results)
```

Arguments

simulation_result	Results object for a single scenario.
simulation_results	Simulation results dataframe.

Details

Summary stats created include: * Mean/Min/Max/Median are calculated for loss events * Median/Max/VaR are calculated for annual loss expected (ALE) * Mean/Median/Max/Min are calculated for single loss expected (SLE) * Mean percentage of threat capability exceeding difficulty on successful threat events * Mean percentage of difficulty exceeding threat capability on defended events * Vulnerability percentage

Value

Dataframe of summary statistics.

Examples

```
data(mc_simulation_results)
# summarize a single scenario
summarize_scenario(mc_simulation_results$results[[1]])

# summarize all scenarios in a data frame
data(mc_simulation_results)
summarize_scenarios(mc_simulation_results)
```

---

summarize_to_disk *Create all summary files and write to disk*

Description

This is a wrapper around summarize_scenario and summarize_domains, calling both functions and writing the dataframes to a location on disk.

Usage

```
summarize_to_disk(simulation_results, results_dir)
```

Arguments

simulation_results	Simulation results dataframe.
results_dir	Directory to place simulation files.

Value

Tibble with paths to the created data files.

Examples

```
## Not run:
data(mc_simulation_results)
summarize_to_disk(mc_simulation_results, results_dir = tempdir())

## End(Not run)
```

---

theme_evaluator *Default ggplot theme used by all Evaluator-supplied graphics*

Description

Returns a standardized ggplot theme used by all built-in Evaluator plots.

Usage

```
theme_evaluator(base_family = "BentonSansRE")
```

Arguments

base_family	Font family.

Value

A ggplot theme object.

Examples

```
library(ggplot2)
p <- ggplot(mtcars) + geom_point(aes(wt, mpg, color = factor(gear))) + facet_wrap(~am)
font_family <- get_base_fontfamily()
p + theme_evaluator(font_family)
```

tidyrisk_factor *Construct a tidyrisk_factor object*

Description

Construct a tidyrisk_factor object

Usage

```
new_tidyrisk_factor(
  samples = double(),
  factor_label = character(),
  details = list()
)

tidyrisk_factor(samples, factor_label, details = list())
```

Arguments

samples	samples
factor_label	fl
details	details

validate_scenarios *Validate qualitative scenario data*

Description

Run a set of basic consistency checks on the key qualitative data inputs (scenarios, capabilities, domains, and mappings).

Usage

```
validate_scenarios(scenarios, capabilities, domains, mappings)
```

Arguments

scenarios	Dataframe of qualitative scenarios.
capabilities	Dataframe of capabilities.
domains	Dataframe of domain mappings.
mappings	Dataframe of qualitative to quantitative mappings.

Details

Checks that:

- All scenarios are distinct
- All controls referenced in scenarios are defined in the controls table
- All controls are distinct

Value

An invisible boolean as to success/failure of validation steps.

Examples

```
## Not run:
validate_scenarios(scenarios, capabilities, domains, mappings)

## End(Not run)
```

`validate_tidyrisk_scenario` *Validates that a scenario object is well formed*

Description

Validates that a scenario object is well formed

Usage

```
validate_tidyrisk_scenario(x)
```

Arguments

x	An object

vec_cast.tidyrisk_factor *Cast a* tidyrisk_factor *vector to a specified type*

Description

Cast a tidyrisk_factor vector to a specified type

Usage

```
## S3 method for class 'tidyrisk_factor'
vec_cast(x, to)
```

Arguments

x	Vectors to cast.
to	Type to cast to. If NULL, x will be returned as is.

vec_ptype_abbr.tidyrisk_scenario *Set an abbreviation when displaying an S3 column in a tibble*

Description

Set an abbreviation when displaying an S3 column in a tibble

Usage

```
vec_ptype_abbr.tidyrisk_scenario(x)
```

Arguments

x	An object

Index

∗ **OpenFAIR helpers**
  compare_tef_vuln, 4
  get_mean_control_strength, 14
  openfair_tef_tc_diff_lm, 25
  sample_diff, 32
  sample_lef, 32
  sample_lm, 33
  sample_tc, 34
  sample_vuln, 35
  select_loss_opportunities, 35
∗ **OpenFAIR models**
  openfair_tef_tc_diff_plm_sr, 26
  sample_tef, 34
∗ **datasets**
  mc_capabilities, 19
  mc_domain_summary, 20
  mc_domains, 19
  mc_mappings, 21
  mc_qualitative_scenarios, 21
  mc_quantitative_scenarios, 22
  mc_scenario_summary, 23
  mc_simulation_results, 24
∗ **result graphs**
  exposure_histogram, 10
  generate_event_outcomes_plot, 10
  generate_heatmap, 11
  loss_exceedance_curve, 17
  loss_scatterplot, 18

as.data.frame.tidyrisk_scenario *(*as_tibble.tidyrisk_scenario*)*, 3
as_tibble.tidyrisk_scenario, 3

calculate_max_losses, 4
compare_tef_vuln, 4, *14*, *26*, *32–36*
create_templates, 5
create_tidyrisk_scenario_skeleton, 6

derive_control_key, 7
derive_controls, 6
dollar_millions, 7

encode_scenarios, 8
explore_scenarios, 9
exposure_histogram, 10, *11*, *17, 18*

generate_event_outcomes_plot, *10*, 10, *11*, *17, 18*
generate_heatmap, *10, 11*, 11, *17, 18*
generate_report, 12
get_base_fontfamily, 13
get_mean_control_strength, *5*, 14, *26*, *32–36*

identify_outliers, 14
import_capabilities, 15, *16*
import_scenarios, *16*, 16
import_spreadsheet, 16
is_tidyrisk_scenario, 17

loss_exceedance_curve, *10, 11*, 17, *18*
loss_scatterplot, *10, 11*, *17*, 18

mc_capabilities, 19
mc_domain_summary, 20
mc_domains, 19
mc_mappings, 21
mc_qualitative_scenarios, 21
mc_quantitative_scenarios, 22
mc_scenario_summary, 23
mc_simulation_results, 24

new_tidyrisk_factor *(*tidyrisk_factor*)*, 41
new_tidyrisk_scenario, 24

openfair_example, 25
openfair_tef_tc_diff_lm, *5*, *14*, 25, *32–36*
openfair_tef_tc_diff_plm_sr, 26, *35*

print.tidyrisk_scenario, 27

rbinom, *35*
read_excel, *36*
read_qualitative_inputs, 27
read_quantitative_inputs, *9*, 28, *29*
risk_dashboard, 29
risk_factory, 30
rnorm, *32*
rpert, *32–34*
run_simulation, 30
run_simulations, 31, *37, 38*

sample_diff, *5*, *14*, *26*, 32, *33–36*
sample_lef, *4, 5*, *14*, *26*, *32*, 32, *33–36*
sample_lm, *5*, *14*, *26*, *32, 33*, 33, *34–36*
sample_tc, *5*, *14*, *26*, *32, 33*, 34, *35, 36*
sample_tef, *27*, 34
sample_vuln, *5*, *14*, *26*, *32–35*, 35, *36*
select_loss_opportunities, *5*, *14*, *26*, *32–35*, 35
split_sheet, 36
summarize_domains, 37, *39*
summarize_iterations, 38
summarize_scenario, 38, *39*
summarize_scenarios *(*summarize_scenario*)*, 38
summarize_to_disk, 39

theme_evaluator, 40
tidyrisk_factor, 41
tidyrisk_scenario, *6*, *22*, *30, 31*
tidyrisk_scenario *(*new_tidyrisk_scenario*)*, 24

validate_scenarios, 41
validate_tidyrisk_scenario, 42
vec_cast.tidyrisk_factor, 43
vec_ptype_abbr.tidyrisk_scenario, 43