Package ‘riskmetric’

June 29, 2023

Type Package
Title Risk Metrics to Evaluating R Packages
Description Facilities for assessing R packages against a number of metrics to help quantify their robustness.
Version 0.2.2
BugReports https://github.com/pharmaR/riskmetric/issues
License MIT + file LICENSE
Encoding UTF-8
Imports backports, utils, tools, xml2, httr, curl, urltools, memoise, BiocManager, cranlogs, covr, vctrs, pillar, tibble, pkgload, devtools
Suggests knitr, rmarkdown, withr, magrittr, dplyr, testthat, webmockr, jsonlite
RoxygenNote 7.2.3
VignetteBuilder knitr
Config/testthat/edition 3
NeedsCompilation no
Author R Validation Hub [aut],
        Doug Kelkoff [aut],
        Marly Gotti [aut],
        Eli Miller [cre, aut],
        Kevin K [aut],
        Yilong Zhang [aut],
        Eric Milliman [aut],
        Juliane Manitz [aut],
        Mark Padgham [ctb],
        PSI special interest group Application and Implementation of Methodologies in Statistics [cph]
Maintainer  Eli Miller <eli.miller@atorusresearch.com>
Repository  CRAN
Date/Publication  2023-06-29 11:30:04 UTC

R topics documented:

all_assessments .................................................. 3
assessment_error_as_warning .................................. 3
assessment_error_empty ........................................ 4
assessment_error_throw ......................................... 5
assess_covr_coverage ........................................... 5
assess_dependencies ............................................. 6
assess_downloads_1yr ............................................ 7
assess_exported_namespace ...................................... 8
assess_export_help ............................................... 8
assess_has_bug_reports_url .................................... 9
assess_has_examples ............................................ 10
assess_has_maintainer .......................................... 10
assess_has_news .................................................. 11
assess_has_source_control ..................................... 12
assess_has_vignettes ........................................... 12
assess_has_website ............................................. 13
assess_last_30_bugs_status .................................... 14
assess_license ................................................... 14
assess_news_current ............................................ 15
assess_remote_checks .......................................... 16
assess_reverse_dependencies ................................... 16
assess_r_cmd_check .............................................. 17
assess_size_codebase ........................................... 18
as_pkg_metric .................................................... 19
get_assessments .................................................. 19
metric_score ..................................................... 20
metric_score.pkg_metric_covr_coverage ....................... 20
metric_score.pkg_metric_dependencies ....................... 21
metric_score.pkg_metric_downloads_1yr ..................... 22
metric_score.pkg_metric_exported_namespace ............... 23
metric_score.pkg_metric_export_help ...................... 24
metric_score.pkg_metric_has_bug_reports_url ............. 24
metric_score.pkg_metric_has_examples ..................... 25
metric_score.pkg_metric_has_maintainer .................... 26
metric_score.pkg_metric_has_news ............................ 26
metric_score.pkg_metric_has_source_control .............. 27
metric_score.pkg_metric_has_vignettes ..................... 28
metric_score.pkg_metric_has_website ....................... 28
metric_score.pkg_metric_last_30_bugs_status .............. 29
metric_score.pkg_metric_license ............................. 30
metric_score.pkg_metric_news_current ..................... 30
**all_assessments**

A default list of assessments to perform for each package

**Description**

A default list of assessments to perform for each package

**Usage**

```r
all_assessments()
```

**Value**

a list of `assess_*` functions exported from riskmetric

---

**assessment_error_as_warning**

_Error handler for assessments to deescalate errors to warnings_

**Description**

Error handler for assessments to deescalate errors to warnings

**Usage**

```r
assessment_error_as_warning(e, name, assessment)
```
Arguments

- `e`: an error raised during a package reference assessment
- `name`: the name of the package whose package reference assessment raised the error
- `assessment`: the name of the assessment function which raised the error

Value

a pkg_metric object of pkg_metric_error subclass

See Also

Other assessment error handlers: `assessment_error_empty()`, `assessment_error_throw()`
**assessment_error_throw**

*Error handler for assessments to throw error immediately*

**Description**

Error handler for assessments to throw error immediately

**Usage**

```r
assessment_error_throw(e, name, assessment)
```

**Arguments**

- `e` an error raised during a package reference assessment
- `name` the name of the package whose package reference assessment raised the error
- `assessment` the name of the assessment function which raised the error

**Value**

the error encountered during assessment

**See Also**

Other assessment error handlers: `assessment_error_as_warning()`, `assessment_error_empty()`

---

**assess_covr_coverage** *Assess a package code coverage using the 'covr' package*

**Description**

Assess a package code coverage using the `covr` package

**Usage**

```r
assess_covr_coverage(x, ...)
```

**Arguments**

- `x` a `pkg_ref` package reference object
- `...` additional arguments passed on to S3 methods, rarely used
assess_dependencies

Value

a pkg_metric containing a list containing fields 'filecoverage' and 'totalcoverage' containing a
named numeric vector of file unit test coverage and a singular numeric value representing overall
test coverage respectively.

See Also

metric_score.pkg_metric_covr_coverage

Examples

## Not run:
asess_covr_coverage(pkg_ref("riskmetric"))

## End(Not run)

assess_dependencies Assessment of dependency footprint for a specific package

Description

Only Depends, Imports and LinkingTo dependencies are assessed because they are required

Usage

assess_dependencies(x, ...)

Arguments

  x a pkg_ref package reference object

  ... additional arguments passed on to S3 methods, rarely used

Details

The more packages a package relies on the more chances for errors exist.

Value

a pkg_metric containing a dataframe of package names and they type of dependency the package
being assess has to them

See Also

metric_score.pkg_metric_dependencies
assess_downloads_1yr

Examples

    ## Not run:
    assess_dependencies(pkg_ref("riskmetric"))

    ## End(Not run)

assess_downloads_1yr Assess a package for the number of downloads in the past year

Description

Assess a package for the number of downloads in the past year

Usage

    assess_downloads_1yr(x, ...)

Arguments

    x a pkg_ref package reference object

    ... additional arguments passed on to S3 methods, rarely used

Details

The more times a package has been downloaded the more extensive the user testing and the greater chance there is of someone finding a bug and logging it.

Value

    a pkg_metric containing a numeric value between [0,1] indicating the volume of downloads

See Also

    metric_score.pkg_metric_downloads_1yr

Examples

    ## Not run:
    assess_downloads_1yr(pkg_ref("riskmetric"))

    ## End(Not run)
assess_exported_namespace

Assess a package’s results from running R CMD check

Description

Assess a package’s results from running R CMD check

Usage

assess_exported_namespace(x, ...)

Arguments

  x a pkg_ref package reference object
  ... additional arguments passed on to S3 methods, rarely used

Value

a pkg_metric containing List of functions and objects exported by a package, excluding S3 methods

See Also

metric_score.pkg_metric_exported_namespace

Examples

## Not run:
assess_exported_namespace(pkg_ref("riskmetric"))

## End(Not run)

assess_export_help

Assess a package for availability of documentation for exported values

Description

Assess a package for availability of documentation for exported values

Usage

assess_export_help(x, ...)

Arguments

  x a pkg_ref package reference object
  ... additional arguments passed on to S3 methods, rarely used
assess_has_bug_reports_url

Value

a pkg_metric containing a logical vector indicating existence of documentation for each namespace export

See Also

metric_score.pkg_metric_export_help

Examples

```r
## Not run:
assess_export_help(pkg_ref("riskmetric"))
## End(Not run)
```
assess_has_examples  Assess a package for the presence of example or usage fields in function documentation

Description
Assess a package for the presence of example or usage fields in function documentation

Usage
assess_has_examples(x, ...)

Arguments
x  a pkg_ref package reference object
...  additional arguments passed on to S3 methods, rarely used

Value
a pkg_metric containing an integer value indicating the proportion of discovered files with examples

See Also
metric_score.pkg_metric_has_examples

Examples
## Not run:
assess_has_examples(pkg_ref("riskmetric"))
## End(Not run)

assess_has_maintainer  Assess a package for an associated maintainer

Description
Assess a package for an associated maintainer

Usage
assess_has_maintainer(x, ...)
assess_has_news

Assess a package for the presence of a NEWS file

Description
Assess a package for the presence of a NEWS file

Usage
assess_has_news(x, ...)

Arguments
x a pkg_ref package reference object
... additional arguments passed on to S3 methods, rarely used

Value
a pkg_metric containing an integer value indicating the number of discovered NEWS files

See Also
metric_score.pkg_metric_has_news

Examples
## Not run:
assess_has_news(pkg_ref("riskmetric"))
## End(Not run)

assess_has_news
Assess a package for the presence of a NEWS file
assess_has_source_control

Assess a package for an associated source control url

Description
Assess a package for an associated source control url

Usage
assess_has_source_control(x, ...)

Arguments
  x          a pkg_ref package reference object
  ...        additional arguments passed on to S3 methods, rarely used

Value
a pkg_metric containing a character vector of source control urls associated with the package

See Also
metric_score.pkg_metric_has_source_control

Examples
## Not run:
assess_has_source_control(pkg_ref("riskmetric"))

## End(Not run)

assess_has_vignettes  Assess a package for the presence of Vignettes files

Description
Assess a package for the presence of Vignettes files

Usage
assess_has_vignettes(x, ...)

Arguments
  x          a pkg_ref package reference object
  ...        additional arguments passed on to S3 methods, rarely used
assess_has_website

Value

a pkg_metric containing an integer value indicating the number of discovered vignettes files

See Also

metric_score.pkg_metric_has_vignettes

Examples

## Not run:
assess_has_vignettes(pkg_ref("riskmetric"))
## End(Not run)

assess_has_website

Assess a package for an associated website url

Description

Assess a package for an associated website url

Usage

assess_has_website(x, ...)

Arguments

x

a pkg_ref package reference object

... additional arguments passed on to S3 methods, rarely used

Value

a pkg_metric containing a character vector of website urls associated with the package

See Also

metric_score.pkg_metric_has_website

Examples

## Not run:
assess_has_website(pkg_ref("riskmetric"))
## End(Not run)
assess_last_30_bugs_status

Assess how many recent BugReports have been closed

Description
Assess how many recent BugReports have been closed

Usage
assess_last_30_bugs_status(x, ...)

Arguments

  x               a pkg_ref package reference object
  ...            additional arguments passed on to S3 methods, rarely used

Value

  a pkg_metric containing a logical vector indicating whether a recent BugReport was closed

See Also
metric_score.pkg_metric_last_30_bugs_status

Examples

## Not run:
assess_last_30_bugs_status(pkg_ref("riskmetric"))

## End(Not run)

assess_license

Assess a package for an acceptable license

Description
Assess a package for an acceptable license

Usage
assess_license(x, ...)

Arguments

  x               a pkg_ref package reference object
  ...            additional arguments passed on to S3 methods, rarely used
assess_news_current

Value

a pkg_metric containing a string indicating the license under which the package is released

See Also

metric_score.pkg_metric_license

Examples

## Not run:
analyse_news(pkg_ref("riskmetric"))
## End(Not run)
assess_reverse_dependencies

assess_remote_checks  Assess package checks from CRAN/Bioc or R CMD check

Description
Assess package checks from CRAN/Bioc or R CMD check

Usage
assess_remote_checks(x, ...)

Arguments
x a pkg_ref package reference object
... additional arguments passed on to S3 methods, rarely used

Value
a pkg_metric containing Tally of R CMD check results run on different OS flavors by BioC or CRAN

See Also
metric_score.pkg_metric_remote_checks

Examples
## Not run:
assess_remote_checks(pkg_ref("riskmetric"))
## End(Not run)

assess_reverse_dependencies

Generate list of Reverse Dependencies for a package

Description
Generate list of Reverse Dependencies for a package

Usage
assess_reverse_dependencies(x, ...)


assess_r_cmd_check

Assess a package’s results from running R CMD check

Description
Assess a package’s results from running R CMD check

Usage
assess_r_cmd_check(x, ...)

Arguments
x a pkg_ref package reference object
... additional arguments passed on to S3 methods, rarely used

Value
a pkg_metric containing Tally of errors, warnings and notes from running R CMD check locally

See Also
metric_score.pkg_metric_r_cmd_check
describe a package for size of code base

Usage

assess_size_codebase(x, ...)

Arguments

  x : a pkg_ref package reference object
  ... : additional arguments passed on to S3 methods, rarely used

Value

  a pkg_metric containing a numeric value for number of lines of code base for a package

See Also

  metric_score.pkg_metric_size_codebase

Examples

  ## Not run:
  assess_size_codebase(pkg_ref("riskmetric"))

  ## End(Not run)
as_pkg_metric

Convert an object to a pkg_metric

Description

Convert an object to a pkg_metric

Usage

as_pkg_metric(x, class = c())

Arguments

x

data to store as a pkg_metric
class

a subclass to differentiate the pkg_metric object

Value

a pkg_metric object

get_assessments

Get a specific set of assess_* functions for pkg_assess

Description

Get a specific set of assess_* functions for pkg_assess

Usage

get_assessments(fxn_string = "")

Arguments

fxn_string

vector of assess functions

Value

a list of specific assess_* functions exported from riskmetric
**metric_score**

*Score a package metric*

**Description**

Convert a package metric into a numeric value between 0 to 1

**Usage**

```r
metric_score(x, ...)
```

**Arguments**

- `x`: A `pkg_metric_*` class object to score
- `...`: Additional arguments unused

**Value**

A score of a package risk metric

---

**metric_score.pkg_metric_covr_coverage**

*Score a package for unit test coverage*

**Description**

Returns the overall test coverage from a covr coverage report

**Usage**

```r
## S3 method for class 'pkg_metric_covr_coverage'
metric_score(x, ...)
```

**Arguments**

- `x`: A `pkg_metric_covr_coverage` package metric object
- `...`: Additional arguments unused

**Value**

A numeric

**Examples**

```r
## Not run: metric_score(assess_covr_coverage(pkg_ref("riskmetric")))
```
**Description**

Calculates a regularized score based on the number of dependencies a package has. Convert the number of dependencies \( \text{NROW}(x) \) into a validation score \([0,1]\)

\[ 1 - \frac{1}{1 + \exp(-0.5 * (\text{NROW}(x) + 4))] \]

**Usage**

```r
## S3 method for class 'pkg_metric_dependencies'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_dependencies` package metric object
- `...`: additional arguments unused

**Details**

The scoring function is the classic logistic curve

\[ \frac{1}{1 + \exp(-k(x - x[0]))} \]

\( x = \text{NROW}(x) \), sigmoid midpoint is 5 reverse dependencies, ie. \( x[0] = 4 \), and logistic growth rate of \( k = 0.5 \).

\[ 1 - \frac{1}{1 + \exp(\text{NROW}(x) - 4))} \]

**Value**

numeric value between 0 (high number of dependencies) and 1 (low number of dependencies)

**Examples**

```r
## Not run: metric_score(assess_dependencies(pkg_ref("riskmetric")))
```
metric_score.pkg_metric_downloads_1yr

Defining an Assessment Scoring Function

Description

Score a package for the number of downloads in the past year regularized. Convert the number of downloads $x$ in the past year into a validation score $[0,1]$

$$1 - 150,000/(x + 150,000)$$

Usage

```r
## S3 method for class 'pkg_metric_downloads_1yr'
metric_score(x, ...)
```

Arguments

- `x`: a `pkg_metric_downloads_1yr` package metric object
- `...`: additional arguments unused

Details

The scoring function is a simplification of the classic logistic curve

$$1/(1 + \exp(-k(x - x[0])))$$

with a log scale for the number of downloads $x = \log(x)$, sigmoid midpoint is 1000 downloads, ie. $x[0] = \log(1,000)$, and logistic growth rate of $k = 0.5$.

$$1 - 1/(1 + \exp(\log(x) - \log(1.5e5))) = 1 - 150,000/(x + 150,000)$$

Value

numeric value between 0 (low) and 1 (high download volume) converting the number of downloads.

Examples

```r
## Not run: metric_score(assess_downloads_1yr(pkg_ref("riskmetric")))
```
Description

Score a package for the number of exported objects it has; regularized. Convert the number of exported objects \( \text{length}(x) \) into a validation score \([0,1]\)

\[
1 / (1 + \exp(-0.5 * (\sqrt{\text{length}(x)} + \sqrt{5})))
\]

Usage

```r
## S3 method for class 'pkg_metric_exported_namespace'
metric_score(x, ...)
```

Arguments

- `x` a `pkg_metric_exported_namespace` package metric object
- `...` additional arguments unused

Details

The scoring function is the classic logistic curve

\[
1 / (1 + \exp(-k(x - x[0])))
\]

with a square root scale for the number of exported objects \( x = \sqrt{\text{length}(x)} \), sigmoid midpoint is 25 exported objects, i.e. \( x[0] = \sqrt{5} \), and logistic growth rate of \( k = 0.25 \).

\[
1 / (1 + \exp(-0.25 * \sqrt{\text{length}(x)} - \sqrt{25}))
\]

Value

numeric value between 0 (high number of exported objects) and 1 (low number of exported objects)

Examples

```r
## Not run: metric_score(assess_exported_namespace(pkg_ref("riskmetric")))
```
### metric_score.pkg_metric_export_help

*Score a package for availability of documentation for exported values*

**Description**

Coerce a logical vector indicating availability of export documentation

**Usage**

```r
## S3 method for class 'pkg_metric_export_help'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_export_help` package metric object
- `...`: additional arguments unused

**Value**

1 if any NEWS files are found, otherwise 0

**Examples**

```r
## Not run: metric_score(assess_export_help(pkg_ref("riskmetric")))
```

---

### metric_score.pkg_metric_has_bug_reports_url

*Score a package for the presence of a bug report url*

**Description**

Score a package for the presence of a bug report url

**Usage**

```r
## S3 method for class 'pkg_metric_has_bug_reports_url'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_has_bug_reports_url` package metric object
- `...`: additional arguments unused
Value

A logical value indicating whether the package has a BugReports field filled in.

Examples

```r
## Not run: metric_score(assess_has_bug_reports_url(pkg_ref("riskmetric")))
```

Description

Coerce a logical vector indicating availability of example or usage documentation.

Usage

```r
## S3 method for class 'pkg_metric_has_examples'
metric_score(x, ...)
```

Arguments

- `x`: a `pkg_metric_has_examples` package metric object
- `...`: additional arguments unused

Value

1 if any example or usage fields are found, otherwise 0.

Examples

```r
## Not run: metric_score(assess_has_examples(pkg_ref("riskmetric")))
```
### `pkg_metric_has_maintainer`

**Score a package for inclusion of an associated maintainer**

**Description**

Coerce a list of maintainers into a numeric value indicating whether the number of listed maintainers is greater than 0.

**Usage**

```r
## S3 method for class 'pkg_metric_has_maintainer'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_has_maintainer` package metric object
- `...`: additional arguments unused

**Value**

1 if any maintainer is provided, otherwise 0

**Examples**

```r
## Not run: metric_score(assess_has_maintainer(pkg_ref("riskmetric")))
```

### `pkg_metric_has_news`

**Score a package for the presence of a NEWS file**

**Description**

Coerce the number of news files to binary indication of valid NEWS files.

**Usage**

```r
## S3 method for class 'pkg_metric_has_news'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_has_news` package metric object
- `...`: additional arguments unused
Value
1 if any NEWS files are found, otherwise 0

Examples
## Not run: metric_score(assess_has_news(pkg_ref("riskmetric")))
### metric_score.pkg_metric_has_vignettes

Score a package for the presence of a Vignettes file

**Description**

Coerce the number of vignettes files to binary indication of valid Vignettes

**Usage**

```r
## S3 method for class 'pkg_metric_has_vignettes'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_has_vignettes` package metric object
- `...`: additional arguments unused

**Value**

1 if any Vignettes files are found, otherwise 0

**Examples**

```r
## Not run: metric_score(assess_has_vignettes(pkg_ref("riskmetric")))
```

---

### metric_score.pkg_metric_has_website

Score a package for inclusion of an associated website url

**Description**

Coerce a list of website urls into a numeric value indicating whether the number of listed urls is greater than 0.

**Usage**

```r
## S3 method for class 'pkg_metric_has_website'
metric_score(x, ...)
```

**Arguments**

- `x`: a `pkg_metric_has_website` package metric object
- `...`: additional arguments unused
Value

1 if any website url is provided, otherwise 0

Examples

```r
## Not run: metric_score(assess_has_website(pkg_ref("riskmetric")))
```

---

**Description**

Score a package for number of recently opened BugReports that are now closed

**Usage**

```r
## S3 method for class 'pkg_metric_last_30_bugs_status'
metric_score(x, ...)
```

**Arguments**

- `x` a `pkg_metric_last_30_bugs_status` package metric object
- `...` additional arguments unused

**Value**

A fractional value indicating percentage of last 30 bug reports that are now closed

**Examples**

```r
## Not run: metric_score(assess_last_30_bugs_status(pkg_ref("riskmetric")))
```
metric_score.pkg_metric_license

Score a package for acceptable license

Description
Maps a license string to a score

Usage
## S3 method for class 'pkg_metric_license'
metric_score(x, ...)

Arguments
x a pkg_metric_license package metric object
...
additional arguments unused

Value
score of metric license

Examples
## Not run: metric_score(assess_license(pkg_ref("riskmetric")))

metric_score.pkg_metric_news_current

Score a package for NEWS files updated to current version

Description
Coerce a logical vector of discovered up-to-date NEWS to a metric score

Usage
## S3 method for class 'pkg_metric_news_current'
metric_score(x, ...)

Arguments
x a pkg_metric_news_current package metric object
...
additional arguments unused
**Value**

1 if any NEWS files are up-to-date, otherwise 0

**Examples**

```r
## Not run: metric_score(assess_news_current(pkg_ref("riskmetric")))
```

---

**metric_score.pkg_metric_remote_checks**

Score a package based on R CMD check results run by BioC or CRAN

**Description**

The scoring function is the number of OS flavors that passed with OK or NOTES + 0.5*the number of OS’s that produced WARNINGS divided by the number of OS’s checked

**Usage**

```r
## S3 method for class 'pkg_metric_remote_checks'
metric_score(x, ...)
```

**Arguments**

- `x` a `pkg_metric_remote_checks` package metric object
- `...` additional arguments unused

**Value**

a fractional value indicating percentage OS flavors that did not produce an error or warning from R CMD check

**Examples**

```r
## Not run: metric_score(assess_remote_checks(pkg_ref("riskmetric")))
```
Description

Score a package for the number of reverse dependencies it has; regularized Convert the number of reverse dependencies length(x) into a validation score [0,1]

\[ \frac{1}{1 + \exp(-0.5 \times (\text{sqrt}(\text{length}(x)) + \text{sqrt}(5)))} \]

Usage

```r
## S3 method for class 'pkg_metric_reverse_dependencies'
metric_score(x, ...)
```

Arguments

- `x` a `pkg_metric_reverse_dependencies` package metric object
- `...` additional arguments unused

Details

The scoring function is the classic logistic curve

\[ \frac{1}{1 + \exp(-k(x - x[0]))} \]

with a square root scale for the number of reverse dependencies \( x = \text{sqrt}(\text{length}(x)) \), sigmoid midpoint is 5 reverse dependencies, ie. \( x[0] = \text{sqrt}(5) \), and logistic growth rate of \( k = 0.5 \).

\[ \frac{1}{1 + -0.5 \times \exp(\text{sqrt}(\text{length}(x)) - \text{sqrt}(5)))} \]

Value

numeric value between 1 (high number of reverse dependencies) and 0 (low number of reverse dependencies)

Examples

```r
## Not run: metric_score(assess_reverse_dependencies(pkg_ref("riskmetric")))
```
**metric_score** 

*Score a package based on R CMD check results run locally*

### Description
The scoring function is the weighted sum of notes (0.1), errors (1) and warnings (0.25), with a maximum score of 1 (no errors, notes or warnings) and a minimum score of 0. Essentially, the metric will allow up to 10 notes, 1 error or 4 warnings before returning the lowest score of 0.

### Usage

```r
## S3 method for class 'pkg_metric_r_cmd_check'
metric_score(x, ...)
```

### Arguments

- `x`: a `pkg_metric_r_cmd_check` package metric object
- `...`: additional arguments unused

### Value
A weighted sum of errors and warnings of all tests performed.

### Examples

```r
## Not run: metric_score(assess_r_cmd_check(pkg_ref("riskmetric")))
```

---

**metric_score** 

*Score a package for number of lines of code*

### Description
Scores packages based on its codebase size, as determined by number of lines of code.

### Usage

```r
## S3 method for class 'pkg_metric_size_codebase'
mometric_score(x, ...)
```

### Arguments

- `x`: a `pkg_metric_size_codebase` package metric object
- `...`: additional arguments unused
pkg_assess

Value

numeric value between 0 (low) and 1 (large number of lines of code) converting the number of downloads.

Examples

```r
## Not run: metric_score(assess_size_codebase(pkg_ref("riskmetric")))
```

pkg_assess

Apply assess_* family of functions to a package reference

Description

By default, use all assess_* funtions in the riskmetric namespace and produce a tibble with one column per assessment applied.

Usage

```r
pkg_assess(
  x,
  assessments = all_assessments(),
  ...,
  error_handler = assessment_error_empty
)
```

Arguments

- **x** A single pkg_ref object or tibble of package references to assess
- **assessments** A list of assessment functions to apply to each package reference. By default, a list of all exported assess_* functions from the riskmetric package.
- **...** additional arguments unused
- **error_handler** A function, which accepts a single parameter expecting the raised error, which will be called if any errors occur when attempting to apply an assessment function.

Value

Either a list_of_pkg_metric object when a single pkg_ref object is passed as x, or a tibble of metrics when a list_of_pkg_ref or tibble is passed as x. When a tibble is returned, it has one row per package reference and a new column per assessment function, with cells of that column as package metric objects returned when the assessment was called with the associated package reference.
**pkg_metric**

### Assessment function catalog

- **assess_last_30_bugs_status**: vector indicating whether BugReports status is closed
- **assess_covr_coverage**: Package unit test coverage
- **assess_size_codebase**: number of lines of code base
- **assess_export_help**: exported objects have documentation
- **assess_r_cmd_check**: Package check results
- **assess_dependencies**: Package dependency footprint
- **assess_reverse_dependencies**: List of reverse dependencies a package has
- **assess_license**: software is released with an acceptable license
- **assess_has_maintainer**: a vector of associated maintainers
- **assess_remote_checks**: Number of OS flavors that passed/warned/errored on R CMD check
- **assess_exported_namespace**: Objects exported by package
- **assess_has_website**: a vector of associated website urls
- **assess_downloads_1yr**: number of downloads in the past year
- **assess_has_news**: number of discovered NEWS files
- **assess_has_vignettes**: number of discovered vignettes files
- **assess_has_examples**: proportion of discovered function files with examples
- **assess_has_source_control**: a vector of associated source control urls
- **assess_has_bug_reports_url**: presence of a bug reports url in repository
- **assess_news_current**: NEWS file contains entry for current version number

---

**pkg_metric**

*A helper for structuring assessment return objects for dispatch with the score function*

---

**Description**

A helper for structuring assessment return objects for dispatch with the score function

**Usage**

```r
pkg_metric(x = NA, ..., class = c())
```

**Arguments**

- **x**: data to store as a pkg_metric
- **...**: additional attributes to bind to the pkg_metric object
- **class**: a subclass to differentiate the pkg_metric object

**Value**

a pkg_metric object
Create a package reference

Description

Create a package reference from package name or filepath, producing an object in which package metadata will be collected as risk assessments are performed. Depending on where the package was found - whether it is found as source code, in a local library or from a remote host - an S3 subclass is given to allow for source-specific collection of metadata. See 'Details' for a breakdown of subclasses. Different sources can be specified by passing a subclass as an argument named 'source', see details.

Usage

```r
pkg_ref(x, ...)
pkg_install(x, lib.loc = NULL)
pkg_source(x)
pkg_cran(x, repos = getOption("repos", "https://cran.rstudio.com"))
pkg_bioc(x)
pkg_missing(x)
pkg_library(lib.loc)
as_pkg_ref(x, ...)
```

Arguments

- `x`: A singular character value, character vector or list of character values of package names or source code directory paths.
- `...`: Additional arguments passed to methods.
- `lib.loc`: The path to the R library directory of the installed package.
- `repos`: URL of CRAN repository to pull package metadata.

Details

Package reference objects are used to collect metadata pertaining to a given package. As data is needed for assessing a package’s risk, this metadata populates fields within the package reference object.

The `pkg_ref` S3 subclasses are used extensively for divergent metadata collection behaviors dependent on where the package was discovered. Because of this, there is a rich hierarchy of subclasses to articulate the different ways package information can be found.
A source argument can be passed using the ‘source’ argument. This will override the logic that riskmetric does when determining a package source. This can be useful when you are scoring the most recent version present on a repository, or testing a specific library.

- **pkg_ref** A default class for general metadata collection.
  - **pkg_source** A reference to a source code directory.
  - **pkg_install** A reference to a package installation location in a package library. A specific library can be passed by passing the path to the library as the parameter ‘lib.loc’
  - **pkg_remote** A reference to package metadata on a remote server.
    * **pkg_cran_remote** A reference to package information pulled from the CRAN repository.
    * **pkg_bioc_remote** A reference to package information pulled from the Bioconductor repository.
    * **pkg_git_remote** A reference to a package source code git repository. (not yet implemented)

**Value**

When a single value is provided, a single **pkg_ref** object is returned, possibly with a subclass based on where the package was found. If a vector or list is provided, a **list_of_pkg_ref** object constructed with the `list_of` is returned, which can be considered analogous to a list. See ‘Details’ for further information about **pkg_ref** subclasses.

**Package Cohorts**

*Experimental!* Package cohorts are structures to determine the risk of a set of packages. ‘pkg_library()’ can be called to create a object containing the **pkg_ref** objects of all packages in a system library.

**Examples**

```r
## Not run:
ref_1 <- pkg_ref("utils")
ref_1$source # returns 'pkg_install'

# lib.loc can be used to specify a library for pkg_install
ref_3 <- pkg_ref("utils", source = "pkg_install", lib.loc = .libPaths()[1])

# You can also override this behavior with a source argument
ref_2 <- pkg_ref("utils", source = "pkg_cran_remote")
ref_2$source # returns 'pkg_cran_remote'
```

## End(Not run)
pkg_ref_cache

S3 generic to calculate a 'pkg_ref' field

Description

Reactively retrieve and cache 'pkg_ref' metadata

Value

a pkg_ref field

Caching Details

pkg_ref class fields: The pkg_ref class structures an environment with special handling for
indexing into the pkg_ref class using the $ or [[ operators. For all intents and purposes, the
pkg_ref class is works conceptually similar to a lazy, immutable list, and uses the pkg_ref_cache
function internally to lazily retrieve package reference fields.

Lazy metadata caching: Laziness in a pkg_ref object refers to the delayed evaluation of the
contents of its fields. Since some metadata is time or computationally intensive to retrieve, and
unnecessary for some assessments, we want to avoid that retrieval until it is needed.
The first time that a field is accessed within a pkg_ref object x, a corresponding pkg_ref_cache
S3 generic is called. For example, when x$description is first accessed, the pkg_ref object
uses the function pkg_ref_cache.description to attempt to retrieve the contents of the corre-
sponding DESCRIPTION file.

Often, the way that this data is collected might be different depending on the subclass of the
pkg_ref. In the case of the description metadata, a reference to a local install might be able
to read in a local file directly, whereas a reference to a remote source of metadata might require
first downloading the file. For this reason, many pkg_ref_cache.* functions are themselves S3
generics that dispatch on the class of the pkg_ref object, allowing for divergent behaviors for
different source of package metadata.

pkg_ref field immutability: Once a field has been calculated, its value is immutable. This
behavior was chosen because of the long time frame over which package metadata changes, ren-
dering it unnecessary to continually reevaluate fields each time they are accessed.
This means that within an assessment, a given field for a package will only ever be calculated
once and preserved for downstream use.

Examples

## Not run:
# implementing a new field called "first_letter" that is consistently derived
# across all pkg_ref objects:

pkg_ref_cache.first_letter <- function(x, name, ...) {
    substring(x$name, 1, 1)
}
pkg_ref_class_hierarchy

The 'pkg_ref' subclass hierarchy, used for pkg_ref object creation with a specified subclass

Description

The 'pkg_ref' subclass hierarchy, used for pkg_ref object creation with a specified subclass

Usage

pkg_ref_class_hierarchy

Format

An object of class list of length 1.
pkg_score

Score a package assessment, collapsing results into a single numeric value

Description

pkg_score() calculates the risk involved with using a package. Risk ranges from 0 (low-risk) to 1 (high-risk).

Usage

pkg_score(x, ..., error_handler = score_error_default)

Arguments

x
A pkg_metric object, whose subclass is used to choose the appropriate scoring method for the atomic metric metadata. Optionally, a tibble can be provided, in which cases all pkg_metric values will be scored.

... Additional arguments passed to summarize_scores when an object of class tbl_df is provided, unused otherwise.

error_handler Specify a function to be called if the class can't be identified. Most commonly this occurs for pkg_metric objects of subclass pkg_metric_error, which is produced when an error is encountered when calculating an associated assessment.

Value

A numeric value if a single pkg_metric is provided, or a tibble with pkg_metric objects scored and returned as numeric values when a tibble is provided.

See Also

score_error_default score_error_zero score_error_NA

Examples

## Not run:

# scoring a single assessment
metric_score(assess_has_news(pkg_ref("riskmetric")))

# scoring many assessments as a tibble
library(dplyr)
pkg_score(pkg_assess(as_tibble(pkg_ref(c("riskmetric", "riskmetric")))))

## End(Not run)
score_error_default

Default score error handling, emitting a warning and returning 0

Description

Default score error handling, emitting a warning and returning 0

Usage

score_error_default(x, ...)

Arguments

x A pkg_metric_* class object to score
...

Value

a value of package score

score_error_NA

Score error handler to silently return NA

Description

Score error handler to silently return NA

Usage

score_error_NA(...) 

Arguments

...

Value

a value of package score
**score_error_zero**  
Score error handler to silently return 0

**Description**
Score error handler to silently return 0

**Usage**

```r
score_error_zero(...)```

**Arguments**

... Additional arguments unused

**Value**
a value of package score

---

**summarize_scores**  
Summarize a default set of assessments into a single risk score

**Description**
This function serves as an example for how a risk score might be derived. Assuming all assessments provided by `riskmetric` are available in a dataset, this function can be used to calculate a vector of risks.

**Usage**

```r
summarize_scores(data, weights = NULL)```

**Arguments**

data a tibble of scored assessments whose column names match those provided by `riskmetric`'s `pkg_assess` function.

weights an optional vector of non-negative weights to be assigned to each assessment.

**Value**
a numeric vector of risk scores
Examples

```r
## Not run:
library(dplyr)
summarize_scores(pkg_score(pkg_assess(as_tibble(pkg_ref("riskmetric")))))

library(dplyr)
pkg_ref("riskmetric") %>%
  pkg_assess() %>%
  pkg_score() %>%
  summarize_scores()

## End(Not run)
```
Index

* assessment error handlers
  assessment_error_as_warning, 3
  assessment_error_empty, 4
  assessment_error_throw, 5
* datasets
  pkg_ref_class_hierarchy, 39

all_assessments, 3
as_pkg_metric, 19
as_pkg_ref(pkg_ref), 36
assess_covr_coverage, 5, 35
assess_dependencies, 6, 35
assess_downloads_1yr, 7, 35
assess_export_help, 8, 35
assess_exported_namespace, 8, 35
assess_has_bug_reports_url, 9, 35
assess_has_examples, 10, 35
assess_has_maintainer, 10, 35
assess_has_news, 11, 35
assess_has_source_control, 12, 35
assess_has_vignettes, 12, 35
assess_has_website, 13, 35
assess_last_30_bugs_status, 14, 35
assess_license, 14, 35
assess_news_current, 15, 35
assess_r_cmd_check, 17, 35
assess_remote_checks, 16, 35
assess_reverse_dependencies, 16, 35
assess_size_codebase, 18, 35
assessment_error_as_warning, 3, 4, 5
assessment_error_empty, 4, 4, 5
assessment_error_throw, 4, 5

get_assessments, 19

list_of, 37

metric_score, 20
metric_score.pkg_metric_covr_coverage, 6, 20
metric_score.pkg_metric_dependencies, 6, 21
metric_score.pkg_metric_downloads_1yr, 7, 22
metric_score.pkg_metric_export_help, 9, 24
metric_score.pkg_metric_exported_namespace, 8, 23
metric_score.pkg_metric_has_bug_reports_url, 9, 24
metric_score.pkg_metric_has_examples, 10, 25
metric_score.pkg_metric_has_maintainer, 11, 26
metric_score.pkg_metric_has_news, 11, 26
metric_score.pkg_metric_has_source_control, 12, 27
metric_score.pkg_metric_has_vignettes, 13, 28
metric_score.pkg_metric_has_website, 13, 28
metric_score.pkg_metric_last_30_bugs_status, 14, 29
metric_score.pkg_metric_license, 15, 30
metric_score.pkg_metric_news_current, 15, 30
metric_score.pkg_metric_r_cmd_check, 17, 33
metric_score.pkg_metric_remote_checks, 16, 31
metric_score.pkg_metric_reverse_dependencies, 17, 32
metric_score.pkg_metric_size_codebase, 18, 33

pkg_assess, 34, 42
pkg_bioc (pkg_ref), 36
pkg_cran (pkg_ref), 36
pkg_install (pkg_ref), 36
pkg_library (pkg_ref), 36
pkg_metric, 35
pkg_missing (pkg_ref), 36
pkg_ref, 34, 36
pkg_ref_cache, 38
pkg_ref_class_hierarchy, 39
pkg_score, 40
pkg_source (pkg_ref), 36

score_error_default, 41
score_error_NA, 41
score_error_zero, 42
summarize_scores, 42

tibble, 34, 40, 42