Package ‘r2dii.match’

March 25, 2024

**Title**  Tools to Match Corporate Lending Portfolios with Climate Data

**Version**  0.2.0

**Description**  These tools implement in R a fundamental part of the software 'PACTA' (Paris Agreement Capital Transition Assessment), which is a free tool that calculates the alignment between financial portfolios and climate scenarios (<https://www.transitionmonitor.com/>). Financial institutions use PACTA to study how their capital allocation decisions align with climate change mitigation goals. This package matches data from corporate lending portfolios to asset level data from market-intelligence databases (e.g. power plant capacities, emission factors, etc.). This is the first step to assess if a financial portfolio aligns with climate goals.

**License**  MIT + file LICENSE

**URL**  https://rmi-pacta.github.io/r2dii.match/
         https://github.com/RMI-PACTA/r2dii.match

**BugReports**  https://github.com/RMI-PACTA/r2dii.match/issues

**Depends**  R (>= 3.4)

**Imports**  data.table, dplyr (>= 0.8.5), glue, lifecycle, magrittr, purrr, r2dii.data (>= 0.4.0), rlang, stringdist, stringi, tibble, tidyr, tidyselect, utils

**Suggests**  covr, rmarkdown, spelling, testthat (>= 2.1.0)

**Config/testthat/edition**  3

**Config/Needs/website**  rmi-pacta/pacta.pkgdown.rmitemplate

**Encoding**  UTF-8

**Language**  en-US

**RoxygenNote**  7.3.1

**NeedsCompilation**  no

**Author**  Alex Axthelm [aut, cre] (https://orcid.org/0000-0001-8579-8565),
           Jackson Hoffart [aut, ctr] (https://orcid.org/0000-0002-8600-5042),
           Mauro Lepore [aut, ctr] (https://orcid.org/0000-0002-1986-7988),
Description

This is a helper to select the minimum loanbook columns you need to run `match_name()`. Using more columns may use too much time and memory.

Usage

```r
crucial_lbk()
```

Value

A character vector.

See Also

Other helpers: `prioritize_level()`

Examples

```r
crucial_lbk()
```
match_name

**match_name**  
Match a loanbook to asset-based company data (abcd) by the name_* columns

---

**Description**

match_name() scores the match between names in a loanbook dataset (columns can be name_direct_loantaker, name_intermediate_parent* and name_ultimate_parent) with names in an asset-based company data (column name_company). The raw names are first internally transformed, and aliases are assigned. The similarity between aliases in each of the loanbook and abcd is scored using stringdist::stringsim().

**Usage**

```r
match_name(
  loanbook, abcd,
  by_sector = TRUE,
  min_score = 0.8,
  method = "jw",
  p = 0.1,
  overwrite = NULL,
  join_id = NULL,
  ...
)
```

**Arguments**

- **loanbook, abcd** data frames structured like r2dii.data::loanbook_demo and r2dii.data::abcd_demo.
- **by_sector** Should names only be compared if companies belong to the same sector?
- **min_score** A number between 0-1, to set the minimum score threshold. A score of 1 is a perfect match.
- **p** Prefix factor for Jaro-Winkler distance. The valid range for p is 0 <= p <= 0.25. If p=0 (default), the Jaro-distance is returned. Applies only to method='jw'.
- **overwrite** A data frame used to overwrite the sector and/or name columns of a particular direct loantaker or ultimate parent. To overwrite only sector, the value in the name column should be NA and vice-versa. This file can be used to manually match loanbook companies to abcd.
- **join_id** A join specification passed to dplyr::inner_join(). If a character string, it assumes identical join columns between loanbook and abcd. If a named character vector, it uses the name as the join column of loanbook and the value as the join column of abcd.
- **...** Arguments passed on to stringdist::stringsim().
Value

A data frame with the same groups (if any) and columns as loanbook, and the additional columns:

- id_2dii - an id used internally by match_name() to distinguish companies
- level - the level of granularity that the loan was matched at (e.g. direct_loantaker or ultimate_parent)
- sector - the sector of the loanbook company
- sector_abcd - the sector of the abcd company
- name - the name of the loanbook company
- name_abcd - the name of the abcd company
- score - the score of the match (manually set this to 1 prior to calling prioritize() to validate the match)
- source - determines the source of the match. (equal to loanbook unless the match is from overwrite)

The returned rows depend on the argument min_value and the result of the column score for each loan: * If any row has score equal to 1, match_name() returns all rows where score equals 1, dropping all other rows. * If no row has score equal to 1, match_name() returns all rows where score is equal to or greater than min_score. * If there is no match the output is a 0-row tibble with the expected column names – for type stability.

Package options

r2dii.match.sector_classifications: Allows you to use your own sector_classifications instead of the default. This feature is experimental and may be dropped and/or become a new argument to match_name().

Assigning aliases

The transformation process used to compare names between loanbook and abcd datasets applies best practices commonly used in name matching algorithms:

- Remove special characters.
- Replace language specific characters.
- Abbreviate certain names to reduce their importance in the matching.
- Spell out numbers to increase their importance.

Handling grouped data

This function ignores but preserves existing groups.

See Also

Other main functions: prioritize()
Examples

```r
# Not run:
library(r2dii.data)
library(tibble)

# Small data for examples
loanbook <- head(loanbook_demo, 50)
abcd <- head(abcd_demo, 50)

match_name(loanbook, abcd)

match_name(loanbook, abcd, min_score = 0.9)

# Use your own `sector_classifications`
your_classifications <- tibble(
  sector = "power",
  borderline = FALSE,
  code = "D35.11",
  code_system = "XYZ"
)

# match on LEI
loanbook <- tibble(
  sector_classification_system = "XYZ",
  sector_classification_direct_loantaker = "D35.11",
  id_ultimate_parent = "UP15",
  name_ultimate_parent = "Won't fuzzy match",
  id_direct_loantaker = "C294",
  name_direct_loantaker = "Won't fuzzy match",
  lei_direct_loantaker = "LEI123"
)

abcd <- tibble(
  name_company = "alpine knits india pvt. limited",
  sector = "power",
  lei = "LEI123"
)

match_name(loanbook, abcd, join_by = c(lei_direct_loantaker = "lei"))

restore <- options(r2dii.match.sector_classifications = your_classifications)

loanbook <- tibble(
  sector_classification_system = "XYZ",
  sector_classification_direct_loantaker = "D35.11",
  id_ultimate_parent = "UP15",
  name_ultimate_parent = "Alpine Knits India Pvt. Limited",
  id_direct_loantaker = "C294",
  name_direct_loantaker = "Yuamen Xinneng Thermal Power Co Ltd"
)

abcd <- tibble(

```
prioritize(prioritize)

prioritize

**Description**

When multiple perfect matches are found per loan (e.g. a match at direct_loantaker level and ultimate_parent level), we must prioritize the desired match. By default, the highest priority is the most granular match (i.e. direct_loantaker).

**Usage**

```r
prioritize(data, priority = NULL)
```

**Arguments**

- `data` A data frame like the validated output of match_name(). See Details on how to validate data.
- `priority` One of:
  - NULL: defaults to the default level priority as returned by prioritize_level().
  - A character vector giving a custom priority.
  - A function to apply to the output of prioritize_level(), e.g. `rev`.
  - A quosure-style lambda function, e.g. `~ rev(.x)`.

**Details**

**How to validate data**

Write the output of match_name() into a .csv file with:

```r
# Writing to current working directory
matched %>%
  readr::write_csv("matched.csv")
```

Compare, edit, and save the data manually:

- Open matched.csv with any spreadsheet editor (Excel, Google Sheets, etc.).
- Compare the columns name and name_abcd manually to determine if the match is valid. Other information can be used in conjunction with just the names to ensure the two entities match (sector, internal information on the company structure, etc.)
- Edit the data:
  - If you are happy with the match, set the score value to 1.
  - Otherwise set or leave the score value to anything other than 1.
- Save the edited file as, say, valid_matches.csv.

Re-read the edited file (validated) with:

```r
# Reading from current working directory
valid_matches <- readr::read_csv("valid_matches.csv")
```

Value

A data frame with a single row per loan, where score is 1 and priority level is highest.

Handling grouped data

This function ignores but preserves existing groups.

See Also

`match_name()`, `prioritize_level()`.

Other main functions: `match_name()`

Examples

```r
library(dplyr)

# styler: off
matched <- tribble(
  ~sector, ~sector_abcd, ~score, ~id_loan, ~level,
  "coal",    "coal",    1,    "aa",    "ultimate_parent",
  "coal",    "coal",    1,    "aa",    "direct_loantaker",
  "coal",    "coal",    1,    "bb",    "intermediate_parent",
  "coal",    "coal",    1,    "bb",    "ultimate_parent",
)
# styler: on
prioritize_level(matched)

# Using default priority
prioritize(matched)

# Using the reverse of the default priority
prioritize(matched, priority = rev)

# Same
prioritize(matched, priority = ~ rev(.x))
```
# Using a custom priority
bad_idea <- c("intermediate_parent", "ultimate_parent", "direct_loantaker")

prioritize(matched, priority = bad_idea)

prioritize_level

---

### Description

Arrange unique level values in default order of priority.

### Usage

```r
prioritize_level(data)
```

### Arguments

- **data**
  
  A data frame, commonly the output of `match_name()`.

### Value

A character vector of the default level priority per loan.

### See Also

Other helpers: `crucial_lbk()`

### Examples

```r
matched <- tibble::tibble(
  level = c(
    "intermediate_parent_1",
    "direct_loantaker",
    "direct_loantaker",
    "direct_loantaker",
    "ultimate_parent",
    "intermediate_parent_2"
  )
)
prioritize_level(matched)
```
Index

* helpers
  crucial_lbk, 2
  prioritize_level, 8

* main functions
  match_name, 3
  prioritize, 6

crucial_lbk, 2, 8
dplyr::inner_join(), 3
match_name, 3, 7
match_name(), 6–8
prioritize, 4, 6
prioritize_level, 2, 8
prioritize_level(), 6, 7
r2dii.data::abcd_demo, 3
r2dii.data::loanbook_demo, 3

stringdist::stringdist-metrics, 3
stringdist::stringsim(), 3