Package 'validateR'

September 11, 2026

Title Simple Data Frame Validation and Quality Checks

Version 0.2.0

Description Checks a data frame for common data quality issues, including missing values, outliers, duplicate rows, and type inconsistencies. Results are returned as a structured 'validation_report' object with 'print' and 'plot' methods for quick inspection.

License MIT + file LICENSE

Encoding UTF-8

Config/roxygen2/version 8.1.0

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports graphics, stats

VignetteBuilder knitr

URL https://github.com/UgyenNorbu/validateR

BugReports https://github.com/UgyenNorbu/validateR/issues

NeedsCompilation no

Author Ugyen Norbu [aut, cre]

Maintainer Ugyen Norbu <unorbu2019@gmail.com>

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2026-09-11 12:20:02 UTC

Contents

`check_date_like` *Check for date-like inconsistencies in character columns*

Description

Examines specified character columns of a data frame and tests whether their non-missing values can be parsed as dates under a given format. If at least 80% (but less than 100%) of a column's non-missing values parse successfully, the column is flagged, and the row positions of the non-parsable ("straggler") values are reported for inspection. Unlike `check_numeric_like()`, this check does not attempt to guess a date format automatically - the caller must supply the expected format for each column to be checked, since date formats are ambiguous (for example, `"03/04/2023"` could mean March 4th or April 3rd depending on convention) and guessing incorrectly could silently produce misleading results.

Usage

```
check_date_like(data, date_columns)
```

Arguments

`data`	A data frame.
`date_columns`	A named character vector, where each name is a column in `data` to check, and each value is the expected date format for that column, using the format codes accepted by `base::as.Date()` (for example, `"%Y-%m-%d"` or `"%m/%d/%Y"`). An error is raised if any name in `date_columns` does not match a column in `data`.

Value

A named list, one element per flagged column (named by column name). Each element is itself a list containing:

expected_type Always `"date"`.

proportion_parsable_to_date Percentage of non-missing values that were successfully parsed under the given format, rounded to 2 decimals.

bad_rows Integer vector of row indices whose values could not be parsed under the given format.

Returns an empty list if no columns are flagged.

Examples

```
df <- data.frame(
  signup_date = c(
    "2023-01-15", "2023-02-20", "not a date",
    "2023-03-10", "2023-04-01"
  ),
  notes = c("a", "b", "c", "d", "e")
)
check_date_like(df, date_columns = c(signup_date = "%Y-%m-%d"))
```

`check_duplicates` *Check for duplicate rows in a data frame*

Description

Identifies exact duplicate rows across all columns of a data frame.

Usage

```
check_duplicates(data)
```

Arguments

`data`	A data frame

Value

A list where each element is an integer vector giving the row indices of a group of duplicate rows. Returns an empty list if no duplicates are found.

Examples

```
df <- data.frame(
  x = c(1, 2, 1, 4, 5, 4, 4),
  y = c("a", "b", "a", "d", "e", "d", "d")
)

check_duplicates(df)
```

check_missing *Check for missing values in a data frame*

Description

Computes the count and percentage of missing (NA) values for each column in a data frame.

Usage

```
check_missing(data)
```

Arguments

data	A data frame.

Value

A data frame with one row per column of data, containing:

column_name Name of the column.

n_missing Number of missing values.

pct_missing Percentage of missing values, rounded to 2 decimals.

Examples

```
df <- data.frame(x = c(1, NA, 3), y = c("a", "b", NA))
check_missing(df)
```

check_numeric_like *Check for type inconsistencies in character columns*

Description

Examines each character column of a data frame and tests whether its non-missing values can be converted to numeric. If at least 80% (but less than 100%) of non-missing values are numeric-looking, the column is flagged as likely intended to be numeric, and the row positions of the non-convertible ("straggler") values are reported for inspection. Columns that are already numeric, or that fall below the 80% threshold, are not flagged.

Usage

```
check_numeric_like(data)
```

Arguments

data	A data frame.

Value

A named list, one element per flagged column (named by column name). Each element is itself a list containing:

expected_type The type the column is likely meant to be (currently always `"numeric"`).

proportion_convertible Percentage of non-missing values that can be converted to numeric, rounded to 2 decimals.

bad_rows Integer vector of row indices whose values could not be converted to numeric.

Returns an empty list if no columns are flagged.

Examples

```
df <- data.frame(
  income = c("50000", "62000", "N/A", "48000", "71000"),
  city = c("Thimphu", "Paro", "Punakha", "Wangdue", "Trongsa"),
  stringsAsFactors = FALSE
)
check_numeric_like(df)
```

---

check_outliers *Check for outlier values in a data frame*

Description

Calculates the count and percentage of outlier values using IQR method (values beyond Q1 - 1.5*IQR or Q3 + 1.5*IQR).

Usage

```
check_outliers(data)
```

Arguments

`data`	A data frame

Value

A data frame with one row per column of `data`, containing:

column_name Name of the column.

n_outliers Number of outlier values.

pct_outliers Percentage of outlier values, rounded to 2 decimals.

Examples

```
df <- data.frame(
  x = c(1, 2, 3, 4, 100),
  y = c(10, 12, 11, 13, 12)
)
check_outliers(df)
```

`new_validation_report` *Construct a validation_report object*

Description

Low-level constructor that assembles already-computed check results into a single `validation_report` object. This function does not run any checks itself — it simply packages results produced elsewhere (typically by `check_missing()`, `check_outliers()`, `check_duplicates()`, `check_numeric_like()`, and `check_date_like()`) into a structured, classed object. In most cases, users should call `validate_df()` instead, which runs all five checks and calls this constructor automatically.

Usage

```
new_validation_report(
  df,
  df_name,
  missing_result,
  outliers_result,
  duplicates_result,
  numeric_like_result,
  date_like_result
)
```

Arguments

`df`	The data frame that was checked. Used only to derive dimensions; not stored directly in the report.
`df_name`	A character string giving the name to display for the data frame in printed output.
`missing_result`	Output of `check_missing()`.
`outliers_result`	Output of `check_outliers()`.
`duplicates_result`	Output of `check_duplicates()`.
`numeric_like_result`	Output of `check_numeric_like()`.
`date_like_result`	Output of `check_date_like()`

Value

An object of class `validation_report`, a list containing:

meta A list with `df_name` (character), `dim` (integer vector of rows and columns), and `timestamp` (the time the report was created).

results A list with four elements, `missing`, `outliers`, `duplicates`, `numeric_like` and `date_like`, containing the corresponding check results.

Examples

```
df <- data.frame(x = c(1, NA, 3), y = c("a", "b", "c"))
report <- new_validation_report(
  df = df,
  df_name = "df",
  missing_result = check_missing(df),
  outliers_result = check_outliers(df),
  duplicates_result = check_duplicates(df),
  numeric_like_result = check_numeric_like(df),
  date_like_result = check_date_like(df, date_columns = c(y = "%Y-%m-%d"))
)
class(report)
```

---

plot.validation_report

Plot a validation_report object

Description

Displays two side-by-side bar charts summarizing a `validation_report`: the count of missing values per column, and the count of outliers per numeric column.

Usage

```
## S3 method for class 'validation_report'
plot(x, ...)
```

Arguments

`x`	A `validation_report` object.
`...`	Further arguments passed to or from other methods (currently unused).

Value

The input `x`, returned invisibly. Called for its side effect of producing a plot.

print.validation_report

Print a validation_report object

Description

Displays a concise summary of a `validation_report`, showing the data frame's dimensions, when it was checked, and how many columns (or groups, for duplicates) were flagged by each check.

Usage

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

Arguments

`x`	A `validation_report` object.
`...`	Further arguments passed to or from other methods (currently unused).

Value

The input `x`, returned invisibly.

validate_df

Validate a data frame for common data quality issues

Description

Runs a full suite of data quality checks on a data frame — missing values, outliers, duplicate rows, and type inconsistencies — and returns a single, structured report summarizing the results. This is the main entry point for the package; most users should start here rather than calling the individual `check_*()` functions directly.

Usage

```
validate_df(data, date_columns = NULL)
```

Arguments

`data`	A data frame to validate.
`date_columns`	Optional. A named character vector mapping column names in `data` to their expected date format (using the format codes accepted by `base::as.Date()`). If supplied, `check_date_like()` is run on those columns as a fifth check. If `NULL` (the default), date checking is skipped entirely, and the report notes it was not checked.

Value

An object of class `validation_report`. See `new_validation_report()` for details on its structure. Printing the result shows a short summary; the full results for each check are available under `report$results$missing`, `report$results$outliers`, `report$results$duplicates`, `report$results$numeric_like` and `report$results$date_like`.

Examples

```
df <- data.frame(
  age = c(25, 30, NA, 45, 200),
  income = c("50000", "62000", "N/A", "48000", "71000"),
  stringsAsFactors = FALSE
)
report <- validate_df(df)
report

# With an optional date check
df2 <- data.frame(
  signup_date = c(
    "2023-01-15", "2023-02-20", "not a date",
    "2023-03-10", "2023-04-01"
  ),
  stringsAsFactors = FALSE
)
report2 <- validate_df(df2, date_columns = c(signup_date = "%Y-%m-%d"))
report2
```

Index