

# Package ‘diffdf’

March 18, 2020

**Type** Package

**Title** Dataframe Difference Tool

**Version** 1.0.4

**Description** Functions for comparing two data.frames against each other. The core functionality is to provide a detailed breakdown of any differences between two data.frames as well as providing utility functions to help narrow down the source of problems and differences.

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.1.2)

**Imports** tibble

**Suggests** testthat, lubridate, knitr, rmarkdown, purrr, dplyr, stringi, stringr, devtools, covr

**RoxygenNote** 7.0.2

**VignetteBuilder** knitr

**License** MIT + file LICENSE

**URL** <https://github.com/gowerc/diffdf>

**BugReports** <https://github.com/gowerc/diffdf/issues>

**NeedsCompilation** no

**Author** Craig Gower-Page [cre, aut],  
Kieran Martin [aut]

**Maintainer** Craig Gower-Page <craig.gower-page@roche.com>

**Repository** CRAN

**Date/Publication** 2020-03-17 23:10:03 UTC

## R topics documented:

as_ascii_table . . . . .	3
as_cropped_char . . . . .	3

cast_variables . . . . .	4
class_merge . . . . .	4
compare_vectors . . . . .	5
compare_vectors.default . . . . .	5
compare_vectors.factor . . . . .	6
compare_vectors.numeric . . . . .	6
construct_issue . . . . .	7
convert_to_issue . . . . .	7
diffdf . . . . .	8
diffdf_has_issues . . . . .	10
diffdf_issuerows . . . . .	10
factor_to_character . . . . .	11
find_difference . . . . .	12
generate_keyname . . . . .	12
get_casted_dataset . . . . .	13
get_casted_vector . . . . .	13
get_issue_dataset . . . . .	14
get_issue_message . . . . .	14
get_print_message . . . . .	15
get_print_message.default . . . . .	15
get_print_message.issue . . . . .	16
get_table . . . . .	16
has_unique_rows . . . . .	17
identify_att_differences . . . . .	17
identify_class_differences . . . . .	18
identify_differences . . . . .	18
identify_extra_cols . . . . .	19
identify_extra_rows . . . . .	19
identify_matching_cols . . . . .	20
identify_mode_differences . . . . .	20
identify_properties . . . . .	21
identify_unsupported_cols . . . . .	21
invert . . . . .	21
is_variable_different . . . . .	22
print.diffdf . . . . .	22
recursive_reduce . . . . .	23
sort_then_join . . . . .	23
string_pad . . . . .	24

---

as_ascii_table	<i>as_ascii_table</i>
----------------	-----------------------

---

### Description

This function takes a data.frame and attempts to convert it into a simple ascii format suitable for printing to the screen. It is assumed all variable values have a as.character() method in order to cast them to character.

### Usage

```
as_ascii_table(dat, line_prefix = " ")
```

### Arguments

dat	Input dataset to convert into a ascii table
line_prefix	Symbols to prefix in front of every line of the table

---

as_cropped_char	<i>as_cropped_char</i>
-----------------	------------------------

---

### Description

Makes any character string above x chars. Reduce down to a x char string with ...

### Usage

```
as_cropped_char(inval, crop_at = 30)
```

### Arguments

inval	a single element value
crop_at	character limit

---

cast_variables	<i>cast_variables</i>
----------------	-----------------------

---

### Description

Function to cast datasets columns if they have differing types Restricted to specific cases, currently integer and double, and character and factor

### Usage

```
cast_variables(  
  BASE,  
  COMPARE,  
  ignore_vars = NULL,  
  cast_integers = FALSE,  
  cast_factors = FALSE  
)
```

### Arguments

BASE	base dataset
COMPARE	comparison dataset
ignore_vars	Variables not to be considered for casting
cast_integers	Logical - Whether integers should be cased to double when compared to doubles
cast_factors	Logical - Whether characters should be casted to characters when compared to characters

---

class_merge	<i>class_merge</i>
-------------	--------------------

---

### Description

Convenience function to put all classes an object has into one string

### Usage

```
class_merge(x)
```

### Arguments

x	an object
---	-----------

---

compare_vectors	<i>compare_vectors</i>
-----------------	------------------------

---

**Description**

Compare two vectors looking for differences

**Usage**

```
compare_vectors(target, current, ...)
```

**Arguments**

target	the base vector
current	a vector to compare target to
...	Additional arguments which might be passed through (numerical accuracy)

---

compare_vectors.default	<i>compare_vectors.default</i>
-------------------------	--------------------------------

---

**Description**

Default method, if the vector is not numeric or factor. Basic comparison

**Usage**

```
## Default S3 method:  
compare_vectors(target, current, ...)
```

**Arguments**

target	the base vector
current	a vector to compare target to
...	Additional arguments which might be passed through (numerical accuracy)

---

```
compare_vectors.factor
      compare_vectors.factor
```

---

**Description**

Compares factors. Sets them as character and then compares

**Usage**

```
## S3 method for class 'factor'
compare_vectors(target, current, ...)
```

**Arguments**

target	the base vector
current	a vector to compare target to
...	Additional arguments which might be passed through (numerical accuracy)

---

```
compare_vectors.numeric
      compare_vectors.numeric
```

---

**Description**

This is a modified version of the all.equal function which returns a vector rather than a message

**Usage**

```
## S3 method for class 'numeric'
compare_vectors(
  target,
  current,
  tolerance = sqrt(.Machine$double.eps),
  scale = NULL
)
```

**Arguments**

target	the base vector
current	a vector to compare target to
tolerance	Level of tolerance for differences between two variables
scale	Scale that tolerance should be set on. If NULL assume absolute

---

construct_issue	<i>construct_issue</i>
-----------------	------------------------

---

**Description**

Make an s3 object with class issue and possible additional class, and assign other arguments to attributes

**Usage**

```
construct_issue(value, message, add_class = NULL)
```

**Arguments**

value	the value of the object
message	the value of the message attribute
add_class	additional class to add

---

convert_to_issue	<i>convert_to_issue</i>
------------------	-------------------------

---

**Description**

converts the count value into the correct issue format

**Usage**

```
convert_to_issue(datin)
```

**Arguments**

datin	data inputted
-------	---------------

diffdf

*diffdf***Description**

Compares 2 dataframes and outputs any differences.

**Usage**

```
diffdf(
  base,
  compare,
  keys = NULL,
  suppress_warnings = FALSE,
  strict_numeric = TRUE,
  strict_factor = TRUE,
  file = NULL,
  tolerance = sqrt(.Machine$double.eps),
  scale = NULL
)
```

**Arguments**

base	input dataframe
compare	comparison dataframe
keys	vector of variables (as strings) that defines a unique row in the base and compare dataframes
suppress_warnings	Do you want to suppress warnings? (logical)
strict_numeric	Flag for strict numeric to numeric comparisons (default = TRUE). If False diffdf will cast integer to double where required for comparisons. Note that variables specified in the keys will never be casted.
strict_factor	Flag for strict factor to character comparisons (default = TRUE). If False diffdf will cast factors to characters where required for comparisons. Note that variables specified in the keys will never be casted.
file	Location and name of a text file to output the results to. Setting to NULL will cause no file to be produced.
tolerance	Set tolerance for numeric comparisons. Note that comparisons fail if $(x-y)/scale > tolerance$ .
scale	Set scale for numeric comparisons. Note that comparisons fail if $(x-y)/scale > tolerance$ . Setting as NULL is a slightly more efficient version of $scale = 1$ .

**Examples**

```
x <- subset( iris, ~Species)
x[1,2] <- 5
COMPARE <- diffdf( iris, x)
print( COMPARE )
print( COMPARE , "Sepal.Length" )

#### Sample data frames

DF1 <- data.frame(
  id = c(1,2,3,4,5,6),
  v1 = letters[1:6],
  v2 = c(NA , NA , 1 , 2 , 3 , NA)
)

DF2 <- data.frame(
  id = c(1,2,3,4,5,7),
  v1 = letters[1:6],
  v2 = c(NA , NA , 1 , 2 , NA , NA),
  v3 = c(NA , NA , 1 , 2 , NA , 4)
)

diffdf(DF1 , DF1 , keys = "id")

# We can control matching with scale/location for example:

DF1 <- data.frame(
  id = c(1,2,3,4,5,6),
  v1 = letters[1:6],
  v2 = c(1,2,3,4,5,6)
)

DF2 <- data.frame(
  id = c(1,2,3,4,5,6),
  v1 = letters[1:6],
  v2 = c(1.1,2,3,4,5,6)
)

diffdf(DF1 , DF2 , keys = "id")
diffdf(DF1 , DF2 , keys = "id", tolerance = 0.2)
diffdf(DF1 , DF2 , keys = "id", scale = 10, tolerance = 0.2)

# We can use strict_factor to compare factors with characters for example:

DF1 <- data.frame(
  id = c(1,2,3,4,5,6),
  v1 = letters[1:6],
  v2 = c(NA , NA , 1 , 2 , 3 , NA),
  stringsAsFactors = FALSE
)

DF2 <- data.frame(
  id = c(1,2,3,4,5,6),
```

```

    v1 = letters[1:6],
    v2 = c(NA , NA , 1 , 2 , 3 , NA)
  )

diffdf(DF1 , DF2 , keys = "id", strict_factor = TRUE)
diffdf(DF1 , DF2 , keys = "id", strict_factor = FALSE)

```

---

diffdf\_has\_issues      *diffdf\_has\_issues*

---

### Description

Utility function which returns TRUE if an diffdf object has issues or FALSE if an diffdf object does not have issues

### Usage

```
diffdf_has_issues(x)
```

### Arguments

x                      diffdf object

### Examples

```

# Example with no issues
x <- diffdf( iris, iris )
diffdf_has_issues(x)

# Example with issues
iris2 <- iris
iris2[2,2] <- NA
x <- diffdf( iris , iris2 , suppress_warnings = TRUE)
diffdf_has_issues(x)

```

---

diffdf\_issuerows      *diffdf\_issuerows*

---

### Description

This function takes a diffdf object and a dataframe and subsets the dataframe for problem rows as identified in the comparison object. If vars has been specified only issue rows associated with those variable(s) will be returned.

**Usage**

```
diffdf_issuerows(df, diff, vars = NULL)
```

**Arguments**

df	dataframe to be subsetted
diff	diffdf object
vars	(optional) character vector containing names of issue variables to subset dataframe on. A value of NULL (default) will be taken to mean available issue variables.

**Details**

Note that `diffdf_issuerows` can be used to subset against any dataframe. The only requirement is that the original variables specified in the `keys` argument to `diffdf` are present on the dataframe you are subsetting against. However please note that if no keys were specified in `diffdf` then the row number is used. This means using `diffdf_issuerows` without a keys against an arbitrary dataset can easily result in nonsense rows being returned. It is always recommended to supply keys to `diffdf`.

**Examples**

```
iris2 <- iris
for ( i in 1:3) iris2[i,i] <- 99
x <- diffdf( iris , iris2, suppress_warnings = TRUE)
diffdf_issuerows( iris , x)
diffdf_issuerows( iris2 , x)
diffdf_issuerows( iris2 , x , vars = "Sepal.Length")
diffdf_issuerows( iris2 , x , vars = c("Sepal.Length" , "Sepal.Width"))
```

---

factor\_to\_character     *factor\_to\_character*

---

**Description**

Takes a dataframe and converts any factor variables to character

**Usage**

```
factor_to_character(dsin, vars = NULL)
```

**Arguments**

dsin	input dataframe
vars	variables to consider for conversion. Default NULL will consider every variable within the dataset

---

find_difference	<i>find_difference</i>
-----------------	------------------------

---

**Description**

This determines if two vectors are different. It expects vectors of the same length and type, and is intended to be used after checks have already been done Initially picks out any nas (matching nas count as a match) Then compares remaining vector

**Usage**

```
find_difference(target, current, ...)
```

**Arguments**

target	the base vector
current	a vector to compare target to
...	Additional arguments which might be passed through (numerical accuracy)

---

generate_keyname	<i>generate_keyname</i>
------------------	-------------------------

---

**Description**

Function to generate a name for the keys if not provided

**Usage**

```
generate_keyname(
  BASE,
  COMP,
  replace_names = c("..ROWNUMBER..", "..RN..", "..ROWN..", "..N..")
)
```

**Arguments**

BASE	base dataset
COMP	comparison dataset
replace_names	a vector of replacement names. Used for recursion, should be edited in function for clarity

---

`get_casted_dataset`      *get\_casted\_dataset*

---

**Description**

Internal utility function to loop across a dataset casting all target variables

**Usage**

`get_casted_dataset(df, columns, whichdat)`

**Arguments**

<code>df</code>	dataset to be casted
<code>columns</code>	columns to be casted
<code>whichdat</code>	whether base or compare is being casted (used for messages)

---

`get_casted_vector`      *get\_casted\_vector*

---

**Description**

casts a vector depending on its type and input

**Usage**

`get_casted_vector(colin, colname, whichdat)`

**Arguments**

<code>colin</code>	column to cast
<code>colname</code>	name of vector
<code>whichdat</code>	whether base or compare is being casted (used for messages)

---

<code>get_issue_dataset</code>	<i>get_issue_dataset</i>
--------------------------------	--------------------------

---

**Description**

Internal function used by `diffdf_issuerows` to extract the dataframe from each a target issue. In particular it also strips off any non-key variables

**Usage**

```
get_issue_dataset(issue, diff)
```

**Arguments**

<code>issue</code>	name of issue to extract the dataset from diff
<code>diff</code>	diffdf object which contains issues

---

<code>get_issue_message</code>	<i>get_issue_message</i>
--------------------------------	--------------------------

---

**Description**

Simple function to grab the issue message

**Usage**

```
get_issue_message(object, ...)
```

**Arguments**

<code>object</code>	inputted object of class issue
<code>...</code>	other arguments

---

`get_print_message`      *get\_print\_message*

---

**Description**

Get the required text depending on type of issue

**Usage**

`get_print_message(object, ...)`

**Arguments**

<code>object</code>	inputted object of class issue
<code>...</code>	other arguments

---

`get_print_message.default`  
*get\_print\_message.default*

---

**Description**

Errors, as this should only ever be given an issue

**Usage**

```
## Default S3 method:  
get_print_message(object)
```

**Arguments**

<code>object</code>	issue
---------------------	-------

---

```
get_print_message.issue  
    get_print_message.issue
```

---

**Description**

Get text from a basic issue, based on the class of the value of the issue

**Usage**

```
## S3 method for class 'issue'  
get_print_message(object)
```

**Arguments**

object            an object of class issue\_basic

---

```
get_table            get_table
```

---

**Description**

Generate nice looking table from a data frame

**Usage**

```
get_table(dsin, row_limit = 10)
```

**Arguments**

dsin            dataset  
row\_limit        Maximum number of rows displayed in dataset

---

has_unique_rows	<i>has_unique_rows</i>
-----------------	------------------------

---

**Description**

Check if a data sets rows are unique

**Usage**

```
has_unique_rows(DAT, KEYS)
```

**Arguments**

DAT	input data set (data frame)
KEYS	Set of keys which should be unique

---

identify_att_differences	<i>identify_att_differences</i>
--------------------------	---------------------------------

---

**Description**

Identifies any attribute differences between two data frames

**Usage**

```
identify_att_differences(BASE, COMP, exclude_cols = "")
```

**Arguments**

BASE	Base dataset for comparison (data.frame)
COMP	Comparator dataset to compare base against (data.frame)
exclude_cols	Columns to exclude from comparison

---

```
identify_class_differences
      identify_class_differences
```

---

**Description**

Identifies any class differences between two data frames

**Usage**

```
identify_class_differences(BASE, COMP)
```

**Arguments**

BASE	Base dataset for comparison (data.frame)
COMP	Comparator dataset to compare base against (data.frame)

---

```
identify_differences  identify_differences
```

---

**Description**

Compares each column within 2 datasets to identify any values which they mismatch on.

**Usage**

```
identify_differences(
  BASE,
  COMP,
  KEYS,
  exclude_cols,
  tolerance = sqrt(.Machine$double.eps),
  scale = NULL
)
```

**Arguments**

BASE	Base dataset for comparison (data.frame)
COMP	Comparator dataset to compare base against (data.frame)
KEYS	List of variables that define a unique row within the datasets (strings)
exclude_cols	Columns to exclude from comparison
tolerance	Level of tolerance for numeric differences between two variables
scale	Scale that tolerance should be set on. If NULL assume absolute

`identify_extra_cols`     *identify\_extra\_cols*

**Description**

Identifies columns that are in a baseline dataset but not in a comparator dataset

**Usage**

`identify_extra_cols(DS1, DS2)`

**Arguments**

DS1	Baseline dataset (data frame)
DS2	Comparator dataset (data frame)

`identify_extra_rows`     *identify\_extra\_rows*

**Description**

Identifies rows that are in a baseline dataset but not in a comparator dataset

**Usage**

`identify_extra_rows(DS1, DS2, KEYS)`

**Arguments**

DS1	Baseline dataset (data frame)
DS2	Comparator dataset (data frame)
KEYS	List of variables that define a unique row within the datasets (strings)

---

```
identify_matching_cols  
    identify_matching_cols
```

---

**Description**

Identifies columns with the same name in two data frames

**Usage**

```
identify_matching_cols(DS1, DS2, EXCLUDE = "")
```

**Arguments**

DS1	Input dataset 1 (data frame)
DS2	Input dataset 2 (data frame)
EXCLUDE	Columns to ignore

---

```
identify_mode_differences  
    identify_mode_differences
```

---

**Description**

Identifies any mode differences between two data frames

**Usage**

```
identify_mode_differences(BASE, COMP)
```

**Arguments**

BASE	Base dataset for comparison (data.frame)
COMP	Comparator dataset to compare base against (data.frame)

---

identify_properties	<i>identify_properties</i>
---------------------	----------------------------

---

**Description**

Returns a dataframe of metadata for a given dataset. Returned values include variable names , class , mode , type & attributes

**Usage**

```
identify_properties(dsin)
```

**Arguments**

dsin	input dataframe that you want to get the metadata from
------	--

---

identify_unsupported_cols	<i>identify_unsupported_cols</i>
---------------------------	----------------------------------

---

**Description**

Identifies any columns for which the package is not setup to handle

**Usage**

```
identify_unsupported_cols(dsin)
```

**Arguments**

dsin	input dataset
------	---------------

---

invert	<i>invert</i>
--------	---------------

---

**Description**

Utility function used to replicated purrr::transpose. Turns a list inside out.

**Usage**

```
invert(x)
```

**Arguments**

x	list
---	------

---

```
is_variable_different  is_variable_different
```

---

### Description

This subsets the data set on the variable name, picks out differences and returns a tibble of differences for the given variable

### Usage

```
is_variable_different(variablename, keynames, datain, ...)
```

### Arguments

variablename	name of variable being compared
keynames	name of keys
datain	Inputted dataset with base and compare vectors
...	Additional arguments which might be passed through (numerical accuracy)

### Value

A boolean vector which is T if target and current are different

---

```
print.diffdf          Print diffdf objects
```

---

### Description

Print nicely formatted version of an diffdf object

### Usage

```
## S3 method for class 'diffdf'
print(x, ..., as_string = FALSE)
```

### Arguments

x	comparison object created by diffdf().
...	Additional arguments (not used)
as_string	Return printed message as an R character vector?

**Examples**

```
x <- subset( iris , -Species )
x[1,2] <- 5
COMPARE <- diffd( iris, x)
print( COMPARE )
print( COMPARE , "Sepal.Length" )
```

---

recursive_reduce	<i>recursive_reduce</i> Utility function used to replicated purrr::reduce. Recursively applies a function to a list of elements until only 1 element remains
------------------	--

---

**Description**

recursive\_reduce

Utility function used to replicated purrr::reduce. Recursively applies a function to a list of elements until only 1 element remains

**Usage**

```
recursive_reduce(.l, .f)
```

**Arguments**

.l	list of values to apply a function to
.f	function to apply to each element of the list in turn i.e. .l[[1]] <- .f( .l[[1]] , .l[[2]] ) ; .l[[1]] <- .f( .l[[1]] , .l[[3]] )

---

sort_then_join	<i>sort_then_join</i>
----------------	-----------------------

---

**Description**

Convenience function to sort two strings and paste them together

**Usage**

```
sort_then_join(string1, string2)
```

**Arguments**

string1	first string
string2	second string

---

string_pad	<i>string_pad</i> Utility function used to replicate <i>str_pad</i> . Adds white space to either end of a string to get it to equal the desired length
------------	--

---

**Description**

string\_pad

Utility function used to replicate *str\_pad*. Adds white space to either end of a string to get it to equal the desired length

**Usage**

```
string_pad(x, width)
```

**Arguments**

x	string
width	desired length

# Index

as\_ascii\_table, 3  
as\_cropped\_char, 3

cast\_variables, 4  
class\_merge, 4  
compare\_vectors, 5  
compare\_vectors.default, 5  
compare\_vectors.factor, 6  
compare\_vectors.numeric, 6  
construct\_issue, 7  
convert\_to\_issue, 7

diffdf, 8  
diffdf\_has\_issues, 10  
diffdf\_issuerows, 10

factor\_to\_character, 11  
find\_difference, 12

generate\_keyname, 12  
get\_casted\_dataset, 13  
get\_casted\_vector, 13  
get\_issue\_dataset, 14  
get\_issue\_message, 14  
get\_print\_message, 15  
get\_print\_message.default, 15  
get\_print\_message.issue, 16  
get\_table, 16

has\_unique\_rows, 17

identify\_att\_differences, 17  
identify\_class\_differences, 18  
identify\_differences, 18  
identify\_extra\_cols, 19  
identify\_extra\_rows, 19  
identify\_matching\_cols, 20  
identify\_mode\_differences, 20  
identify\_properties, 21  
identify\_unsupported\_cols, 21  
invert, 21

is\_variable\_different, 22

print.diffdf, 22

recursive\_reduce, 23

sort\_then\_join, 23  
string\_pad, 24