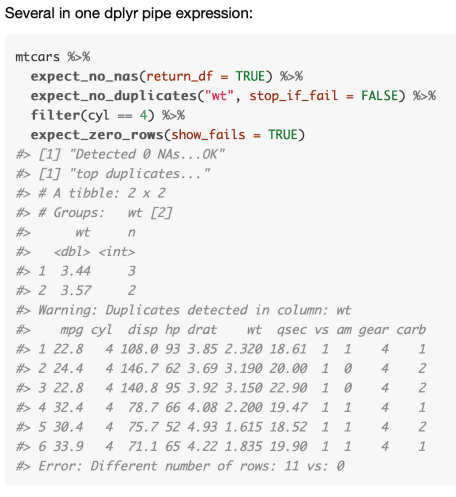
Expectdata is an R package that makes it easy to test assumptions about a data frame before conducting analyses. Below is a concise tour of some of the data assumptions expectdata can test for you. For example,



**Check for unexpected duplication**

library(expectdata)

expect\_no\_duplicates(mtcars, "cyl")

#> [1] "top duplicates..."

#> # A tibble: 3 x 2

#> # Groups: cyl [3]

#> cyl n

#>

#> 1 8 14

#> 2 4 11

#> 3 6 7

#> Error: Duplicates detected in column: cyl

The default return\_df == TRUE option allows for using these function as part of a dplyr piped expression that is stopped when data assumptions are not kept.

library(dplyr, warn.conflicts = FALSE)

library(ggplot2)

mtcars %>%

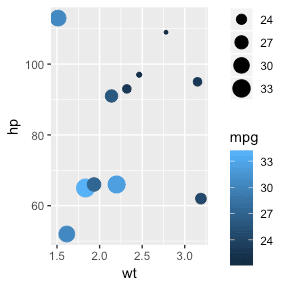
filter(cyl == 4) %>%

expect\_no\_duplicates("wt", return\_df = TRUE) %>%

ggplot(aes(x = wt, y = hp, color = mpg, size = mpg)) +

geom\_point()

#> [1] "no wt duplicates...OK"



If there are no expectations violated, an “OK” message is printed.

After joining two data sets you may want to verify that no unintended duplication occurred. Expectdata allows comparing pre- and post- processing to ensure they have the same number of rows before continuing.

expect\_same\_number\_of\_rows(mtcars, mtcars, return\_df = FALSE)

#> [1] "Same number of rows...OK"

expect\_same\_number\_of\_rows(mtcars, iris, show\_fails = FALSE, stop\_if\_fail = FALSE, return\_df = FALSE)

#> Warning: Different number of rows: 32 vs: 150

# can also compare to no df2 to check is zero rows

expect\_same\_number\_of\_rows(mtcars, show\_fails = FALSE, stop\_if\_fail = FALSE, return\_df = FALSE)

#> Warning: Different number of rows: 32 vs: 0

Can see how the stop\_if\_fail = FALSE option will turn failed expectations into warnings instead of errors.

**Check for existance of problematic rows**

Comparing a data frame to an empty, zero-length data frame can also be done more explicitly. If the expectations fail, cases can be shown to begin the next step of exploring why these showed up.

expect\_zero\_rows(mtcars[mtcars$cyl == 0, ], return\_df = TRUE)

#> [1] "No rows found as expected...OK"

#> [1] mpg cyl disp hp drat wt qsec vs am gear carb

#> <0 rows> (or 0-length row.names)

expect\_zero\_rows(mtcars$cyl[mtcars$cyl == 0])

#> [1] "No rows found as expected...OK"

#> numeric(0)

expect\_zero\_rows(mtcars, show\_fails = TRUE)

#> mpg cyl disp hp drat wt qsec vs am gear carb

#> Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4

#> Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4

#> Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1

#> Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1

#> Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2

#> Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

#> Error: Different number of rows: 32 vs: 0

This works well at the end of a pipeline that starts with a data frame, runs some logic to filter to cases that should not exist, then runs expect\_zero\_rows() to check no cases exist.

# verify no cars have zero cylindars

mtcars %>%

filter(cyl == 0) %>%

expect\_zero\_rows(return\_df = FALSE)

#> [1] "No rows found as expected...OK"

Can also check for NAs in a vector, specific columns of a data frame, or a whole data frame.

expect\_no\_nas(mtcars, "cyl", return\_df = FALSE)

#> [1] "Detected 0 NAs...OK"

expect\_no\_nas(mtcars, return\_df = FALSE)

#> [1] "Detected 0 NAs...OK"

expect\_no\_nas(c(0, 3, 4, 5))

#> [1] "Detected 0 NAs...OK"

#> [1] 0 3 4 5

expect\_no\_nas(c(0, 3, NA, 5))

#> Error: Detected 1 NAs

Several in one dplyr pipe expression:

mtcars %>%

expect\_no\_nas(return\_df = TRUE) %>%

expect\_no\_duplicates("wt", stop\_if\_fail = FALSE) %>%

filter(cyl == 4) %>%

expect\_zero\_rows(show\_fails = TRUE)

#> [1] "Detected 0 NAs...OK"

#> [1] "top duplicates..."

#> # A tibble: 2 x 2

#> # Groups: wt [2]

#> wt n

#>

#> 1 3.44 3

#> 2 3.57 2

#> Warning: Duplicates detected in column: wt

#> mpg cyl disp hp drat wt qsec vs am gear carb

#> 1 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1

#> 2 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2

#> 3 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2

#> 4 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1

#> 5 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2

#> 6 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1

#> Error: Different number of rows: 11 vs: 0