In this post, We’ll see 3 functions from tidyr that’s useful for handling Missing Values (NAs) in the dataset. Please note: This post isn’t going to be about Missing Value Imputation.

**tidyr**

The goal of tidyr is to help you create tidy data. Tidy data is data where:

+ Every column is variable.

+ Every row is an observation..

+ Every cell is a single value.

Let’s start with loading tidyr library. tidyr is also one of the packages present in tidyverse.

library(tidyr)

**tidyr functions**

Following are the 3 tidyr functions that are handy for processing Missing Values

* drop\_na()
* fill()
* replace\_na()

**Dataset with Missing Value**

To get a dataset with missing values, let’s take mtcars and make some missing values in it.

df <- mtcars

df$hp[2] <- NA

df$cyl[5] <- NA

df$gear[5] <- NA

df$mpg[10] <- NA

# counting number of missing values

paste("Number of Missing Values", sum([is.na](http://is.na)(df)))

## [1] "Number of Missing Values 4"

# dimensions

paste("Number of Rows",nrow(df))

## [1] "Number of Rows 32"

paste("Number of Columns",ncol(df))

## [1] "Number of Columns 11"

Now that we’ve got a dataset with Missing Values (NAs) in it.

**drop\_na()**

drop\_na() drops/removes the rows/entries with Missing Values

library(dplyr) #just in-case if we need to some dplyr verbs

## Warning: package 'dplyr' was built under R version 3.5.2

##

## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':

##

## filter, lag

## The following objects are masked from 'package:base':

##

## intersect, setdiff, setequal, union

df\_no\_na <- drop\_na(df)

# counting number of missing values

paste("Number of Missing Values", sum([is.na](http://is.na)(df\_no\_na)))

## [1] "Number of Missing Values 0"

# dimensions

paste("Number of Rows",nrow(df\_no\_na))

## [1] "Number of Rows 29"

paste("Number of Columns",ncol(df\_no\_na))

## [1] "Number of Columns 11"

**fill()**

fill() fills the NAs (missing values) in selected columns (dplyr::select() options could be used like in the below example with everything()).

It also lets us select the .direction either down (default) or up or updown or downup from where the missing value must be filled.

Quite Naive, but could be handy in a lot of instances like let’s say Time Series data.

df\_na\_filled <- df %>%

fill(

dplyr::everything()

)

# counting number of missing values

paste("Number of Missing Values", sum([is.na](http://is.na)(df\_na\_filled)))

## [1] "Number of Missing Values 0"

# dimensions

paste("Number of Rows",nrow(df\_na\_filled))

## [1] "Number of Rows 32"

paste("Number of Columns",ncol(df\_na\_filled))

## [1] "Number of Columns 11"

**replace\_na()**

replace\_na() is to be used when you have got the replacement value which the NAs should be filled with.

Below is an example of how we have replaced all NAs with just zero (0)

df\_na\_replaced <- df %>%

mutate\_all(replace\_na,0)

# counting number of missing values

paste("Number of Missing Values", sum([is.na](http://is.na)(df\_na\_replaced)))

## [1] "Number of Missing Values 0"

# dimensions

paste("Number of Rows",nrow(df\_na\_replaced))

## [1] "Number of Rows 32"

paste("Number of Columns",ncol(df\_na\_replaced))

## [1] "Number of Columns 11"

Alternatively, We could’ve simply identified numeric / continous values and replaced their values with NAs like this:

df\_na\_replaced <- df %>%

mutate\_if(is.numeric, replace\_na,0)

Hopefully, this post would have thrown some light on those three functions of tidyr to handle missing values: drop\_na(), fill(), replace\_na().