Four Page dplyr

Andy Grogan-Kaylor

2018-10-16

Contents

1 Background

`dplyr` is a very powerful R library for managing and processing data.

While `dplyr` is very powerful, learning to use `dplyr` can be very confusing. This guide aims to present some of the most common `dplyr` functions and commands in the form of a brief cheatsheet.

```
library(dplyr)
```

2 Sample Data

year	x	y	z
2005	NA	Group B	116.4
2016	32.12	Group A	94.42
2006	37.7	Group A	94.07

year	x	y	z
2006	29.04	Group A	78.47
2003	31.99	Group A	87.77

3 Piping

Pipes %>% connect pieces of a command e.g. *data* to *data wrangling* to a *graph command*.

4 Select A Subset of Variables: `select()`

```
mynewdata <- mydata %>% select(x,y) # select only x and y
```

x	y
NA	Group B
32.12	Group A
37.7	Group A
29.04	Group A
31.99	Group A

5 Filter A Subset of Rows: `filter()`

```
mynewdata <- mydata %>%
  filter(year > 2010) # filter on year
```

year	x	y	z
2016	32.12	Group A	94.42

6 Create New Variables: `mutate()`

```
mynewdata <- mydata %>%
  mutate(myscale = x + z) # create a new variable e.g. a scale
```

year	x	y	z	myscale
2005	NA	Group B	116.4	NA
2016	32.12	Group A	94.42	126.5
2006	37.7	Group A	94.07	131.8

year	x	y	z	myscale
2006	29.04	Group A	78.47	107.5
2003	31.99	Group A	87.77	119.8

7 Recode Variables: `mutate()`

```
mynewdata <- mydata %>%
  mutate(zcategorical = cut(z, # use mutate to recode
                            breaks=c(-Inf, 100, Inf),
                 labels = c("low", "high")))
```

year	x	y	z	zcategorical
2005	NA	Group B	116.4	high
2016	32.12	Group A	94.42	low
2006	37.7	Group A	94.07	low
2006	29.04	Group A	78.47	low
2003	31.99	Group A	87.77	low

8 Rename Variables: `rename()`

```
newdata <- mydata %>%
  rename(age = x, # rename
         mental_health = z)
```

year	age	y	mental_health
2005	NA	Group B	116.4
2016	32.12	Group A	94.42
2006	37.7	Group A	94.07
2006	29.04	Group A	78.47
2003	31.99	Group A	87.77

9 Drop Missing Values: `filter()`

```
newdata <- mydata %>%
  filter(!is.na(x)) # filter by x is not missing
```

year	x	y	z
2016	32.12	Group A	94.42
2006	37.7	Group A	94.07
2006	29.04	Group A	78.47
2003	31.99	Group A	87.77

10 Connecting To Other Packages Like `ggplot`

Notice how, in the code below, I never actually create the new data set `mynewdata`. I simply pipe `mydata` into a `dplyr` command, and pipe the result directly to `ggplot2`.

```
library(ggplot2)

mydata %>% # my data
  mutate(myscale = x + z) %>% # dplyr command
  ggplot(aes(x = year, # the rest is ggplot
             y = myscale)) +
  geom_point() +
  geom_smooth(se = FALSE) + # smoother without confidence interval
  labs(title = "My Scale By Year") +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
  theme(axis.text.x = element_text(angle = 90))
```