

Five Page dplyr

Andy Grogan-Kaylor

2019-05-09

Contents

1	Background	1
2	Simulated Data	1
3	Piping	2
4	Aggregate Data: <code>group_by()</code> & <code>summarise()</code>	2
5	Select A Subset of Variables: <code>select()</code>	2
6	Filter A Subset of Rows: <code>filter()</code>	2
7	Create New Variables: <code>mutate()</code>	3
8	Recode Variables: <code>mutate()</code>	3
8.1	Continuous Into Categorical: <code>mutate()</code> & <code>cut()</code>	3
8.2	Categorical Into Categorical: <code>mutate()</code> & <code>recode()</code>	3
9	Rename Variables: <code>rename()</code>	4
10	Drop Missing Values: <code>filter()</code>	4
11	Random Sample	4
12	Connecting To Other Packages Like <code>ggplot</code>	5

1 Background

`dplyr` is a very powerful R library for managing and processing data.¹

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.

¹ The origins of the name `dplyr` seem somewhat obscure, but I sometimes think of this package as the *data plyers*.

```
library(dplyr)
```

2 Simulated Data

year	x	y	z
2015	NA	Group A	104.1
2015	48.32	Group C	103.4
2015	36.89	Group A	115

year	x	y	z
2016	23.89	Group B	102.5
2016	40.86	Group A	96.26

3 Piping

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

4 Aggregate Data: `group_by()` & `summarise()`

```
mynewdata <- mydata %>%
  group_by(year) %>% # group by y
  summarise(mean_x = mean(x), # mean of x
            n = n()) # count up
```

year	mean_x	n
2015	NA	3
2016	32.38	2

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

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

x	y
NA	Group A
48.32	Group C
36.89	Group A
23.89	Group B
40.86	Group A

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

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

year	x	y	z
2015	NA	Group A	104.1
2015	48.32	Group C	103.4

year	x	y	z
2015	36.89	Group A	115
2016	23.89	Group B	102.5
2016	40.86	Group A	96.26

7 Create New Variables: `mutate()`

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

year	x	y	z	myscale
2015	NA	Group A	104.1	NA
2015	48.32	Group C	103.4	151.7
2015	36.89	Group A	115	151.9
2016	23.89	Group B	102.5	126.4
2016	40.86	Group A	96.26	137.1

8 Recode Variables: `mutate()`

8.1 Continuous Into Categorical: mutate() & cut()

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

year	x	y	z	zcategorical
2015	NA	Group A	104.1	high
2015	48.32	Group C	103.4	high
2015	36.89	Group A	115	high
2016	23.89	Group B	102.5	high
2016	40.86	Group A	96.26	low

8.2 Categorical Into Categorical: mutate() & recode()

```
mynewdata <- mydata %>%
  mutate(yrecoded = dplyr::recode(y, # recode values
    "Group A" = "Red Group",
    "Group B" = "Blue Group",
    .default = "Other"))
```

year	x	y	z	yrecoded
2015	NA	Group A	104.1	Red Group
2015	48.32	Group C	103.4	Other
2015	36.89	Group A	115	Red Group
2016	23.89	Group B	102.5	Blue Group
2016	40.86	Group A	96.26	Red Group

9 Rename Variables: `rename()`

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

year	age	y	mental_health
2015	NA	Group A	104.1
2015	48.32	Group C	103.4
2015	36.89	Group A	115
2016	23.89	Group B	102.5
2016	40.86	Group A	96.26

10 Drop Missing Values: `filter()`

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

year	x	y	z
2015	48.32	Group C	103.4
2015	36.89	Group A	115
2016	23.89	Group B	102.5
2016	40.86	Group A	96.26

11 Random Sample

```
newdata <- mydata %>% sample_frac(0.5) # fraction of data to sample
```

year	x	y	z
2015	36.89	Group A	115
2016	40.86	Group A	96.26

12 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 to make new variable
  ggplot(aes(x = year, # the rest is ggplot
             y = myscale)) +
  geom_point() + # points
  geom_smooth(se = FALSE) + # smoother without confidence interval
  labs(title = "My Scale By Year") + # labels
  theme(axis.text.x = element_text(size = 10, # tweak theme
                                     angle = 90))
```

