Homework 03: Use dplyr to manipulate and explore data (also use ggplot2)

Cody

September 26, 2017

```
library("tidyverse")
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
library("gapminder")
```

Overview

Consult the general homework guidelines. You'll be submitting this homework to the repo you submitted Homework 02 too.

Due Tuesday 2017-10-03 at midnight.

The goal is to manipulate and explore a dataset with the `dplyr` package, complemented by visualizations made with `ggplot2`.

Your homework should serve as your own personal cheatsheet in the future for ways to manipulate a dataset and produce companion figures. Give yourself the cheatsheet you deserve! Check out the sampler concept for inspiration.

Gapminder data

Work with the Gapminder excerpt. *If you really want to, you can explore a different dataset but get permission from Jenny. Self-assess the suitability of your dataset by reading this issue.*

Your mission, high-level

Pick at least three of the tasks below and attack each with a table and figure.

- dplyr should be your data manipulation tool
- ggplot2 should be your visualization tool

Make observations about what your tables/figures show and about the process.

If you want to do something comparable but different, i.e. swap one quantitative variable for another, be my guest! If you are feeling inspired and curious, then we're doing this right. Go for it.

Relax about the following things:

- Tidying/reshaping is NOT your assignment. Many of your tables will be awkwardly shaped in the report. That's OK.
- Table beauty is not a big deal. Simply printing to "screen" is fine. You could also try the `knitr::kable()` function. Assuming `my_df` is a data.frame, here's an R chunk that should print it as a decent-looking table:

"'{r results = 'asis'} knitr::kable(my_df)

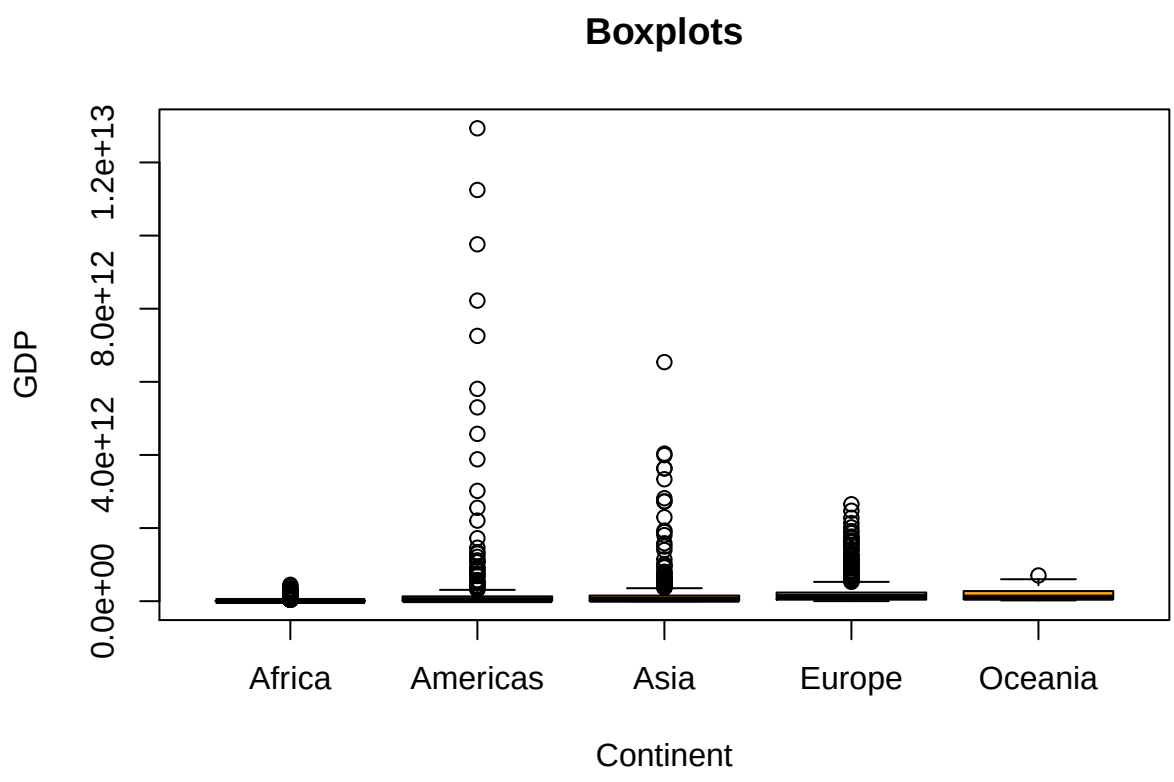
- For all things, graphical and tabular, if you're dissatisfied with a result, discuss the problem, what you've tried and move on.

Task menu

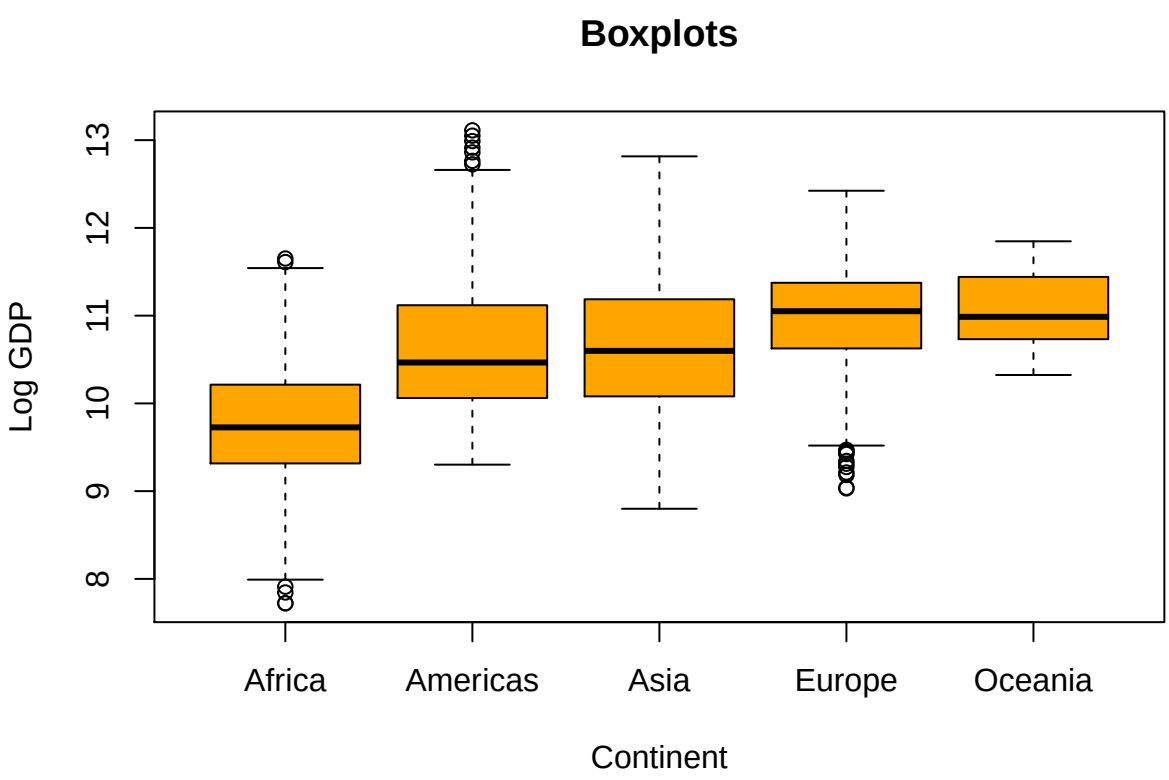
Get the maximum and minimum of GDP per capita for all continents.

```
newdata <- gapminder %>%
          mutate(GDP=gdpPercap*pop,logGDP=log10(gdpPercap*pop))

boxplot(GDP~continent,data=newdata,col="orange",
        xlab="Continent",ylab="GDP",main="Boxplots")
```




```
boxplot(logGDP~continent,data=newdata,col="orange",
        xlab="Continent",ylab="Log GDP",main="Boxplots")
```




```
newdata %>%
  group_by(continent) %>%
  summarise_each(funs(min,max),GDP,logGDP)
```

```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over a selection of variables, use `summarise_at()`
```

```
## # A tibble: 5 x 5
##   continent     GDP_min logGDP_min      GDP_max logGDP_max
##      <fctr>       <dbl>      <dbl>        <dbl>      <dbl>
## 1    Africa    52784691   7.722508 4.479709e+11   11.65125
## 2  Americas  2003975797   9.301892 1.293446e+13   13.11175
## 3      Asia   629774980   8.799185 6.539501e+12   12.81554
## 4    Europe  1075341715   9.031546 2.650871e+12   12.42339
## 5   Oceania 21058193787  10.323421 7.036584e+11   11.84736
```

I have plotted both the regular boxplots as well as the logarithmic boxplots. In the regular case, we can hardly see any kind of distribution as the values are too large to be compared in a reasonable window. Instead, considering the logged plots, there is a bit more of a sense of distribution and even a bit of normality. But the table then produces the max and min in both the regular and logged case. Note: The strange warning message that says summarise_each is deprecated. Not quite sure what this means but things seem to be functioning. I used summarise_each because I wanted to map over several functions and the other commands didn't seem to do what I was looking for.

Look at the spread of GDP per capita within the continents.

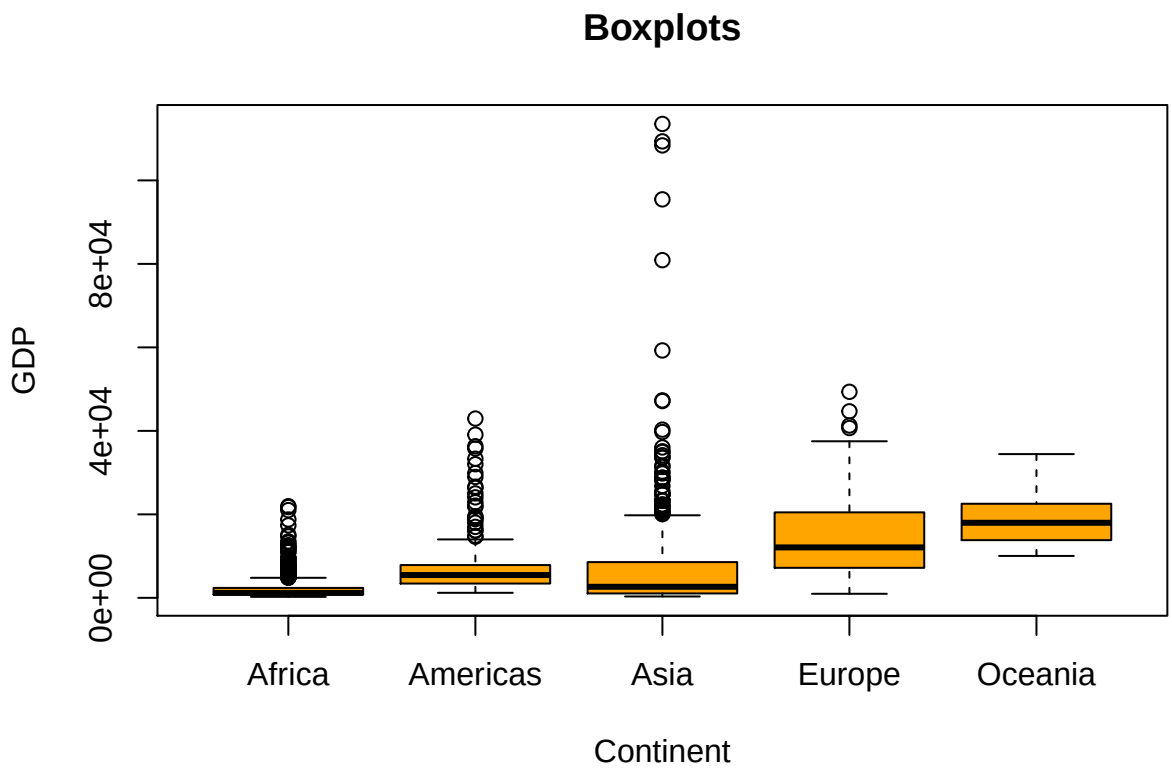
```
newdata <- gapminder %>%
             mutate(loggdpPercap=log10(gdpPercap))

newdata %>%
  group_by(continent) %>%
  summarise_each(funs(IQR,sd),gdpPercap,loggdpPercap)
```

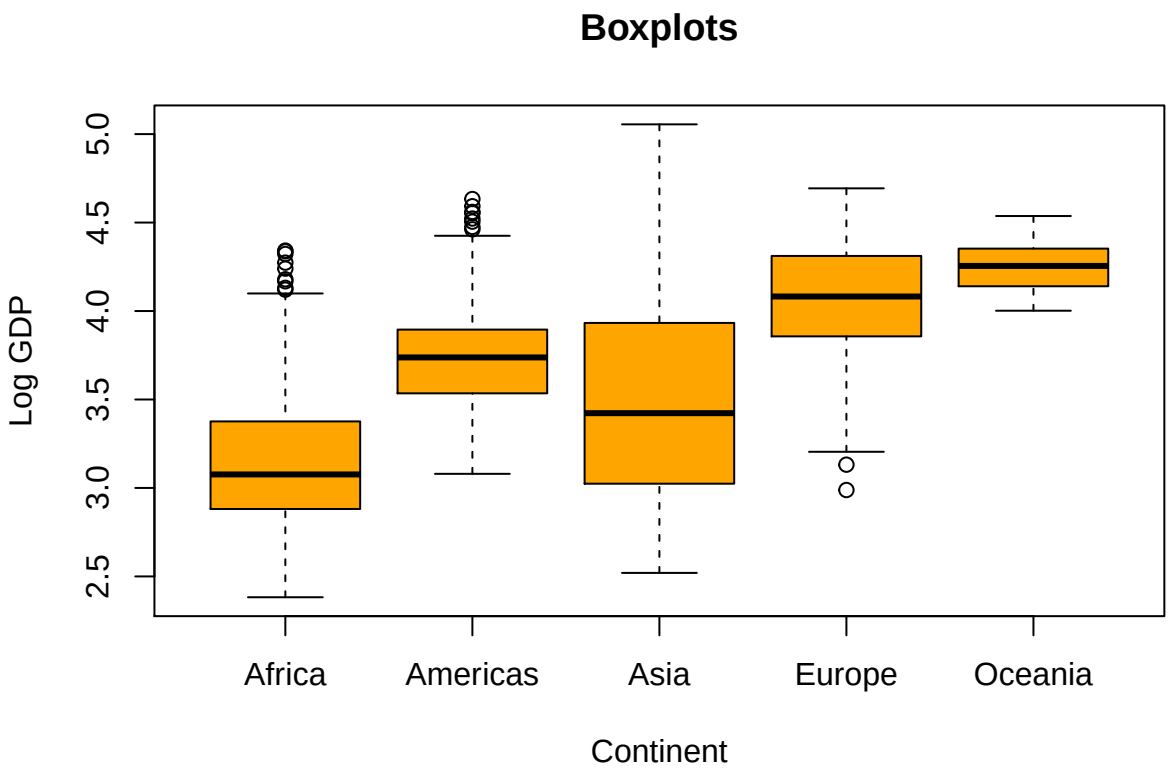
```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over a selection of variables, use `summarise_at()`
```

```
## # A tibble: 5 x 5
##   continent gdpPercap_IQR loggdpPercap_IQR gdpPercap_sd loggdpPercap_sd
##      <fctr>         <dbl>            <dbl>        <dbl>           <dbl>
## 1    Africa      1616.170        0.4945803     2827.930       0.3733077
## 2  Americas      4402.431        0.3587552     6396.764       0.2965612
## 3      Asia      7492.262        0.9078538    14045.373       0.5680604
## 4    Europe     13248.301        0.4528202     9355.213       0.3188679
## 5   Oceania      8072.258        0.1963343     6358.983       0.1451969
```

```
boxplot(gdpPercap~continent,data=newdata,col="orange",
        xlab="Continent",ylab="GDP",main="Boxplots")
```



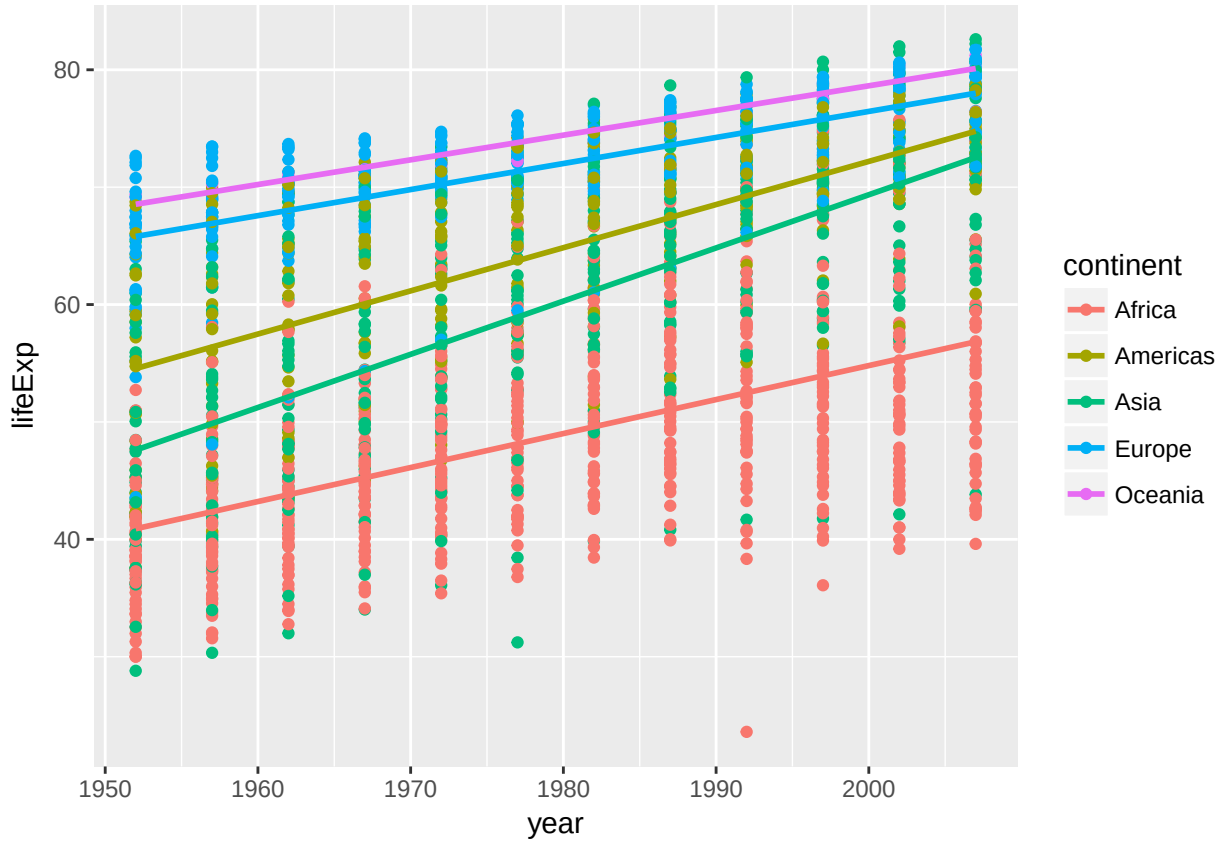

```
boxplot(loggdpPercap~continent,data=newdata,col="orange",
        xlab="Continent",ylab="Log GDP",main="Boxplots")
```



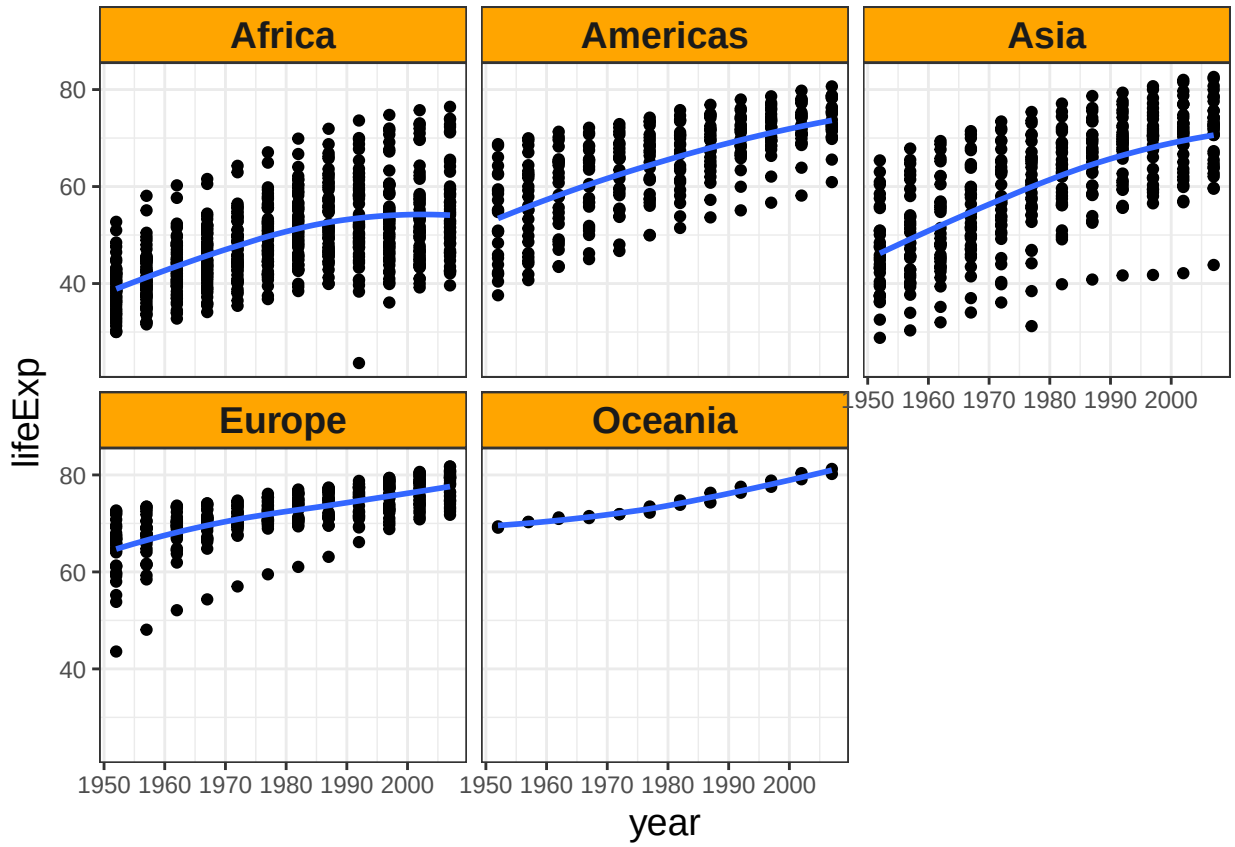
Once again, I have used both a regular and logged version to see a bit more of a distribution. I have also reported two values to indicate the spread of the data, both the IQR and the standard deviation. Certain continents (like Africa or the Americas) seem to have a few outliers as well as the gdpPercap values are quite large and this will have impact on the standard deviation. So instead, reporting the IQR will serve as perhaps a better measure of spread. Note: Same warning message. Again, the code and plots seem to function how I want them to. This will be a good thing to ask about.

How is life expectancy changing over time on different continents?

```
ggplot(gapminder, aes(year, lifeExp,
                      colour=continent)) +
    geom_point() +
    geom_smooth(method="lm", se=FALSE)
```




```
ggplot(gapminder, aes(year, lifeExp)) +
    facet_wrap(~ continent) +
    geom_point() +
    geom_smooth(se=FALSE, span=1, method="loess")+
    theme_bw()+
    theme(strip.background = element_rect(fill="orange"),
          axis.title = element_text(size=14),
          strip.text = element_text(size=14, face="bold"))
```



Here I have given two seperate plots that tell similar stories yet have a slight different conclusion. * The first is a scatterplot by year where the colors per continent. Over this we have a simple linear regression analysis that essentially predicts all continents to have a higher life expectancy the more recent the year. But we can see that Asia has the steepest curve and perhaps Oceania and Africa tied with one another for slowest increase. * In the second plot, there is a separate analysis done per continent and we again see a general increase in life expectancy per year but we have a smoother added ontop of the points and here we do not see the trend to continue. For example, it seems that Asia is slowing down in its life expectancy where Africa is all but plateauing in it's trend. These are more or less the same plots but separated, but the different models on top of them do have slightly different things to say.

Report your process

I used a function "summarise_each" to map over functions to create multiple columns that report different statistics. But when I use this command, I seem to be getting a strange warning message. If time wasn't a pressure I would sit down and figure out what "deprecated" really means here. But research calls :'(.