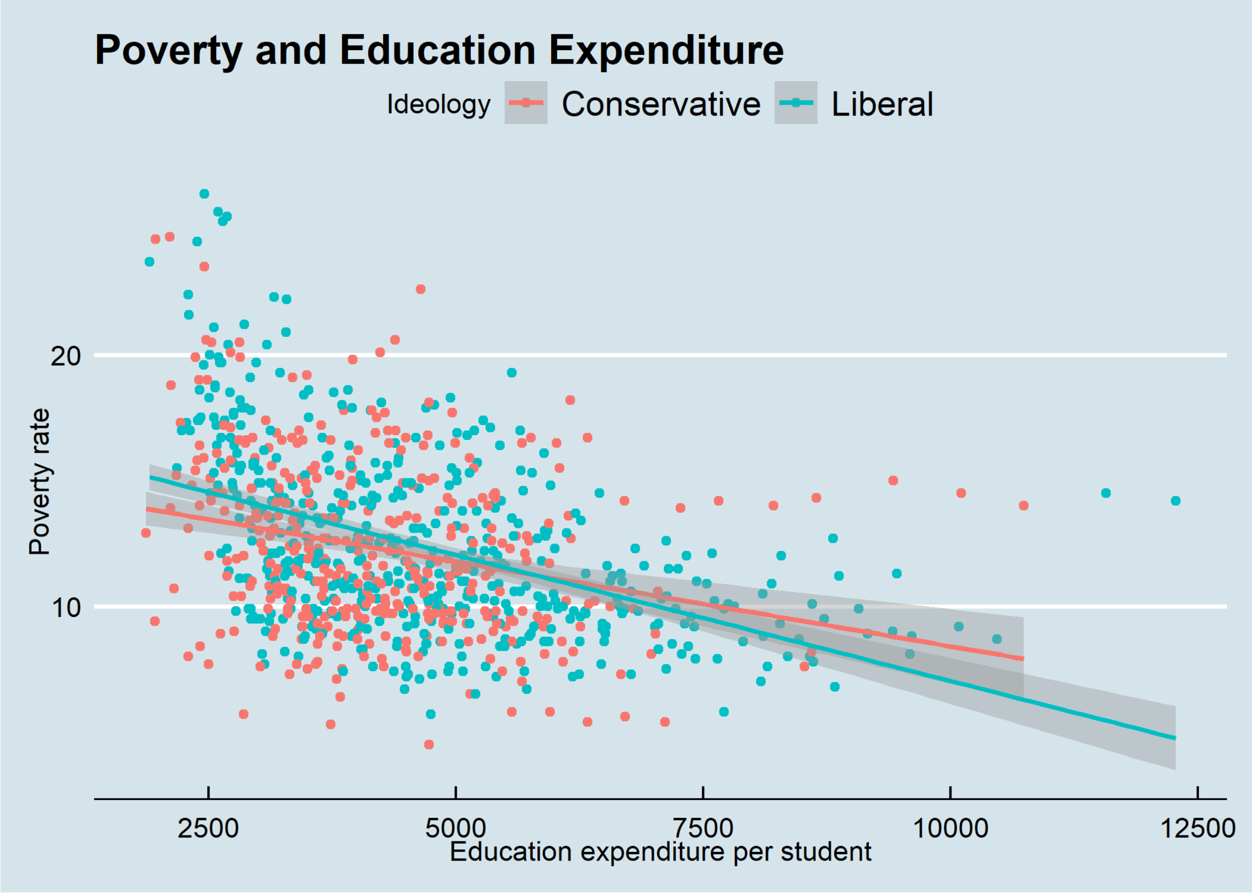
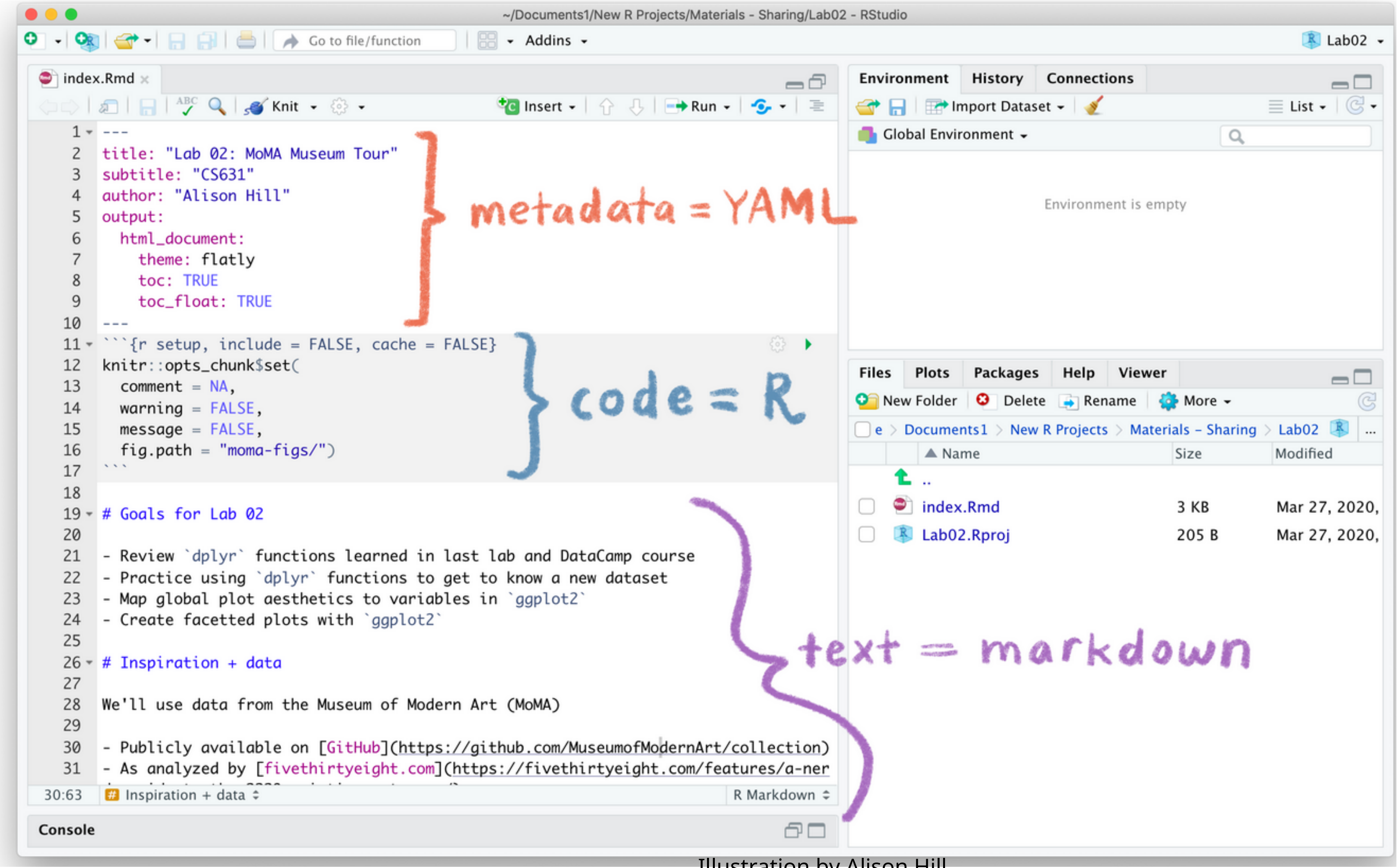
R Workshop

Pilar Manzi and Max Weylandt

10/21/2020

Where we're going



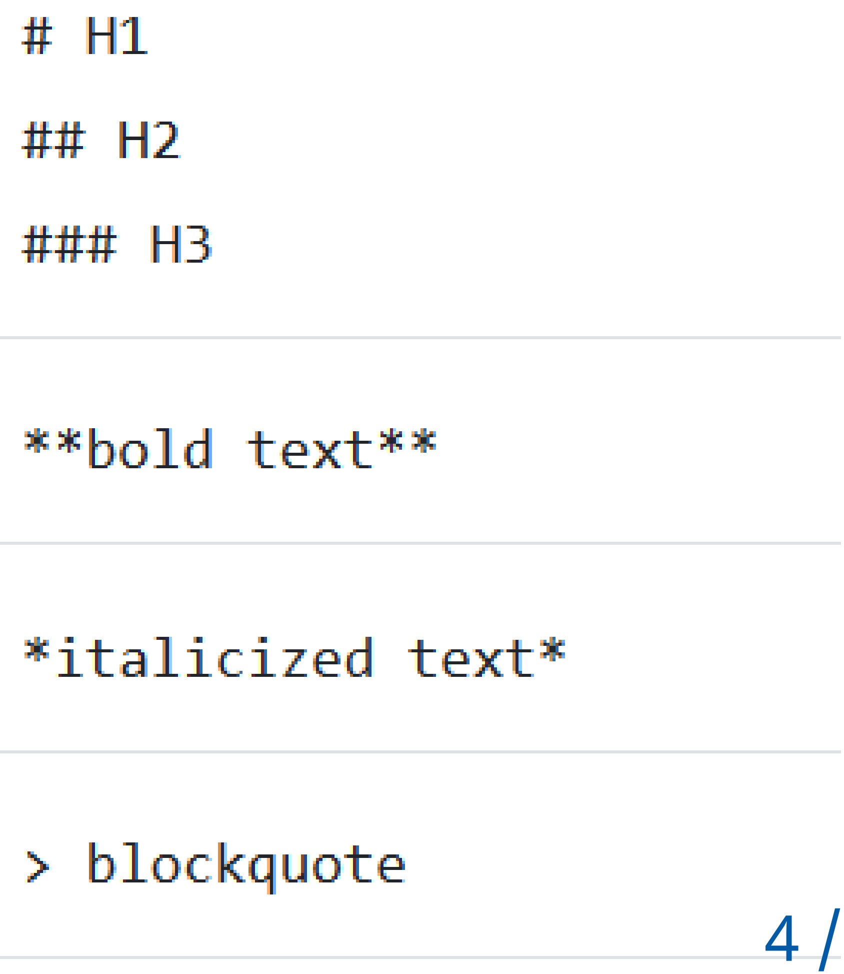


~~~
---
title: "Lab 02: MoMA Museum Tour"
subtitle: "CS631"
author: "Alison Hill"
output:
  html_document:
    theme: flatly
    toc: TRUE
    toc_float: TRUE
---
```{r setup, include = FALSE, cache = FALSE}
knitr::opts_chunk$set(
  comment = NA,
  warning = FALSE,
  message = FALSE,
  fig.path = "moma-figs/")
```


# Goals for Lab 02

- Review `dplyr` functions learned in last lab and DataCamp course
- Practice using `dplyr` functions to get to know a new dataset
- Map global plot aesthetics to variables in `ggplot2`
- Create facetted plots with `ggplot2`

# Inspiration + data

We'll use data from the Museum of Modern Art (MoMA)

- Publicly available on [GitHub](https://github.com/MuseumofModernArt/collection)
- As analyzed by [fivethirtyeight.com](https://fivethirtyeight.com/features/a-ner
~~~

Illustration by Alison Hill

R Markdown vs. regular R Script

- Combine text and code into attractive output of various formats



```
# H1

## H2

### H3

**bold text**

*italicized text*

> blockquote
```

Look out:

- All code runs every time you "knit" - so beware!
- Code needs to be contained within the grey "chunks"
- Chunks can be set to be hidden in output, or to show but not execute

Recap

The Grammar of R:

- Verb(Nouns, Adjectives)
  - `function(object, other arguments)`

Accessing and manipulating Data in R

Reading Data into R

```
library(readr)
states_data <- read_csv("correlates_state.csv")
```

- Do not use the Graphical User Interface
- It runs the code in the console
- This means the data won't import next time

states_data <- read_csv("correlates_state.csv")

Function depends on data type:

- .csv - `read_csv("filename.csv")` (this requires `library(readr)`)
- .xlsx - `read_excel("filename.xlsx")` (this requires `library(readxl)`)
- .RDS - `readRDS("filename.RDS")`
- .dta - `read_dta("filename.dta")` (this requires `library(haven)`)

states_data <- read_csv("correlates_state.csv")

- R uses relative filepaths
- In an .RMD file, the working directory is where your .rmd file is located
- if the dataset is in the same directory, just give its name
- if it's in a subdirectory (say `data`) write it `read_csv("data/file.csv")`

Getting a first look at the data:

```r
# Info on no. observations, column names,
# variable types, and some values
str(states_data)

# Shows first 5 rows of the dataset
head(states_data)
```

Dplyr for Data Manipulation

1) select

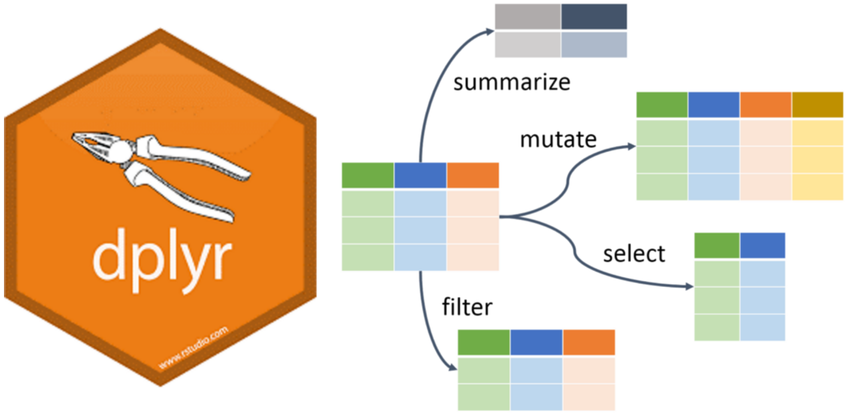
2) filter

3) mutate

4) summariz(s)e

5) group_by

6) rename



Dplyr: Pipes %>%

- Funnel the output of any function forward into the next function
- Easier to read than regular R code with its nested functions

e.g.`select()` function syntax: `select(datasetname, column1, column2, col3:col7)`

With Pipe: `datasetName %>% select(column1, column2, col3:col7)`

Select

- pull out specified columns
- syntax: `datasetName %>% select(column1, column2, col3:col7)`

```
data_selected <- states_data %>%
  select(1:6, year, region, ideo,inst6014_nom,
         povrate, edinstruct_expend_pstud)
```

Select

We can select by:

- name (`select(country, year)`)
- position of the column(`select(c(1,3,6))`)
- range: (`select(country:year)` or `select(1:3)`).

See markdown for 'helper functions'

Filter

Instead of picking out whole columns (i.e. variables), we can pick out **observations** that comply with given condition(s).

```r
data_filtered <- states_data %>%
  filter(year<2010)
```

Filter and R's Logical Operators

Operator	Description
<	less than
<=	less than or equal to
==	exactly equal to
!=	not equal to
!x	Not x
x & y	x AND y
x |y	x OR y
isTRUE(x)	test if X is TRUE

Filter

We can combine as many conditions as we want:

```
data_filtered <- states_data %>%
  filter((year > 2010 & povrate<4) |
        (year >= 2010 & state == "IL"))
```

If we want to exclude DC from our data:

```
data_filtered <- states_data %>%
  filter(state != "District of Columbia")
```

Pipes

We can funnel the output on one function to another without creating a new object at each step:

```
states_final <- states_data %>%
  select(1:6, region, ideo,inst6014_nom,
         povrate, edinstruct_expend_pstud) %>%
  filter(state != "District of Columbia")
```

(Note the keyboard shortcut: Ctrl+Shift+M)

Mutate

- Create new variables based on the transformation of an existing one.
    - e.g. Create the log of educational spending (`edinstruct_expend_pstud`)
- syntax: `dataset %>% mutate(newVarName = contentOfVariable)`

```
states_final <- states_final %>%
  mutate(log_expend = log(edinstruct_expend_pstud))
```

Mutate

`Mutate` can also be used to recode variables together with `case_when()` Based on the Nominate Score of state ideology (continuous, from 0 to 100), we can create a dummy variable that groups states into Liberal and Conservative:

```r
states_final <- states_final %>%
  mutate(RedBlue = case_when(
    inst6014_nom >= 49 ~ "Liberal",
    inst6014_nom < 49 ~ "Conservative")
    )
```

Summariz(s)e

- reduces observations to a single value based on certain functions:
  - mean, standard deviation, minimun, maximum, etc.

```
states_final %>%
  summarise(mean_ideo =
              mean(edinstruct_expend_pstud,
                   na.rm = TRUE))
```

```
## # A tibble: 1 x 1
##   mean_ideo
##       <dbl>
## 1     4473.
```

Summarize

We can add as many descriptive statistics as we need:

```
states_final %>%
  summarise(mean_expend =
              mean(edinstruct_expend_pstud, na.rm = TRUE),
            sd_expend =
              sd(edinstruct_expend_pstud,na.rm = TRUE),
            maz_expend =
              max(edinstruct_expend_pstud, na.rm = TRUE))
```

```
## # A tibble: 1 x 3
##   mean_expend sd_expend maz_expend
##         <dbl>     <dbl>      <dbl>
## 1       4473.     1552.      12276
```

Group_by and Summarize

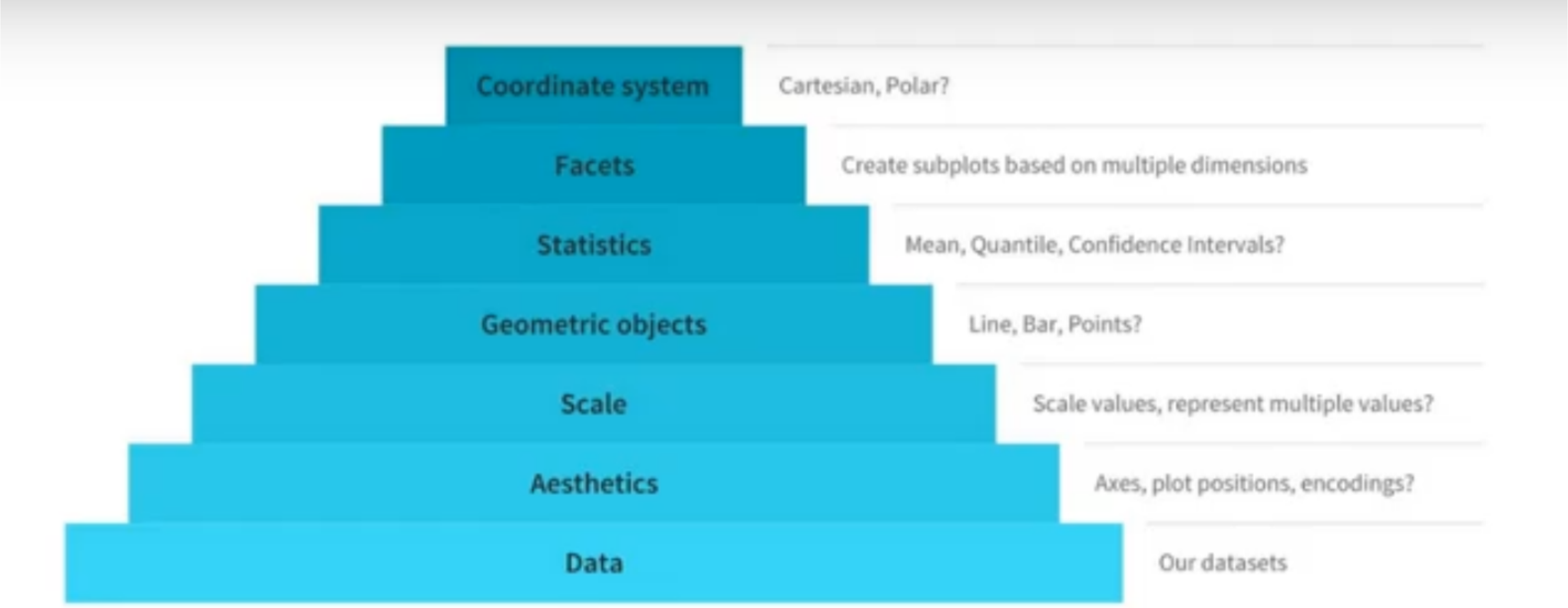
- Analyze data across groups (regions, gender, age group, political party, etc.).

```
states_final %>% group_by(RedBlue) %>%
  summarize(mean_expend =
                  mean(edinstruct_expend_pstud, na.rm = TRUE),
                mean_pov =
                  mean(povrate, na.rm=TRUE))
```

```
## # A tibble: 2 x 3
##   RedBlue      mean_expend mean_pov
##   <chr>              <dbl>    <dbl>
## 1 Conservative       4218.     13.0
## 2 Liberal            4669.     12.5
```

ggplot: The Grammar of Graphics

The Grammar of Graphics


Coordinate system
Cartesian, Polar?
Facets
Create subplots based on multiple dimensions
Statistics
Mean, Quantile, Confidence Intervals?
Geometric objects
Line, Bar, Points?
Scale
Scale values, represent multiple values?
Aesthetics
Axes, plot positions, encodings?
Data
Our datasets

Source: Nick Huntington-Klein

The Basic Structure

```
ggplot(name of dataset), aes (x= name
variable x, y= name variable y, ...) +
geom_something()
```

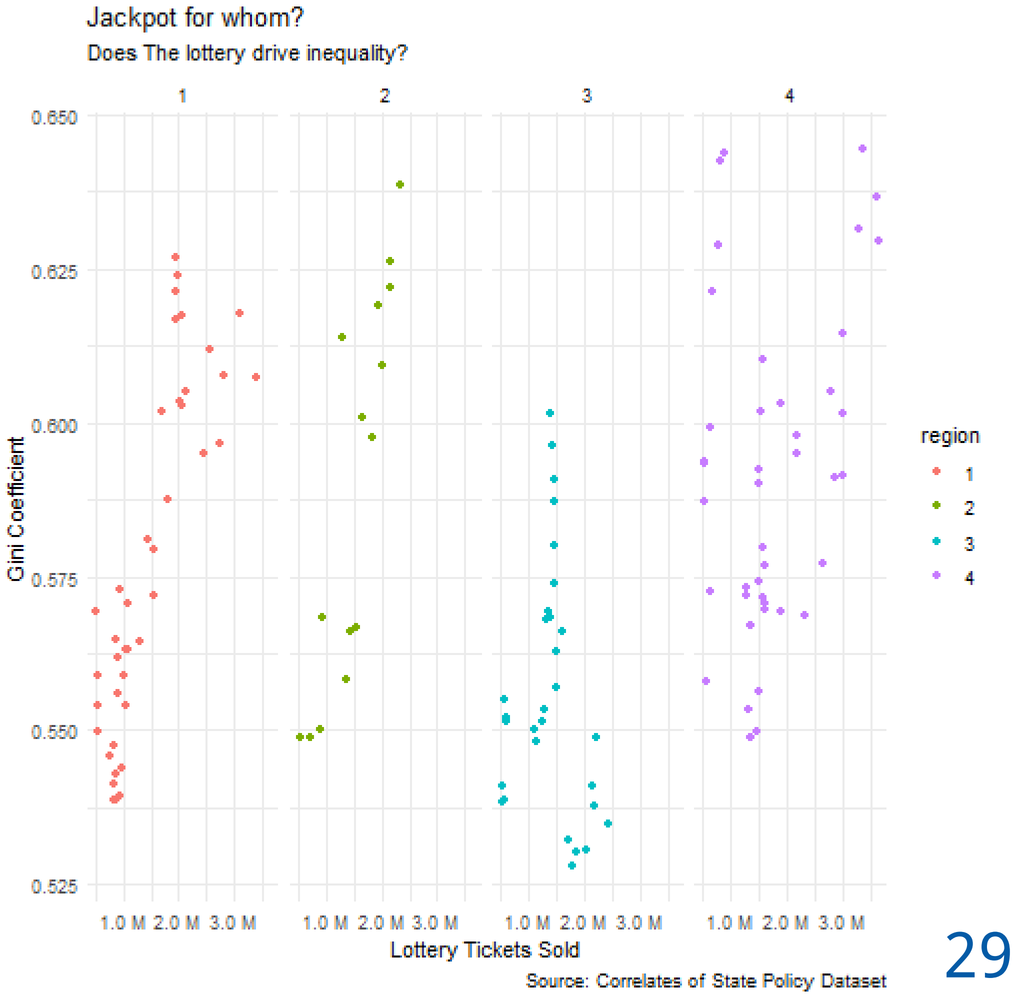
- aes(): links *what* you want to plot (variables) to what you want to *see* (color, fill, shape, linetype, and size). aes stands for 'aesthetics'.

The Basic Structure

```
ggplot(name of dataset), aes (x= name
variable x, y= name variable y, ...) +
geom_something()
```

- geom_something: *how* you want to plot it. Geometric objects are the actual marks we put on a plot. You need at least one but can have as many as you want! Common examples: geom_point, geom_line, geom_boxplot

```r
ggplot(states_data)+
  aes(x = lotticksales)+
  aes(y = gini_coef)+
  scale_x_continuous(label
  geom_point()+
  geom_point(aes(color=re
  labs(title="Jackpot for
  labs(subtitle = "Does Th
  labs(caption="Source: Co
  xlab("Lottery Tickets So
  ylab("Gini Coefficient")
  theme_minimal()+
  facet_grid(~region)
#scale_y_continuous(label
```


Jackpot for whom?

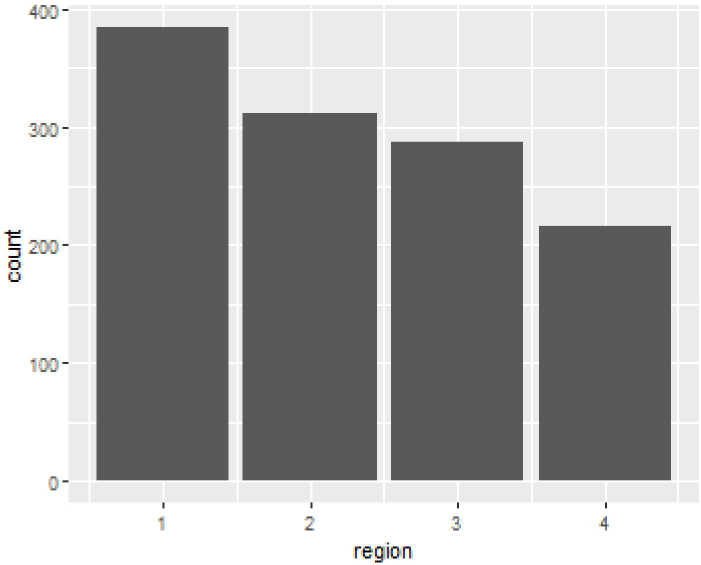
Does The lottery drive inequality?

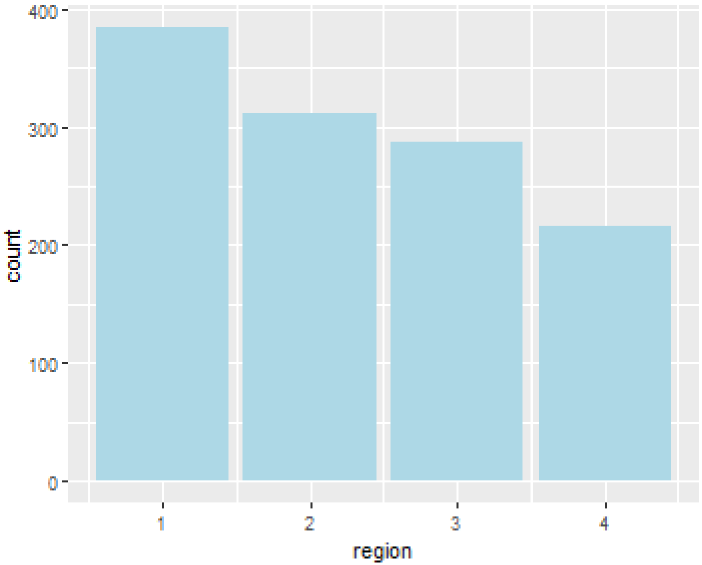
Source: Correlates of State Policy Dataset

Categorical variables

```
ggplot(data = states_final)+
  aes(x= region)+
  geom_bar()
```

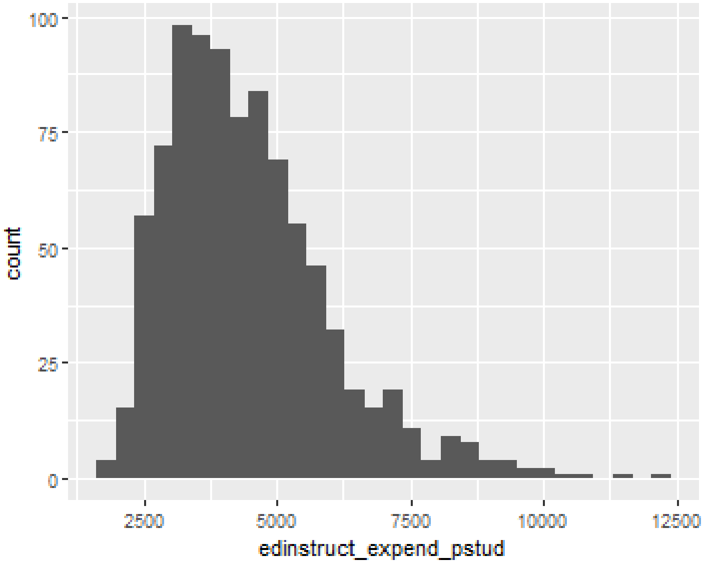
```
ggplot(data = states_final)+
  aes(x= region)+
  geom_bar(fill="light blue")
```



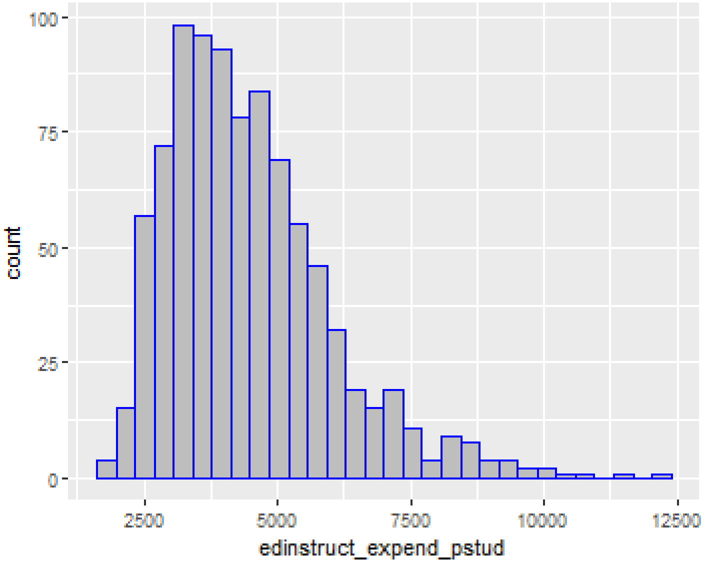


Continuous variable

```
ggplot(data = states_final)+
  aes(x= edinstruct_expend_pstu
  geom_histogram()
```



```
ggplot(data = states_final)+
  aes(x= edinstruct_expend_pstu
  geom_histogram(fill="grey",
                 color="blue")
```



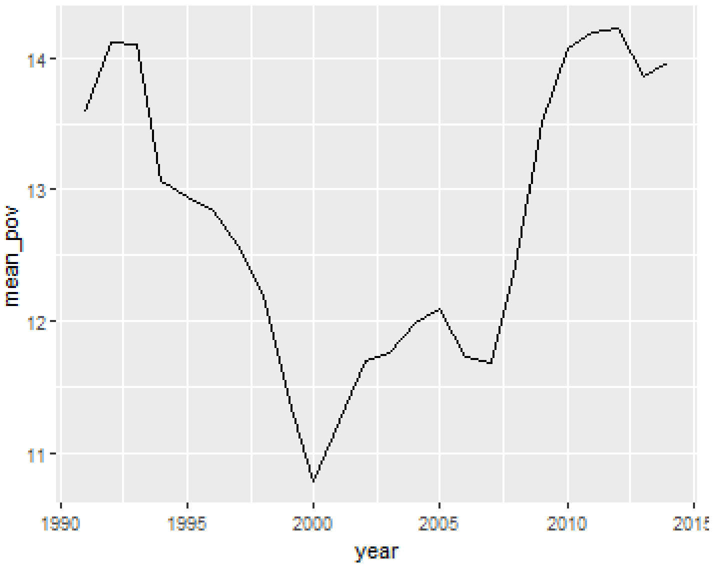
Categorical and continuous

See Markdown for boxplots

Time series

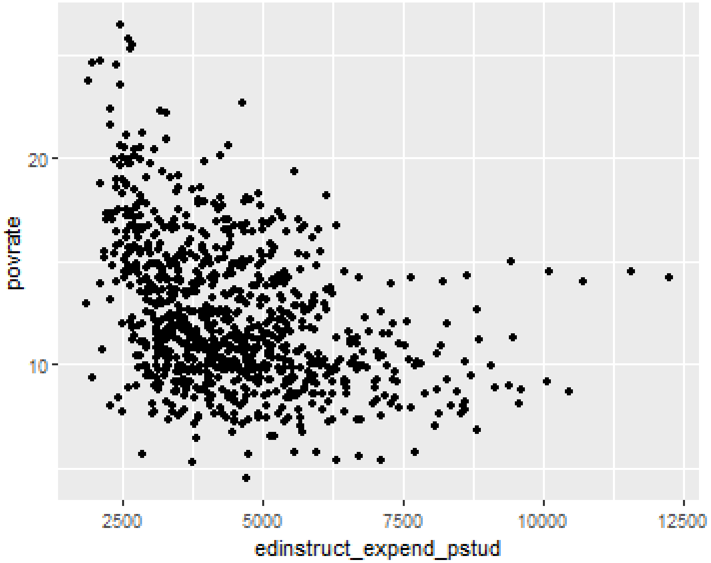
When we want to plot change over time, we use line charts.

```
states_final %>%
  group_by(year) %>%
  summarize(mean_pov =
                mean(povrate)) %>
  ggplot(aes(x= year,
               y=mean_pov)) +
  geom_line()
```

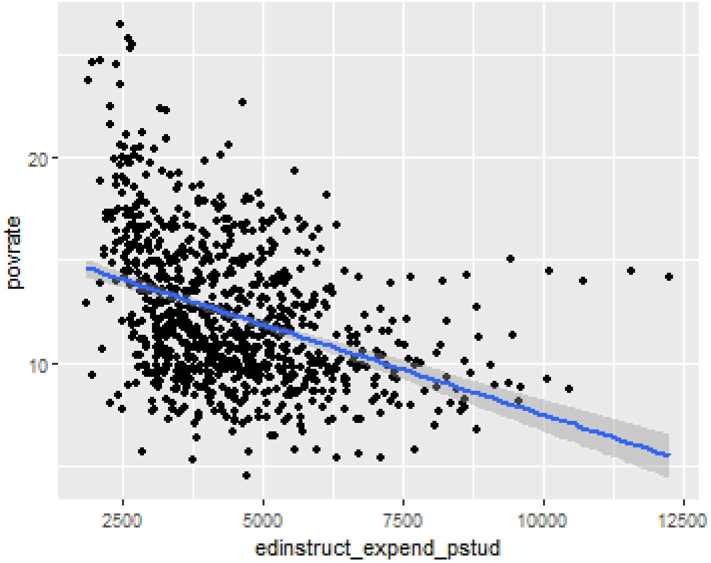


Two continuous variables

```
ggplot(data = states_final)+
  aes(x= edinstruct_expend_pstu
      y=povrate)+
  geom_point()
```



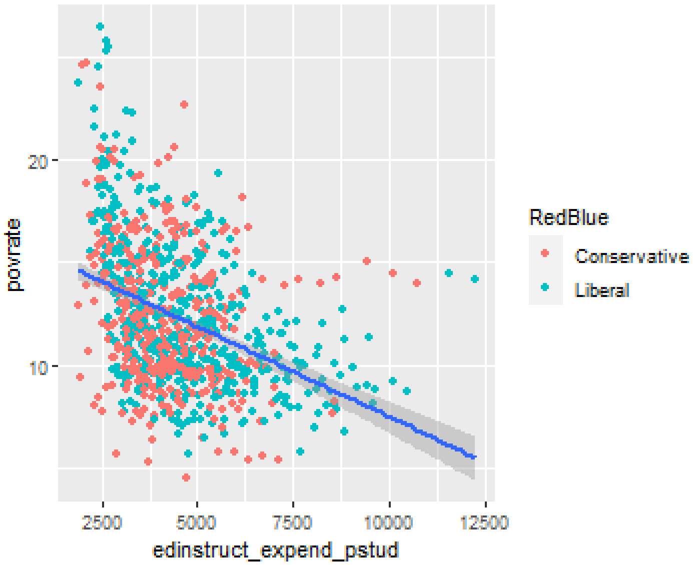
```
ggplot(data = states_final) +
  aes(x= edinstruct_expend_pstu
      y=povrate) +
  geom_point() +
  geom_smooth(method="lm")
```



Two continous variables

```
ggplot(data = states_final)+
  aes(x=edinstruct_expend_pstud
      y=povrate)+
  geom_point(aes(color=RedBlue)
  geom_smooth(method="lm")
```

```
ggplot(data = states_final)+
  aes(x=edinstruct_expend_pstud
      y=povrate,
      color=RedBlue)+
  geom_point()+
  geom_smooth(method="lm")
```



Graph styling: labels

```
ggplot(data = states_final)+
  aes(x=edinstruct_expend_pstud,
      y=povrate,
      color=RedBlue)+
  geom_point()+
  geom_smooth(method="lm") +
  labs(title = "Poverty and Education Expenditure",
       x ="Education expenditure per student",
       y = "Poverty rate",
       color = "Ideology")
```

Graph styling: themes

```
plot + theme_dark()
```

```
plot + theme_bw()
```

Graph styling: themes

External packages like `ggthemr` and `ggthemes` contain even more option. See Markdown document for an example.

Note on Color

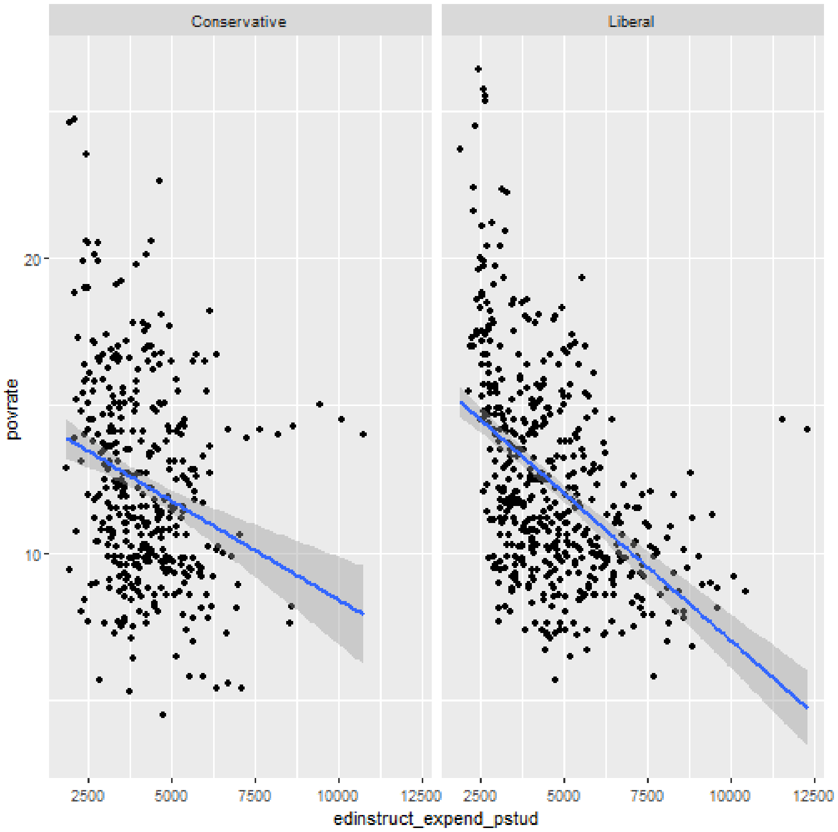
A Note on Colors

Colors

```r
library(RColorBrewer)
display.brewer.all(colorblindFriendly = TRUE)
```

Facetting

```
ggplot(data = states_final) +
  aes(x=edinstruct_expend_pstud
      y=povrate) +
  geom_point()+
  geom_smooth(method="lm") +
  facet_wrap(~RedBlue)
```

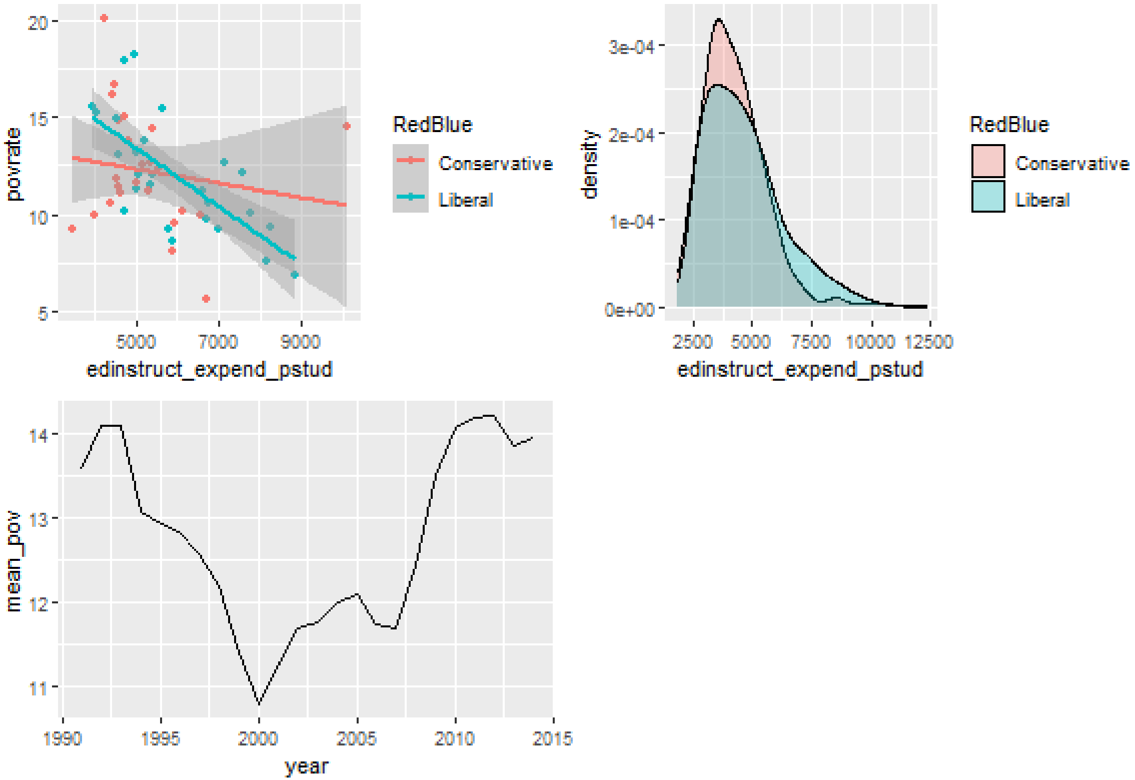


Combining plots

```
library(cowplot)

plot1 <- states_final %>% filter(year==2005) %>%
ggplot(aes(x=edinstruct_expend_pstud, y=povrate, color= RedBlue))+
  geom_point()+
  geom_smooth(method="lm")

plot2<- ggplot(data = states_final)+
  aes(x= edinstruct_expend_pstud, fill=RedBlue)+
  geom_density(alpha=0.3 )

plot3 <- states_final %>% group_by(year) %>% summarize(mean_pov = m
  geom_line()
```

Combining plots

```r
plot_grid(plot1, plot2, plot3, ncol=2, nrow=2)
```


povrate
edinstruct_expend_pstud
RedBlue
Conservative
Liberal
3e-04
2e-04
1e-04
0e+00
density
edinstruct_expend_pstud
RedBlue
Conservative
Liberal
mean_pov
year