Intro to R

Functions, Objects and Visualization

Prof. Bisbee

Seoul National University

Slides Updated: 2024-06-30

Agenda

1. Recap of last lecture
   - Using packages: `install.packages()` & `require()`
   - Loading and manipulating data: `read_rds()` and `%>%`
2. `tidyverse` functions
   - `filter` and `select`
   - `summarize` and `mutate`
   - `group_by`

Loading Packages & Data

- Create an `.Rmd` file and save to your `code` folder
    - Accept defaults, Save As... (with a good name), then `knit`
- Load the `tidyverse` package

```
require(tidyverse)
```

- Load the data from the course github page directly using `read_rds()`
    - We **create** an "object" to store the data using a left-arrow: `<-`

```
df<-
read_rds("https://github.com/jbisbee1/ISP_Data_Science_2024/raw/main/da
```

Tabular Data

- Data comes in many different formats
- **Structured data**: standardized, well-defined structure, easily accessed
  - I.e., tables, databases
  - In my YouTube example, the survey we gave was **structured**
- **Unstructured data**: messy, organic, disorganized, hard to use
  - I.e., web pages, images, videos
  - In my YouTube example, the scraped HTML code of a list of recommendations was **unstructured**
- In this class, we will always be working with **structured** data...specifically "tabular data frames"
- This still requires work to prepare!

Tabular Data Frame

- AKA a "tibble"
- These are "square" (although actually rectagular)
- Rows: **units of observation** (i.e., the entities we are studying)
    - People (each row is a survey respondent, athlete, etc.)
    - Places (each row is a state, county, country, etc.)
    - Things (each row is a tweet, firm, product, etc.)
- Columns: **variables of interest** (i.e., attributes we are studying)
    - Beliefs / behaviors / etc. (i.e., where rows are people)
    - Rainfall / crimes / etc. (i.e., where rows are places)
    - Likes / profits / etc. (i.e., where rows are things)

Looking at Data

- We now have the contents of `sc_debt.Rds` stored in the object `df`
- We can look at this object directly

```
df
```

```
## # A tibble: 2,546 × 16
##    unitid instnm stabbr grad_debt_mdn control region preddeg
##     <int> <chr>  <chr>          <int> <chr>   <chr>  <chr>  
##  1 100654 Alaba… AL             33375 Public  South… Bachel…
##  2 100663 Unive… AL             22500 Public  South… Bachel…
##  3 100690 Amrid… AL             27334 Private South… Associ…
##  4 100706 Unive… AL             21607 Public  South… Bachel…
##  5 100724 Alaba… AL             32000 Public  South… Bachel…
##  6 100751 The U… AL             23250 Public  South… Bachel…
##  7 100760 Centr… AL             12500 Public  South… Associ…
##  8 100812 Athen… AL             19500 Public  South… Bachel…
##  9 100830 Aubur… AL             24826 Public  South… Bachel…
## 10 100858 Aubur… AL             21281 Public  South… Bachel…
## # ℹ 2,536 more rows
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
```

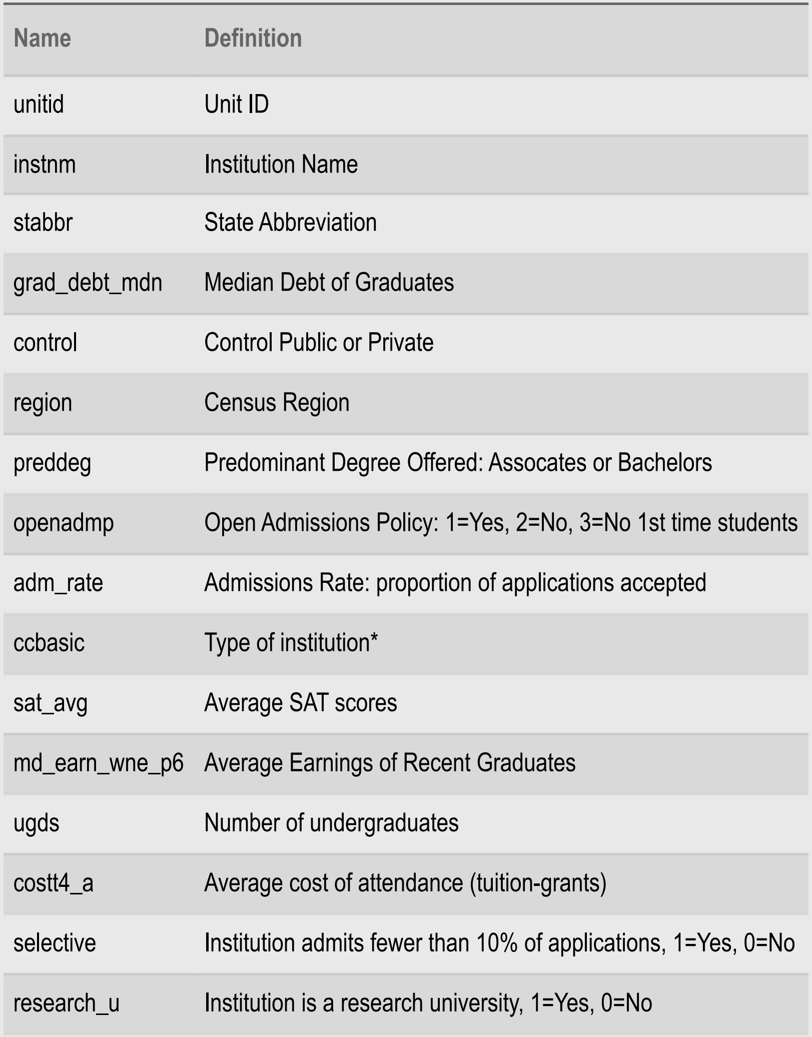
Looking at Data

- What is our **unit of observation**?
  - Academic institutions: each row is a single school
- What are our **variables of interest**?
  - Let's look!

```r
colnames(df) # Prints the variable names
```

```
##  [1] "unitid"        "instnm"          "stabbr"
##  [4] "grad_debt_mdn" "control"         "region"
##  [7] "preddeg"       "openadmp"        "adm_rate"
## [10] "ccbasic"       "sat_avg"         "md_earn_wne_p6"
## [13] "ugds"          "costt4_a"        "selective"
## [16] "research_u"
```

Good Data has Codebooks!



Name	Definition
unitid	Unit ID
instnm	Institution Name
stabbr	State Abbreviation
grad_debt_mdn	Median Debt of Graduates
control	Control Public or Private
region	Census Region
preddeg	Predominant Degree Offered: Assocates or Bachelors
openadmp	Open Admissions Policy: 1=Yes, 2=No, 3=No 1st time students
adm_rate	Admissions Rate: proportion of applications accepted
ccbasic	Type of institution*
sat_avg	Average SAT scores
md_earn_wne_p6	Average Earnings of Recent Graduates
ugds	Number of undergraduates
costt4_a	Average cost of attendance (tuition-grants)
selective	Institution admits fewer than 10% of applications, 1=Yes, 0=No
research_u	Institution is a research university, 1=Yes, 0=No

*

Looking at data

- Looking at data is **crucial**

```r
# First 6 rows
df %>% head()
```

```
## # A tibble: 6 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>  
## 1 100654 Alabam… AL             33375 Public  South… Bachel…
## 2 100663 Univer… AL             22500 Public  South… Bachel…
## 3 100690 Amridg… AL             27334 Private South… Associ…
## 4 100706 Univer… AL             21607 Public  South… Bachel…
## 5 100724 Alabam… AL             32000 Public  South… Bachel…
## 6 100751 The Un… AL             23250 Public  South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

- (Same as `head(df)`)

Looking at data

- Looking at data is **crucial**

```
# Last 6 rows
df %>% tail()
```

```
## # A tibble: 6 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>  
## 1 493716 Yeshiv… NJ                NA Private North… Associ…
## 2 493725 Univer… AR                NA Public  South… Bachel…
## 3 493822 Colleg… RI                NA Private New E… Bachel…
## 4 494630 Christ… TX                NA Private South… Bachel…
## 5 494685 Urshan… MO                NA Private Plains Bachel…
## 6 494737 Yeshiv… NY                NA Private North… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

- (Same as `tail(df)`)

Manipulating the Data

- Last lecture, we wanted to know…

    1. Where is `Vanderbilt University`?

```
df %>%
  filter(instnm == "Vanderbilt University") # Only select rows with
Vandy
```

```
## # A tibble: 1 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>
## 1 221999 Vander… TN             14962 Private South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

Manipulating the Data

- What if we don't know precisely how Vandy is spelled in these data?
- `str_detect()` and `grepl()` to the rescue!

```
df %>%
  filter(str_detect(instnm,'Vand'))
```

```
## # A tibble: 2 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>
## 1 149639 Vander… IL             27000 Private Great… Bachel…
## 2 221999 Vander… TN             14962 Private South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

Manipulating the Data

- What if we don't know precisely how Vandy is spelled in these data?
- `str_detect()` and `grepl()` to the rescue!

```
df %>%
  filter(grepl('Vand',instnm))
```

```
## # A tibble: 2 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>
## 1 149639 Vander… IL             27000 Private Great… Bachel…
## 2 221999 Vander… TN             14962 Private South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

Manipulating the Data

- We can go deeper with this logic
  - "or" denoted with |
  - "and" denoted with &

```
df %>%
  filter(str_detect(instnm,"Vand") | str_detect(instnm,"Tenn"))
```

```
## # A tibble: 12 × 16
##    unitid instnm stabbr grad_debt_mdn control region preddeg
##     <int> <chr>  <chr>          <int> <chr>   <chr>  <chr>
##  1 149639 Vande… IL             27000 Private Great… Bachel…
##  2 220075 East … TN             20500 Public  South… Bachel…
##  3 220978 Middl… TN             21500 Public  South… Bachel…
##  4 221485 South… TN                NA Public  South… Associ…
##  5 221731 Tenne… TN             21500 Private South… Bachel…
##  6 221740 The U… TN             20635 Public  South… Bachel…
##  7 221759 The U… TN             20500 Public  South… Bachel…
##  8 221768 The U… TN             22500 Public  South… Bachel…
##  9 221838 Tenne… TN             27000 Public  South… Bachel…
## 10 221847 Tenne… TN             17000 Public  South… Bachel…
```

Manipulating the Data

- We can go deeper with this logic
  - "or" denoted with |
  - "and" denoted with &

```
df %>%
  filter(str_detect(instnm,"Vand") & str_detect(instnm,"Univ"))
```

```
## # A tibble: 1 × 16
##   unitid instnm  stabbr grad_debt_mdn control region preddeg
##    <int> <chr>   <chr>          <int> <chr>   <chr>  <chr>
## 1 221999 Vander… TN             14962 Private South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

Manipulating the Data

- Can also put `|` or `&` in a single `str_detect()`

```
df %>%
  filter(str_detect(instnm,'Vand|Tenn'))
```

```
## # A tibble: 12 × 16
##    unitid instnm stabbr grad_debt_mdn control region preddeg
##     <int> <chr>  <chr>          <int> <chr>   <chr>  <chr>  
##  1 149639 Vande… IL             27000 Private Great… Bachel…
##  2 220075 East … TN             20500 Public  South… Bachel…
##  3 220978 Middl… TN             21500 Public  South… Bachel…
##  4 221485 South… TN                NA Public  South… Associ…
##  5 221731 Tenne… TN             21500 Private South… Bachel…
##  6 221740 The U… TN             20635 Public  South… Bachel…
##  7 221759 The U… TN             20500 Public  South… Bachel…
##  8 221768 The U… TN             22500 Public  South… Bachel…
##  9 221838 Tenne… TN             27000 Public  South… Bachel…
## 10 221847 Tenne… TN             17000 Public  South… Bachel…
## 11 221999 Vande… TN             14962 Private South… Bachel…
## 12 487010 The U… TN             13500 Public  South… Bachel…
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
```

Manipulating the Data

- But **can't** do the same with &

```
df %>%
  filter(str_detect(instnm,'Vand&Univ'))
```

```
## # A tibble: 0 × 16
## # ℹ 16 variables: unitid <int>, instnm <chr>, stabbr <chr>,
## #   grad_debt_mdn <int>, control <chr>, region <chr>,
## #   preddeg <chr>, openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
## #   research_u <dbl>
```

Manipulating the Data

- Negations are handled with `!`
    - Literally means "not"
- Drop rows with "of" in the school name

```
df %>%
  filter(!str_detect(instnm,"of"))
```

```
## # A tibble: 2,025 × 16
##    unitid instnm stabbr grad_debt_mdn control region preddeg
##     <int> <chr>  <chr>          <int> <chr>   <chr>  <chr>  
##  1 100654 Alaba… AL             33375 Public  South… Bachel…
##  2 100690 Amrid… AL             27334 Private South… Associ…
##  3 100724 Alaba… AL             32000 Public  South… Bachel…
##  4 100760 Centr… AL             12500 Public  South… Associ…
##  5 100812 Athen… AL             19500 Public  South… Bachel…
##  6 100830 Aubur… AL             24826 Public  South… Bachel…
##  7 100858 Aubur… AL             21281 Public  South… Bachel…
##  8 100937 Birmi… AL             25773 Private South… Bachel…
##  9 101028 Chatt… AL             11931 Public  South… Associ…
## 10 101161 Coast… AL             11000 Public  South… Associ…
```

Manipulating the Data

- (same as…)

```
df %>%
  filter(!grepl("of",instnm))
```

```
## # A tibble: 2,025 × 16
##    unitid instnm stabbr grad_debt_mdn control region preddeg
##     <int> <chr>  <chr>          <int> <chr>   <chr>  <chr>
##  1 100654 Alaba… AL             33375 Public  South… Bachel…
##  2 100690 Amrid… AL             27334 Private South… Associ…
##  3 100724 Alaba… AL             32000 Public  South… Bachel…
##  4 100760 Centr… AL             12500 Public  South… Associ…
##  5 100812 Athen… AL             19500 Public  South… Bachel…
##  6 100830 Aubur… AL             24826 Public  South… Bachel…
##  7 100858 Aubur… AL             21281 Public  South… Bachel…
##  8 100937 Birmi… AL             25773 Private South… Bachel…
##  9 101028 Chatt… AL             11931 Public  South… Associ…
## 10 101161 Coast… AL             11000 Public  South… Associ…
## # ℹ 2,015 more rows
## # ℹ 9 more variables: openadmp <int>, adm_rate <dbl>,
## #   ccbasic <int>, sat_avg <int>, md_earn_wne_p6 <int>,
## #   ugds <int>, costt4_a <int>, selective <dbl>,
```

Manipulating: select()

- Still TMI!
- Before, I only cared about the admissions rate (adm_rate), the SAT scores (sat_avg), and the future earnings (md_earn_wne_p6)
- select will select **columns**

```
df %>%
  filter(instnm == "Vanderbilt University") %>%
  select(instnm,adm_rate,sat_avg,md_earn_wne_p6) # Select variables
of interest
```

```
## # A tibble: 1 × 4
##   instnm                adm_rate sat_avg md_earn_wne_p6
##   <chr>                    <dbl>   <int>          <int>
## 1 Vanderbilt University   0.0912    1515          53400
```

Manipulating: select()

- We can use matches() function with select() in a manner similar to str_detect()

```
df %>%
  select(matches("_"))
```

```
## # A tibble: 2,546 × 6
##    grad_debt_mdn adm_rate sat_avg md_earn_wne_p6 costt4_a
##            <int>    <dbl>   <int>          <int>    <int>
##  1         33375    0.918     939          25200    23053
##  2         22500    0.737    1234          35100    24495
##  3         27334   NA          NA          30700    14800
##  4         21607    0.826    1319          36200    23917
##  5         32000    0.969     946          22600    21866
##  6         23250    0.827    1261          37400    29872
##  7         12500   NA          NA          23100    10493
##  8         19500   NA          NA          33400       NA
##  9         24826    0.904    1082          30100    19849
## 10         21281    0.807    1300          39500    31590
## # ℹ 2,536 more rows
## # ℹ 1 more variable: research_u <dbl>
```

Stepping back

- RQ: How might admissions and SAT scores be **related**?
  - Theory: selective schools have stricter criteria
  - Hypothesis: admissions and SAT scores should be **negatively** related
- How can we test this hypothesis?

Summarizing Data: `summarise()` + `mean()`

- We can combine base `R` functions with `tidyverse` functions!
  - Base `R`: `mean()`
  - `tidyverse`: `summarise()` (aka `summarize()`)
- Overall average SAT scores

```
df %>%
  summarise(mean_sat = mean(sat_avg,na.rm=T)) # Average SAT scores for entire data
```

```
## # A tibble: 1 × 1
##   mean_sat
##      <dbl>
## 1    1141.
```

Summarizing Data

- Let's unpack this

```
df %>%
  summarise(mean_sat = mean(sat_avg,na.rm=T))
```

- Create new variable `mean_sat` that contains the `mean()` of every school's average SAT score

- `na.rm=T` means we want to ignore missing data. If not?

```
df %>%
  summarise(mean_sat = mean(sat_avg))
```

```
## # A tibble: 1 × 1
##   mean_sat
##      <dbl>
## 1       NA
```

Summarizing Data

- Recall we want see if more selective schools have higher SAT scores

```
df %>%
  filter(adm_rate < .1) %>% # Only schools who accept < 10%
  summarise(mean_sat_LT10 = mean(sat_avg,na.rm=T)) # Average SAT
```

```
## # A tibble: 1 × 1
##   mean_sat_LT10
##           <dbl>
## 1         1510.
```

```
df %>%
  filter(adm_rate > .1) %>% # Only schools who accept > 10%
  summarise(mean_sat_GT20 = mean(sat_avg,na.rm=T)) # Average SAT
```

```
## # A tibble: 1 × 1
##   mean_sat_GT20
##           <dbl>
## 1         1135.
```

Adding / changing variables: mutate()

- mutate() creates a new variable

```
df %>%
  mutate(newvar = 1) %>%
  select(instnm,newvar)
```

```
## # A tibble: 2,546 × 2
##    instnm                              newvar
##    <chr>                                <dbl>
##  1 Alabama A & M University                 1
##  2 University of Alabama at Birmingham      1
##  3 Amridge University                       1
##  4 University of Alabama in Huntsville      1
##  5 Alabama State University                 1
##  6 The University of Alabama                1
##  7 Central Alabama Community College        1
##  8 Athens State University                  1
##  9 Auburn University at Montgomery          1
## 10 Auburn University                        1
```

Object Assignment Operator: <-

- Thus far, nothing we have done has changed df
- Use object assignment operator <- to **overwrite** an existing object

```
df <- df %>%
  mutate(adm_rate_pct = adm_rate*100)
```

- Did it work?

```
df %>%
  summarise(adm_rate_pct = mean(adm_rate_pct,na.rm=T),
            adm_rate = mean(adm_rate,na.rm=T))
```

```
## # A tibble: 1 × 2
##   adm_rate_pct adm_rate
##          <dbl>    <dbl>
## 1         67.9    0.679
```

Logic: `ifelse()`

- 3 inputs:
    - Logical statement (labeled `test`)
    - Value if the logic is `TRUE` (labeled `yes`)
    - Value if the logic is `FALSE` (labeled `no`)
- `ifelse([LOGIC],[VALUE IF TRUE],[VALUE IF FALSE])`

Logic: `ifelse()`

- Say it out loud: "Create a new variable called `sel` that records if the school is selective or not. If the admissions rate is less than 10% (0.1), record the school as `sel = 1`. Otherwise, record the school as `sel = 0`."

```
df %>%
  mutate(sel = ifelse(test = [LOGIC],
                      yes = [VALUE IF TRUE],
                      no = [VALUE IF FALSE]))
```

Logic: `ifelse()`

- Say it out loud: "Create a new variable called `sel` that records if the school is selective or not. **If the admissions rate is less than 10% (0.1)**, record the school as `sel = 1`. Otherwise, record the school as `sel = 0`."

```
df %>%
  mutate(sel = ifelse(test = adm_rate < 0.1, # This is the logic
                      yes = [VALUE IF TRUE],
                      no = [VALUE IF FALSE]))
```

Logic: `ifelse()`

- Say it out loud: "Create a new variable called `sel` that records if the school is selective or not. If the admissions rate is less than 10% (0.1), **record the school as** `sel = 1`. Otherwise, record the school as `sel = 0`."

```r
df %>%
  mutate(sel = ifelse(test = adm_rate < 0.1, # This is the logic
                      yes = 1, # This is the value if TRUE
                      no = [VALUE IF FALSE]))
```

Logic: ifelse()

- Say it out loud: "Create a new variable called sel that records if the school is selective or not. If the admissions rate is less than 10% (0.1), record the school as sel = 1. **Otherwise, record the school as sel = 0**."

```
df %>%
  mutate(sel = ifelse(test = adm_rate < 0.1, # This is the logic
                      yes = 1, # This is the value if TRUE
                      no = 0)) # This is the value if FALSE
```

Logic: `ifelse()` + `mutate()`

- Remember that if we want to keep this, we need the **assignment operator** `<-`

```
df <- df %>%
  mutate(sel = ifelse(test = adm_rate < 0.1, # This is the logic
                      yes = 1, # This is the value if TRUE
                      no = 0)) # This is the value if FALSE
```

Summarizing Data: group_by()

- One final tidyverse function: group_by()
- Let's use the newly created selective variable which is either 1 or 0

```
df %>%
  select(instnm,selective,adm_rate)
```

```
## # A tibble: 2,546 × 3
##    instnm                              selective adm_rate
##    <chr>                                   <dbl>    <dbl>
##  1 Alabama A & M University                    0    0.918
##  2 University of Alabama at Birmingham         0    0.737
##  3 Amridge University                         NA   NA    
##  4 University of Alabama in Huntsville         0    0.826
##  5 Alabama State University                    0    0.969
##  6 The University of Alabama                   0    0.827
##  7 Central Alabama Community College          NA   NA    
##  8 Athens State University                    NA   NA    
##  9 Auburn University at Montgomery             0    0.904
## 10 Auburn University                           0    0.807
## # ℹ 2,536 more rows
```

Summarizing Data: group_by()

- Instead of running two separate filter() commands, use group_by()

```r
df %>%
  # Group the data by selective (either 1 or 0)
  group_by(selective) %>%
  # Calculate average SAT for each group
  summarise(mean_sat = mean(sat_avg,na.rm=T))
```

```
## # A tibble: 3 × 2
##   selective mean_sat
##       <dbl>    <dbl>
## 1         0    1135.
## 2         1    1510.
## 3        NA     NaN
```

Results

- Do more selective schools have higher SAT scores?
- Yes
- This Result **confirms** our Hypothesis and **answers** our Research Question

Conclusion

- What we've done today is a microcosm of data science

  1. Opened data (`readRDS`)

  2. Looked at data (`tidyverse` + `select()`, `filter()`, `arrange()`)

  3. Generated hypotheses (Admissions versus SAT scores)

  4. Tested hypotheses (`summarise()` + `mean()`)

Advanced Logic: filter()

If no time, jump to end

- filter() command with other logical operators
  - >, <: greater than, less than (>=, <=)
  - !: not (i.e., != means "not equal to")
  - &: and
  - |: or

```
df %>%
  # Schools EXCEPT Vandy
  filter(instnm != "Vanderbilt University") %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 2,545 × 4
##    instnm                              stabbr adm_rate sat_avg
##    <chr>                               <chr>     <dbl>   <int>
##  1 Alabama A & M University            AL        0.918     939
##  2 University of Alabama at Birming…   AL        0.737    1234
##  3 Amridge University                  AL       NA          NA
##  4 University of Alabama in Huntsvi…   AL        0.826    1319
##  5 Alabama State University            AL        0.969     946
```

Advanced Logic: str_detect()

- filter() command with other functions
  - str_detect([VAR],[PATTERN]): detect a string
  - grepl([PATTERN],[VAR]): also detects a string

```
df %>%
  filter(str_detect(instnm,"Vanderbilt")) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 1 × 4
##   instnm                stabbr adm_rate sat_avg
##   <chr>                 <chr>     <dbl>   <int>
## 1 Vanderbilt University TN       0.0912    1515
```

Advanced Logic: str_detect()

- String detection is case sensitive!

```
df %>%
  filter(str_detect(instnm,"VAND")) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 0 × 4
## # ℹ 4 variables: instnm <chr>, stabbr <chr>,
## #   adm_rate <dbl>, sat_avg <int>
```

```
df %>%
  filter(str_detect(instnm,"anderbil")) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 1 × 4
##   instnm                stabbr adm_rate sat_avg
##   <chr>                 <chr>     <dbl>   <int>
## 1 Vanderbilt University TN       0.0912    1515
```

Advanced Logic: & (and), | (or)

```
df %>%
  filter(str_detect(instnm,"Colorado")) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 12 × 4
##    instnm                            stabbr adm_rate sat_avg
##    <chr>                             <chr>     <dbl>   <int>
##  1 University of Colorado Denver/An… CO        0.673    1124
##  2 University of Colorado Colorado … CO        0.872    1136
##  3 University of Colorado Boulder    CO        0.784    1276
##  4 Colorado Christian University     CO       NA          NA
##  5 Colorado College                  CO        0.135      NA
##  6 Colorado School of Mines          CO        0.531    1342
##  7 Colorado State University-Fort C… CO        0.814    1204
##  8 Colorado Mesa University          CO        0.782    1063
##  9 University of Northern Colorado   CO        0.908    1096
## 10 Colorado State University Pueblo  CO        0.930    1047
## 11 Western Colorado University       CO        0.842    1114
## 12 Colorado State University-Global… CO        0.986    1048
```

Advanced Logic: & (and), | (or)

```
df %>%
  filter(grepl("Colorado",instnm) & grepl(' of ',instnm)) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 5 × 4
##   instnm                                  stabbr adm_rate sat_avg
##   <chr>                                   <chr>     <dbl>   <int>
## 1 University of Colorado Denver/Ans… CO        0.673    1124
## 2 University of Colorado Colorado S… CO        0.872    1136
## 3 University of Colorado Boulder     CO        0.784    1276
## 4 Colorado School of Mines           CO        0.531    1342
## 5 University of Northern Colorado    CO        0.908    1096
```

Advanced Logic: & (and), | (or)

```
df %>%
  filter(grepl("Colorado",instnm) | grepl('Vermont',instnm)) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 16 × 4
##    instnm                                stabbr adm_rate sat_avg
##    <chr>                                 <chr>     <dbl>   <int>
##  1 University of Colorado Denver/An… CO        0.673    1124
##  2 University of Colorado Colorado … CO        0.872    1136
##  3 University of Colorado Boulder      CO        0.784    1276
##  4 Colorado Christian University       CO       NA          NA
##  5 Colorado College                    CO        0.135      NA
##  6 Colorado School of Mines            CO        0.531    1342
##  7 Colorado State University-Fort C… CO        0.814    1204
##  8 Colorado Mesa University            CO        0.782    1063
##  9 University of Northern Colorado     CO        0.908    1096
## 10 Colorado State University Pueblo    CO        0.930    1047
## 11 Western Colorado University         CO        0.842    1114
## 12 Community College of Vermont        VT       NA          NA
## 13 Northern Vermont University         VT        0.778      NA
## 14 Vermont Technical College           VT        0.670      NA
## 15 University of Vermont               VT        0.673    1287
```

Advanced Logic: & (and), | (or)

```r
df %>%
  filter((grepl("Colorado",instnm) | grepl('Vermont',instnm)) &
grepl(' of ',instnm)) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 7 × 4
##   instnm                                 stabbr adm_rate sat_avg
##   <chr>                                  <chr>     <dbl>   <int>
## 1 University of Colorado Denver/Ans… CO         0.673    1124
## 2 University of Colorado Colorado S… CO         0.872    1136
## 3 University of Colorado Boulder     CO         0.784    1276
## 4 Colorado School of Mines           CO         0.531    1342
## 5 University of Northern Colorado    CO         0.908    1096
## 6 Community College of Vermont       VT        NA          NA
## 7 University of Vermont              VT         0.673    1287
```

Advanced Logic: & (and), | (or)

- & can be separated into multiple `filter()` commands

```
df %>%
  filter((grepl("Colorado",instnm) | grepl('Vermont',instnm))) %>%
  filter(grepl(' of ',instnm)) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 7 × 4
##   instnm                             stabbr adm_rate sat_avg
##   <chr>                              <chr>     <dbl>   <int>
## 1 University of Colorado Denver/Ans… CO        0.673    1124
## 2 University of Colorado Colorado S… CO        0.872    1136
## 3 University of Colorado Boulder     CO        0.784    1276
## 4 Colorado School of Mines           CO        0.531    1342
## 5 University of Northern Colorado    CO        0.908    1096
## 6 Community College of Vermont       VT       NA          NA
## 7 University of Vermont              VT        0.673    1287
```

Advanced Logic: & (and), | (or)

- | can be moved into the `str_detect()` or `grepl()` commands

```
df %>%
  filter(grepl("Colorado|Vermont",instnm)) %>%
  filter(grepl(' of ',instnm)) %>%
  select(instnm,stabbr,adm_rate,sat_avg)
```

```
## # A tibble: 7 × 4
##   instnm                            stabbr adm_rate sat_avg
##   <chr>                             <chr>     <dbl>   <int>
## 1 University of Colorado Denver/Ans… CO        0.673    1124
## 2 University of Colorado Colorado S… CO        0.872    1136
## 3 University of Colorado Boulder     CO        0.784    1276
## 4 Colorado School of Mines           CO        0.531    1342
## 5 University of Northern Colorado    CO        0.908    1096
## 6 Community College of Vermont       VT       NA         NA
## 7 University of Vermont              VT        0.673    1287
```

Quick Test

- Filter schools from Texas with the word "community" in their name

```
# INSERT CODE HERE
```

Advanced Logic: select()

- select can be paired with matches() or contains() for similar flexibility (equivalent to str_detect() or grepl() for filter())

```
df %>%
  select(contains('inst'))
```

```
## # A tibble: 2,546 × 1
##    instnm
##    <chr>
##  1 Alabama A & M University
##  2 University of Alabama at Birmingham
##  3 Amridge University
##  4 University of Alabama in Huntsville
##  5 Alabama State University
##  6 The University of Alabama
##  7 Central Alabama Community College
##  8 Athens State University
##  9 Auburn University at Montgomery
## 10 Auburn University
## # ℹ 2,536 more rows
```

Advanced Logic: select()

- matches can work with |

```
df %>%
  select(!matches('_|inst'))
```

```
## # A tibble: 2,546 × 10
##     unitid stabbr control region    preddeg  openadmp ccbasic
##      <int> <chr>  <chr>   <chr>     <chr>       <int>   <int>
##  1 100654 AL     Public  Southeast Bachelo…        2      18
##  2 100663 AL     Public  Southeast Bachelo…        2      15
##  3 100690 AL     Private Southeast Associa…        1      20
##  4 100706 AL     Public  Southeast Bachelo…        2      16
##  5 100724 AL     Public  Southeast Bachelo…        2      19
##  6 100751 AL     Public  Southeast Bachelo…        2      15
##  7 100760 AL     Public  Southeast Associa…        1       2
##  8 100812 AL     Public  Southeast Bachelo…       NA      22
##  9 100830 AL     Public  Southeast Bachelo…        2      18
## 10 100858 AL     Public  Southeast Bachelo…        2      15
## # ℹ 2,536 more rows
## # ℹ 3 more variables: ugds <int>, selective <dbl>,
## #   sel <dbl>
```

Advanced Logic: select()

- select can also work with where to find classes

```
df %>%
  select(where(is.numeric))
```

```
## # A tibble: 2,546 × 13
##    unitid grad_debt_mdn openadmp adm_rate ccbasic sat_avg
##     <int>         <int>    <int>    <dbl>   <int>   <int>
##  1 100654         33375        2    0.918      18     939
##  2 100663         22500        2    0.737      15    1234
##  3 100690         27334        1   NA          20      NA
##  4 100706         21607        2    0.826      16    1319
##  5 100724         32000        2    0.969      19     946
##  6 100751         23250        2    0.827      15    1261
##  7 100760         12500        1   NA           2      NA
##  8 100812         19500       NA   NA          22      NA
##  9 100830         24826        2    0.904      18    1082
## 10 100858         21281        2    0.807      15    1300
## # ℹ 2,536 more rows
## # ℹ 7 more variables: md_earn_wne_p6 <int>, ugds <int>,
## #   costt4_a <int>, selective <dbl>, research_u <dbl>,
## #   adm_rate_pct <dbl>, sel <dbl>
```

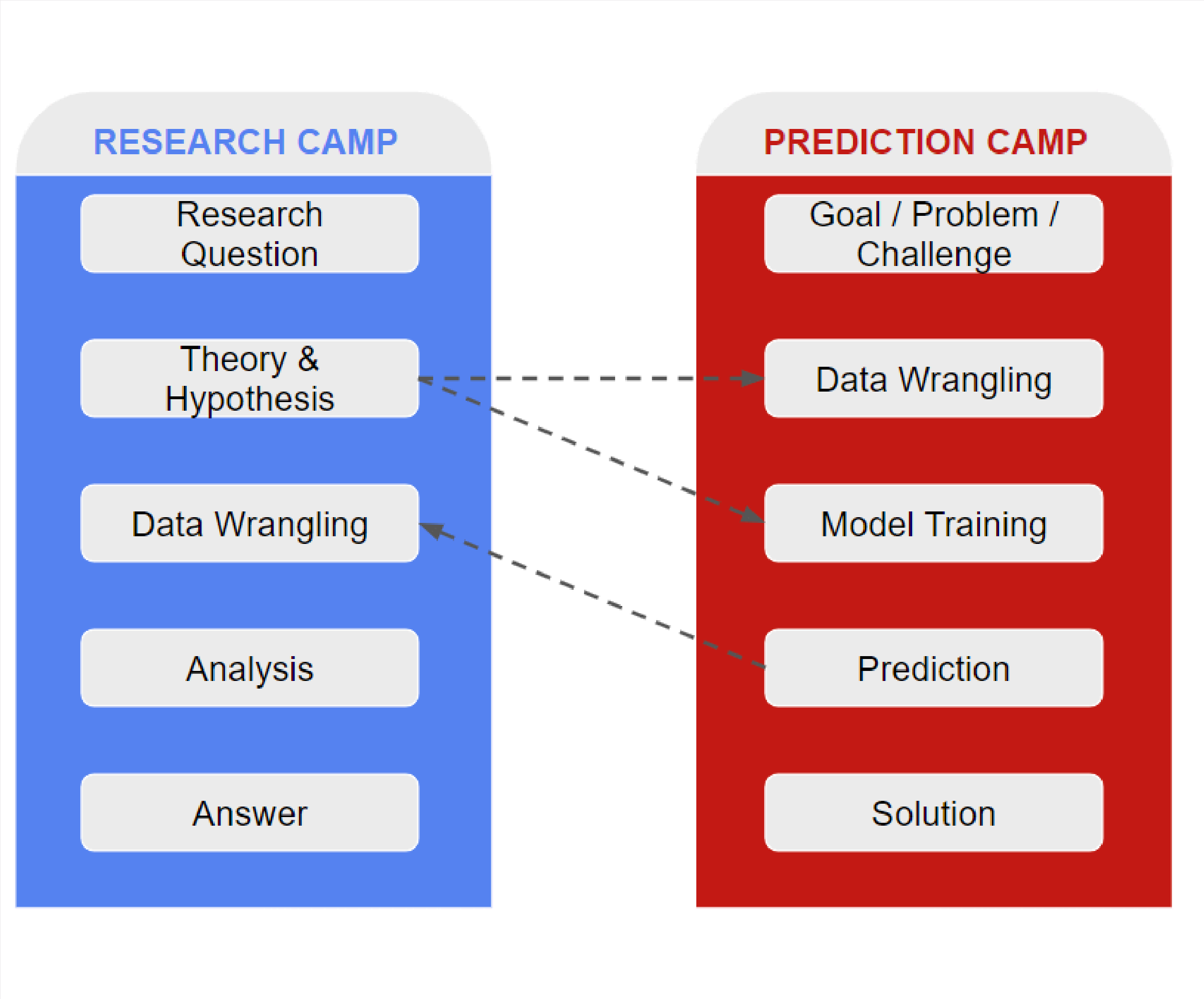
Quick Test

- Filter to only schools in California and select only character columns

```
# INSERT CODE HERE
```

BREAK

The Two Camps



The Research Camp

- RQ: How might admissions and SAT scores be **related**?
  - Theory: selective schools have stricter criteria
  - Hypothesis: admissions and SAT scores should be **negatively** related
- How can we test this hypothesis?

Previously: summarise()

- We can combine base R functions with tidyverse functions!
    - Base R: mean()
    - tidyverse: summarise() (aka summarize())
- Overall average SAT scores

```
df %>%
  summarise(mean_sat = mean(sat_avg,na.rm=T))
```

```
## # A tibble: 1 × 1
##   mean_sat
##      <dbl>
## 1    1141.
```

Previously: summarise()

- Let's unpack this

```
df %>%
  summarise(mean_sat = mean(sat_avg,na.rm=T))
```

- Create new variable mean_sat that contains the mean() of every school's average SAT score

- na.rm=T means we want to ignore missing data. If not?

```
df %>%
  summarise(mean_sat = mean(sat_avg))
```

```
## # A tibble: 1 × 1
##   mean_sat
##      <dbl>
## 1       NA
```

summarise() + filter()

- Recall we want see if more selective schools have higher SAT scores

```
df %>%
  filter(adm_rate < .1) %>%
  summarise(mean_sat_LT10 = mean(sat_avg,na.rm=T))
```

```
## # A tibble: 1 × 1
##   mean_sat_LT10
##           <dbl>
## 1         1510.
```

```
df %>%
  filter(adm_rate > .1 & adm_rate < .2) %>%
  summarise(mean_sat_1020 = mean(sat_avg,na.rm=T))
```

```
## # A tibble: 1 × 1
##   mean_sat_1020
##           <dbl>
## 1         1424.
```

summarise() + group_by()

- One final tidyverse function: group_by()

```
df %>%
  group_by(selective) %>%
  summarise(mean_sat = mean(sat_avg,na.rm=T))
```

```
## # A tibble: 3 × 2
##   selective mean_sat
##       <dbl>    <dbl>
## 1         0    1135.
## 2         1    1510.
## 3        NA      NaN
```

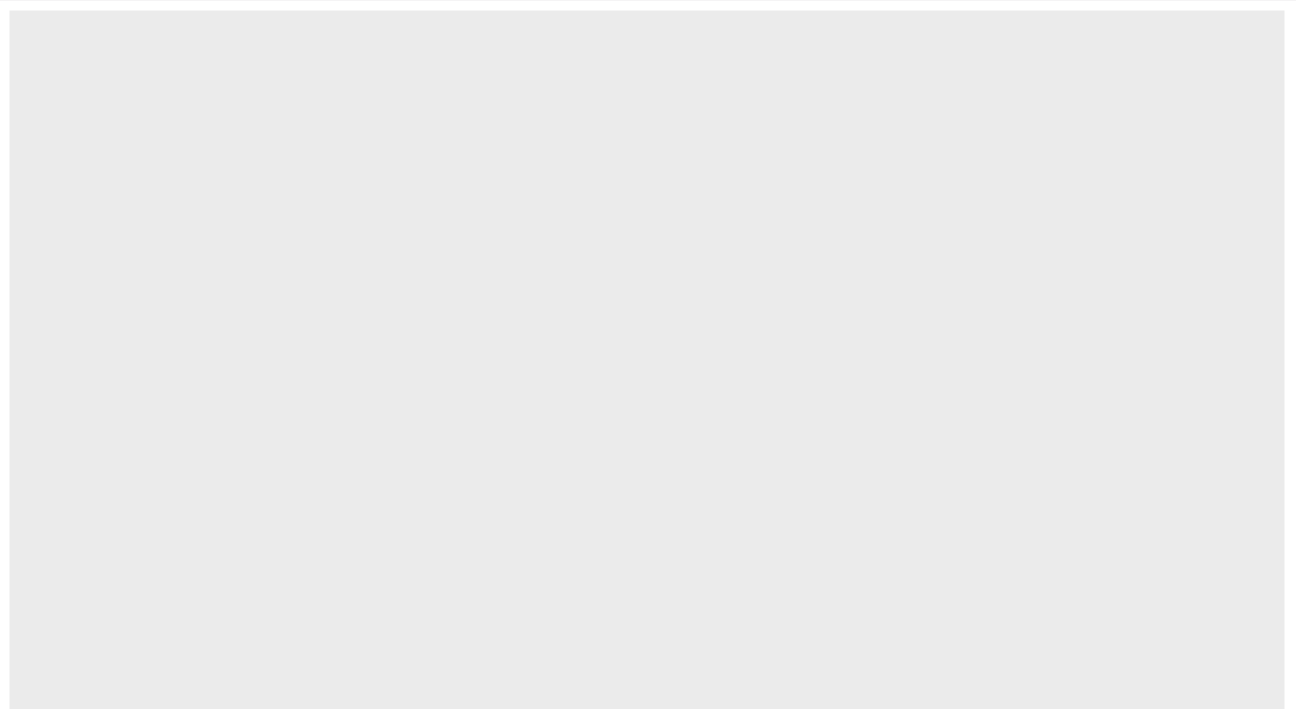
Plotting data

- Let's plot the data instead of writing many of these `summarise()` functions
- Visualization in `R` uses `ggplot()` function
    - Inputs: `aes(x,y,...)` (elipses `...` indicates many more inputs)
    - `x` is the x-axis (horizontal)
    - `y` is the y-axis (vertical)

ggplot()

- Attach `ggplot()` to your data with `%>%`

```
df %>%
  ggplot()
```

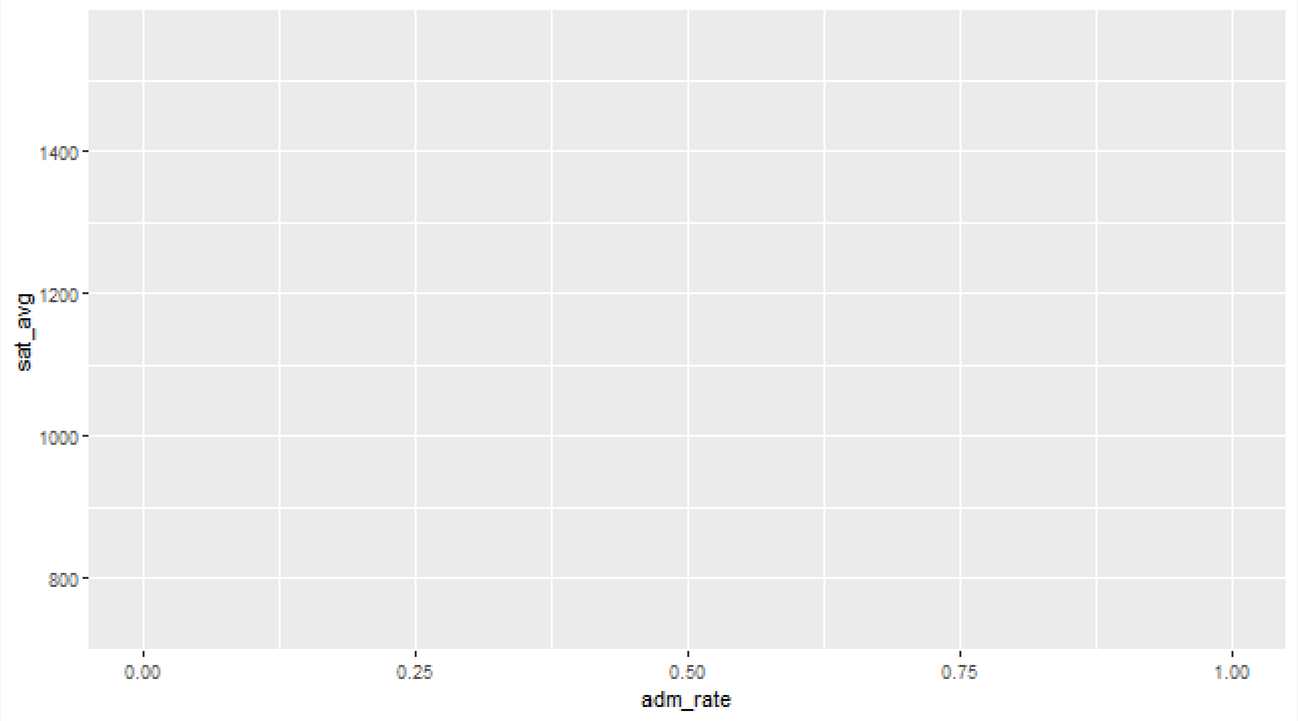


`ggplot()`

- Then tell it what to put in the x-axis and y-axis
- What should go on these axes?
- Theory: Selective schools choose higher scoring students
    - Selective schools **explain** higher scores
    - Selective schools: **independent variable** / **explanatory variable** / **predictor** / X
    - Higher scores: **dependent variable** / **outcome variable** / Y
- Selective schools go on the x-axis, SAT scores go on the y-axis

ggplot()

```
df %>%
  ggplot(aes(x = adm_rate,y = sat_avg))
```



`ggplot()`

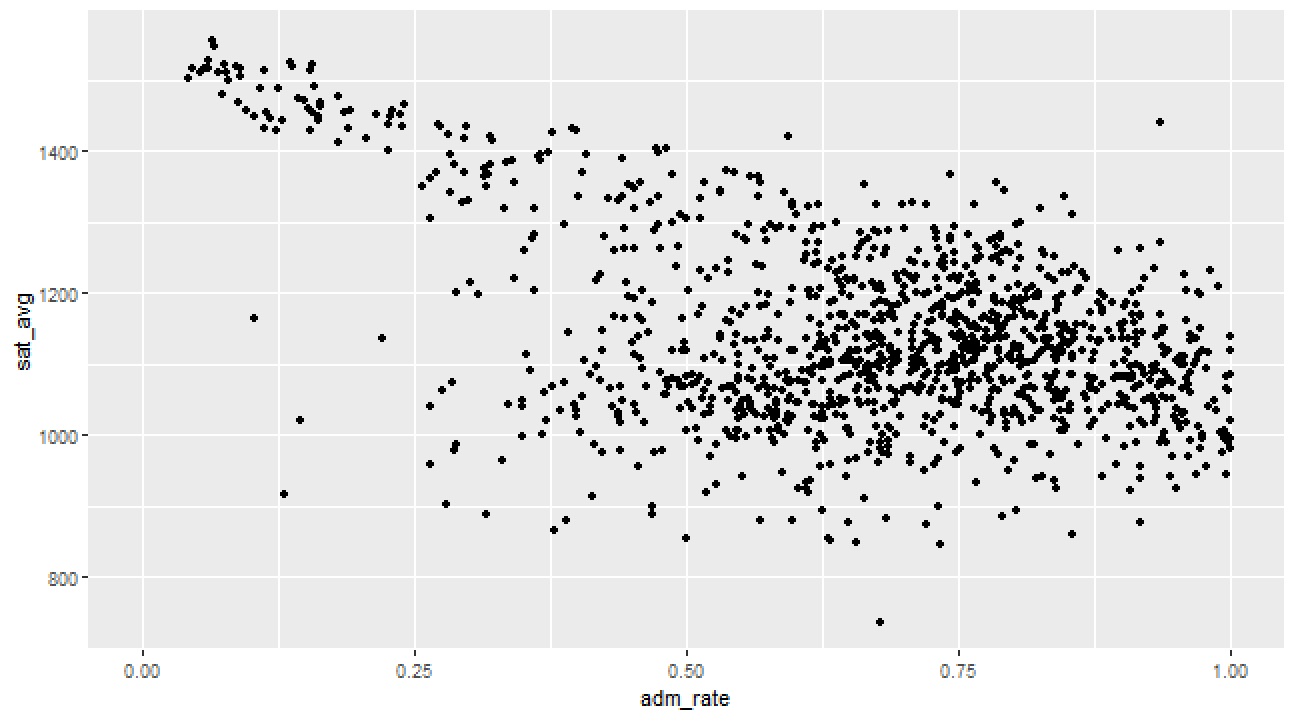
- This gives us an empty plot
- We have the correct variables on the correct axes...
- ...but we need to choose how to display them
- There are many different `ggplot()` functions to choose from
  - `geom_point()` creates one point for each x and y coordinate
  - `geom_bar()` creates a barplot
  - `geom_histogram()` creates a histogram
  - `geom_density()` creates a density plot
  - `geom_boxplot()` creates a box-and-whisker plot

`ggplot()`

- We **add** a second `ggplot()` function to the first with a plus sign `+`
    - **NB:** This is JUST LIKE THE PIPE OPERATOR `%>%` in `tidyverse`!
- Since `adm_rate` (the x-axis variable) and `sat_avg` (the y-axis variable) are both numeric ("continuous") measures, we will use `geom_point()`
    - We will come back to **variable types** and how to visualize them later

ggplot()

```
df %>%
  ggplot(aes(x = adm_rate,y = sat_avg)) +
  geom_point()
```



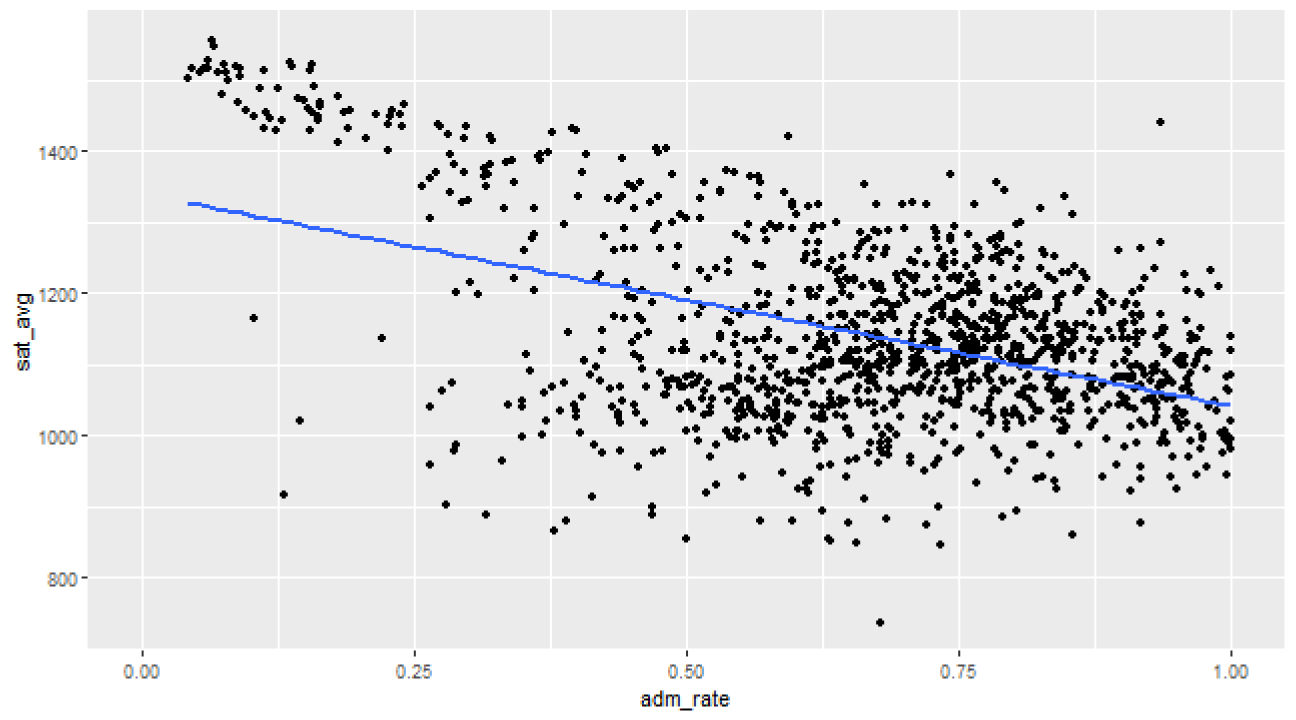
Plotting data

- Let's unpack this
  - `aes(x,y)` sets the basic aesthetics for the plot
  - `geom_point()` tells `ggplot()` how to visualize those aesthetics
  - These two parts are linked with the `+`. Similar to...?
  - ...the `%>%` in `tidyverse`!

Interpreting the plot

- We **hypothesized** that admissions and SAT scores are negatively related
    - Is this supported in the data?
- Let's add a line of best fit with `geom_smooth()`

```
df %>%
  ggplot(aes(x = adm_rate,y = sat_avg)) +
  geom_point() +
  geom_smooth(method = 'lm',se = F)
```


1400
1200
1000
800
sat_avg
0.00
0.25
0.50
0.75
1.00
adm_rate

The Research Camp

- RQ: How might future earnings and SAT scores be **related**?
  - Theory: SATs measure student ability.
  - Theory: Student ability is valued by the labor market.
  - Theory: Firms pay more for students with higher SAT scores.
  - Hypothesis: Earnings and SAT scores should be **positively** related

Plotting Quiz

- Which variable goes on the x-axis?
  - **SAT scores**
- Which variable goes on the y-axis?
  - **Earnings**
- In our theory, SAT scores **cause** earnings
- Why might this **not** be the case?
  - Spurious 1: SAT scores **and** earnings are caused by student ability
  - Spurious 2: SAT scores **and** earnings are caused by socio-economic privilege

Let's Plot!

```r
df %>%
  ggplot(aes(x = sat_avg,y = md_earn_wne_p6)) + # Build axes
  geom_point() +  # Add points
  geom_smooth(method = 'lm',se = F) # Add line of best fit
```

Outliers

- Which schools are furthest from the line?
  - These are **outliers**
  - These schools are the **furthest** from our theory

```
df %>%
  mutate(out = ifelse(md_earn_wne_p6 > 100000,
                       instnm,  # Value if TRUE
                       NA)) %>% # Value if FALSE
  drop_na(out,sat_avg) %>%
  select(instnm,md_earn_wne_p6,sat_avg)
```

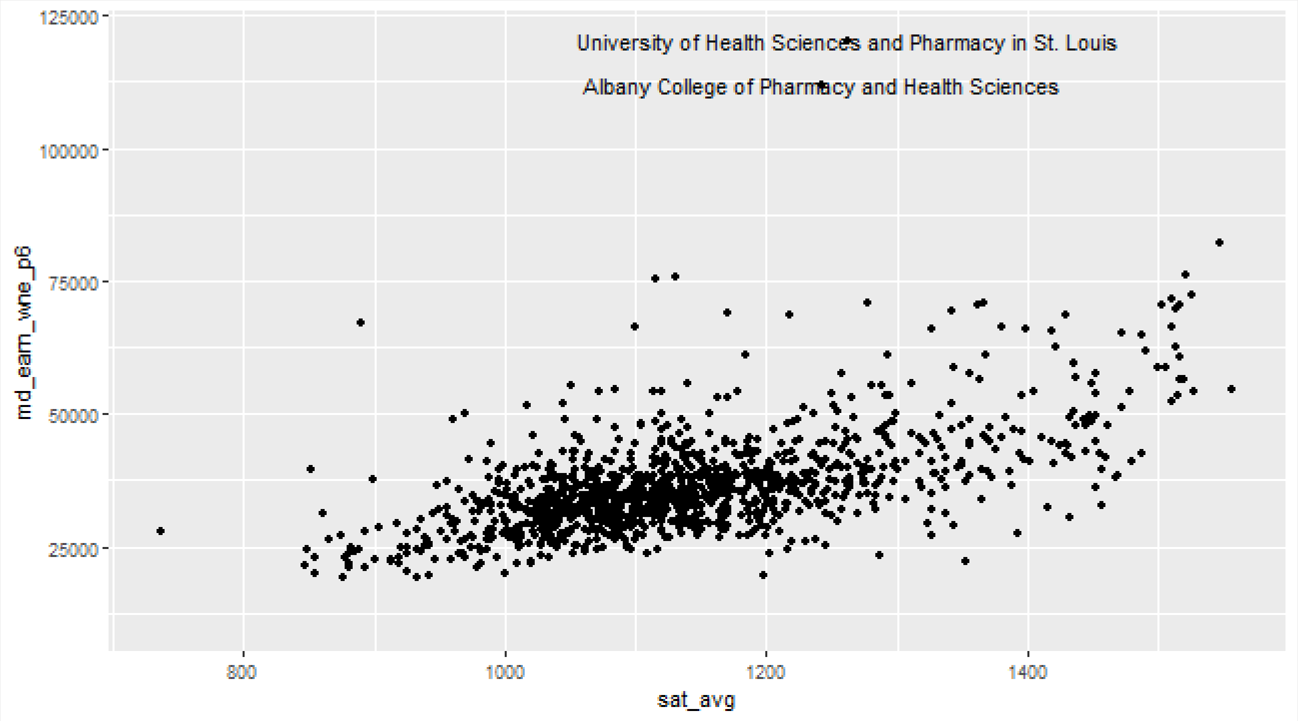
```
## # A tibble: 2 × 3
##   instnm                                    md_earn_wne_p6 sat_avg
##   <chr>                                              <int>   <int>
## 1 University of Health Sciences and …               120400    1262
## 2 Albany College of Pharmacy and Hea…               112100    1242
```

Plotting data

- We can add these as labels!

```
df %>%
   mutate(out = ifelse(md_earn_wne_p6 > 100000,
                          instnm,  # Value if TRUE
                          NA)) %>% # Value if FALSE
  ggplot(aes(x = sat_avg,y = md_earn_wne_p6,
             label = out)) +
  geom_point() +
  geom_text()
```

Plotting data

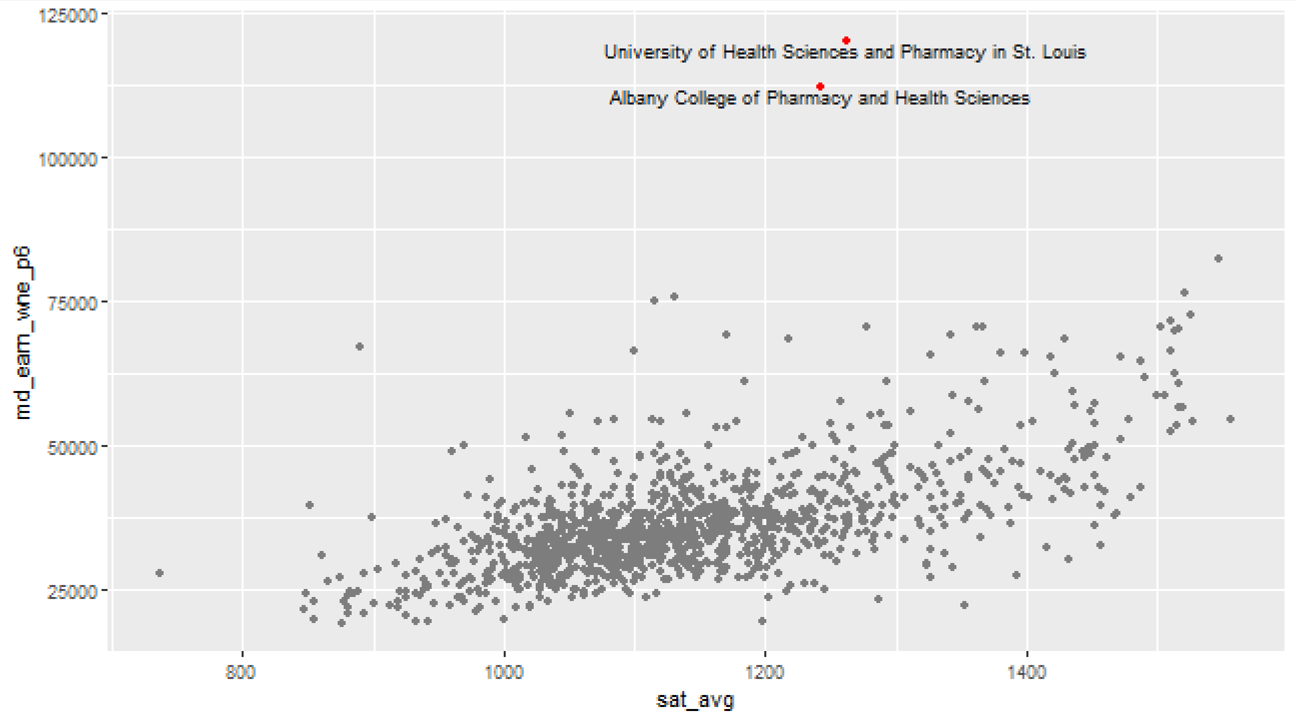


Plotting data

- Let's accentuate the outlier more with color

```
p <- df %>%
   mutate(out = ifelse(md_earn_wne_p6 > 100000,
                          instnm,  # Value if TRUE
                          NA)) %>% # Value if FALSE
  drop_na(sat_avg) %>%
  ggplot(aes(x = sat_avg,y = md_earn_wne_p6,
             label = out,color = out)) +
  geom_point() +
  scale_color_manual(name = "Outlier",values =
c('red','red','black')) +
  geom_text(hjust = .5,vjust = 1,color = 'black',size = 3)
```

Plotting data



Categorical Data

- Thus far, plotting two continuous variables with `geom_point()`
- What if we wanted to see which state has the most selective schools?
- Use `group_by()` and `summarise()`

```
df %>%
  group_by(stabbr) %>%
  summarise(selective_avg = mean(adm_rate,na.rm=T))
```

```
## # A tibble: 51 × 2
##    stabbr selective_avg
##    <chr>          <dbl>
##  1 AK             0.827
##  2 AL             0.654
##  3 AR             0.676
##  4 AZ             0.843
##  5 CA             0.592
##  6 CO             0.768
##  7 CT             0.589
##  8 DC             0.529
##  9 DE             0.627
```

Categorical Data

- Two variables (stabbr and selective_avg), but one of them is now a character type

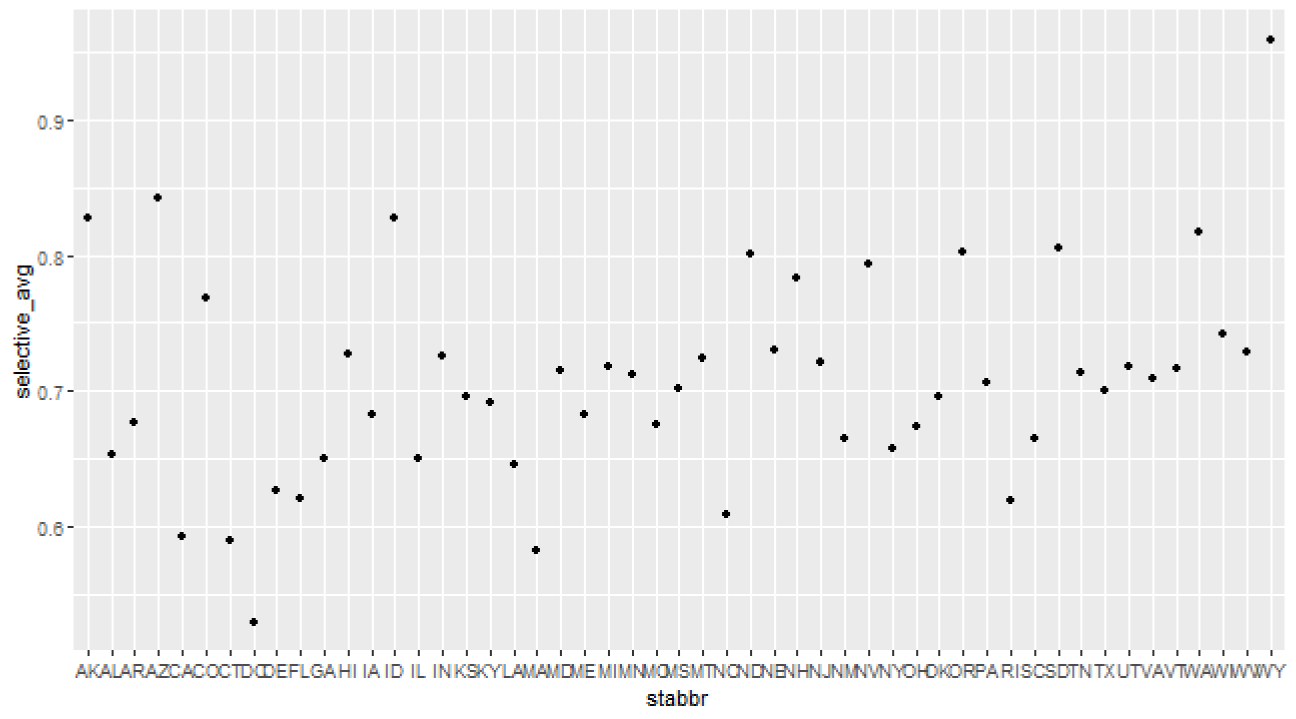
- Can we plot this as a scatterplot?

```
p <- df %>%
  group_by(stabbr) %>%
  summarise(selective_avg = mean(adm_rate,na.rm=T)) %>%
  ggplot(aes(x = stabbr,y = selective_avg)) +
  geom_point()
```

Categorical Data

- Yes...but it isn't very pretty

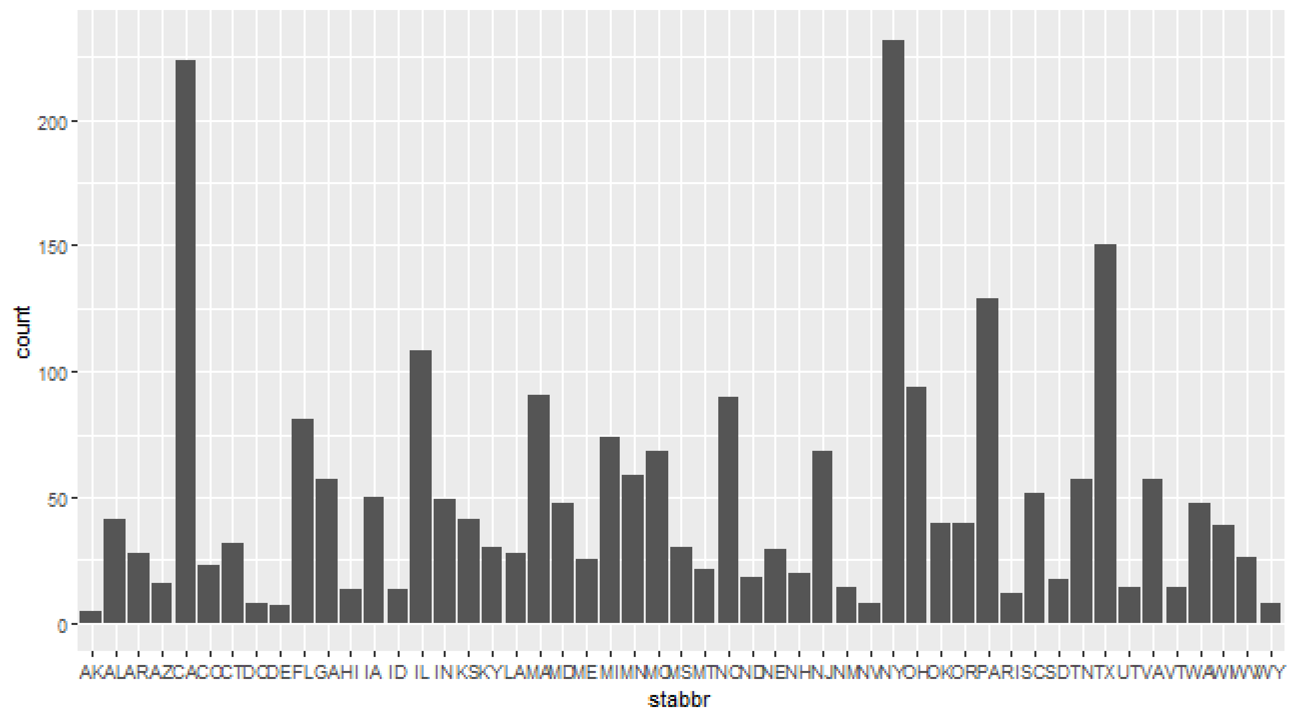
```
p
```



Categorical Data: `geom_bar()`

- NB: `geom_bar()` will automatically count the values on the x-axis

```
df %>%
  ggplot(aes(x = stabbr)) +
  geom_bar()
```



Categorical Data: `geom_bar()`

- This is fine if we just want to know which states have the most schools in our data

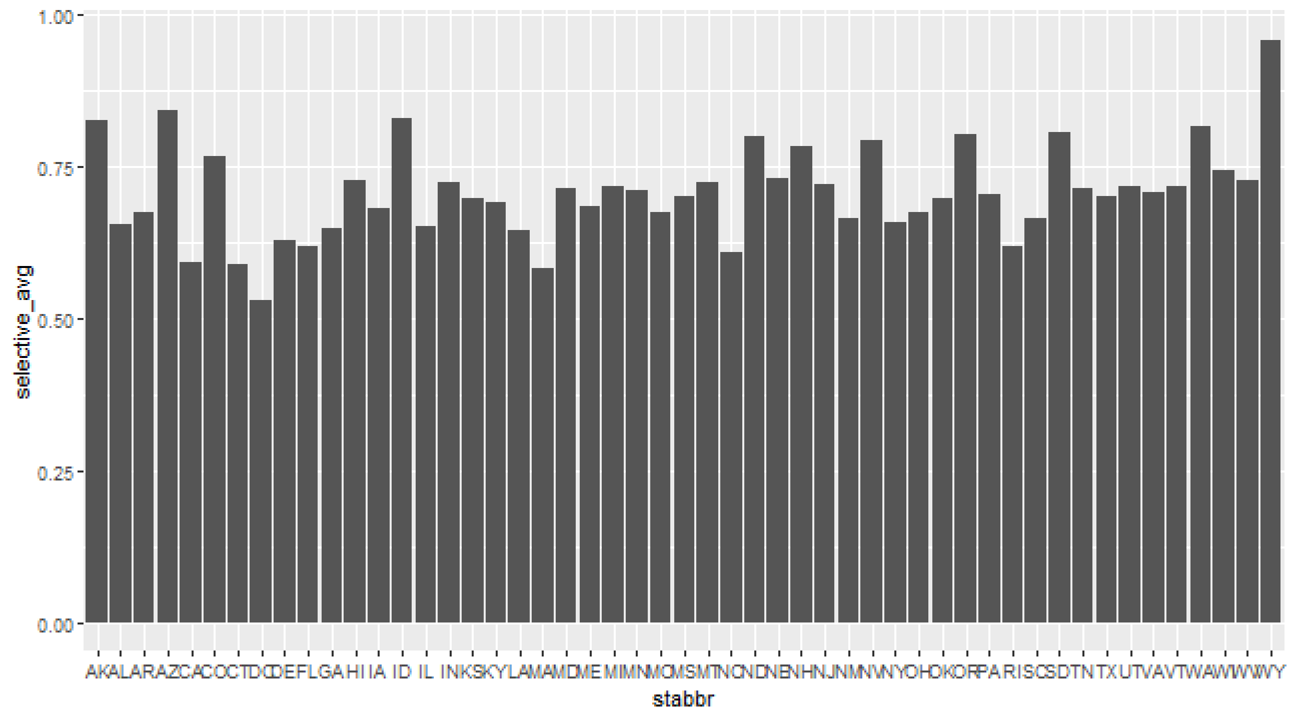
- But we want to put the average admissions rate on the y-axis instead

  - Need to **override** `geom_bar()` default behavior

```
p <- df %>%
  group_by(stabbr) %>%
  summarise(selective_avg = mean(adm_rate,na.rm=T)) %>%
  ggplot(aes(x = stabbr,y = selective_avg)) +
  geom_bar(stat = 'identity')
```

Categorical Data

```
p
```



Categorical Data

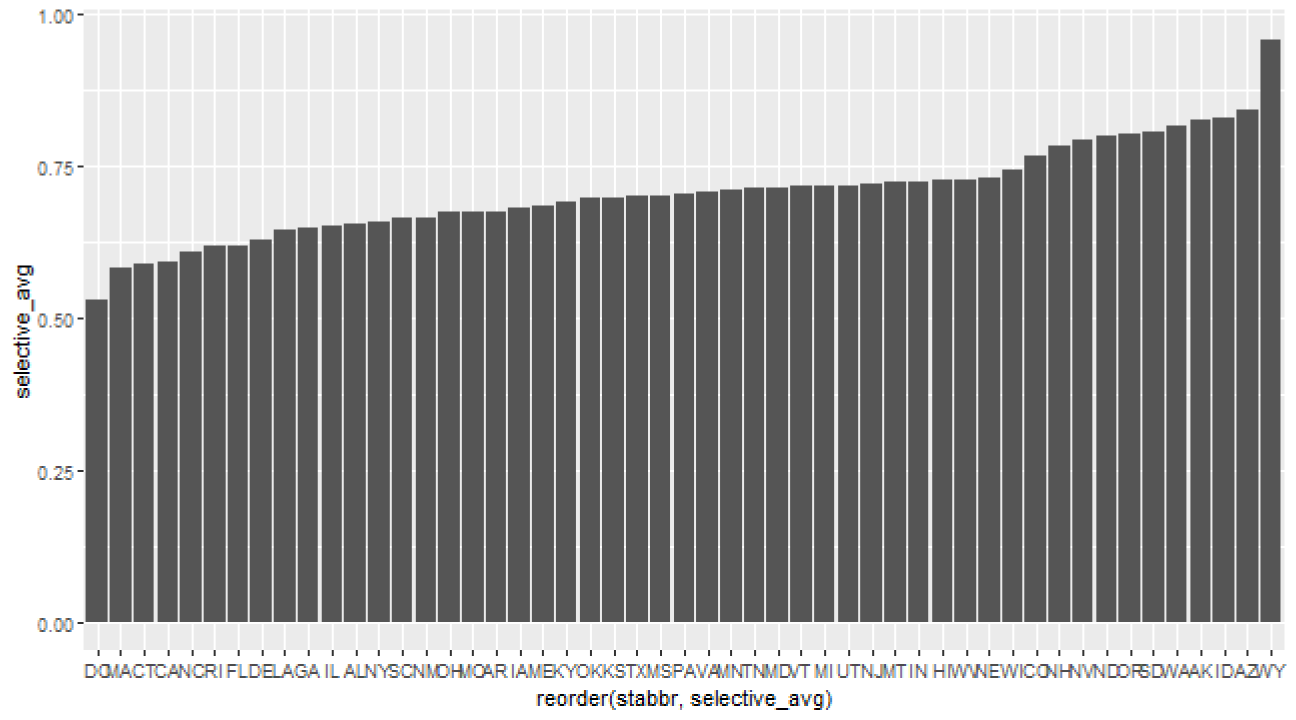
- Getting a little better, but still ugly
- Use `reorder()` to sort the x-axis values by the y-axis

```
p <- df %>%
  group_by(stabbr) %>%
  summarise(selective_avg = mean(adm_rate,na.rm=T)) %>%
  ggplot(aes(x = reorder(stabbr,selective_avg),y = selective_avg)) +
  geom_bar(stat = 'identity')
```

Categorical Data

- Even better!

```
p
```



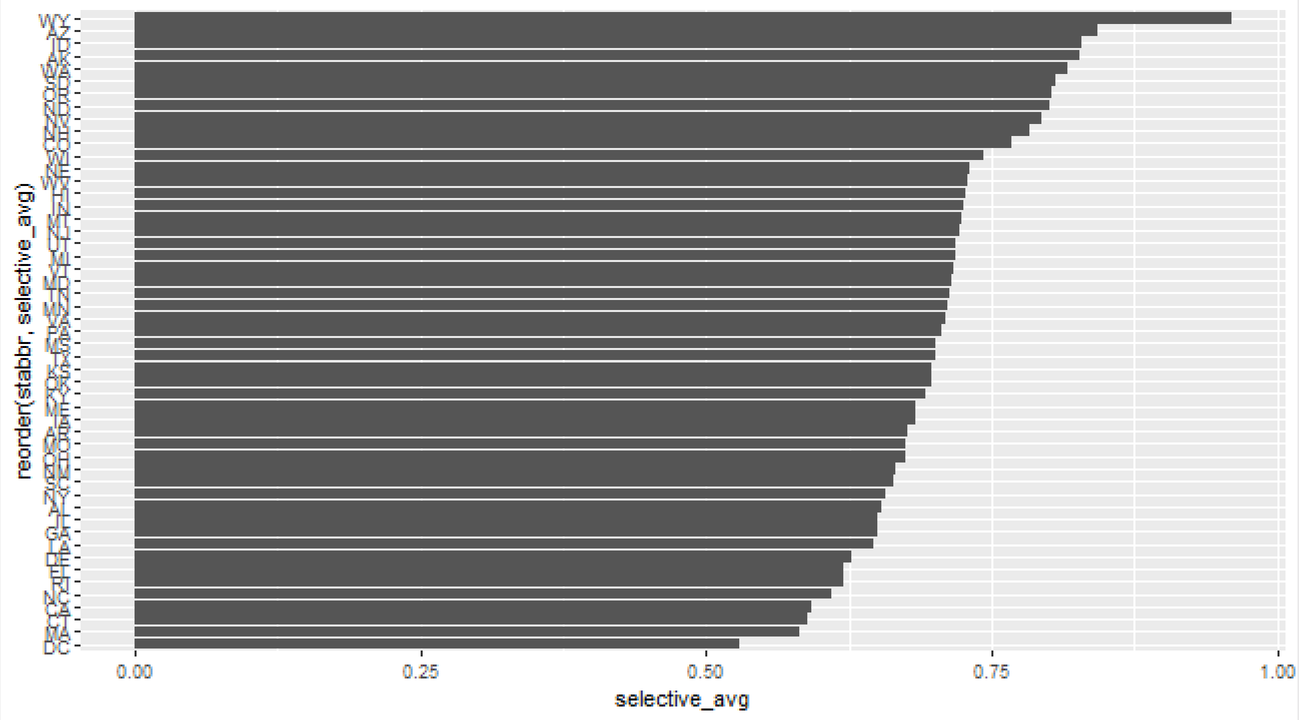
Plot Tweaking

- We could go even further and swap the x and y-axes (although this isn't always a good idea!)

```r
p <- df %>%
  group_by(stabbr) %>%
  summarise(selective_avg = mean(adm_rate,na.rm=T)) %>%
  ggplot(aes(y = reorder(stabbr,selective_avg),x = selective_avg)) +
  geom_bar(stat = 'identity')
```

Plot Tweaking

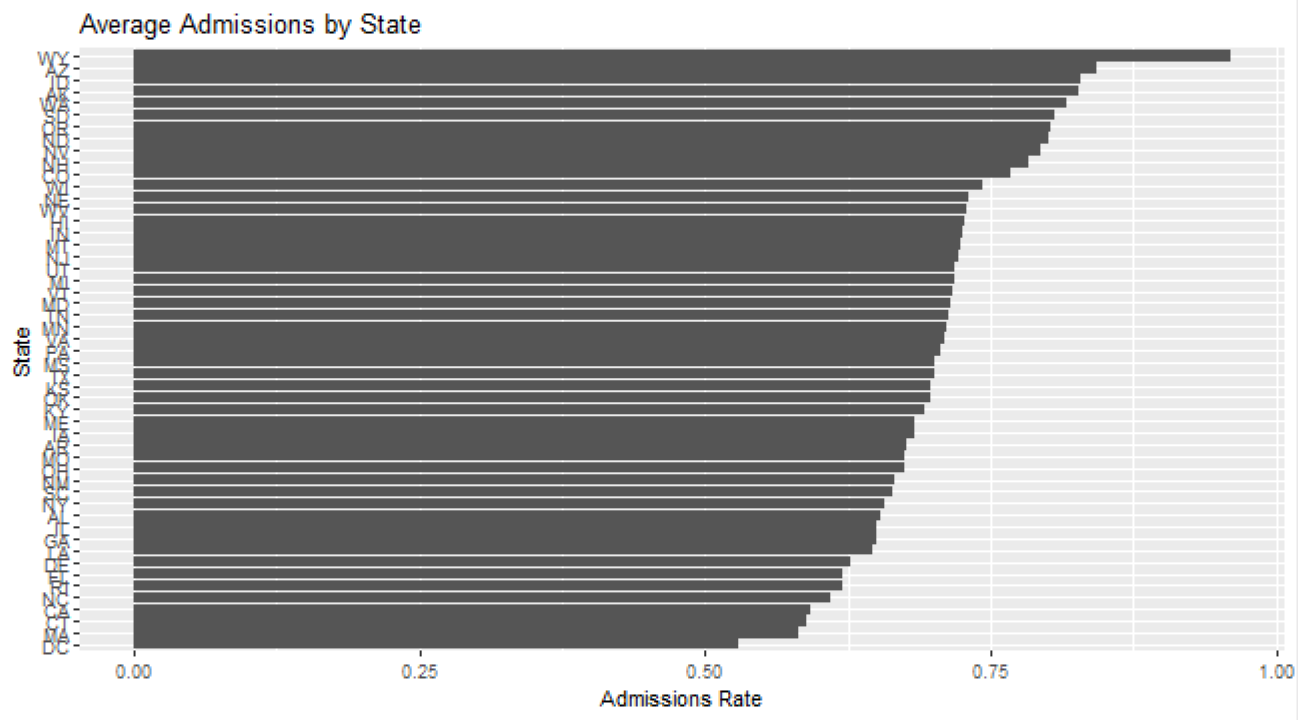
```
p
```



- Still ugly though! We want to tweak the labels with `labs()`

Plot Tweaking

```
p +
  labs(title = "Average Admissions by State",
       x = "Admissions Rate",
       y = "State")
```



Conclusion

- What to take away
  1. Which variables go on which axes
  2. How to put these on a `ggplot()` figure
  3. How to create a visualization of these variables
- This wraps up the crash course in `R`
  - **REMEMBER**: This class is *inherently* challenging because of `R`
  - The course is graded leniently to reflect the inherent difficulty of the material