# Skills Problem Set 1

Fernanda Sobrino

3/14/2021

Due Thursday April 8, midnight Central Time.

Upload your pdf to canvas.

Push your code to your repo on Github Classroom.

This submission is my work alone and complies with the 30535 integrity policy. Add your initials to indicate your agreement: **ES**

Add names of anyone you discussed this problem set with: **Guillermo Antonio Trefogli Wong**

Late coins used this pset: 0. Late coins left after submission: 9.

Name your submission files `skills_ps_1.Rmd` and `skills_ps_1.pdf`. (10 points)

## 1 Setup

### 1.1 Installation (10 points)

1. If you do not have R and RStudio installed:watch and follow the video on how to install them
2. If you do not have a github account, set up one now
3. Download Github Desktop here. If you are familiar with using `git` through the command line you are welcome to do so.
4. Initialize your ps1 repository and download the pset_template here. Please read the README file which is visible on the repo's homepage.
5. Make sure you already installed these packages in R: `tidyverse`, `markdown`, and `dslabs`

```
# install.packages("tidyverse")
# install.packages("markdown")
# install.packages("dslabs")

## First specify the packages of interest
list.of.packages <- c("tidyverse", "markdown", "dslabs")

## Now load or install&load all
package.check <- lapply(
  list.of.packages,
  FUN = function(x) {
    if (!require(x, character.only = TRUE)) {
      install.packages(x, dependencies = TRUE)
      library(x, character.only = TRUE)
    }
```

```
  }
)
```

```
## Loading required package: markdown
```

```
## Warning: package 'markdown' was built under R version 4.0.5
```

1. Run a line of code which tests the packages are installed using Stackoverflow instructions posted here.

```
list.of.packages <- c("tidyverse", "markdown", "dslabs")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
```

Put the output in your problem set. This lets us know which packages successfully installed and which ones didn't 1. What is your github id?

My github id is guccimane457

1. Add and commit your code. Push it to github with commit message "start-up completed"
2. Now we'll practice reverting.
    a. Add the following text to you homework: "Why did the code on Github delete tindr?"
    b. Now push the code to Github.
    c. Now revert to the previous state of the code. (Now that the code is uncommitted, maybe it'll join tindr again.)

# 2 R for Data Science Exercises

## 2.1 First Steps (10 points)

Load `tidyverse` and `dslabs`

```
library(tidyverse)
library(dslabs)
```

We will be using `polls_us_election_2016` data that is in the `dslabs` package

1. How many rows are there in `polls_us_election_2016`? How many columns? What do the rows represent? How about the columns

```
glimpse(polls_us_election_2016)
```

```
## Rows: 4,208
## Columns: 15
## $ state            <fct> U.S., U.S., U.S., U.S., U.S., U.S., U.S., U.S., Ne...
## $ startdate        <date> 2016-11-03, 2016-11-01, 2016-11-02, 2016-11-04, 2...
## $ enddate          <date> 2016-11-06, 2016-11-07, 2016-11-06, 2016-11-07, 2...
## $ pollster         <fct> ABC News/Washington Post, Google Consumer Surveys,...
```

```
## $ grade            <fct> A+, B, A-, B, B-, A, A-, A-, NA, A-, A+, A-, A+, B...
## $ samplesize       <int> 2220, 26574, 2195, 3677, 16639, 1295, 1426, 1282, ...
## $ population       <chr> "lv", "lv", "lv", "lv", "rv", "lv", "lv", "lv", "l...
## $ rawpoll_clinton  <dbl> 47.00, 38.03, 42.00, 45.00, 47.00, 48.00, 45.00, 4...
## $ rawpoll_trump    <dbl> 43.00, 35.69, 39.00, 41.00, 43.00, 44.00, 41.00, 4...
## $ rawpoll_johnson  <dbl> 4.00, 5.46, 6.00, 5.00, 3.00, 3.00, 5.00, 6.00, 6....
## $ rawpoll_mcmullin <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ adjpoll_clinton  <dbl> 45.20163, 43.34557, 42.02638, 45.65676, 46.84089, ...
## $ adjpoll_trump    <dbl> 41.72430, 41.21439, 38.81620, 40.92004, 42.33184, ...
## $ adjpoll_johnson  <dbl> 4.626221, 5.175792, 6.844734, 6.069454, 3.726098, ...
## $ adjpoll_mcmullin <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
```

```
# conversely, you can also use ncol() and nrow() to find out how many rows and
# columns there are.

view(polls_us_election_2016)

summary(polls_us_election_2016)
```

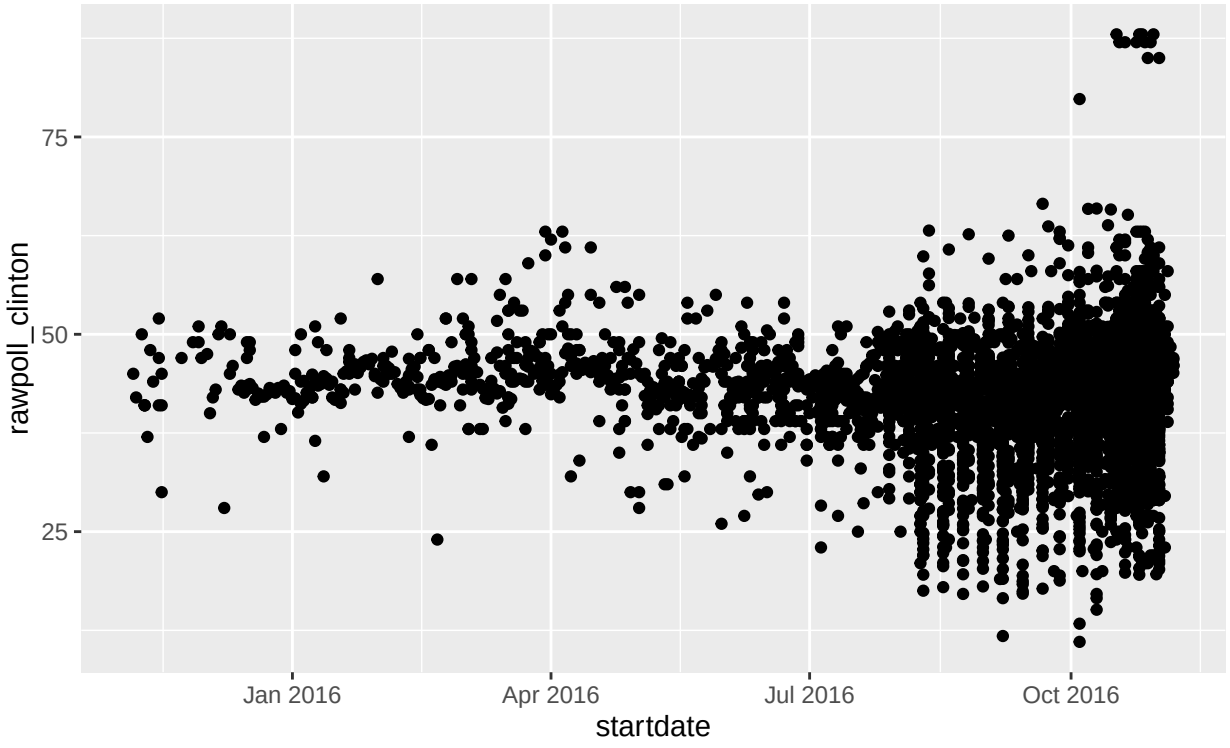
```
##             state        startdate             enddate
##  U.S.          :1106   Min.   :2015-11-06   Min.   :2015-11-08
##  Florida       : 148   1st Qu.:2016-08-10   1st Qu.:2016-08-21
##  North Carolina: 125   Median :2016-09-23   Median :2016-09-30
##  Pennsylvania  : 125   Mean   :2016-08-31   Mean   :2016-09-06
##  Ohio          : 115   3rd Qu.:2016-10-20   3rd Qu.:2016-10-28
##  New Hampshire : 112   Max.   :2016-11-06   Max.   :2016-11-07
##  (Other)       :2477
##                                      pollster        grade
##  Ipsos                                   : 919   A-     :1085
##  Google Consumer Surveys                 : 743   B      :1011
##  SurveyMonkey                            : 660   C-     : 693
##  YouGov                                  : 130   C+     : 329
##  Rasmussen Reports/Pulse Opinion Research: 125   B+     : 204
##  USC Dornsife/LA Times                   : 121   (Other): 457
##  (Other)                                 :1510   NA's   : 429
##    samplesize       population        rawpoll_clinton rawpoll_trump
##  Min.   :   35.0   Length:4208        Min.   :11.04   Min.   : 4.00
##  1st Qu.:  447.5   Class :character   1st Qu.:38.00   1st Qu.:35.00
##  Median :  772.0   Mode  :character   Median :43.00   Median :40.00
##  Mean   : 1148.2                      Mean   :41.99   Mean   :39.83
##  3rd Qu.: 1236.5                      3rd Qu.:46.20   3rd Qu.:45.00
##  Max.   :84292.0                      Max.   :88.00   Max.   :68.00
##  NA's   :1
##  rawpoll_johnson  rawpoll_mcmullin adjpoll_clinton adjpoll_trump
##  Min.   : 0.000   Min.   : 9.0     Min.   :17.06   Min.   : 4.373
##  1st Qu.: 5.400   1st Qu.:22.5     1st Qu.:40.21   1st Qu.:38.429
##  Median : 7.000   Median :25.0     Median :44.15   Median :42.765
##  Mean   : 7.382   Mean   :24.0     Mean   :43.32   Mean   :42.674
##  3rd Qu.: 9.000   3rd Qu.:27.9     3rd Qu.:46.92   3rd Qu.:46.290
##  Max.   :25.000   Max.   :31.0     Max.   :86.77   Max.   :72.433
##  NA's   :1409     NA's   :4178
##  adjpoll_johnson  adjpoll_mcmullin
##  Min.   :-3.668   Min.   :11.03
##  1st Qu.: 3.145   1st Qu.:23.11
```

```
##  Median : 4.384   Median :25.14
##  Mean   : 4.660   Mean   :24.51
##  3rd Qu.: 5.756   3rd Qu.:27.98
##  Max.   :20.367   Max.   :31.57
##  NA's   :1409     NA's   :4178
```

```
# there are 4208 rows in 'polls_us_election_2016'
# there are 15 columns in 'polls_us_election_2016'

# the rows or observations represent individual polls and their characteristics
# in the U.S. / U.S. States in 2016.

# the columns represent the variables included in the dataset.
# For example, grade and population.
```

1. Make a scatter-plot of startdate vs rawpoll_clinton

```
library(ggplot2)
qplot(x = 'startdate', y = 'rawpoll_clinton', data = polls_us_election_2016)
```




1. What does the variable grade describes? Use the help '?polls_us_election_2016 to find out.

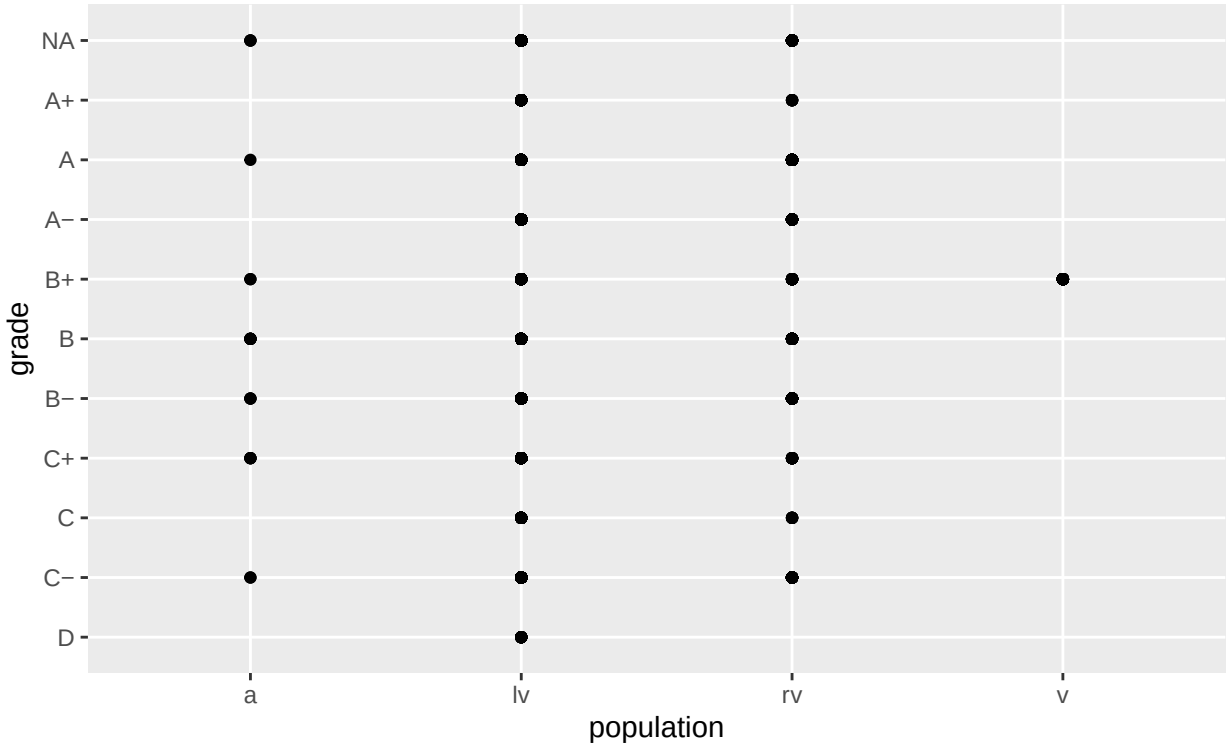
```
?polls_us_election_2016
```

```
## starting httpd help server ... done
```

```
# According to R documentation, the variable 'grade' describes Grade assigned by
# fivethirtyeight to the pollster.
```

1. What happens if you make a scatter-plot of population and grade? Why is the plot not useful?

```
library(ggplot2)
qplot(x = 'population', y = 'grade', data = polls_us_election_2016)
```




```
# this scatter plot of population and grade is not useful because we cannot
# observe the distribution for either variable across pollsters, or some period
# range. These variables are dependent on pollsters, and they characterize
# pollsters. To observe the distribution, we must match these variables with an
# independent variable (for example, startdate and pollster)
```

## 2.2 Grammar of graphics: mapping data to aesthetics (20 points)

1. Run `?polls_us_election_2016` to see the documentation for the data set. Run `head(polls_us_election_2016)` to see the first 6 rows of this data frame. Run `colnames(polls_us_election_2016)` to inspect which variables we have for each poll

```
?polls_us_election_2016

head(polls_us_election_2016)
```

```
##   state  startdate    enddate
```

```
## 1  U.S. 2016-11-03 2016-11-06
## 2  U.S. 2016-11-01 2016-11-07
## 3  U.S. 2016-11-02 2016-11-06
## 4  U.S. 2016-11-04 2016-11-07
## 5  U.S. 2016-11-03 2016-11-06
## 6  U.S. 2016-11-03 2016-11-06
##                                                 pollster grade samplesize
## 1                              ABC News/Washington Post    A+       2220
## 2                               Google Consumer Surveys     B      26574
## 3                                                 Ipsos    A-       2195
## 4                                               YouGov     B       3677
## 5                                        Gravis Marketing    B-      16639
## 6 Fox News/Anderson Robbins Research/Shaw & Company Research     A       1295
##   population rawpoll_clinton rawpoll_trump rawpoll_johnson rawpoll_mcmullin
## 1         lv           47.00         43.00            4.00               NA
## 2         lv           38.03         35.69            5.46               NA
## 3         lv           42.00         39.00            6.00               NA
## 4         lv           45.00         41.00            5.00               NA
## 5         rv           47.00         43.00            3.00               NA
## 6         lv           48.00         44.00            3.00               NA
##   adjpoll_clinton adjpoll_trump adjpoll_johnson adjpoll_mcmullin
## 1        45.20163      41.72430        4.626221               NA
## 2        43.34557      41.21439        5.175792               NA
## 3        42.02638      38.81620        6.844734               NA
## 4        45.65676      40.92004        6.069454               NA
## 5        46.84089      42.33184        3.726098               NA
## 6        49.02208      43.95631        3.057876               NA
```

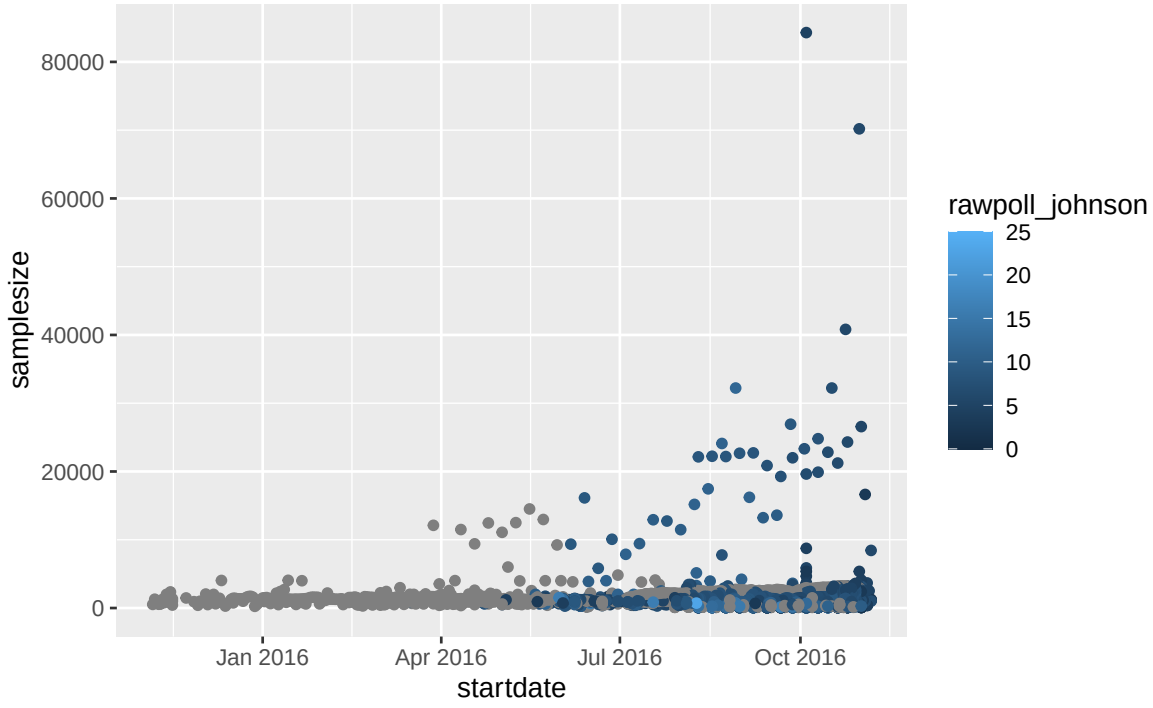
```r
colnames(polls_us_election_2016)
```

```
##  [1] "state"            "startdate"        "enddate"          "pollster"
##  [5] "grade"            "samplesize"       "population"       "rawpoll_clinton"
##  [9] "rawpoll_trump"    "rawpoll_johnson"  "rawpoll_mcmullin" "adjpoll_clinton"
## [13] "adjpoll_trump"    "adjpoll_johnson"  "adjpoll_mcmullin"
```

1. Compare the following scatter-plots. Why are the two graphs different? Which graph is better representation of the data? (*You do not need to graph then to answer these questions.*)

```r
# Graph 1
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = samplesize,
                           color = rawpoll_johnson))
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```




```
# Graph 2
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = samplesize,
                           color = as.character(rawpoll_johnson)))
```

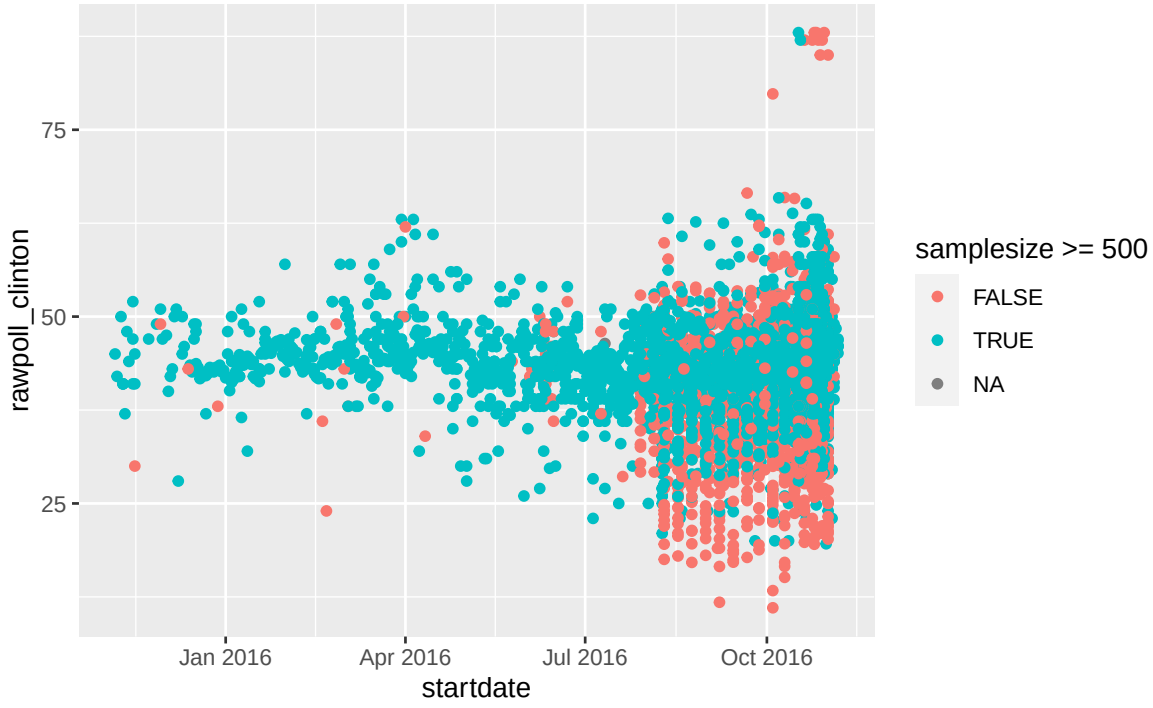
```
## Warning: Removed 1 rows containing missing values (geom_point).
```

| | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | ● | 3.37 | ● | 3.96 | ● | 4.4 | ● | 4.8 | ● | 5.12 | ● | 5.46 | ● | 5.74 | ● | 6.06 | ● | 6.36 | ● | 6.61 |
| 0 | ● | 3.4 | ● | 3.97 | ● | 4.46 | ● | 4.84 | ● | 5.14 | ● | 5.47 | ● | 5.77 | ● | 6.07 | ● | 6.37 | ● | 6.62 |
| | ● | 3.5 | ● | 3.98 | ● | 4.48 | ● | 4.85 | ● | 5.15 | ● | 5.48 | ● | 5.78 | ● | 6.08 | ● | 6.38 | ● | 6.65 |
| 5 | ● | 3.51 | ● | 3.99 | ● | 4.5 | ● | 4.86 | ● | 5.16 | ● | 5.49 | ● | 5.8 | ● | 6.1 | ● | 6.39 | ● | 6.66 |
| | ● | 3.53 | ● | 4 | ● | 4.54 | ● | 4.87 | ● | 5.17 | ● | 5.5 | ● | 5.82 | ● | 6.13 | ● | 6.4 | ● | 6.67 |
| 47 | ● | 3.58 | ● | 4.03 | ● | 4.55 | ● | 4.88 | ● | 5.19 | ● | 5.52 | ● | 5.83 | ● | 6.15 | ● | 6.42 | ● | 6.68 |
| | ● | 3.59 | ● | 4.04 | ● | 4.56 | ● | 4.89 | ● | 5.2 | ● | 5.53 | ● | 5.85 | ● | 6.17 | ● | 6.43 | ● | 6.69 |
| | ● | 3.6 | ● | 4.05 | ● | 4.57 | ● | 4.9 | ● | 5.21 | ● | 5.54 | ● | 5.86 | ● | 6.18 | ● | 6.46 | ● | 6.7 |
| | ● | 3.64 | ● | 4.07 | ● | 4.59 | ● | 4.92 | ● | 5.24 | ● | 5.57 | ● | 5.87 | ● | 6.19 | ● | 6.47 | ● | 6.71 |
| | ● | 3.69 | ● | 4.1 | ● | 4.6 | ● | 4.93 | ● | 5.25 | ● | 5.6 | ● | 5.88 | ● | 6.2 | ● | 6.48 | ● | 6.75 |
| | ● | 3.7 | ● | 4.12 | ● | 4.64 | ● | 4.94 | ● | 5.26 | ● | 5.61 | ● | 5.89 | ● | 6.21 | ● | 6.49 | ● | 6.76 |
| | ● | 3.71 | ● | 4.14 | ● | 4.65 | ● | 4.95 | ● | 5.27 | ● | 5.62 | ● | 5.9 | ● | 6.22 | ● | 6.5 | ● | 6.77 |
| 1 | ● | 3.77 | ● | 4.15 | ● | 4.68 | ● | 4.98 | ● | 5.29 | ● | 5.64 | ● | 5.91 | ● | 6.23 | ● | 6.51 | ● | 6.79 |
| 4 | ● | 3.79 | ● | 4.17 | ● | 4.69 | ● | 5 | ● | 5.3 | ● | 5.65 | ● | 5.93 | ● | 6.24 | ● | 6.52 | ● | 6.8 |
| | ● | 3.8 | ● | 4.2 | ● | 4.7 | ● | 5.03 | ● | 5.31 | ● | 5.66 | ● | 5.97 | ● | 6.3 | ● | 6.54 | ● | 6.82 |
| . | ● | 3.81 | ● | 4.23 | ● | 4.72 | ● | 5.04 | ● | 5.33 | ● | 5.67 | ● | 5.99 | ● | 6.31 | ● | 6.55 | ● | 6.85 |
| | ● | 3.86 | ● | 4.27 | ● | 4.73 | ● | 5.07 | ● | 5.36 | ● | 5.69 | ● | 6 | ● | 6.32 | ● | 6.56 | ● | 6.86 |

```
# The two graphs differ in how they assign colors to the data points of
# rawpoll_johnson. Graph 1 assigns a blue color to rawpoll_johnson that gets
# lighter in color as value increases. Graph 2 assigns a new color to each
# unique 'character' in the variable rawpoll_johnson. Graph 1 is a better
# representation of the data because it shows the raw poll scores fit in a
# chart of start date versus sample size.
```

2. What happens if you map an aesthetic to something other than a variable name, like `aes(color = samplesize >= 500)`

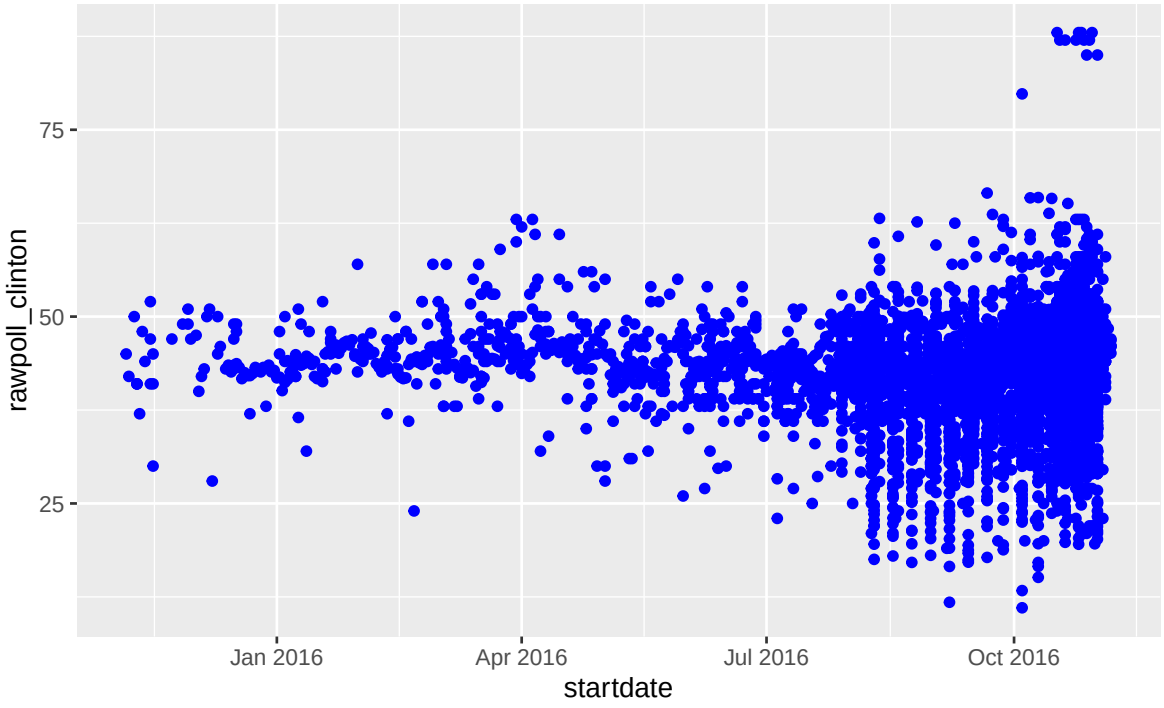
```
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = rawpoll_clinton,
                           color = samplesize >= 500))
```




```
# When an aesthetic is mapped to something other than a variable name, such
# as 'color = samplesize >= 500', you can differentiate the data by color
# which pollsters had sample size over a specific size. This means you can
# create a threshold and visually observe differences by color in the plot.
```

3. Common bugs: What's gone wrong with this code? Fix the code so the points are blue.

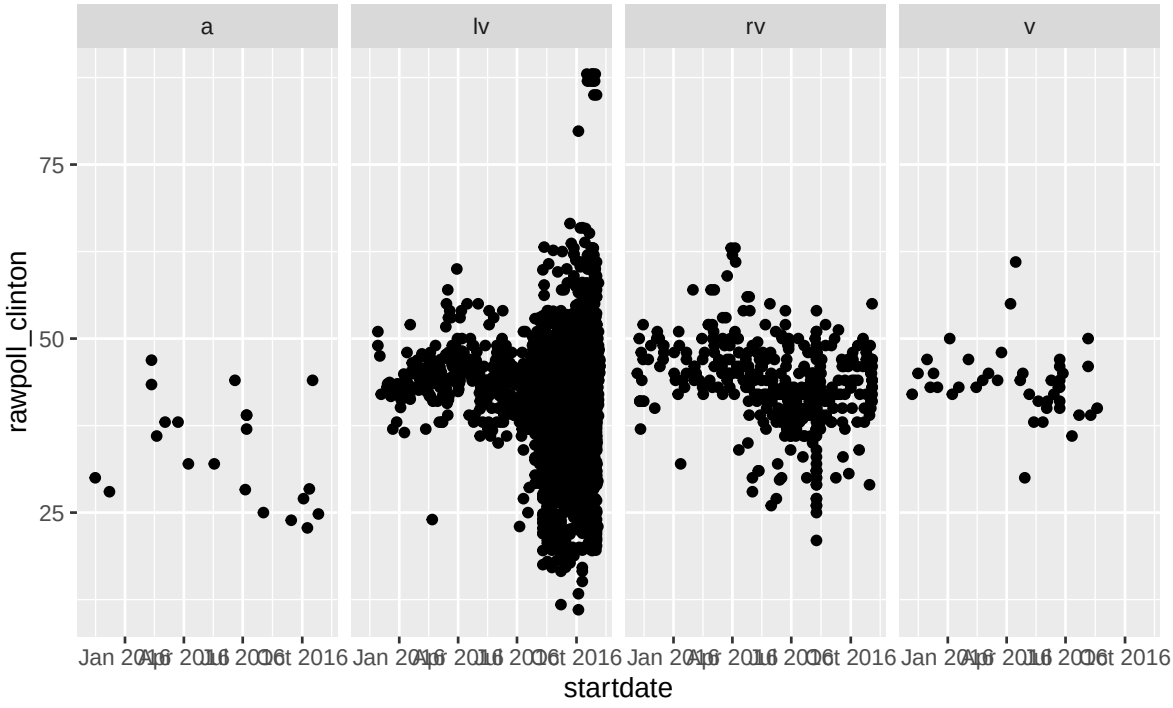
```
# ggplot(data = polls_us_election_2016) +
#  geom_point(mapping = aes(x = startdate,
#                           y = rawpoll_clinton,
#                           color = "blue"))

ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = rawpoll_clinton),
                           color = "blue")
```



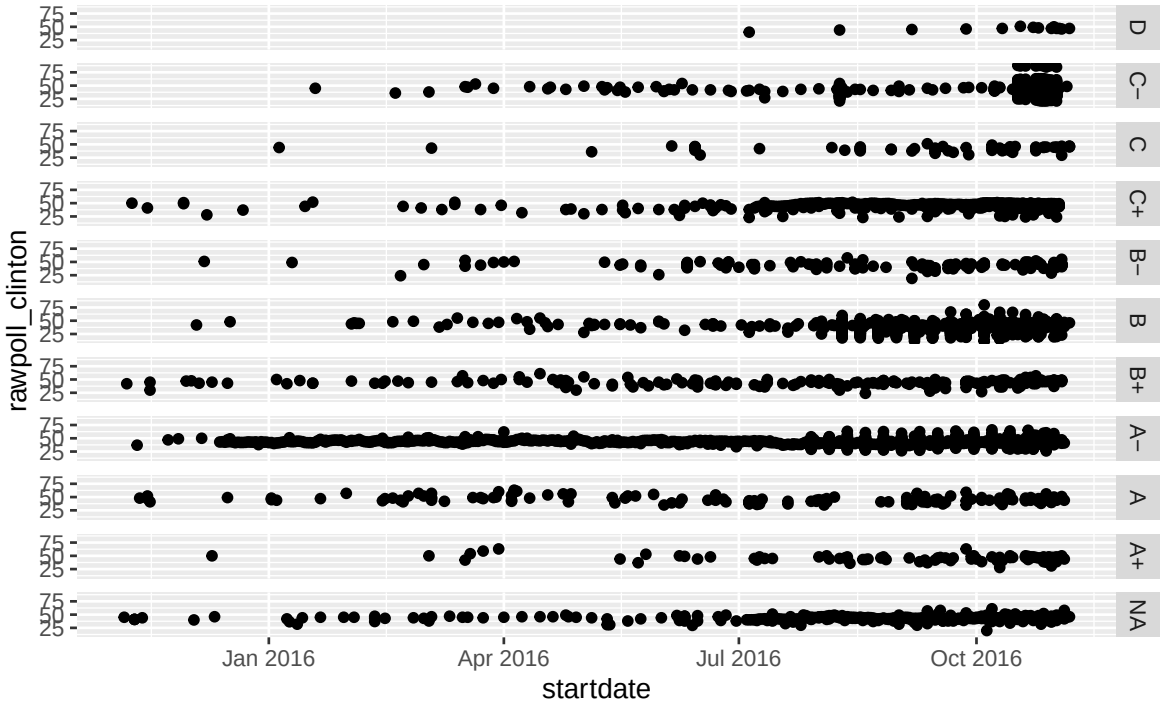

## 2.3 grammar of graphics: Facets (20 points)

1. Make the following plots. How does `facet_grid()` decide the layout of the grid?

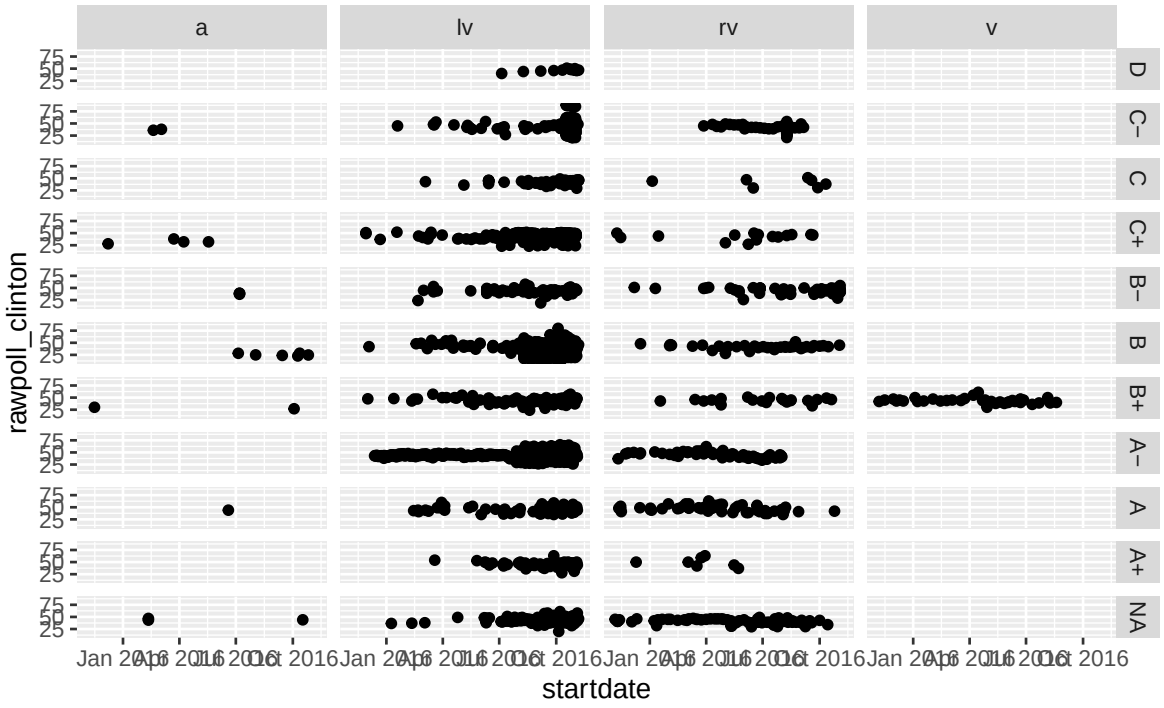
```
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = rawpoll_clinton)) +
  facet_grid(cols = vars(population))
```




```
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = rawpoll_clinton)) +
  facet_grid(rows = vars(grade))
```



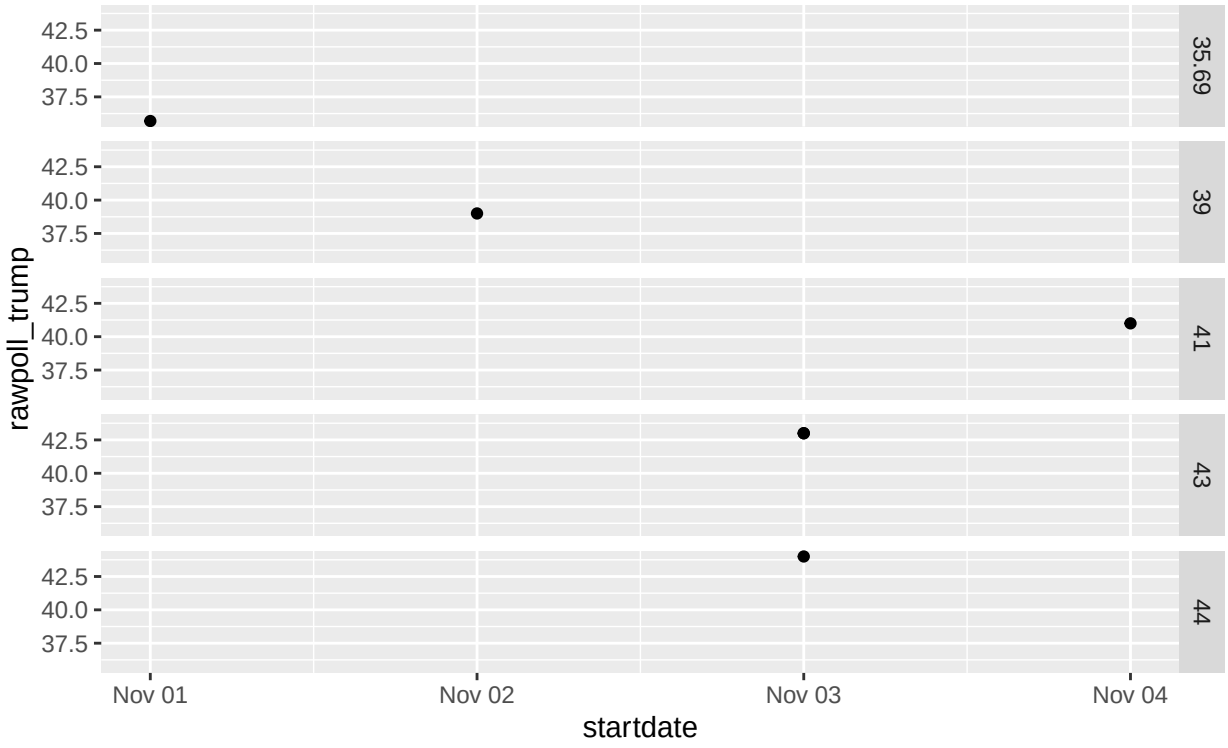
```r
ggplot(data = polls_us_election_2016) +
  geom_point(mapping = aes(x = startdate,
                           y = rawpoll_clinton)) +
  facet_grid(rows = vars(grade), cols = vars(population))
```




```r
# 'facet_grid()' decide the layout of the grid using vars(), which tells
# which variables to include/exclude in the plot as a column or row.
```

2. What happens if you facet a continuous variable? Provide an example

```r
# if you facet a continuous variable, it can create very many segments of
# rows/columns depending on what you assign in facet_grid(). The following code
# in comment is my example of faceting a continuous variable (not run due to
# crashing Rstudio)

#    ggplot(data = polls_us_election_2016) +
#      geom_point(mapping = aes(x = startdate,
#                               y = population)) +
#      facet_grid(rows = vars(rawpoll_trump))
#
# we adjust our code to only see a segment data to facet a continuous variable

facet_continuous_variable <- head(polls_us_election_2016)

    ggplot(data = facet_continuous_variable) +
      geom_point(mapping = aes(x = startdate,
                               y = rawpoll_trump)) +
      facet_grid(rows = vars(rawpoll_trump))
```

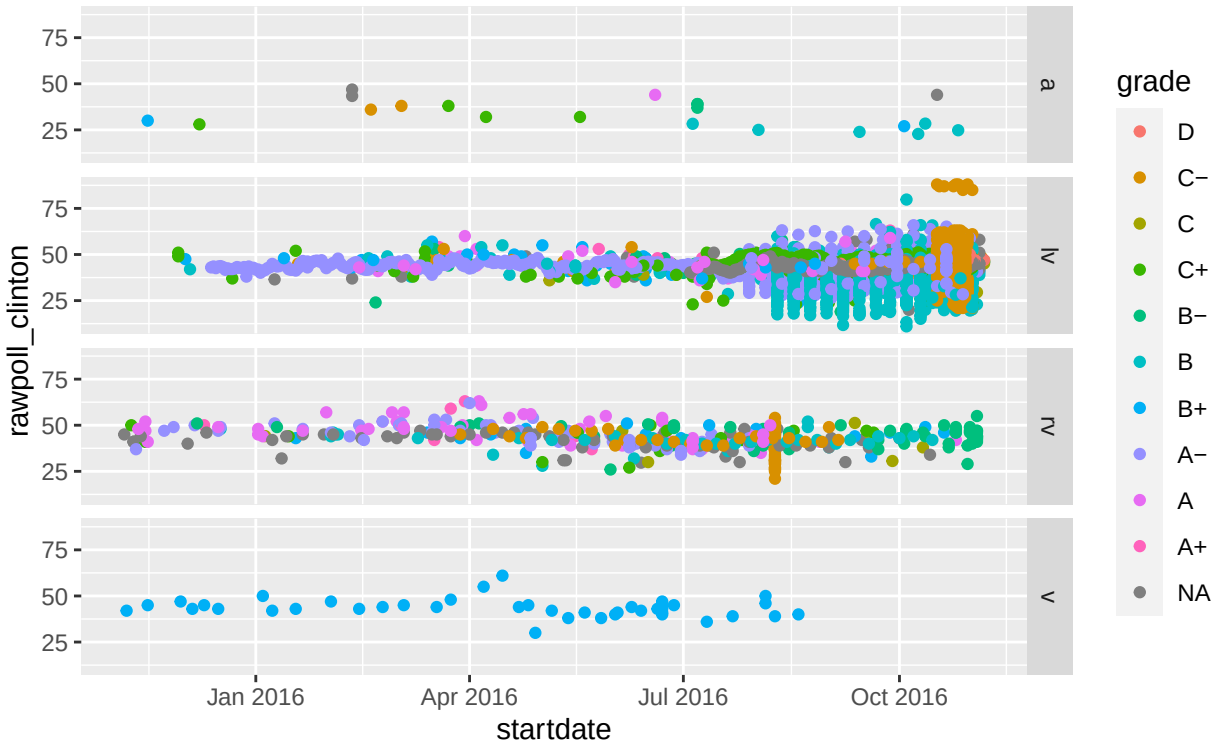



1. Reproduce the following graph

```r
ggplot(data = polls_us_election_2016) +
  geom_point(aes(x = startdate,
                 y = rawpoll_trump,
                 color = grade)) +
  facet_grid(rows = vars(population))
```

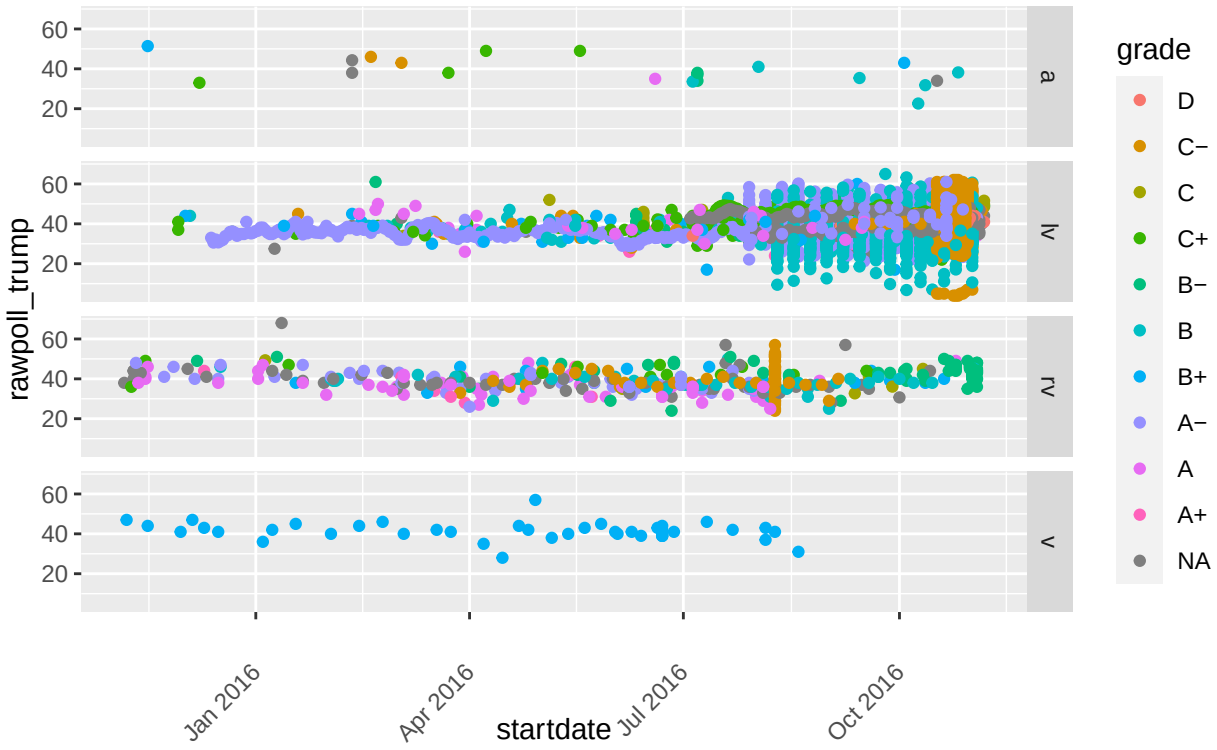
```
![](pset_template_12285784_files/figure-latex/unnamed-chunk-14-1.pdf)<!-- -->
```

```r
# Similarly, the graph produced from the above code can also be produced for
    # Clinton and other candidates

        ggplot(data = polls_us_election_2016) +
      geom_point(aes(x = startdate,
                     y = rawpoll_clinton,
                     color = grade)) +
      facet_grid(rows = vars(population))
```




1. Rotate 45 degrees the `startdate` labels from the previous plot. You can use Google, include `ggplot` in your search to get more relevant answers. Remember to cite any code you gather from the internet.

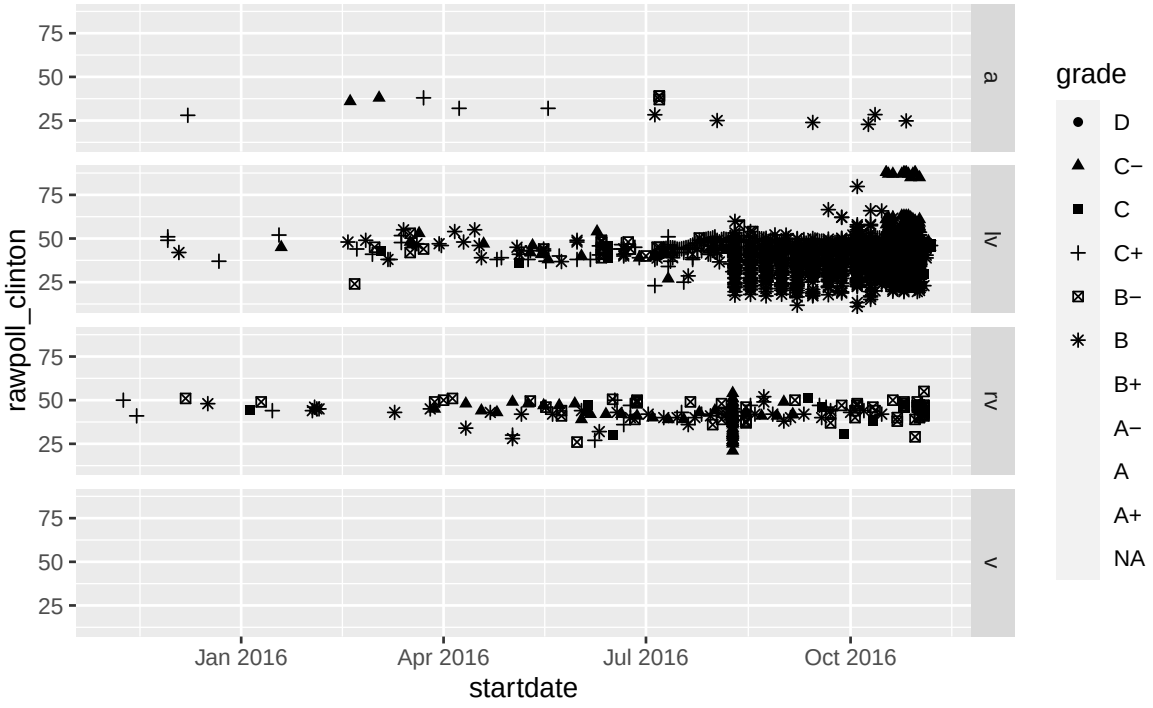
```
#our goal is to rotate x axis labels 45 degrees
#code_source:
#https://stackoverflow.com/questions/1330989/rotating-and-spacing-axis-labels-in-ggplot2

    ggplot(data = polls_us_election_2016) +
      geom_point(aes(x = startdate,
                     y = rawpoll_trump,
                     color = grade)) +
      facet_grid(rows = vars(population)) + theme(
        axis.text.x = element_text(angle = 45, vjust = 0.5, hjust=1))
```




1. Reproduce the following graph. Why are there grades missing?

```
ggplot(data = polls_us_election_2016) +
  geom_point(aes(x = startdate,
                 y = rawpoll_clinton,
                 shape = grade)) +
  facet_grid(rows = vars(population))
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 10. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 1961 rows containing missing values (geom_point).
```




```
# There appears to be limited number of shapes stored in R, less than the
# total number of grades.
```

## 2.4 Grammar of graphics: geoms (10 pts)

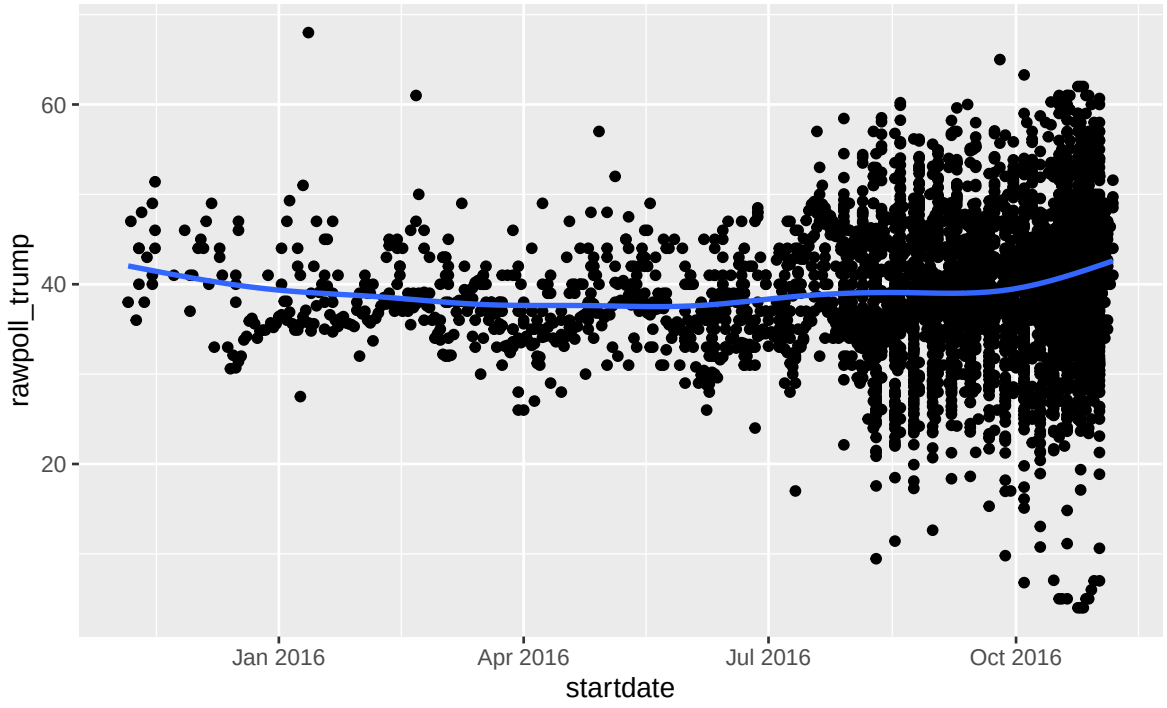
1. What geom would you use to draw a line chart? A boxplot? A histogram? An area chart?

```
# to draw a line chart, use geom_line()
# to draw a boxplot, use geom_boxplot()
# to draw a histogram, use geom_histogram()
# to draw an area chart, use geom_area()
```

1. Will these two graphs look different? Why/why not?

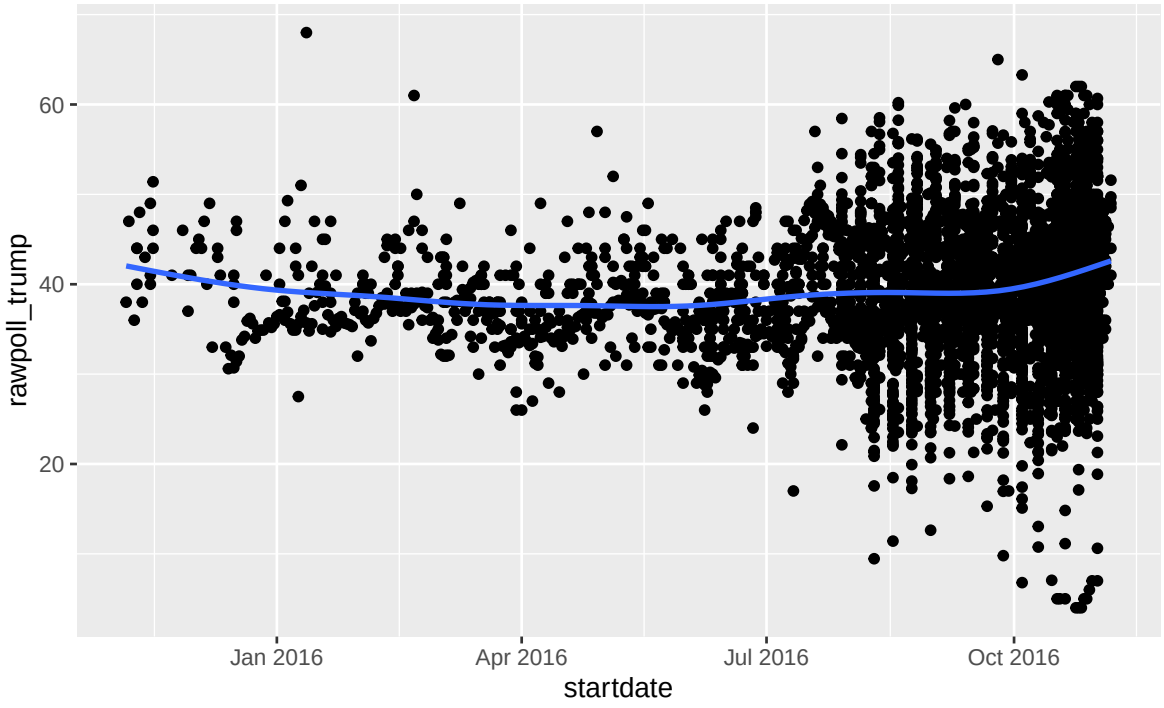
```
ggplot(data = polls_us_election_2016,
       mapping = aes(x = startdate,
                     y = rawpoll_trump)) +
  geom_point() +
  geom_smooth(se=FALSE)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
ggplot() +
  geom_point(data = polls_us_election_2016,
             mapping = aes(x = startdate,
                           y = rawpoll_trump)) +
  geom_smooth(data = polls_us_election_2016,
              mapping = aes(x = startdate,
                            y = rawpoll_trump), se=FALSE)
```

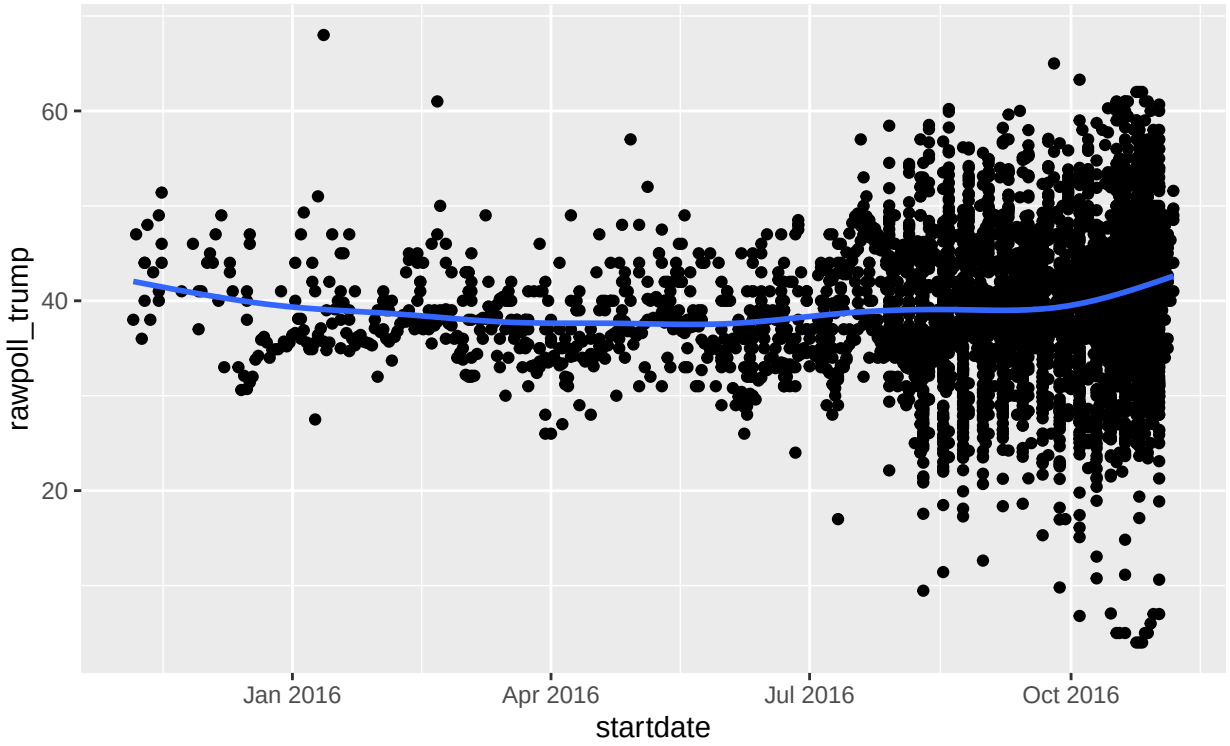
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
# these two graphs are identical even if their codes are slightly different.
# Duplicative code can be removed to achieve the same plot. In this case,
# data = polls_us_election_2016,
#              mapping = aes(x = startdate,
#                            y = rawpoll_trump
# is duplicative and only has to be coded into one of the following to work:
# ggplot(), geom_point(), and geom_smooth().
```

```
# Below is the previous code with the data syntax entered into all three
# functions, producing the identical plot still.

 ggplot(data = polls_us_election_2016,
                mapping = aes(x = startdate,
                              y = rawpoll_trump)) +
      geom_point(data = polls_us_election_2016,
                mapping = aes(x = startdate,
                              y = rawpoll_trump)) +
      geom_smooth(data = polls_us_election_2016,
                  mapping = aes(x = startdate,
                                y = rawpoll_trump), se=FALSE)
```

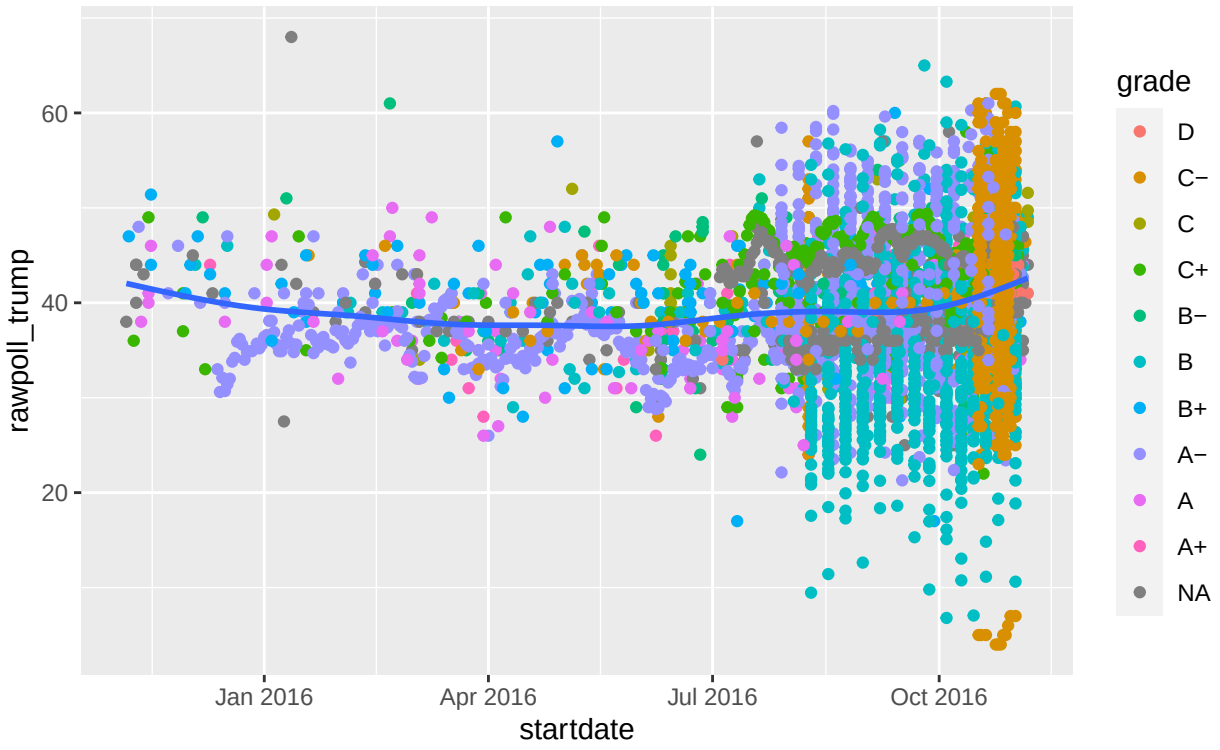
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




1. You are trying to figure out if there is a relationship between Trump poll numbers and the quality of the polls. Write code to make this graph.

```
    ggplot(data = polls_us_election_2016,
           mapping = aes(x = startdate,
                         y = rawpoll_trump)) +
        geom_point(aes(color = grade)) +
        geom_smooth(se = FALSE)
```

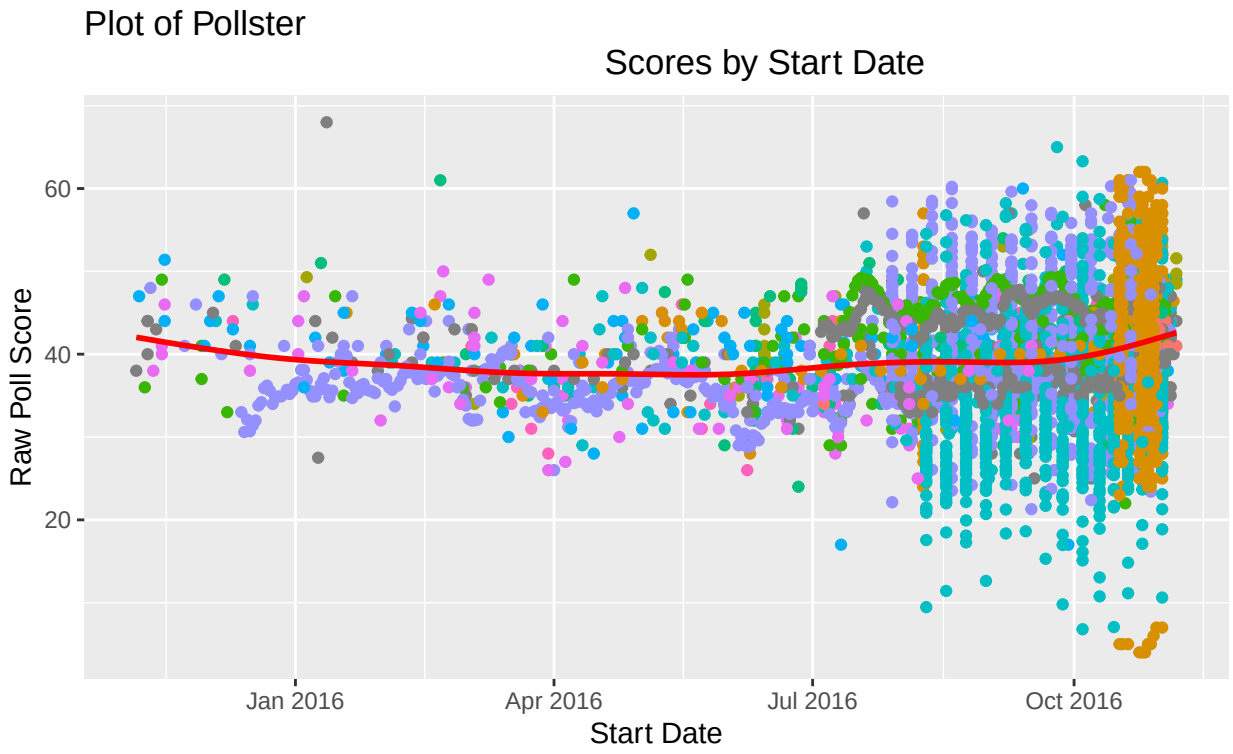
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




1. Make some changes to this graph:
   - make line red
   - make the x- and y-axes labels more informative using `+ labs()`
   - use an informative title
   - remove the legend (Google might be helpful to learn how) Are all four changes improvements? Which change made the plot worse and why?

```
ggplot(data = polls_us_election_2016,
       mapping = aes(x = startdate,
                     y = rawpoll_trump)) +
    geom_point(aes(color = grade), show.legend = FALSE) +
    geom_smooth(se = FALSE, color = "Red") + labs(title="Plot of Pollster
                                                  Scores by Start Date",
    x ="Start Date", y = "Raw Poll Score")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

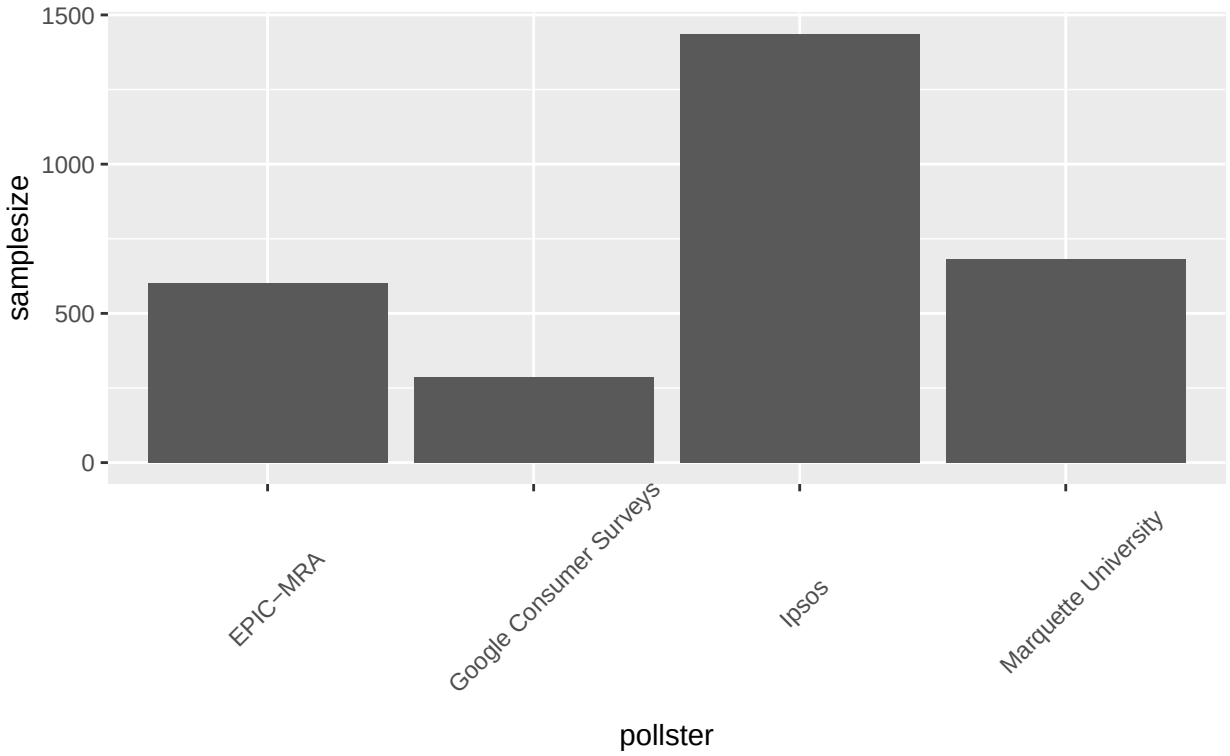



```
# All changes except removal of the legend I think are improvements to make the
# data and chart clearer.
# Removing the legend was not an improvement because by removing it we lose the
# association of each color to each grade.
```

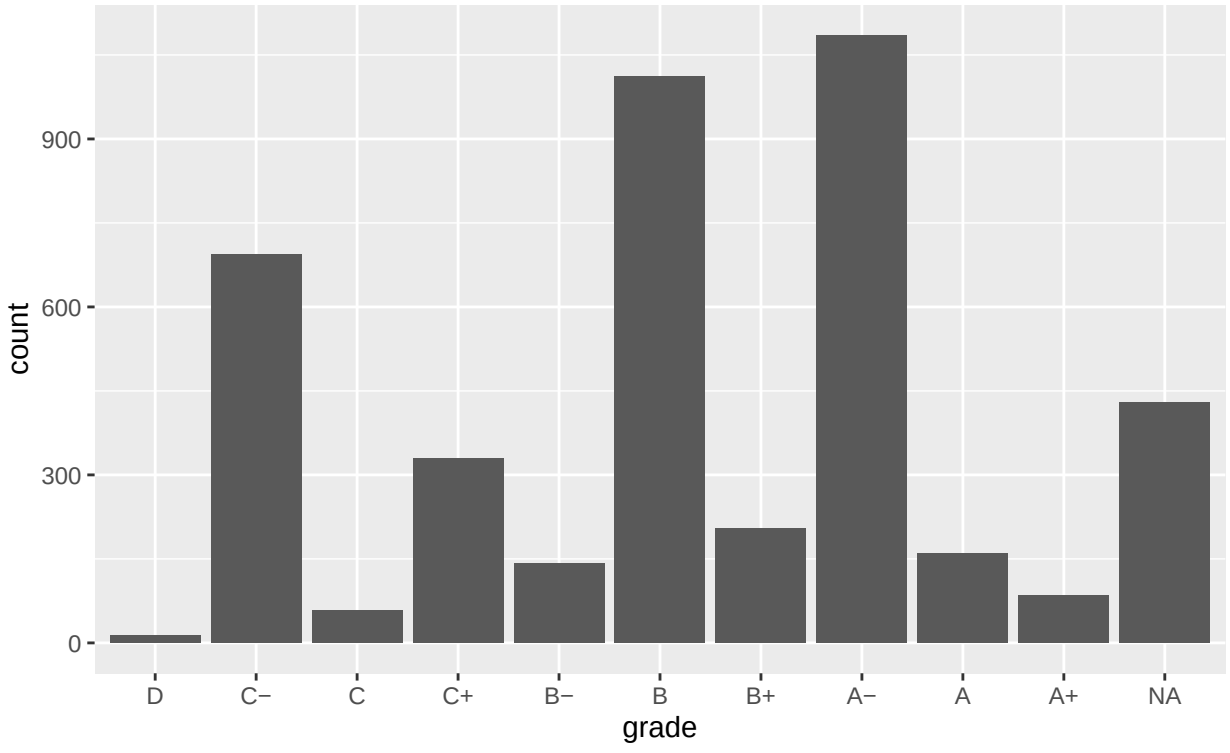
### 2.4.1 grammar of graphics: Statistical transformations (10 pts)

1. What does `geom_col()` do? How is it different from `geom_bar()`?

```
#link: https://ggplot2.tidyverse.org/reference/geom_bar.html
?geom_bar()
?geom_col()

# geom_col and geom_bar are two types of bar charts in R. In my own words,
# with geom_col you can plot values of any x variable against any y variable.
# With geom_bar, you can easily plot the total count of each unique observation
# for a variable (or the number of cases at each x position). This is made
# possible by what each stat function each geom uses by default. By default,
# geom_bar uses stat_count and geom_col uses stat_identity.

example <- tail(polls_us_election_2016)

ggplot(data=example, aes(x=pollster, y = samplesize)) + geom_col()+
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5, hjust=0.4))
```

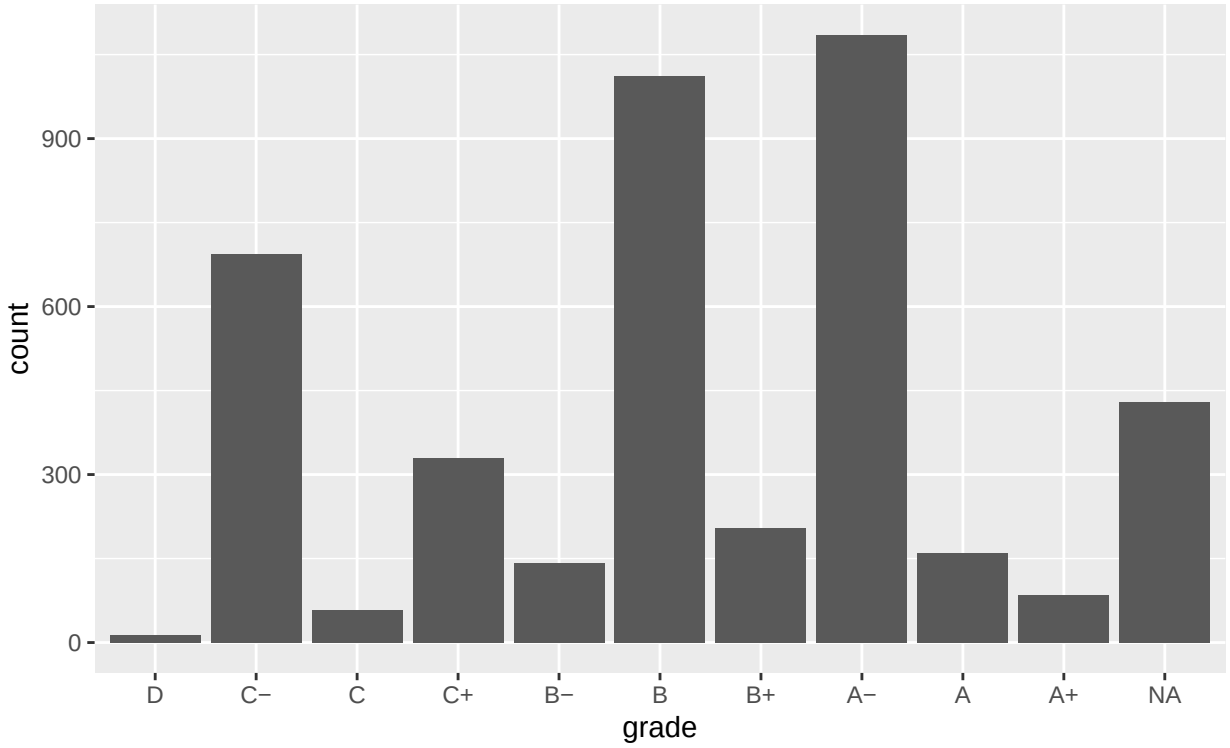



```
ggplot(data=polls_us_election_2016, aes(x=grade)) + geom_bar()
```

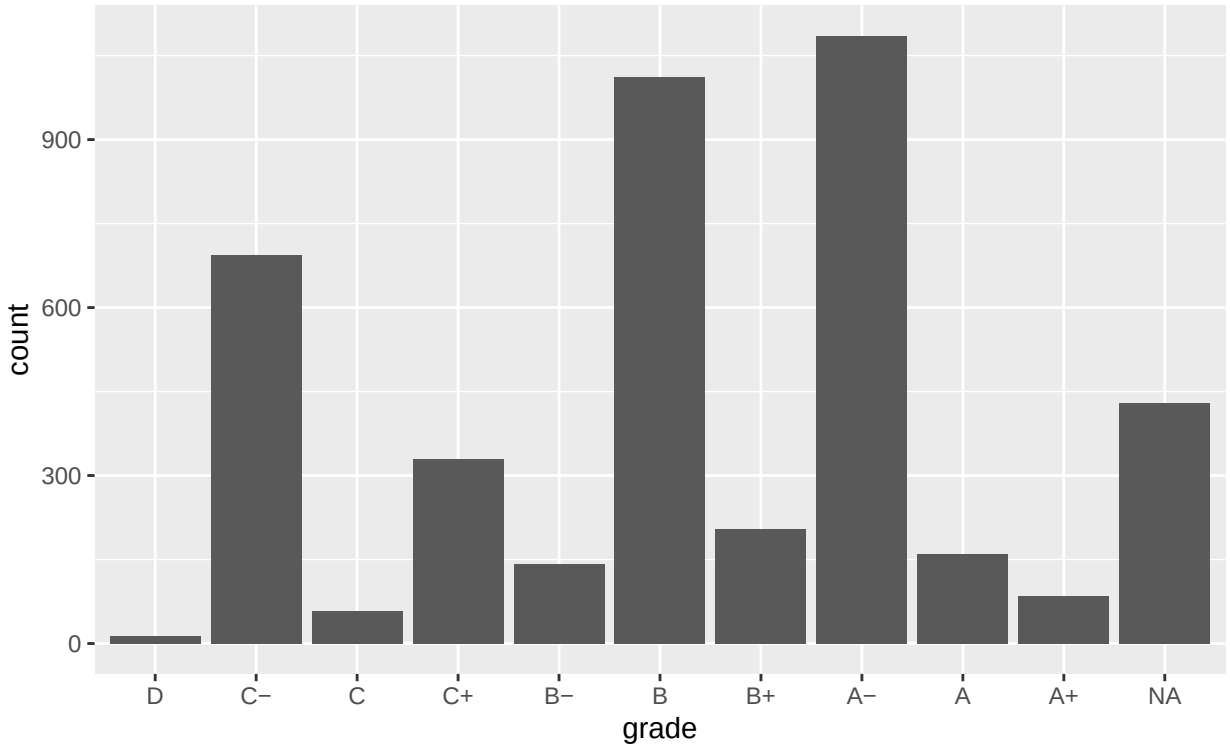



1. Plot `ggplot(data=polls_us_election_2016, aes(x=grade)) + geom_bar()`. Replace the `geom` with a `stat` to make the same graph.

```
ggplot(data=polls_us_election_2016, aes(x=grade)) + geom_bar()
```




```
ggplot(data=polls_us_election_2016, aes(x=grade)) + stat_count()
```

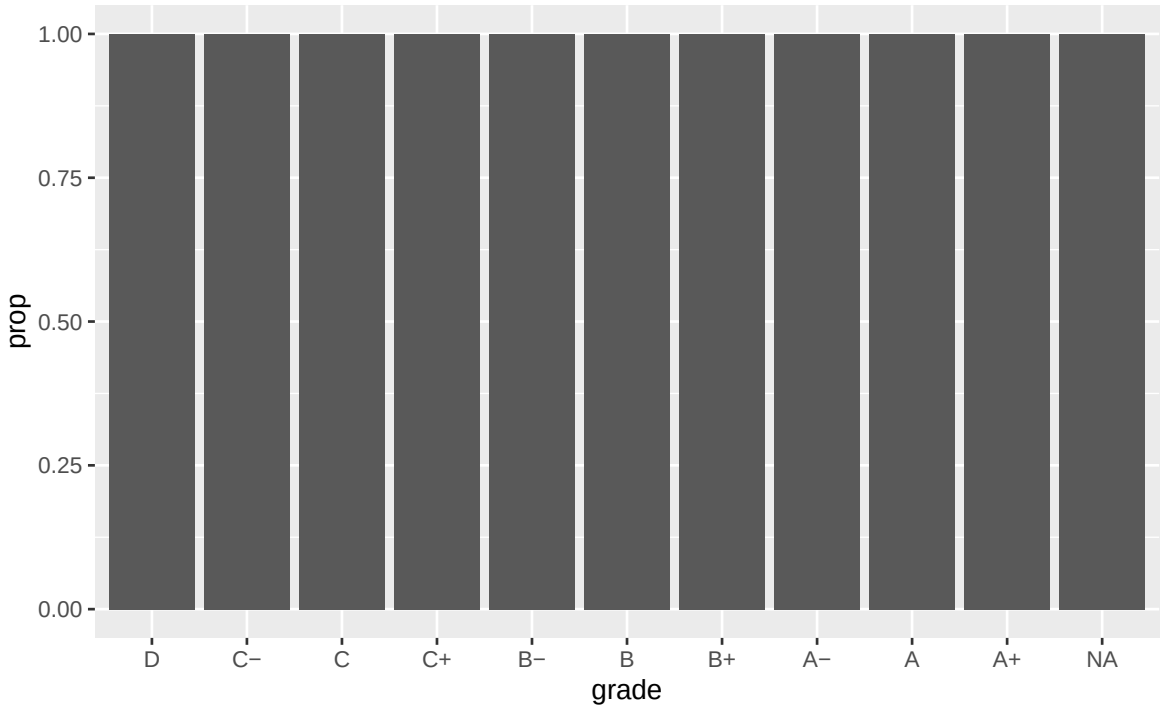


1. Which 4 variables does `stat_smooth()` compute? How are these variables displayed on a graph made with `geom_smooth()`? What parameters (i.e. inputs to the function) control its behavior?

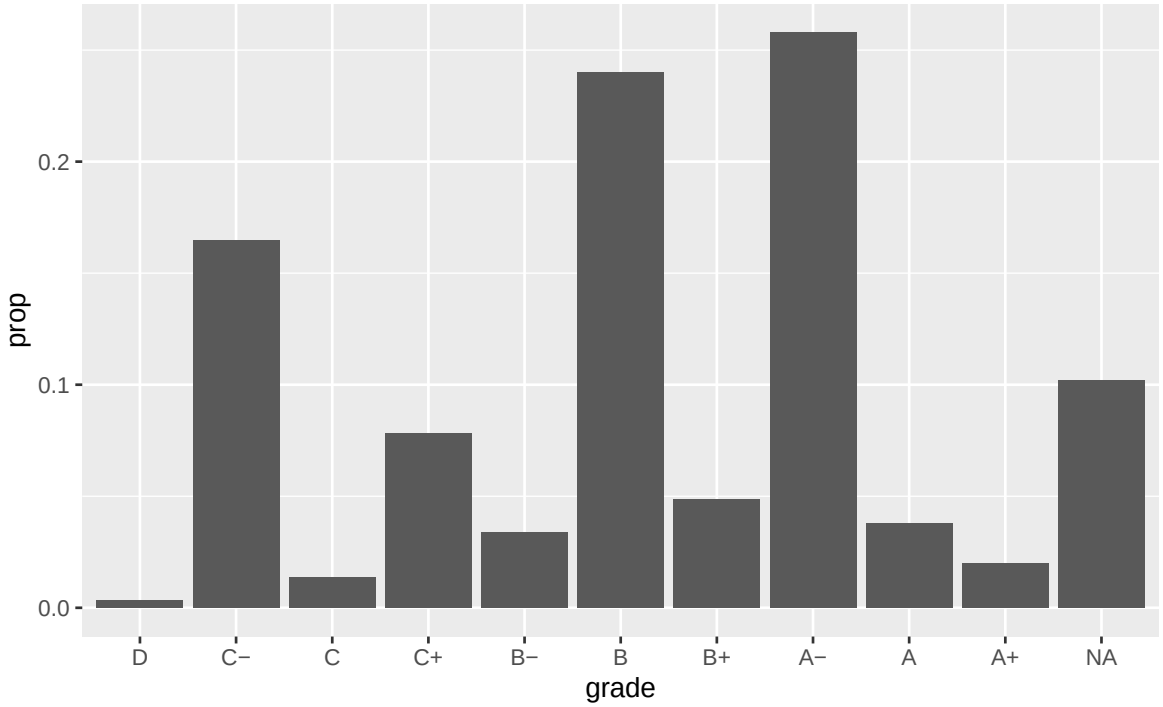
```
?stat_smooth()
?geom_smooth()

# Computed variables are the same for both functions.
## y = predicted value

## ymin = lower pointwise confidence interval around the mean

## ymax = upper pointwise confidence interval around the mean

## se = standard error

# If we want to display our results with a non-standard geom, we use
# stat_smooth().

# Their parameters (inputs) that control its behavior are shown in full below:

geom_smooth(
  mapping = NULL,
  data = NULL,
  stat = "smooth",
  position = "identity",
  ...,
  method = NULL,
  formula = NULL,
  se = TRUE,
  na.rm = FALSE,
  orientation = NA,
  show.legend = NA,
  inherit.aes = TRUE
)

stat_smooth(
  mapping = NULL,
  data = NULL,
  geom = "smooth",
  position = "identity",
  ...,
  method = NULL,
  formula = NULL,
  se = TRUE,
  n = 80,
  span = 0.75,
  fullrange = FALSE,
  level = 0.95,
  method.args = list(),
  na.rm = FALSE,
  orientation = NA,
  show.legend = NA,
  inherit.aes = TRUE
)
```

1. What is wrong with the following graph? Do we need to add `group = 1` to it? What denominator is ggplot using to determine proportions?

```
ggplot(data = polls_us_election_2016) +
  geom_bar(mapping = aes(x = grade, y = ..prop..))
```



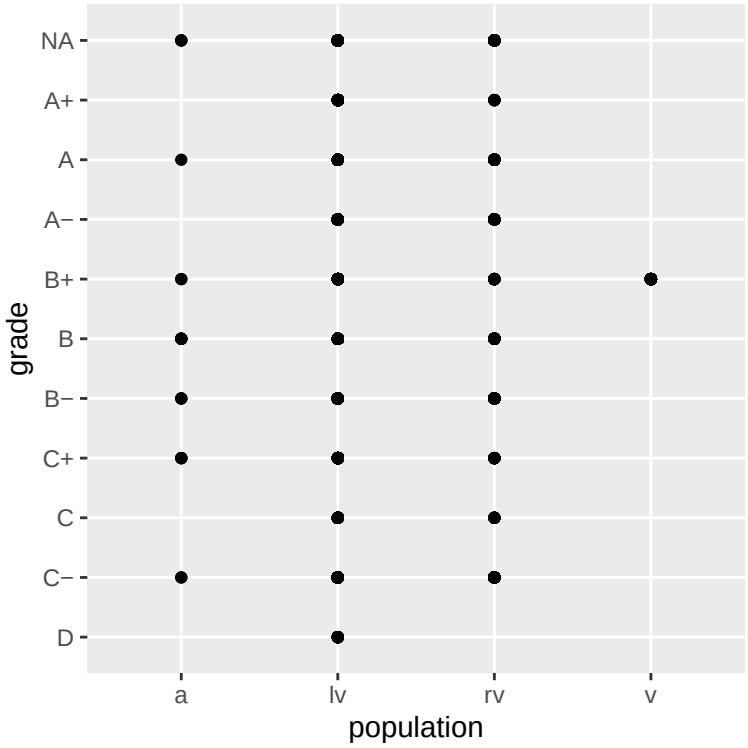

```
# the graph before adding 'group = 1' does not take into account each grades
# proportion in terms of the total dataset. It simply tells us proportion
# of each grade in terms of its own subtotal. We need to add 'group = 1', to
# observe proportion of each grade in relation to the total count of grades.

ggplot(data = polls_us_election_2016) +
  geom_bar(mapping = aes(x = grade, y = ..prop.., group = 1))
```



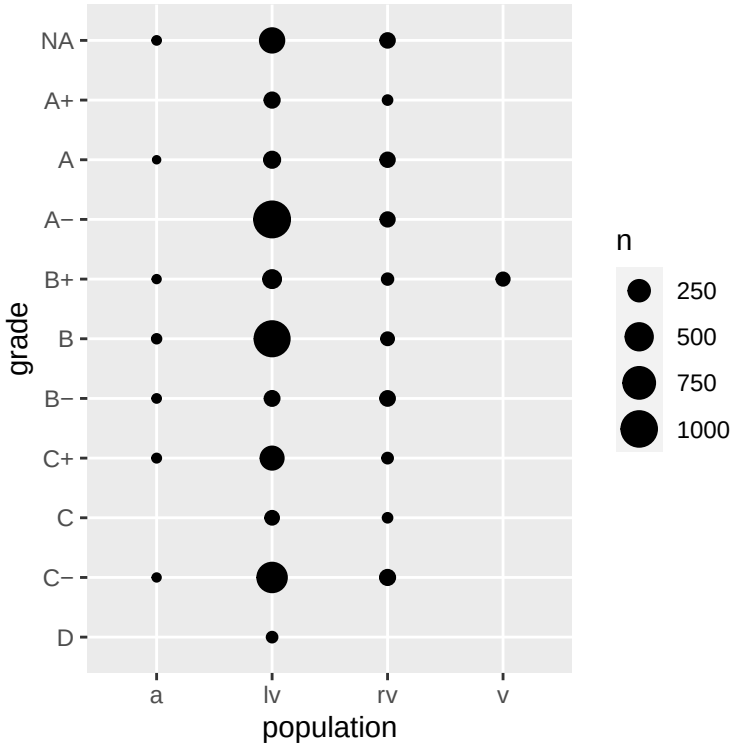

## grammar of graphics: Positional adjustments (5 pts)

2. What is the problem with this plot? How could you improve it? (You already made this plot in this Pset)

```
ggplot(data = polls_us_election_2016,
       mapping = aes(x = population, y = grade)) +
  geom_point()
```

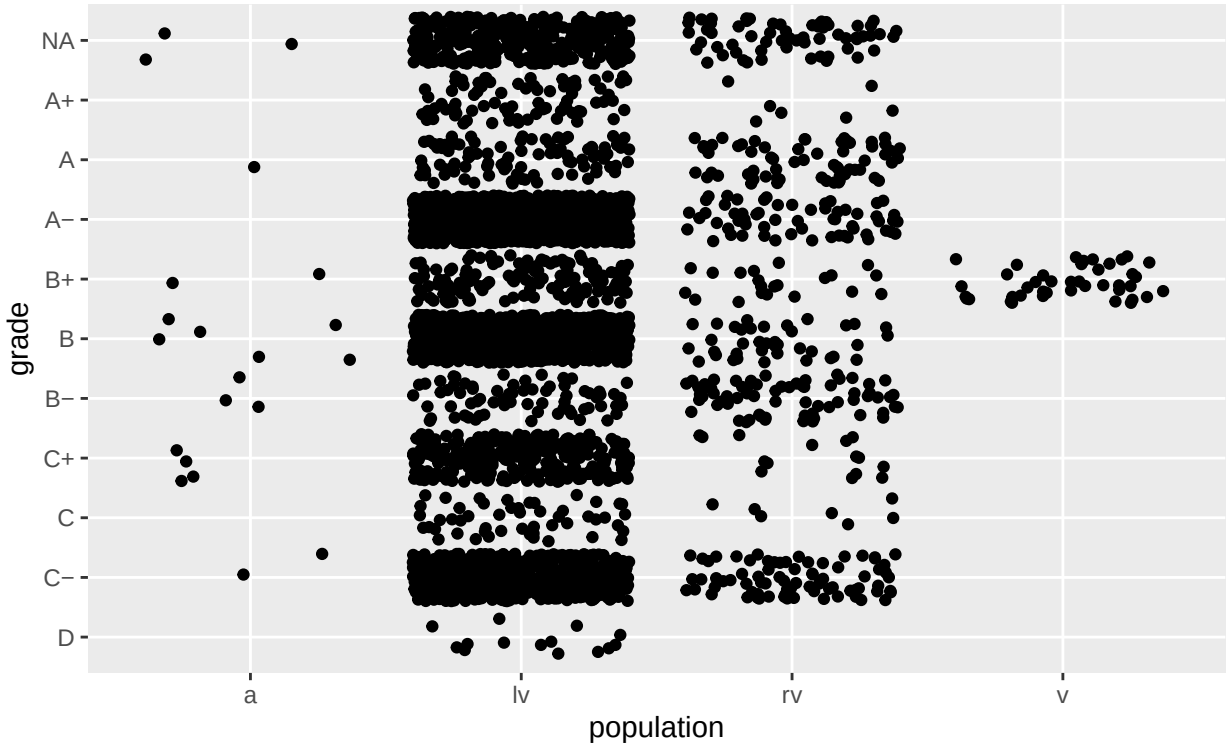



```
# The problem with the plot from above code is that geom_point cannot show
# overlapping points, or it cannot map the number of observations at each
# point location. To improve it, we can replace geom_point for geom_count.

    ggplot(data = polls_us_election_2016,
       mapping = aes(x = population, y = grade)) +
  geom_count()
```




3. Compare and contrast `geom_jitter()` with `geom_count()`. Use vocabulary from the "grammar of graphics" to support your argument.

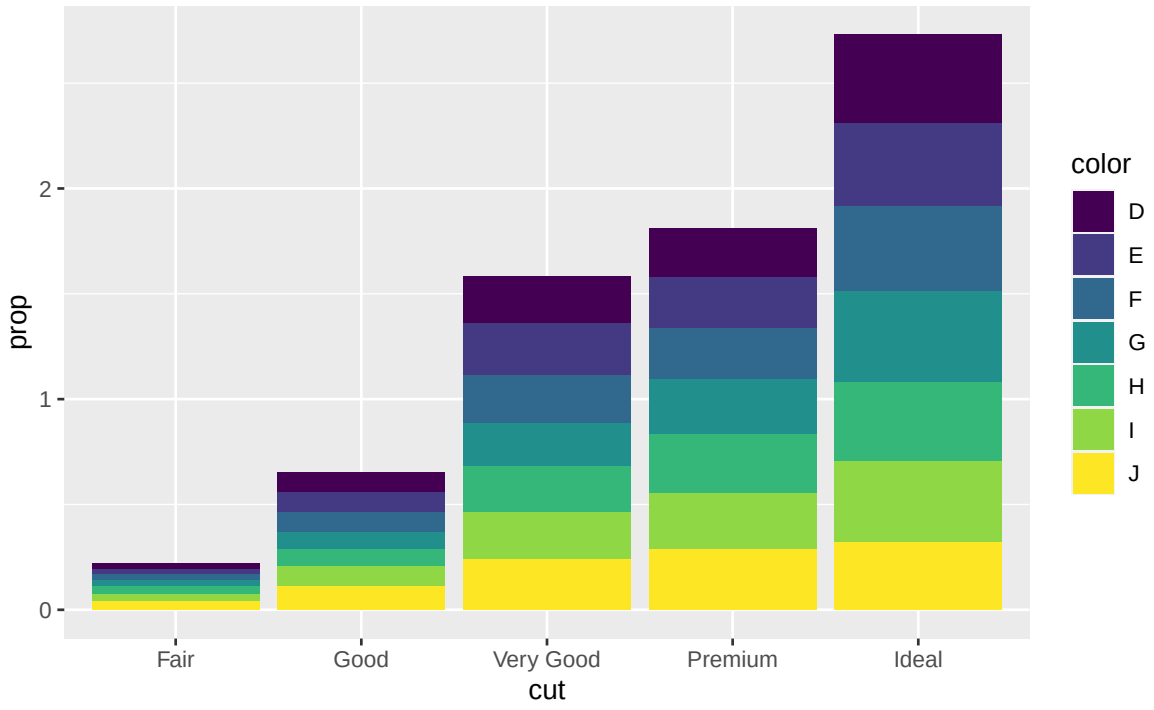
```
  ggplot(data = polls_us_election_2016,
     mapping = aes(x = population, y = grade)) +
geom_jitter()
```



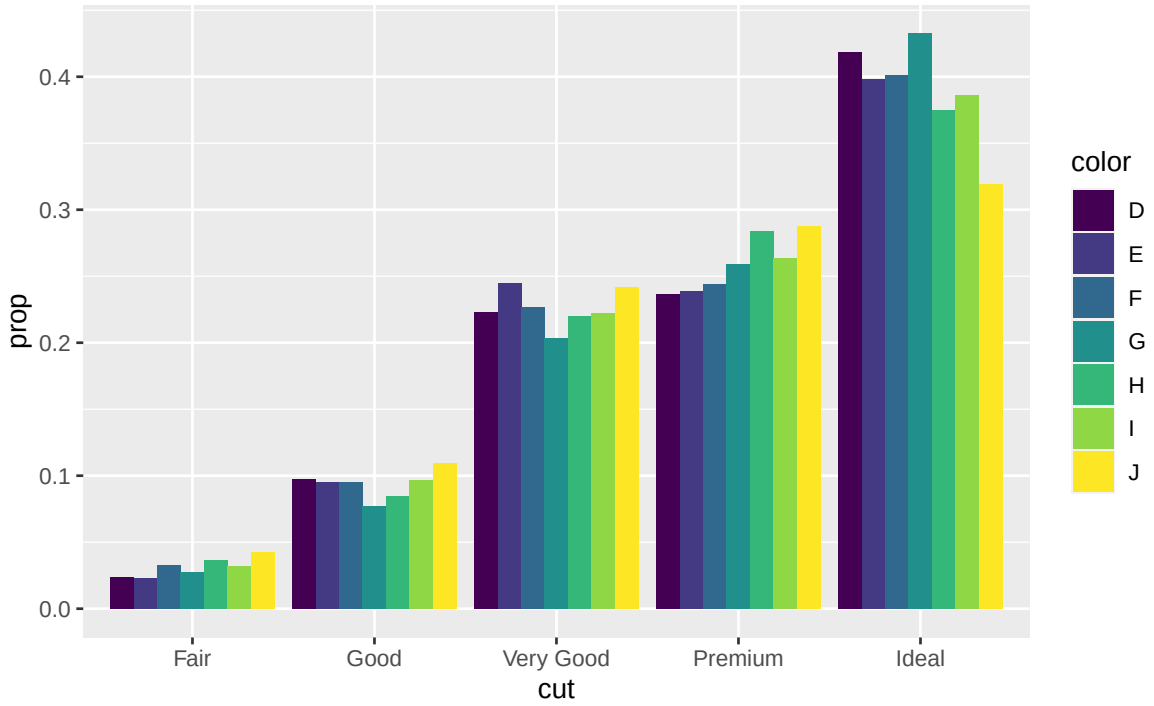

```
# We can best compare and contrast `geom_jitter()` with `geom_count()` by first
# writing a little bit of code to produce plots for both. Overlapping is a
# challenge for our data, because there is a large number of individual grades
# and a small number of unique population values. The first plot we observe
# geom_count mapping the number of observations at each point location. This is
# helpful but it is difficult to visualize individual data points at each
# population value. We can see this better in the geom_jitter plot below, where
# a small amount of random variation is added to the location of each point.
```

1. What's the default position adjustment for `geom_bar()`? What did we add to the code to change the default behavior of geom_bar? Here we are using the diamonds data set again

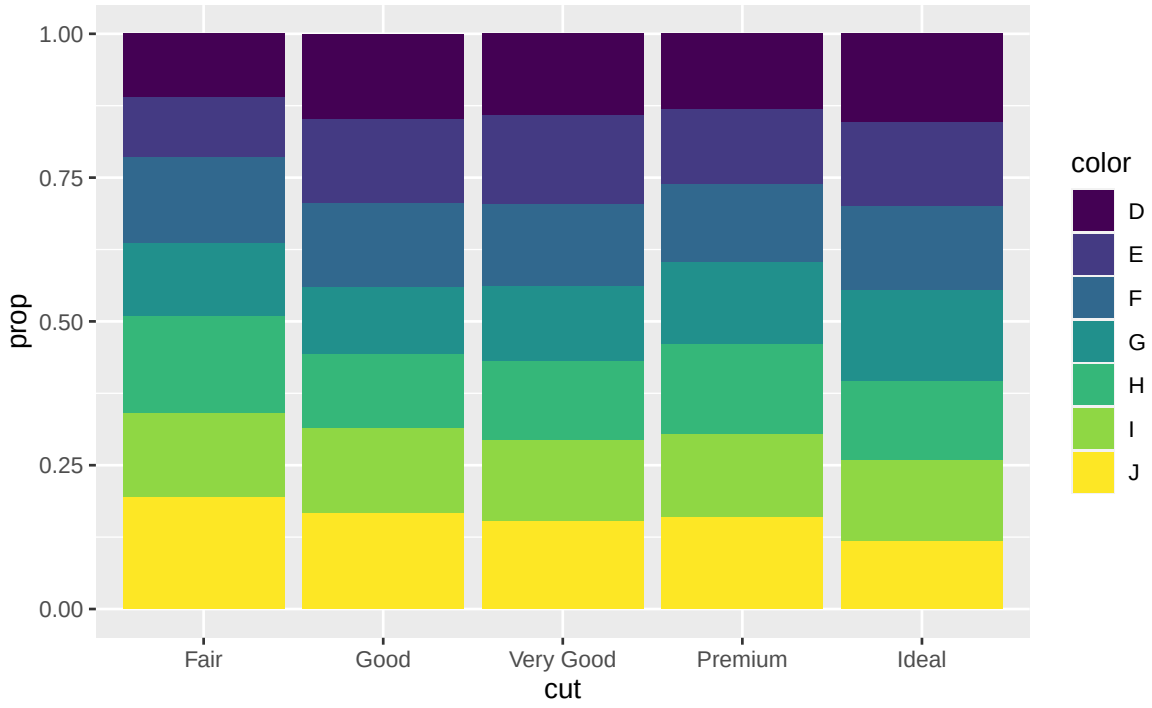
```
# reference link: https://ggplot2.tidyverse.org/reference/geom_bar.html

# the default position adjustment for `geom_bar()` is stacking. The bars are
# automatically stacked (vertically). Each color represents a unique
# observation of a variable.
# When you want to view overlapping data as side-by-side bars, you add in
# position= "dodge"
# When you want to view overlapping data as stacked bars with equal height,
# use position= "fill"

ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = color, y = ..prop.., group= color))
```




```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = color, y = ..prop.., group= color),
           position="dodge")
```



```
  ggplot(data = diamonds) +
geom_bar(mapping = aes(x = cut, fill = color, y = ..prop.., group= color),
         position="fill")
```



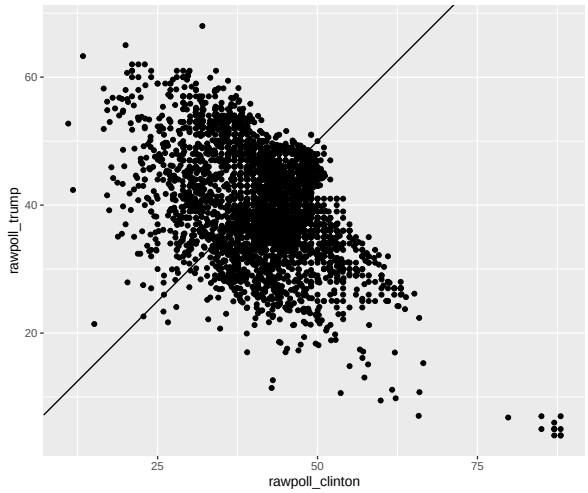

## 2.5 grammar of graphics: Coordinate systems (5 pts)

1. What happens when you use `coord_flip()`?

```
?coord_flip

# coord_flip switches the cartesian coordinates of a plot, so x axis and y axis
# switch places. This is primarily useful for converting geoms and
# statistics which display y conditional on x, to x conditional on y. It also
# can be very useful for creating boxplots and other geoms in the horizontal
# instead of vertical position.
```

1. What is this plot telling us? What does `geom_abline()` do? Why is `coord_fixed()` important?

```
ggplot(data = polls_us_election_2016,
       mapping = aes(x = rawpoll_clinton,
                     y = rawpoll_trump)) +
  geom_point() +
  geom_abline() +
  coord_fixed()
```




```
?geom_abline()
?coord_fixed()

# This plot is telling us the raw poll scores for both Clinton and Trump for
# each pollster.

# What `geom_abline()` does is adds a reference line to the plot. Here we
# are not adding parameters to this reference line, so it starts off in its
# default slope and intercept positions, which are 1 and 0 respectively.

# The `coord_fixed()` function is important because it provides us with the
# ability to specify a ratio for the units on the x and y axes. This allows
# to stretch or widen the plot so that the resulting plot is clearer
# visually or magnified without losing integrity or accuracy in the units.
```