MPA 634  
Data Science for Managers  
Final Exam Key : Winter 2021

# Definitions and Concepts

1. (3 Points) Smoothing is a very important concept for data visualization. Explain how smoothing is affected in the following cases:
   1. binwidth and number of bins on histograms and freqpoly

Smooth histograms and frequency polygons result when you have a small number of wide bins.

* 1. bandwidth (bw) on densities and violin plots

Increasing the bandwidth increases the number points that are averaged. More points in an average gives smoother graphs. The larger number of points means that each one receives a smaller weight in the averaging process.

* 1. span on loess smoothed lines

Increasing the span increases the number points that are averaged. More points in an average gives smoother graphs. The larger number of points means that each one receives a smaller weight in the averaging process.

1. (3 Points) Explain the difference between mutating and filtering joins.

All joins involve two different tibbles. Join structures the information in resulting tibbles by adding information to the first tibble or by selecting information from the first tibble

In dplyr, filtering chooses rows in a tibble. A filtering join does the same thing. It selects rows from the first tibble based on whether there are matches in the keys in the second tibble. A semi-join selects those rows in the first tibble where there is a match in the keys in the second tibble. An anti-join excludes those rose in the first tibble where there is a match in the keys.

In dplyr, mutating adds columns to a tibble. There are two general types of mutating joins: outer and inner. The outer join is like a union because it contains all rows from both tibbles being joined. Inner joins are of three different types: inner, left, and right. The inner join is like intersection because it includes only those observations where the keys match in both tibbles. The left\_join takes all the rows in the first tibble and adds columns from the second tibble where the keys match. The right\_join is seldom used because we can always switch places between the first and second tibbles.

1. (3 Points) Explain how class and method are related to each other. Use the summary method as an example to explain and illustrate your answer.

Methods are a special kind of function. This generic function checks the class of an object that is being used as an argument to the function. Based on the class of the object, it chooses from among the family of functions included in the generic category. For example, the generic function or method summary can be applied to over 89 different classes of objects. If we ask R to summarize a data frame, then it finds the specific function summary.data.frame. If we ask R to summarize an lm object, then it finds the specific function summary.lm to perform the operation. We can see the different methods for the generic summary call by using methods(summary).

1. (3 Points) First define a tibble and then explain why a tibble is a list but not all lists are tibbles.

A tibble is a collection of atomic vectors that are arranged in a list. Each of the atomic vectors must have the same length in a tibble. This means that tibbles are rectangular lists. This means that a tibble is a special case of a list.

In R, vectors are one dimensional holders of information. Vectors have slots or positions in which information and be stored. The information can be referenced by specifying its slot number or name. Although not everyone agrees with Hadley Wickham, he states that there are two kinds of vectors:

* atomic vectors
* lists or recursive vectors

In most programming languages, lists are heterogeneous. This means that they can hold information of a variety of different classes: logical, integer, double, character, etc. The objects in a list can also have a variety of lengths. Lists can even contain other lists. Using lists within lists is a common practice.

1. (3 points) The following code includes each of the seven different parts of the grammar of graphics and creates the accompanying bar chart. Use the code to define and explain each of the seven parts of the grammar of graphics.

## 

A picture containing screenshot

Description automatically generated

1. Data: in this case line 1 identifies mpg as the data frame or tibble from which to get data.

2. Aesthetics: line 2 assigns class to the x axis and fill to the type of drive train.

3. Geometric objects or geom: Line 3 choose a bar chart for the geometric object.

4. Stats: Encompassed in geom\_bar is a counting function which counts the number of observations in each category or level

5. Position: also in the geom\_bar. This locates the bars side by side rather than stacked

6. Coordinate system: The x and y axes are flipped in line 4

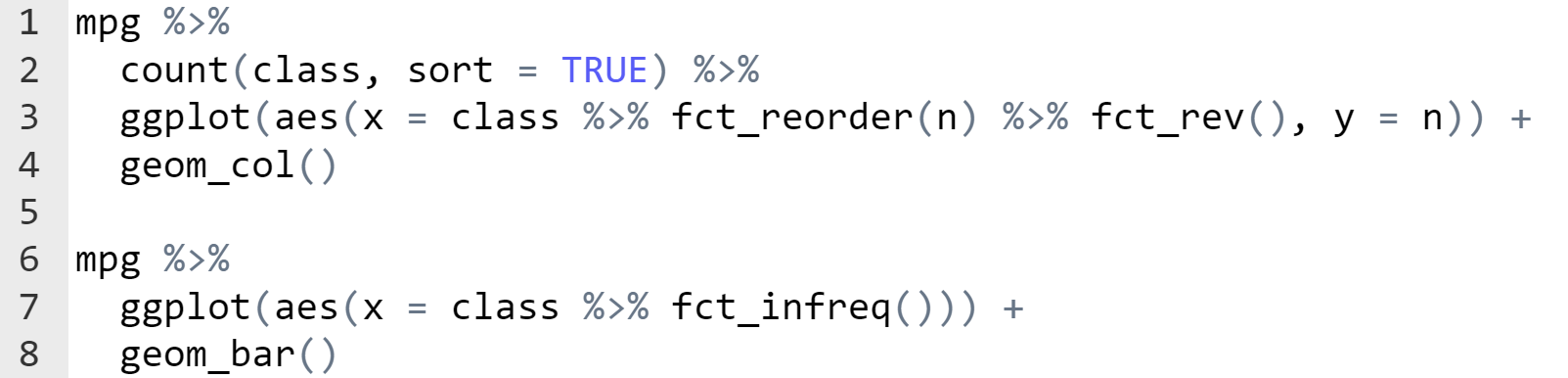
7. Faceting: multiple graphs, one for each year are created by line 5.

# Line by Line Code Interpretation

Please explain the following code chunks. By referring to line numbers, carefully explain the code and describe what it is accomplishing. Please avoid just repeating the R commands that are in the code.

## Code Chunk I: Pareto Charts

Here are two alternative ways to create Pareto charts. Please explain lines 2 – 5 and 8- 9 then compare and contrast the two different approaches to creating Pareto charts.



Line 2: Creates a new tibble with two columns. The first is a category that includes an observation for each class of vehicle. The second, which is named n, is the number of observations in each class. The sort = TRUE argument to the count function sorts the rows in descending order based on the number of observations n.

Line 3: Receives the tibble created in lines 1 – 2. It identifies the class variable in the aesthetic statement. It reorders the factor levels of class based on the number of observations. The order is from smallest to largest initially but the fct\_rev() changes that so that the factors levels are ordered from largest to smallest.

Line 4: Because we have already counted the number of observations for each class, we use geom\_col to put the only layer on our graph, which is a bar chart.

Lines 6 – 7: Identifies mpg as the data source and pipes it into ggplot. The aes identifies the factor class as the x-variable and then reorders the factor levels based on their frequency using fct\_infreq(). The factor levels are ordered from largest number of observations to smallest number of observations

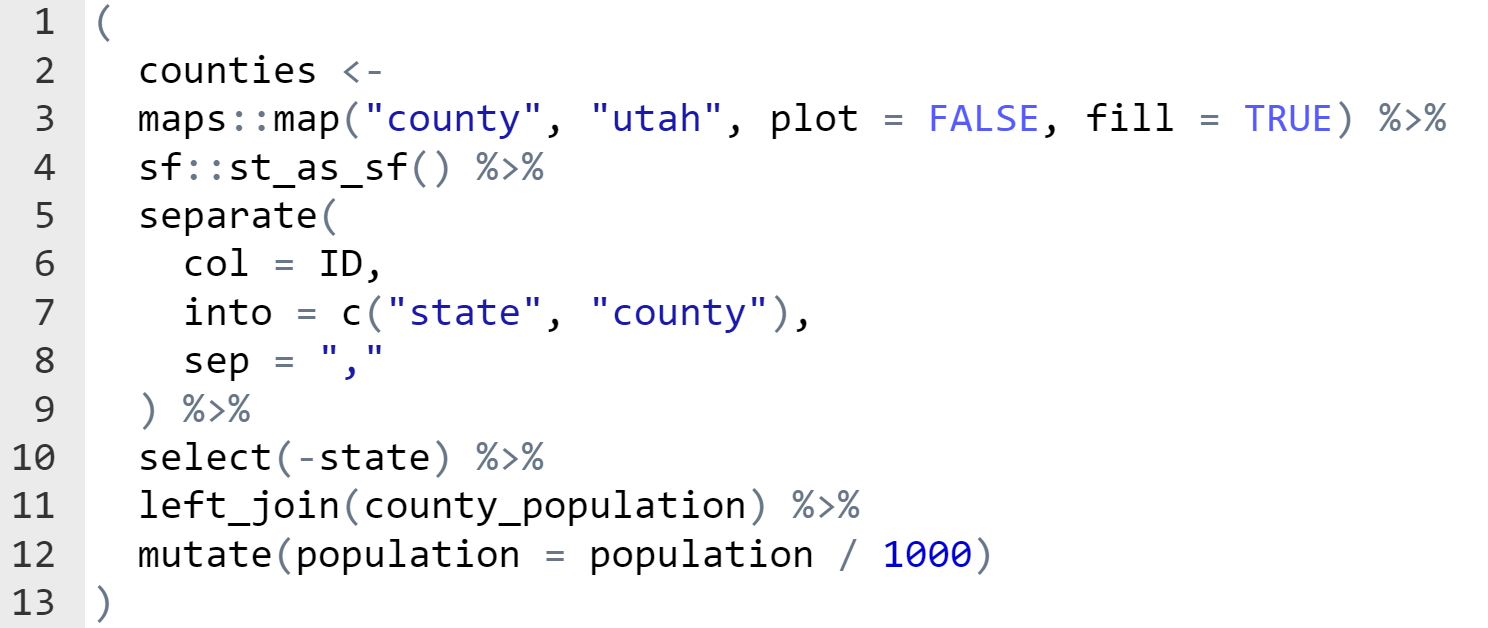
Line 9: The geom\_bar counts the number of observations in each class again and then creates a bar chart. The bars are ordered based on the previous count completed by fct\_infreq.

Both approaches involve counting the frequencies and reordering the factor levels based the size of the frequency for each class. The second approach offers simpler code but is less efficient since it calculates the frequencies twice. Computer cycles are cheap so the simpler code will probably be preferred by many.

## Code Chunk II: Geospatial data

Code chunks II and III create a map of Utah with counties shaded by the size of their population. The tibble county\_population contains two variables, the county name and its population.

Interpret lines 2 – 12 of the following script that creates the tibble needed for code chunk III.



Lines 2 and 3: Creates a new object (counties) which contains the map shapes from the maps library. The maps::map part of the code uses the map function from the maps library. The arguments “county” and “utah” get the county outlines for Utah. The map function contains the outlines for all the counties in all the states.

Line 4: uses the st\_as\_sf function from the sf library to convert the map shapes into an sf (which is like a special tibble) object which will be needed later when we use ggplot.

Line 5 through 9: the ID column in the sf or tibble object is a conglomerate of state and county. We split or separate this variable into two variables: state and county. In the ID variable, the state and county are separated by a comma.

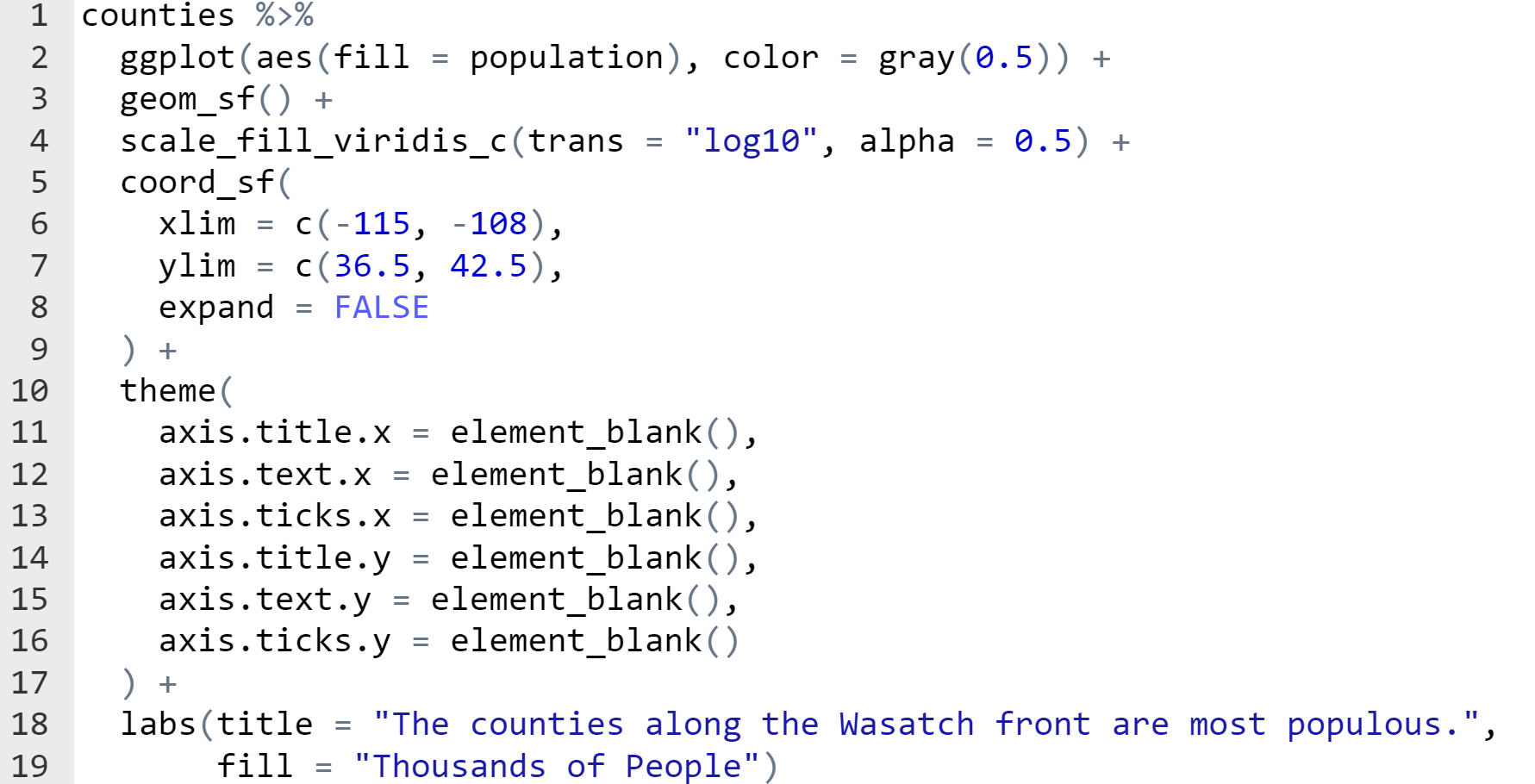
Line 10: Drops the state variable from the sf object

Line 11: This is a mutating join since we are adding the number of people in each county to the sf object. Since we are using a left\_join, we first replicate the tibble created in the previous lines and add a column population from the county\_population tibble.

Line 12: Divides the population variable by 1000 so that the numbers will print out in a more readable form.

## Code Chunk III: Geospatial

Explain all the lines of code which starts with the counties tibble created in code chunk II.



Lines 1 and 2: Use the counties sf object created in the previous code chunk. The fill depends on the numerical variable population. The outline color is assigned a shade of gray

Line 3: Creates a layer in our graph using geom\_sf. Since we are passing ggplot an sf object, it knows how to find the map shapes and plot the outlines in this layer of our plot.

Line 4: We want to fill the counties using the viridis color scheme so that those who have trouble distinguishing colors can use our graph. We use the viridis\_c palette because population is a numerical variable. When we map the population to a color, we first take the logarithm base 10 of the population numbers. We use a transparency of 50% when we will the counties.

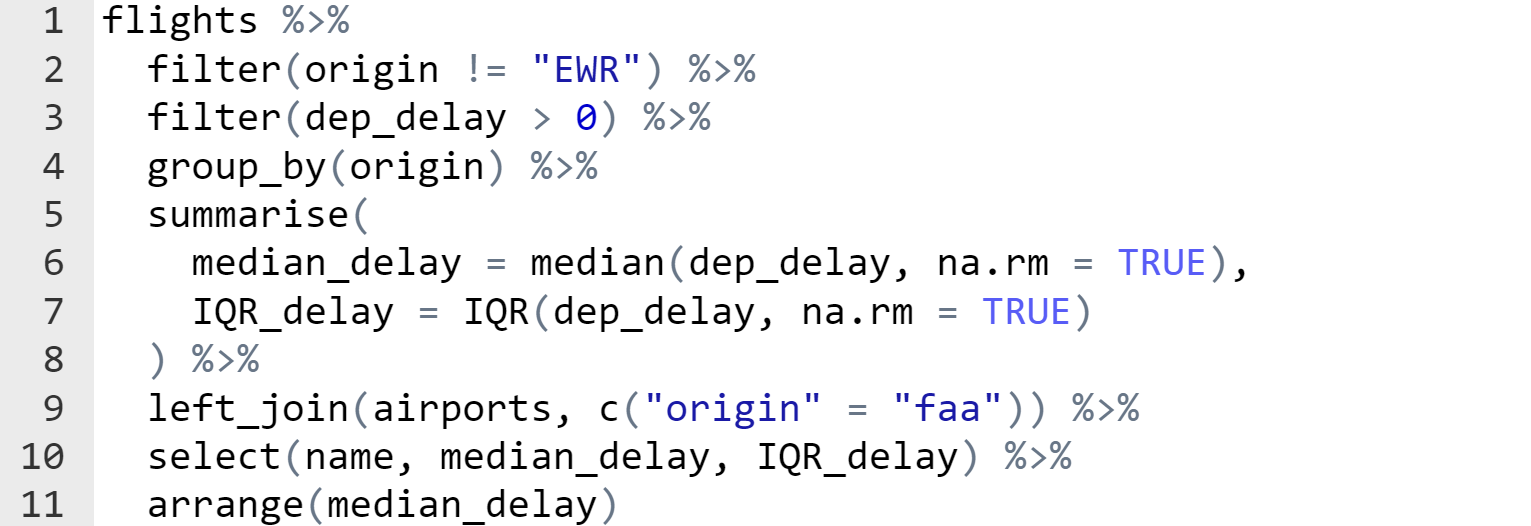
Lines 5 through 9: Focuses the map on Utah by limiting the longitude on the x-axis between -115 and -108 and the latitude to between 36.5 and 42.5

Lines 10 through 17: Suppress the title, text, and ticks on both the x and y axes.

Lines 18 and 19: Gives the title to the graph and the legend.

## Code Chunk IV: dplyr

Explain lines 2 - 11

  
  
Lines 1 – 2: Chooses all of the rows from the flights tibble that don’t originate from Newark (EWR)

Line 3: Chooses only those observations that are delayed.

Line 4: Groups the observations into categories defined by origin (JFK and LGA)

Lines 5 – 8: Calculates the median and interquartile range for JFK and LGA. The result is a tibble that has origin airport as its rows and median\_delay and IQR\_delay as columns

Line 9: Adds information from the airports tibble to the tibble that has been created in lines 1 through 8. The primary key from the created tibble is origin and the foreign key from the airports tibble is faa

Line 10: Selects the columns name, median\_delay, and IQR\_delay

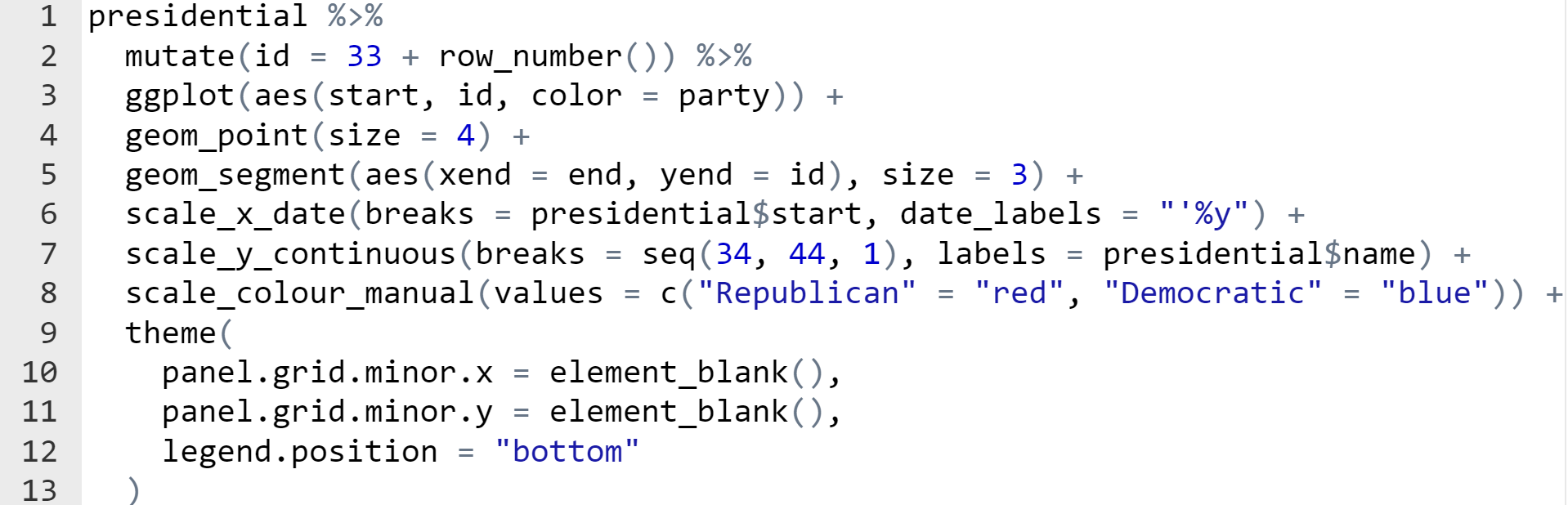
Line 11: Sorts the rows of the resulting tibble by median\_delay from smallest to largest

## Code Chunk V: Communication Graphics

The presidential tibble has four variables initially:

* name of president
* start date of administration
* end date of administration
* political party

Interpret lines 2 - 12



Lines 1 – 2: Calculates a new variable called id by adding the row number of each observation to 33. This gives the number of the president administration.

Line 3: Assigns x as the date of the start of the administration and y as the id created in the previous step. Color is scaled or assigned to the political party.

Line 4: Created a scatter plot where each point has a size of 4.

Line 5: Geom\_segment requires an x and y value for the start and an x and y value for the end of the line segment. In this case, it gets the start x and y from the aes declaration in the ggplot call. The aes in the geom\_segment specifies the ending x and y values. They are the end variable and the id variable.

Line 6: Specifies that the x axis is a date and the tick marks appear at the start of each president’s administration. The format of the data is an apostrophe by the last two digits of the year.

Line 7: Creates tick marks at the integers between 34 and 44 and then labels the tick marks by the president’s name.

Line 8: Defines that the color which was assigned in line 3 should be red for Republican and blue for democrat.

Lines 10 – 11: Excludes the minor grids for the x and y axes

Line 12: Positions the legend at the bottom.