MPA 634  
Data Science for Managers  
Final Exam: Winter 2020

# Definitions and Concepts

1. First explain distribution of a numerical random variable, then define each of the four characteristics of a distribution, and finally detail how you measure and detect each of the four characteristics.  
     
   Distribution:

* The possible values for the random variable
* Probability or frequency of observation

Location

* Central tendency
* Mean and Median

Scale

* Spread
* Standard deviation and IQR

Symmetry

* Skewness
* Relationship of mean and median
* Location of median in the box part of a boxplot
* Length of whiskers

Outliers

* Deviant or extreme observations
* z-scores
* Greater than 1.5 IQR from the hinges of the boxplot

1. Explain the difference between hypothesis generation and hypothesis confirmation.   
     
   Each observation can be used for exploration or confirmation, but not both.

You can use an observation as many times as you want in exploration but only once in hypothesis confirmation. As soon as an observation gets used more than once, we have transitioned from confirmation to exploration.  
  
We can separate our data into training sets and testing sets. This way we can use our training set data as many times as we want when exploring our data.

We then use the resulting model to test the accuracy of our model once on the testing set.

A new practice is to create a query set to test our training set repeatedly before we do the final, one test using the testing set.

1. 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. Compare and contrast functions and loops. How are they related to the map functions?  
     
   A function is a body of code that we repeatedly execute. A function has a name, arguments, and a copy. Functions are only executed whenever they are called.

A loop is also a body of code that we repeatedly execute. A loop has a place to store results, an index over which we iterate, and a body of code to execute over and over again.  
  
Functions and loops both have bodies of code that are executed repeatedly. Functions only execute the code once when the function is called. In order to execute a body of code repeatedly without any intervention or supervision, it must be placed within a loop. Often we write functions and include them within the body of a loop.

1. Explain why a tibble is a list but not all lists are tibbles.

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.

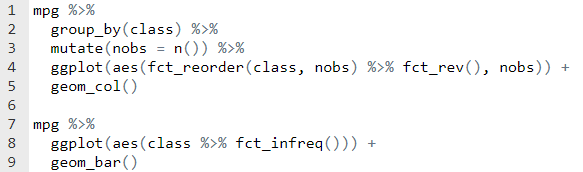
A tibble is a collection of atomic vectors that are arranged in a list. Each of the atomic vectors must have the same length in a tibble. This means that tibbles are rectangular lists. This means that a tibble is a special case of a list.

# 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.



Lines 1 – 2: Divides the tibble mpg into groups by the class of the vehicle

Line 3: Counts the number of observations in each class and creates a new variable called nobs

Line 4: Receives the tibble created in lines 1 – 3. 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.

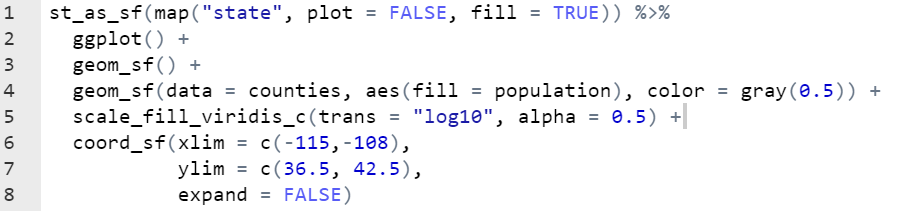
Line 5: Because we have already counted the number of observations for each class, we use geom\_col to put the only layer on our graph, which is a bar chart.

Lines 7 – 8: 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: Interpret lines 1 – 8 of the following script that creates a map of Utah with counties shaped by the size of their population



Line 1: Uses the map function in the maps library to find the spatial polygons for each of the states in the US. The st\_as\_sf function converts the object into a sf (simple features) object that ggplot knows how to process

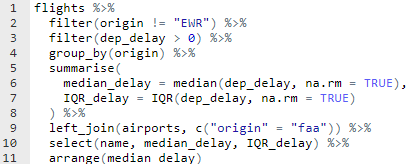
Lines 2 – 3: Receives the object from line 1 and then outlines a map of the states in the United States in a new layer.

Line 4: Uses data from an sf object called counties to create a second map layer on top of the first which contains the outlines of counties within the states shown on the map. The aes(fill = population) fills each county based on the population. The color = gray(0.5) outlines each county in gray at an opacity of 50%.

Line 5: Uses the viridis palette to determine the scale of colors to use for filling the counties. We also use a log base 10 transformation of the population values to make the colors on the map more distinguishable. Finally, we fill with a transparency of 50%.

Lines 6 – 8: Focuses the map of the US on the area with longitudes of -115 and -108 and latitudes of 36.5 to 42.5.

Code Chunk III: Explain lines 2 – 11 of transformations using dplyr

  
  
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)

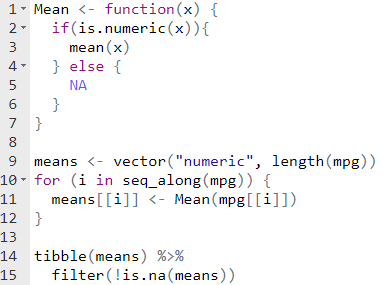
Lines 5 – 8: Calculates the median and interquartile range for JFK and LGA. The result is a tibble that has origin airport as its rows and median\_delay and IQR\_delay as columns

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 IV: Functions and Iteration. Explain lines 1 – 5, 9 – 11, and 14-15



Line 1: Names the function and identifies x as its argument

Lines 2 – 7: This if else structure first tests to see if x is numeric. If it is numeric then it calculates and returns the mean. If x is not numeric, then the function assigns the value as missing and returns an NA value.

Line 9: Initializes a numeric atomic vector called means. Its length is the number of columns in the mpg tibble.

Line 10: Sets up a for loop. The index variable I takes on the values between 1 and the number of columns in mpg.

Lines 11 – 12: Assigns the result of our applying our Mean function to each column in the mpg tibble. The result is put into the ith slot of the vector that we created in line 9.

Line 14: Created a tibble from the means vector that was initialized in line 9 and calculated in line 11.

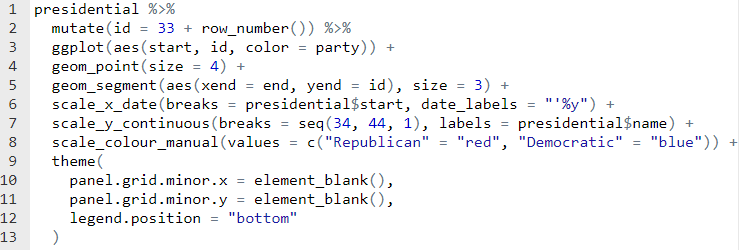
Line 15: Removes the missing observations from the tibble

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.