IST 687 Descriptive Statistics & Functions

Corey Jackson

2021-01-27 19:51:57

Today's Agenda

- Announcements
- Review of Week 2 (Async; Chapters 4-6)
- Breakout Session
  - Complete Lab 3
  - Project Update I
- Homework 3 Tips
- Next week's agenda

Announcements

- HW 1 grades/feedback on LMS
- Questions/concerns?

Office Hours
Office Hours: 6-7pm ET Wednesday (Link on 2U)
Slack: via direct message or @Corey Jackson (Instructor)
Email: cjacks04@syr.edu

Week 2: Using R to manipulate data

Overview of Week 2: (Using R to manipulate data)

- Working with vectors
- Introduction to data frames/manipulating data frames

Week 2: Working with Vectors

- Useful functions to explore your data: `sum()`,`max()`,`min()`
- Working with `which()` returns the index values matching the conditional

```
weather <- c("hot","cold","cold","cold")

which(weather=="cold")
```

```
## [1] 2 3 4
```

Week 2: Working with Vectors

- Warning when using `length()` vs `sum()`
- `length()` counts the number of index positions returned when evaluating its argument
  `length(which(weather=="cold"))`

```
## [1] 3
```

- `sum()` sums up the index values returned from evaluating the conditional (Proably not what you intended)
  `sum(which(weather=="cold"))`

```
## [1] 9
```

Week 2: Working with Accessors

- Accessors get data from R objects
- Working with accessors i.e., `[ ]`,`[[ ]]`,`$`

`mtcars$mpg` (interact with)

```
## [1] 21.0 21.0 22.8 21.4 18.7 18.1
```

`mtcars[1:2,]` (subset with)

```
##                mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4       21   6  160 110  3.9 2.620 16.46  0  1    4    4
## Mazda RX4 Wag   21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

Week 2: Dataframes

- Subsetting columns and rows
- `df[ first position (row) , second position (column), ]` additional resource "RC-Cola"
  - 1st and 2nd row and 2nd and 3rd column: `mtcars[1:2, 2:3]`
  - 1st and 2nd row and assume all columns:`mtcars[1:2,]`
  - assume all rows and 2nd and 3rd column:`mtcars[,c(2,3)]`

Week 2: Dataframes

- Extracting `colnames()` and `rownames()`
- Creates vector of column/row names

```
rownames(mtcars)
```

```
##  [1] "Mazda RX4"           "Mazda RX4 Wag"       "Datsun 710"
##  [4] "Hornet 4 Drive"      "Hornet Sportabout"   "Valiant"
##  [7] "Duster 360"          "Merc 240D"           "Merc 230"
## [10] "Merc 280"            "Merc 280C"           "Merc 450SE"
## [13] "Merc 450SL"          "Merc 450SLC"         "Cadillac Fleetwood
## [16] "Lincoln Continental" "Chrysler Imperial"   "Fiat 128"
## [19] "Honda Civic"         "Toyota Corolla"      "Toyota Corona"
## [22] "Dodge Challenger"    "AMC Javelin"         "Camaro Z28"
## [25] "Pontiac Firebird"    "Fiat X1-9"           "Porsche 914-2"
## [28] "Lotus Europa"        "Ford Pantera L"      "Ferrari Dino"
## [31] "Maserati Bora"       "Volvo 142E"
```

Week 2: Dataframes

```
carnames <- rownames(mtcars)
MyCars$cars <- carnames
```

```
##    qsec vs am gear carb              cars
## 1 16.46  0  1    4    4         Mazda RX4
## 2 17.02  0  1    4    4     Mazda RX4 Wag
## 3 18.61  1  1    4    1        Datsun 710
## 4 19.44  1  0    3    1    Hornet 4 Drive
## 5 17.02  0  0    3    2 Hornet Sportabout
## 6 20.22  1  0    3    1           Valiant
```

Week 2: Operating on Dataframes

- A few useful functions to summarize your data: `str()` and `summary`
- Subsetting dataframes with more complex conditionals

```
MyCars2 <- MyCars[which(MyCars$mpg > 20), ]
MyCars2
```

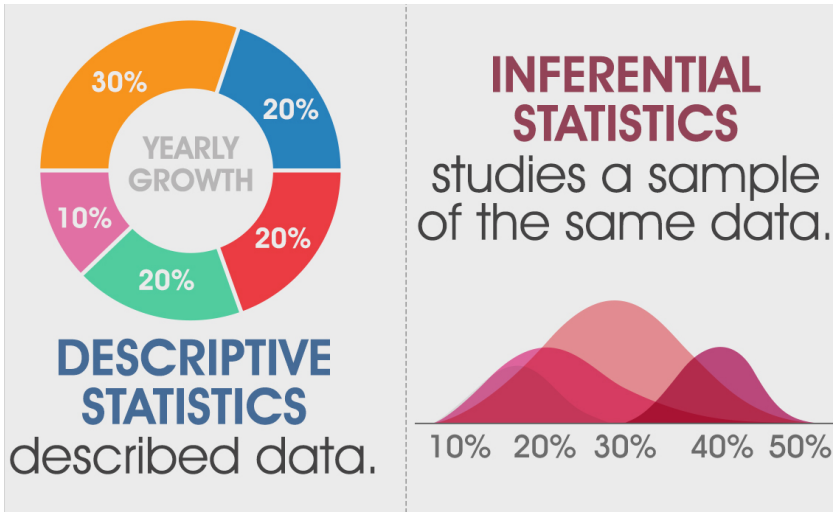
```
##    mpg cyl  disp  hp drat    wt  qsec vs am gear carb           cars
## 1 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4      Mazda RX4
## 2 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4  Mazda RX4 Wag
## 3 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1     Datsun 710
## 4 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1 Hornet 4 Drive
## 8 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2      Merc 240D
## 9 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2       Merc 230
```

Week 3: Descriptive Stats & Functions

Week 3: Descriptive Stats & Functions

The goal for this module will be to introduce you to descriptive statistics used to summarize your data and inferential statistics used to draw conclusions about a sample from the population.

- Descriptive Statistcs
- Data Distributions
- Writing functions



Descriptive Statistcs

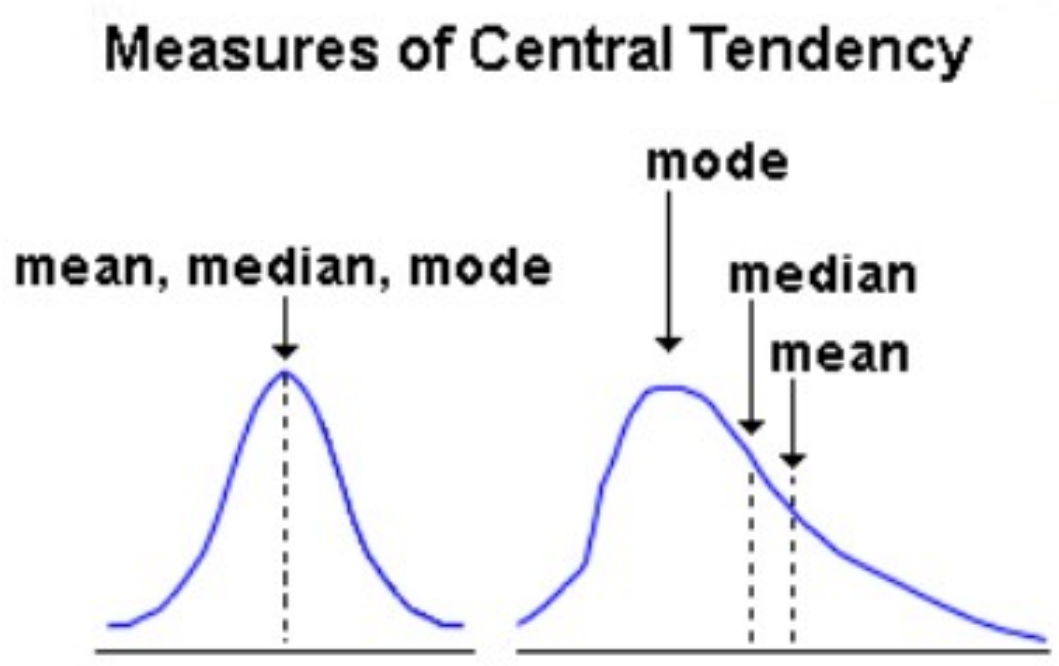
A descriptive statistic is a summary statistic that quantitatively describes or summarizes features of data collected.

Two primary means of describing data:
1. Central tendency: a central or typical value for a distribution
2. Spread or Variance: the extent to which a distribution is stretched or squeezed.

Descriptive Statistcs: Central tendency

Central tendency is a central or typical value for a distribution. Also called center or location

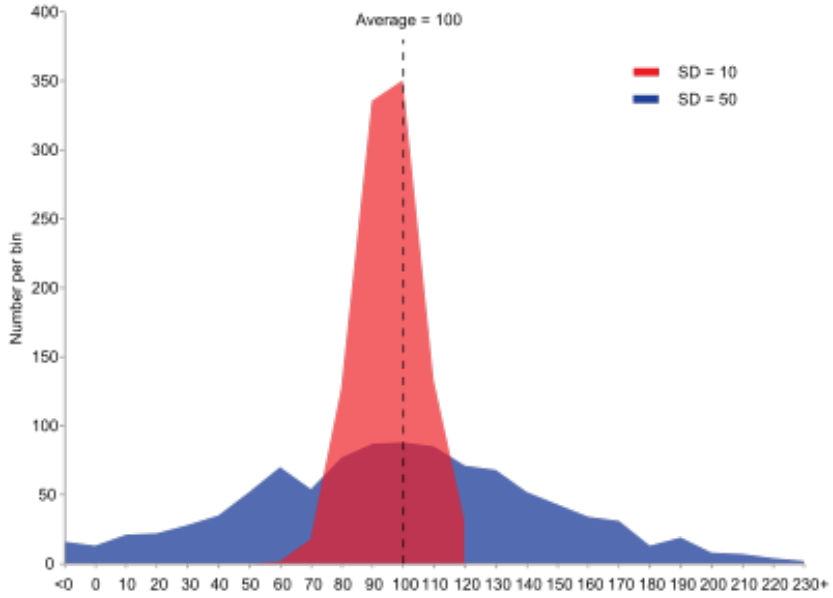
The most common measures of central tendency are:
- *arithmetic mean*: the numerical average of all values
- *median*: the value directly in the middle of the data set
- *mode*. the most frequent value in the data set



Descriptive Statistcs: Spread or Variance

Spread (dispersion or variability) is the extent to which a distribution is stretched or squeezed.

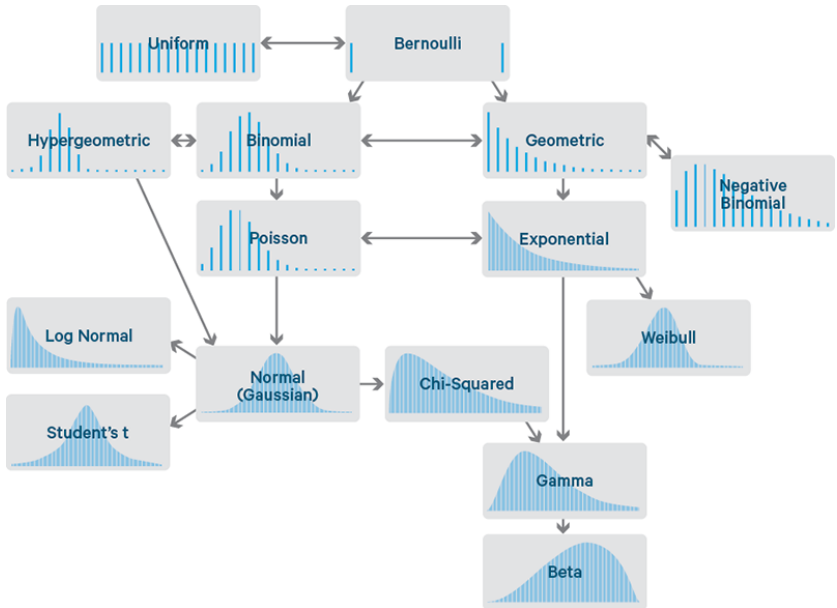
The most common measures of statistical dispersion
- *variance*: the average of the squared differences from the mean
- *standard deviation*: the square root of the variance
- *inter-quartile range (IQR)*: the distance between the 1st quartile and 3rd quartile and gives us the range of the middle 50% of our data



Data distributions

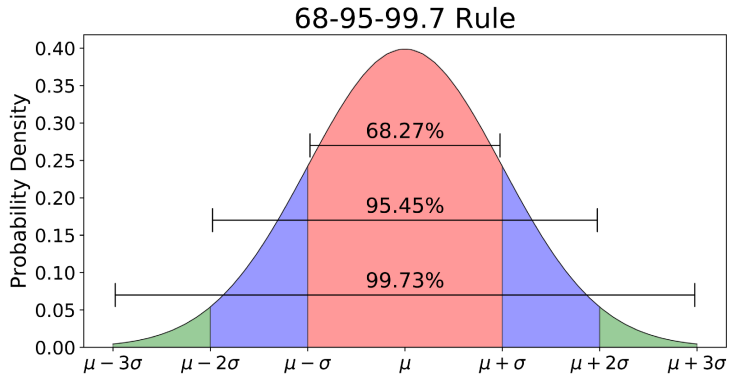
A distribution contains information about the probabilities associated with the data points.

- Thousands of data distributions



Data distributions

- Visualizing data distributions in R



- Why is knowing the distirbutions of data helpful?

Example: Simulating a normal distributions in R

R allows you to simulate different distributions using functions and arguments as parameters.

Task: Generate 1000 values of a normal distribution, with a mean of 85

- Normal distribution: `rnorm()`

```
testdatasim <- rnorm(1000,85)
```

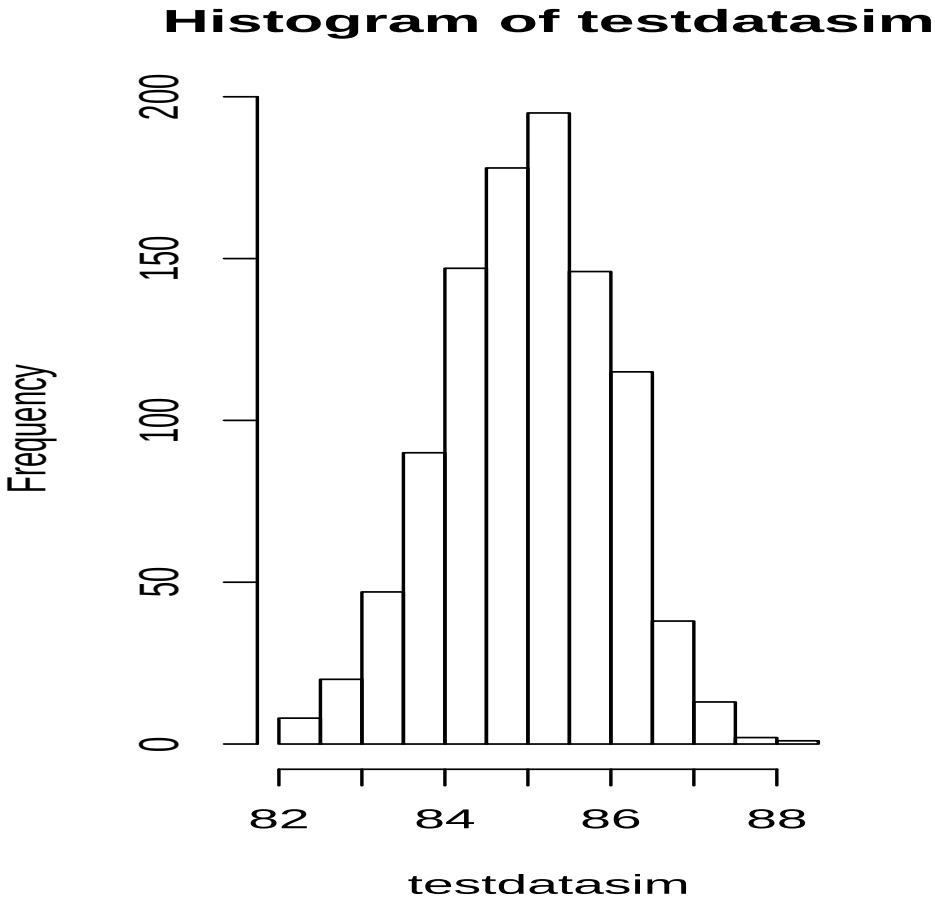
```
## [1] 84.77568 85.36390 84.60846 84.65081 84.38255 86.90651
```

```
mean(testdatasim)
```

```
## [1] 84.98958
```

Example: Visualizing a normal distributions in R

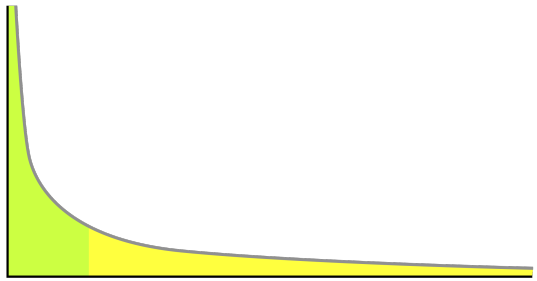
```
hist(testdatasim)
```



Lab Assignment

Lab 3: Simulating and visualizing a Pareto distribution

- A common distribution found in data science in the "long-tail" e.g., Pareto



In lab you need to simulate a Pareto distribution: `rpareto(n, m, s)`

1. Install VGAM: `install.packages("VGAM")`
2. Read about rpareto using help: `??rpareto`
3. Set `m` to 560000 (about the population size of Wyoming), play around with the `s` parameter

Functions

- Basic components of functions: function name, arguments, function body, and return value

```
function_name <- function(arg_1, arg_2, ...) {
   Function body
}
```

- R has many *in-built* functions which can be directly called in the program without defining them first: mean(), max(), length(x), but we can also create *user defined* functions.

Function Example

Write a function that takes two arguments - a vector of numbers (v) and a random number (w) and returns the count of numbers in v greater than w

```
function_name <- function(arg_1, arg_2, ...) {
   Function body
}
```

Function Example: Step-wise function writing

- Create a vector of numbers v: `v <- c(112,54,10,3,152,55)`

```
## [1] 112  54  10   3 152  55
```

- Create a random number w: `w <- 25`

```
## [1] 25
```

Function Example: Step-wise function writing

Which are elements in v that are greater than w: `v > w`

```
## [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE
```

Return only the elements in v that are greater than w: `which(v>w)`

```
## [1] 1 2 5 6
```

Return the count of the elements in v that are greater than w
`length(which(v>w))`

```
## [1] 4
```

- Store that value in a variable: `greater_numbers <- length(which(v>w))`

Function Example: Step-wise function writing

```
myfirstfunction <- function(arg,arg,..)
{
   BODY
}
```

Write a function that takes two arguments - a vector of numbers (v) and a random number (w) and returns the count of numbers in v greater than w

```
myfirstfunction <- function(vector,constraint)
{
   greater_numbers <- which(vector > constraint)
   count_numbers <- length(greater_numbers)
   return(count_numbers)
}
```

Function Example: myfirstfunction()

- Vector of values as argument
  `myfirstfunction(c(112,54,10,3,152,55),55)`

```
## [1] 2
```

- R object as arguments
  `myfirstfunction(v,w)`

```
## [1] 4
```

- In a dataframe
  `myfirstfunction(mtcars$hp,225)`

```
## [1] 5
```

Lab 3: Writing a function

Write a function that takes three arguments – a vector, a min and a max, and returns the percentage of elements in the vector that are between the min and max (including the min and max)

Build in a stepwise manner

1. Compute the number of elements in the vector that are greater than min and less than max.
2. Using the number that was returned in the previous line, divide the number by the total number of elements in the vector

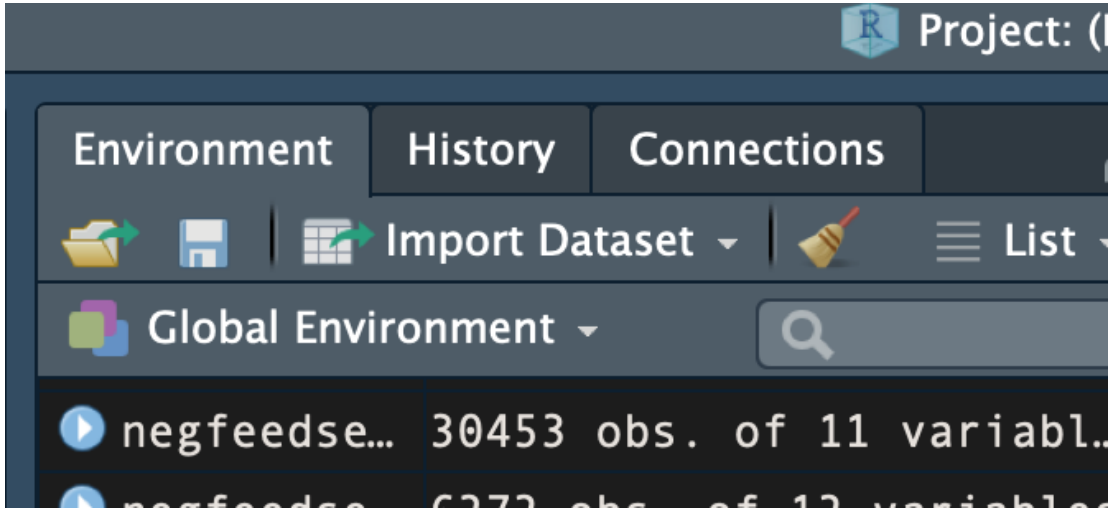
Code hints: `which()` and `length()` or `sum()` and logical operators from Week 1

Project Update I

Homework Tips

Homework 3 Tips

- Data importing
  - Manually importing data vs. the data import wizard



Homework 3 Tips

- Changing column names using `colnames()` function
- Replacing characters in strings using the `gsub()` function (type ??gsub or search for gsub() to see its arguments)
- Note: R does not accept commas for numeric datatypes. e.g., 1343 not 1,343
- Same function you created in lab needed with slight modifications
- Error in original HW file. Here's the correct data: https://www2.census.gov/programs-surveys/popest/tables/2010-2011/state/totals/nst-est2011-02.csv

Next Week

- Asynchronous
  - Week 4: Inferential statistics; Read chapter 10
  - HW 3 and Lab 3 due Monday, 11:59 AOE
- Synchronous
  - Lab 4: Sampling & Decisions Pt. 2