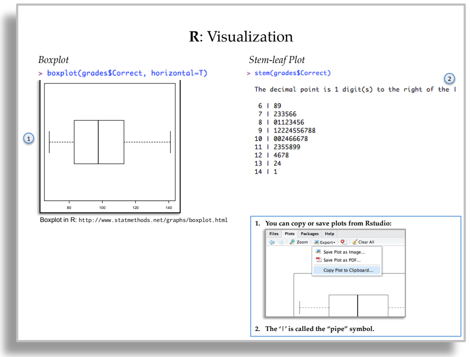
Business Analytics Using Computational Statistics

Week 1
Computation and Statistics

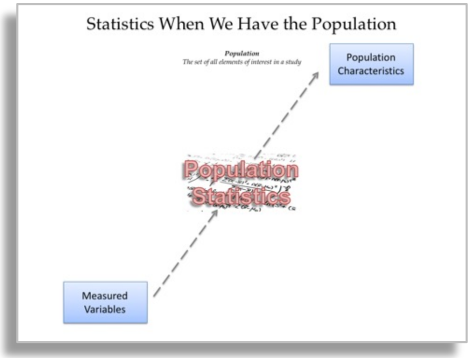
Week 2
Description and Simulation

Week 3
Intervals and Resampling

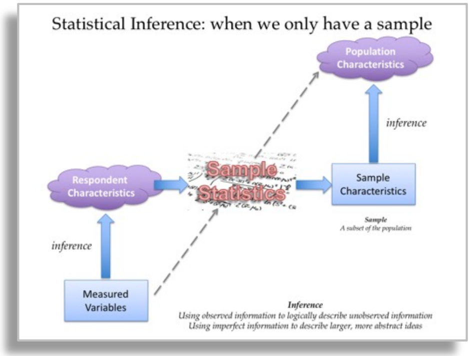
Describing and Visualizing



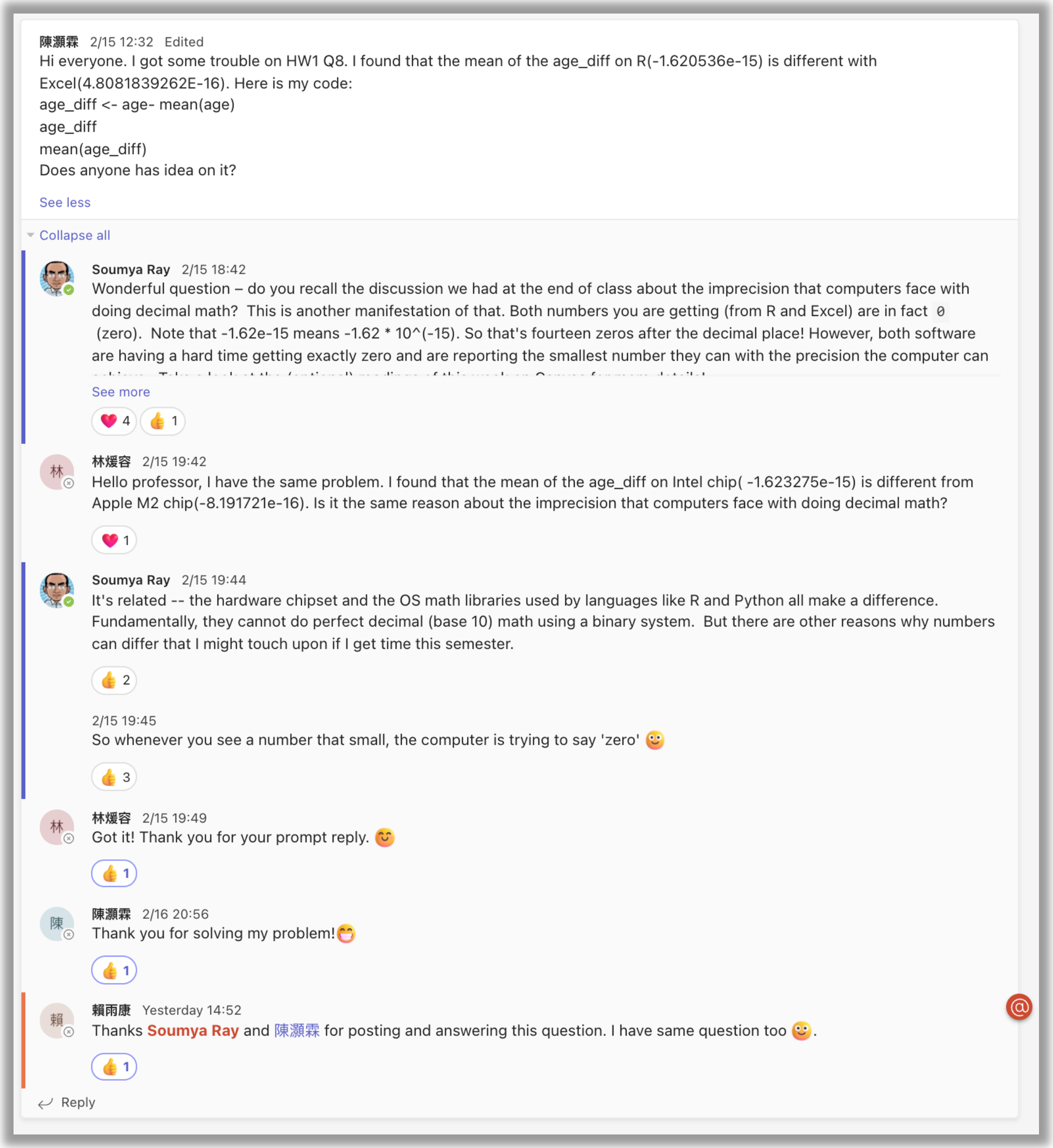
Population Statistics



Population & Sample Statistics



Social Coding is Easier



Asking questions *benefits you and others who might have the same question*

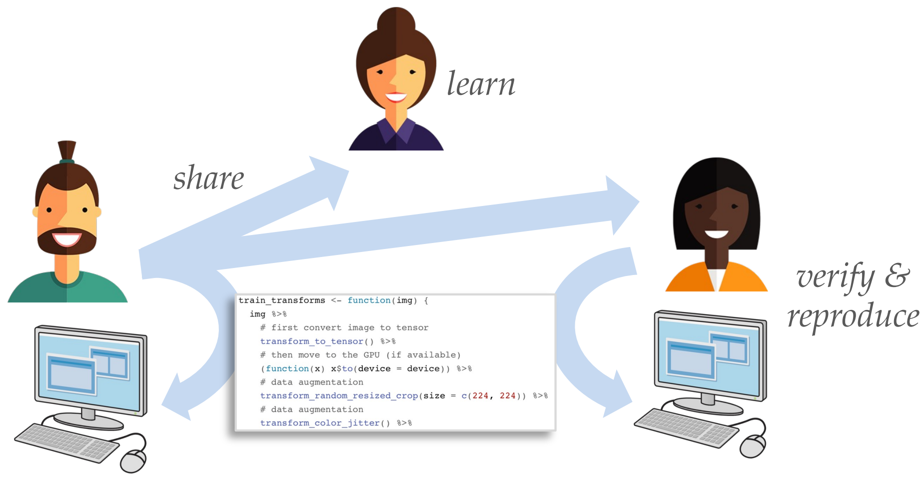
Answering questions *benefits others, and helps buid your own understanding*

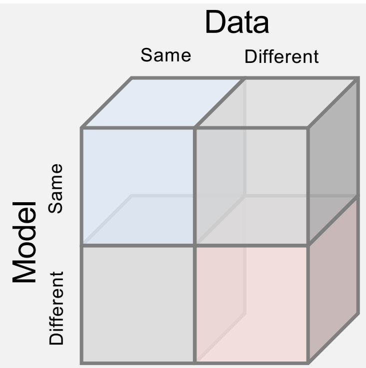
Coding can seem scary *and it helps to know that others are facing the same issues!*

Social Coding is More Powerful

Sharable
Give your solution to others to learn from

Verifiable
Others can ensure it does the right things



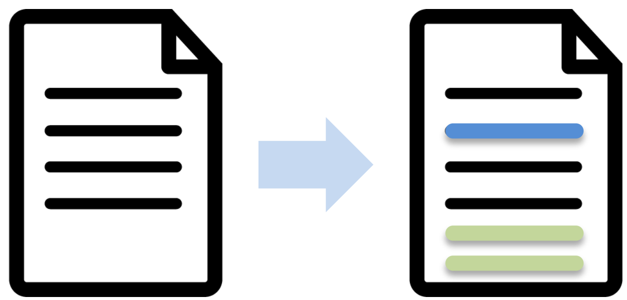


Reproducible
Others can recreate your results

Reapplicable
Others can reapply your process

Publishable
Publish your code & data as part of your paper

Extensible
Alter and improve on other studies!



Limits of Computation

Differences in Arithmetic

```
0.1 == 0.1                TRUE or FALSE?
0.1 + 0.1 == 0.2          TRUE or FALSE?
0.1 + 0.1 + 0.1 == 0.3    TRUE or FALSE?
```

```
(0.1 * 3) - 0.3
[1] -5.551115e-17
```

$$\underbrace{-5.551115e\text{-}17} \quad -5.551115\times10^{-17}$$

$-0.00000000000000005551115$

```
0.1 * 4  ==  0.4 ?  TRUE or FALSE
0.1 * 5  ==  0.5 ?  TRUE or FALSE
0.1 * 6  ==  0.6 ?  TRUE or FALSE
0.1 * 7  ==  0.7 ?  TRUE or FALSE
0.1 * 8  ==  0.8 ?  TRUE or FALSE
0.1 * 9  ==  0.9 ?  TRUE or FALSE
0.1 * 10 ==  1.0 ?  TRUE or FALSE
```

⚠

When Math Fails You
https://dev.to/jdsteinhauser/when-math-fails-you-2if8

Why 0.1 Does Not Exist In Floating-Point
https://www.exploringbinary.com/why-0-point-1-does-not-exist-in-floating-point/

Finding Zero

```
ages - mean(ages)
 [1]   2.1929825  22.1929825  -5.8070175  26.1929825  -1.8070175  24.1929825   3.1929825
 [8]  -3.8070175  23.1929825 -14.8070175   0.1929825  30.1929825  17.1929825   3.1929825
```

```
mean(ages - mean(ages))
 [1] -1.623275e-15
```

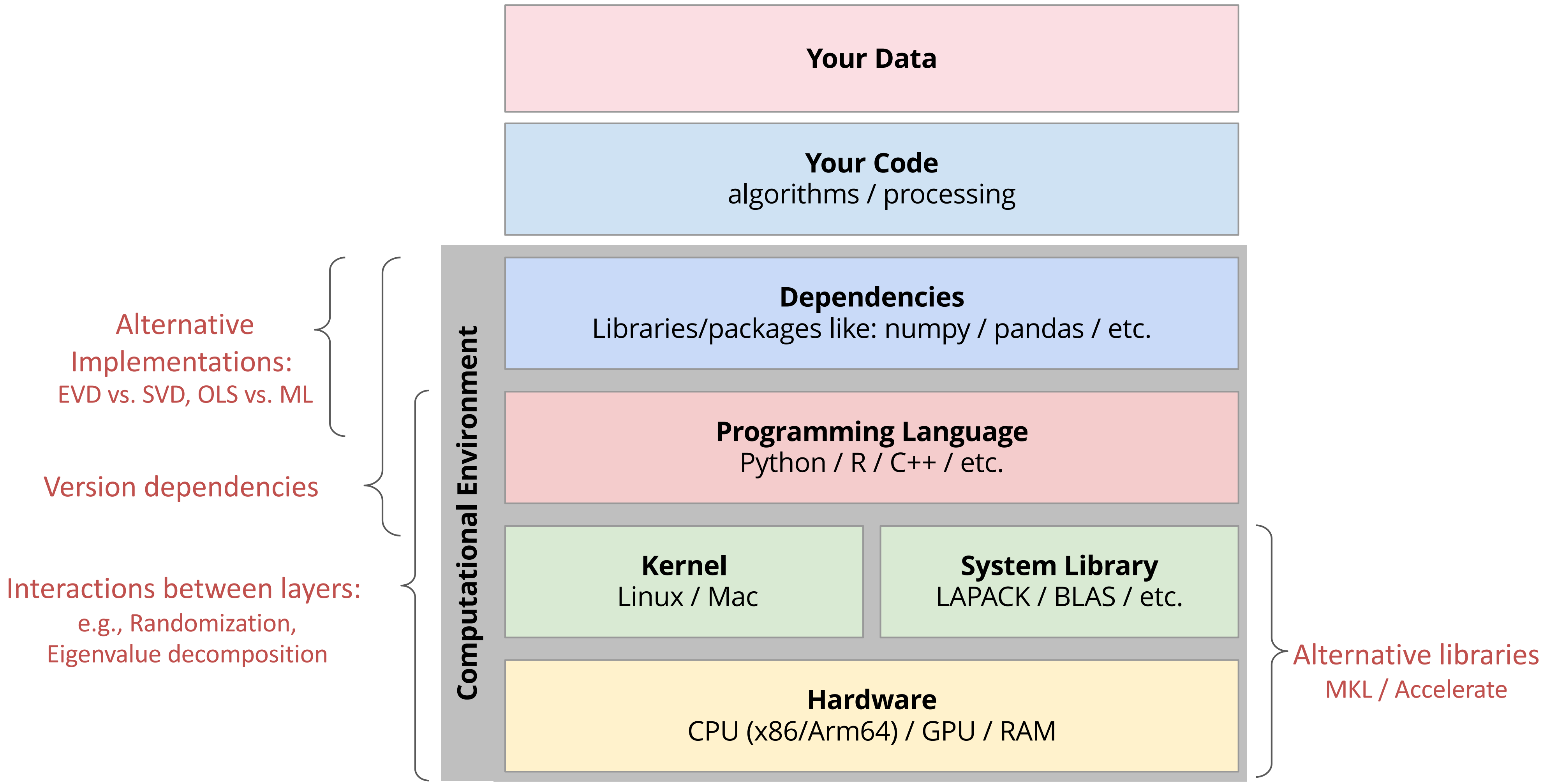
-1.623275×10^{-15}

-0.000000000000001623275

😧

It make a difference:
- *which **language** we use*
- *which **operating system** libraries we use*
- *which **hardware (CPU)** we use*
- *and more…*

Computational Stack for Analytics



Each layer of this stack can affect the precision or other aspect of computational solutions

Variables vs. Values

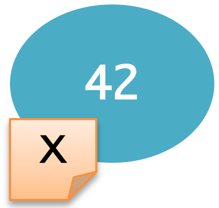
variable *value*

```
x <- 42

x
# [1] 42
```

Boxes? or **Labels?**

variable is a container
values are put in variables

variable is a "sticky note" label
value is an object in memory



Variables vs. Values

Boxes? or Labels?

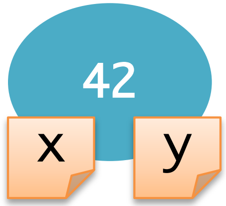
Boxes? — *variable is a container*
values are put in variables

Labels? — *variable is a "sticky note" label*
value is an object in memory

variable *value*

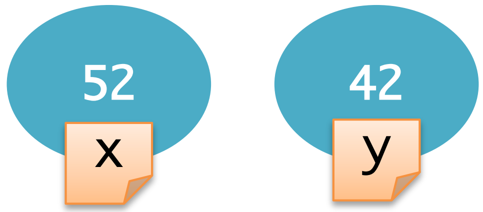
```
x <- 42

x
# [1] 42
```


```
y <- x

x
# [1] 42

y
# [1] 42
```




```
x <- 52

x
# [1] 52

y
# [1] 42
```

New number 52 created *Old number 42*



This variable-as-a-box metaphor does not match our results

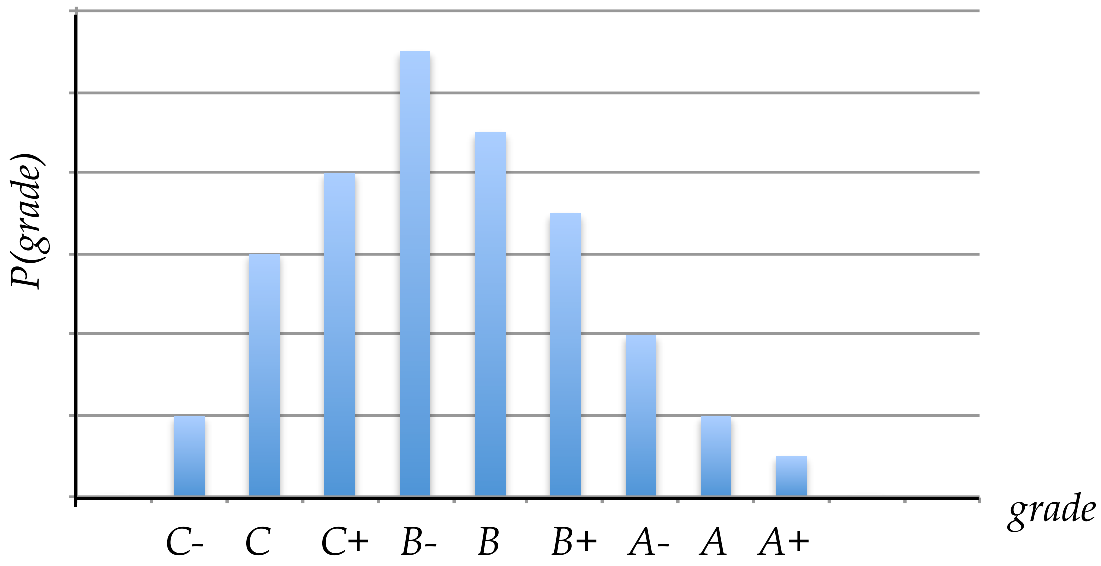
This variable-as-a-label metaphor is consistent with our results

Data Distributions

Discrete probability distribution

(useful for interval scale)

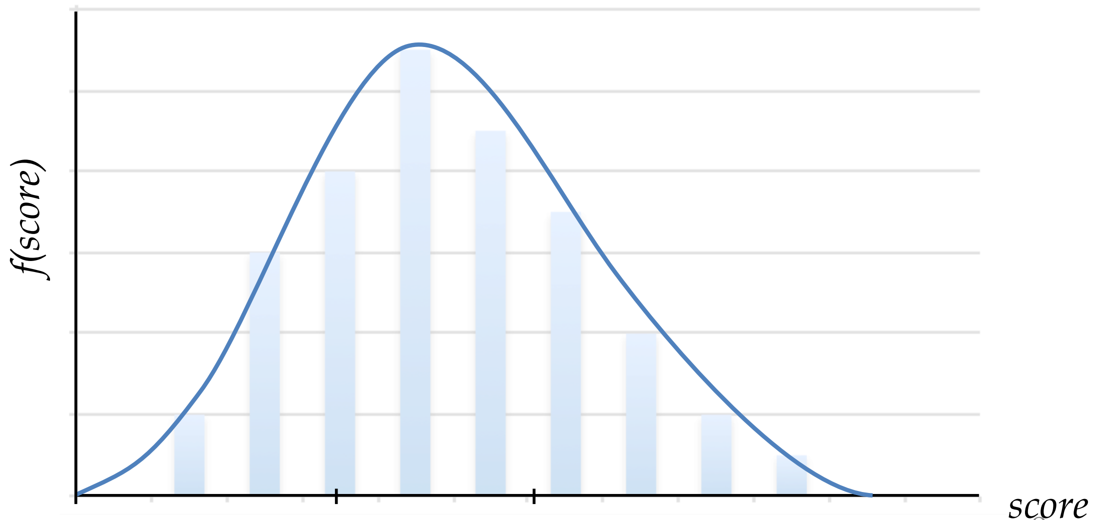
$$P(x) = \frac{occurences\ of\ an\ event}{total\ number\ of\ all\ events}$$



Continuous probability distribution

(more useful for ratio scale)

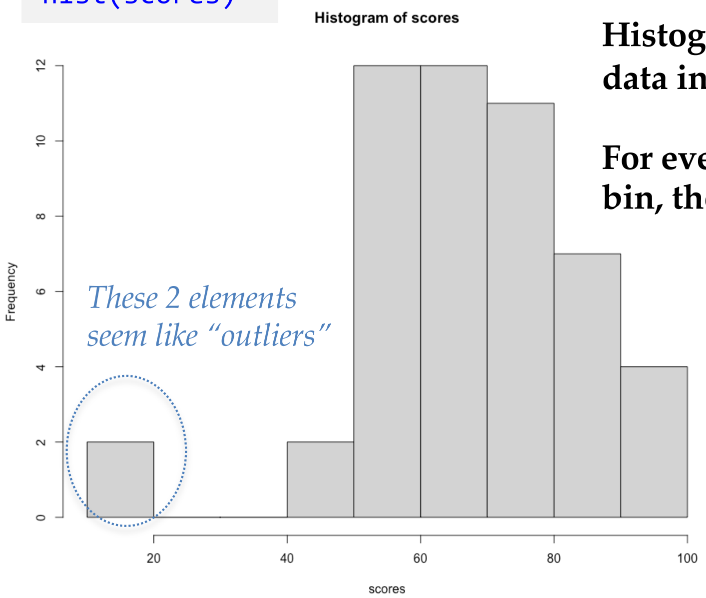
$$f(x): probability\ density\ function$$



Visualization: **Histograms**

```
exam <- read.table("exam_results.txt", header = TRUE)
scores <- exam$scores
```

```
hist(scores)
```

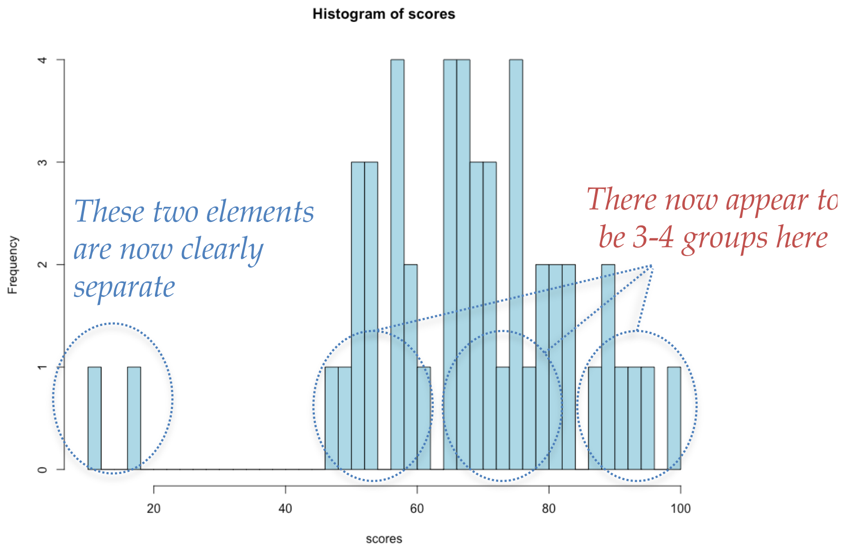


Histograms are created by *breaking* data into *bins* of equal length.

For every data element that falls in a bin, the bin's bar is 1 unit higher

`col` *changes the main color of a plot*

```
hist(scores, breaks = 50, col = "lightblue")
```




```
?hist
```

`breaks` one of:

- a vector giving the breakpoints between histogram cells,
- a function to compute the vector of breakpoints,
- a single number giving the number of cells for the histogram,
- a character string naming an algorithm to compute the number of cells (see 'Details'),
- a function to compute the number of cells.

🤔

How old is the 'histogram' visualization?

How should you find the optimal number of bins?

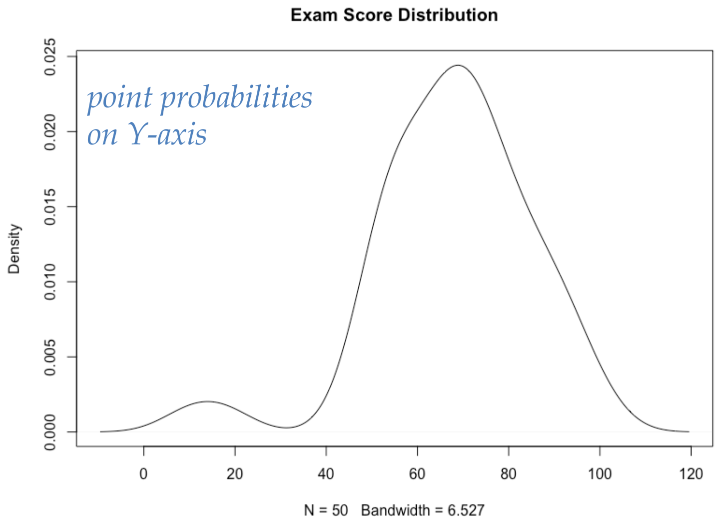
```
hist(scores, breaks = seq(0, 100, 10))

h <- hist(scores)

h$breaks            # [1]  10  20  30  40  50  60  70  80  90 100

h$counts            # [1]   2   0   0   2  12  12  11   7   4
```

🤔

*What's the **probability** of 80-90 grade?*

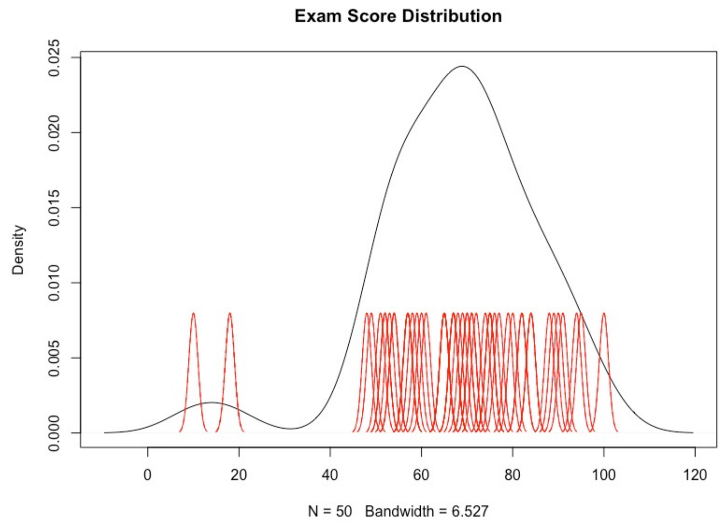
Visualization: **Density Plot**

```
plot( density(scores), main = "Exam Score Distribution")
```



Kernel density **plots show the estimated probability density function of a data series.**

Kernel: a curve function that has **area of 1**

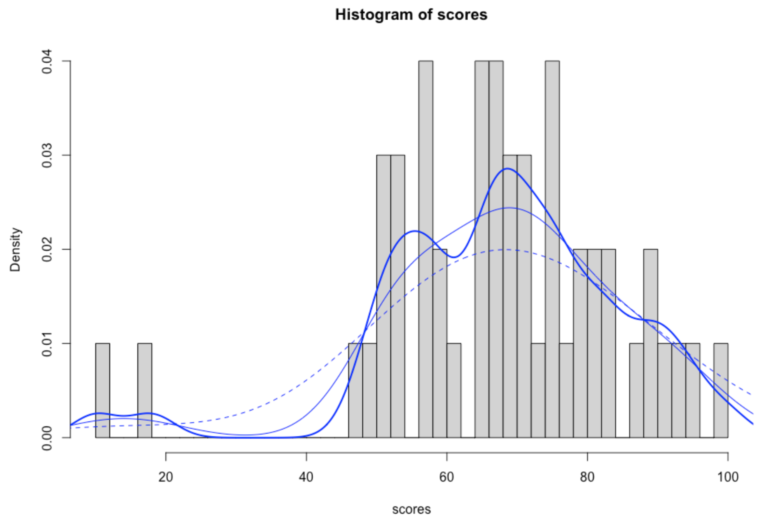


Individual kernels of each data point are combined to estimate overall kernel density.

The smoothness of a density plot can be *adjusted* by scaling the *bandwidth* of the kernel.

```
hist(scores, breaks=50, prob=TRUE)
lines(density(scores, adjust=0.5), col="blue", lwd=2)
lines(density(scores, adjust=1), col="blue")
lines(density(scores, adjust=2), col="blue", lty="dashed")
```

`lines(…)` similar *`plot(…)`* but *plots on top of existing graphics*



How should we find the optimal smoothness of bandwidth?

When should we use histogram vs. density plots?

source: http://en.wikipedia.org/wiki/Kernel_density_estimation

ggplot2 package

Follows formal "grammar of graphics" to separate graphics into layers: data, geometries, etc.

Install the package (*only needed once*)

```
install.packages("ggplot2")
```

Install the package (*every time you start the project*)

```
library(ggplot2)
```

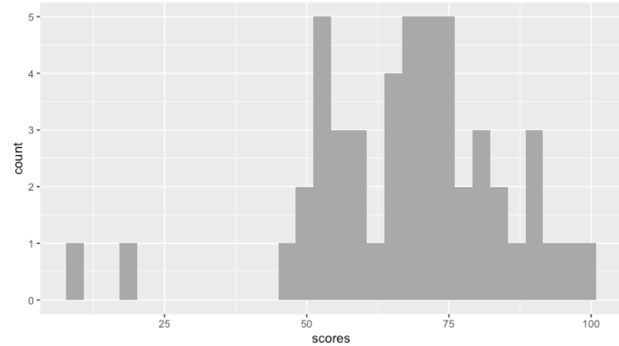
Specify the plot

```
ggplot(exam, aes(x=scores)) + geom_histogram(fill="darkgray")
```

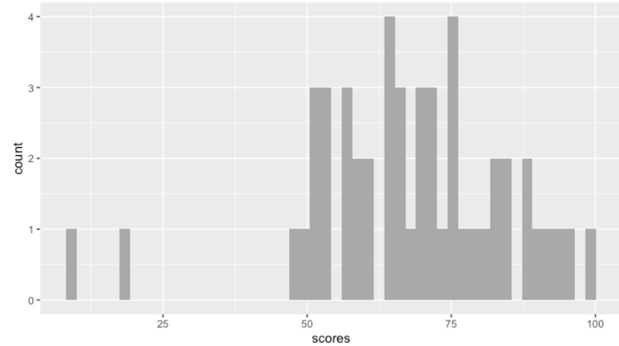
data — `ggplot(exam, aes(x=scores))`

geometric object — `geom_histogram(fill="darkgray")`

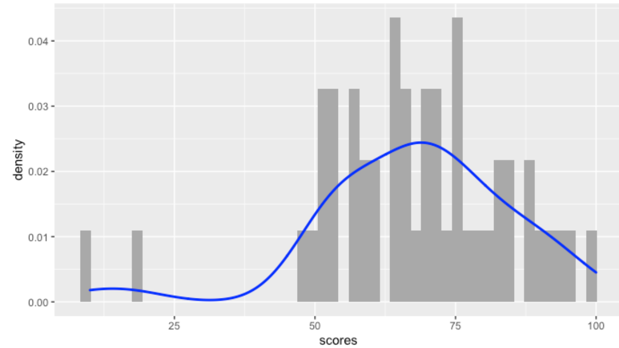
***aes**thetics define visual attributes*




```
ggplot(exam, aes(x=scores)) + geom_histogram(bins=50, fill="darkgray")
```

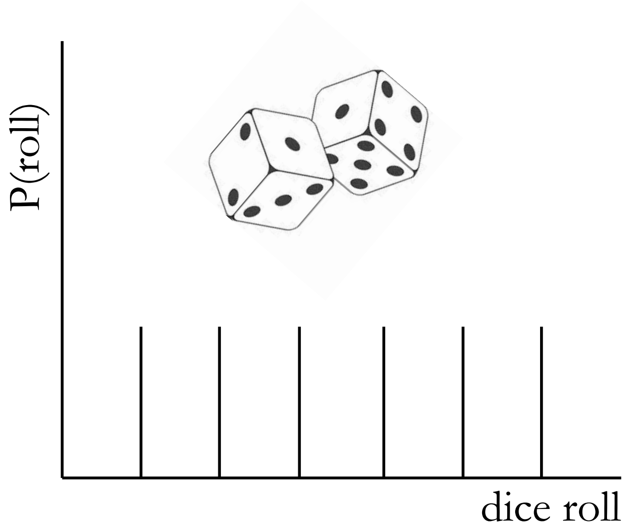



```
ggplot(exam, aes(x=scores)) +
  geom_histogram(aes(y=after_stat(density)), bins=50, fill="darkgray") +
  geom_density(color="blue", linewidth=1)
```



learn more: https://ggplot2.tidyverse.org/

Simulation: Uniform Distributions

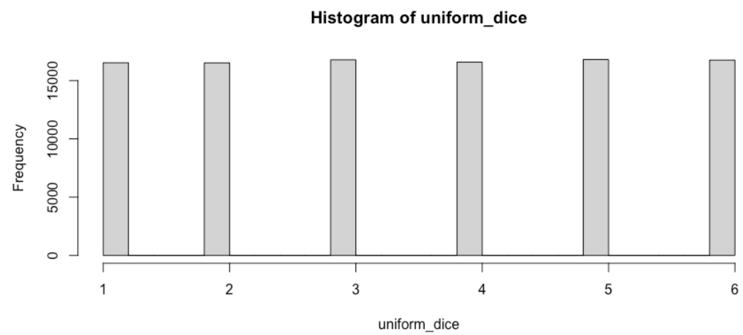


Random Uniform

```
runif(n=10)
```

```
 [1] 0.50660837 0.82407258 0.23201429 0.99551556 0.48578535
 [6] 0.80276338 0.72508096 0.08728901 0.83528430 0.80663412
```

```
uniform_dice <- round(runif(n=100000, min=0.5, max=6.5))

hist(uniform_dice)
```



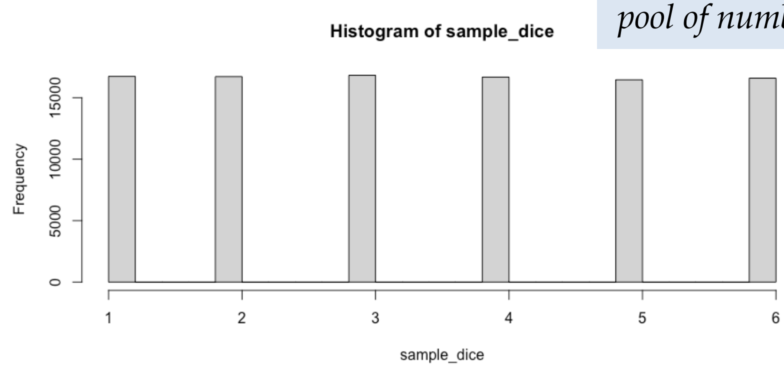
Random Sampling

sampling with replacement

The `sample()` *function uniformly picks elements from a set*

```
sample_dice <- sample(1:6, size=100000, replace=TRUE)
hist(sample_dice)
```

puts the number back in the pool of numbers to choose from



Simulation: Binomial Distribution

Imitates the behavior of a model

Binary outcomes: Experiments with **p** probability of success/failure
Repeated Attempts: Number of success in **n** experiments

number of cases

probability on each attempt

```
rbinom(n, size, prob)
```

simulates binary experiments

number of attempts each

Number of heads if **1 person** flips **1 coin**:

```
rbinom(1, 1, 0.5)
# [1] 1
```

Number of heads if **1 person** flips **5 coins**:

```
rbinom(1, 5, 0.5)
# [1] 4
```

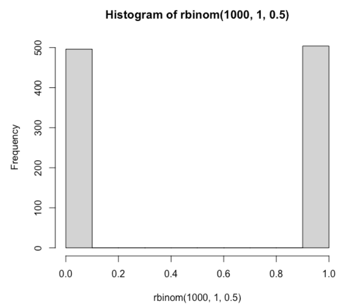
Number of heads if **10 people** flips **5 coins**:

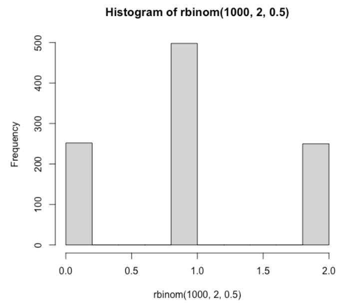
```
rbinom(10, 5, 0.5)
# [1] 3 3 3 1 2 4 3 2 0 2
```

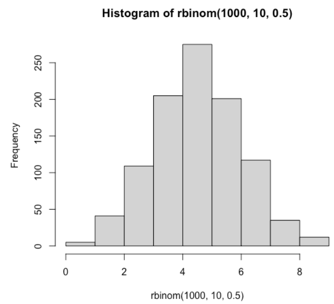
Visualizing the binomial distribution:

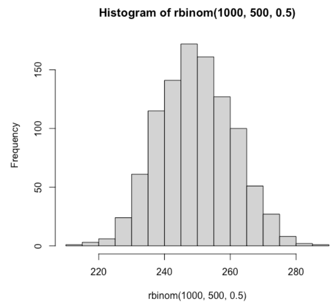
```
hist(rbinom(1000, 1,   0.5))
hist(rbinom(1000, 2,   0.5))
hist(rbinom(1000, 10,  0.5))
hist(rbinom(1000, 500, 0.5))
```

Why does a "bell shape" start to appear?





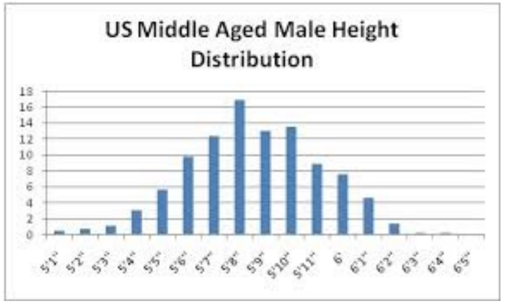




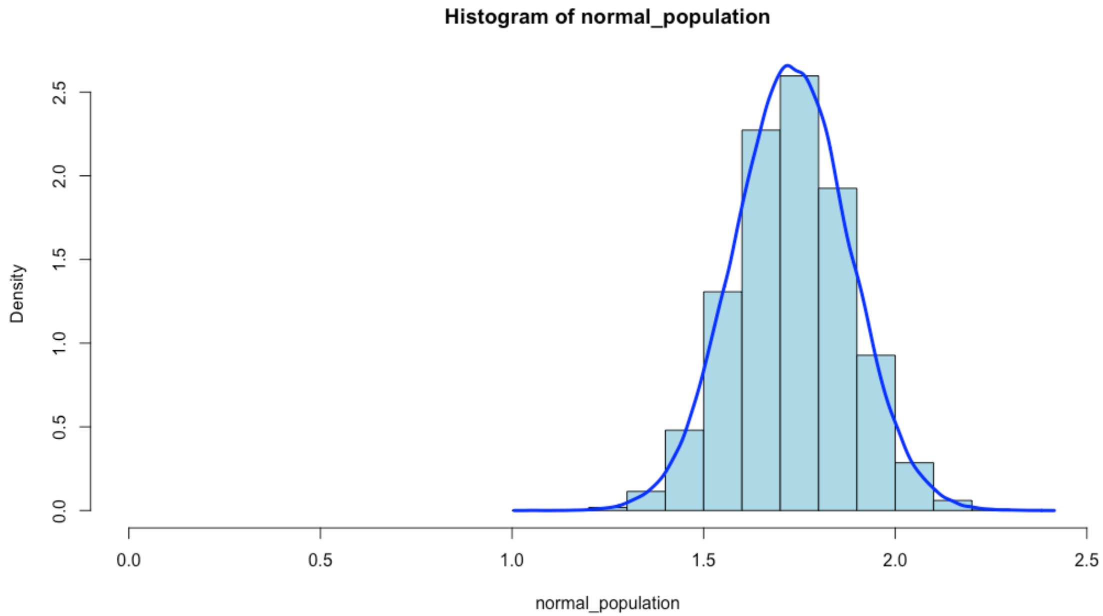
Simulation: Normal Distribution

Random Normal

Let's assume this is a normal distribution, and make another distribution like it…




```
normal_population <-  rnorm(100000, mean=1.73, sd=0.15)

hist(normal_population, xlim=c(0,2.5), col="lightblue", probability=TRUE)
lines(density(normal_population), col = "blue", lwd=3)
```

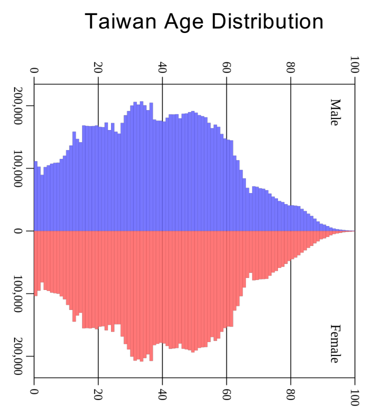


Simulation: Composite Distributions

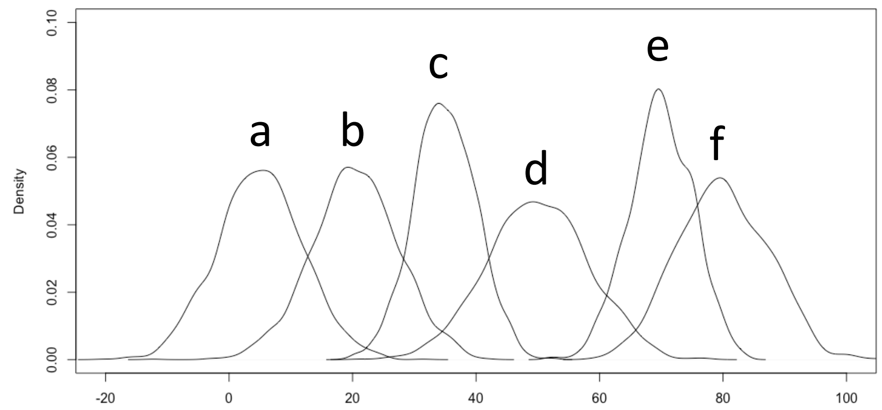
$\underbrace{\hspace{6em}}$

Made up of various smaller parts

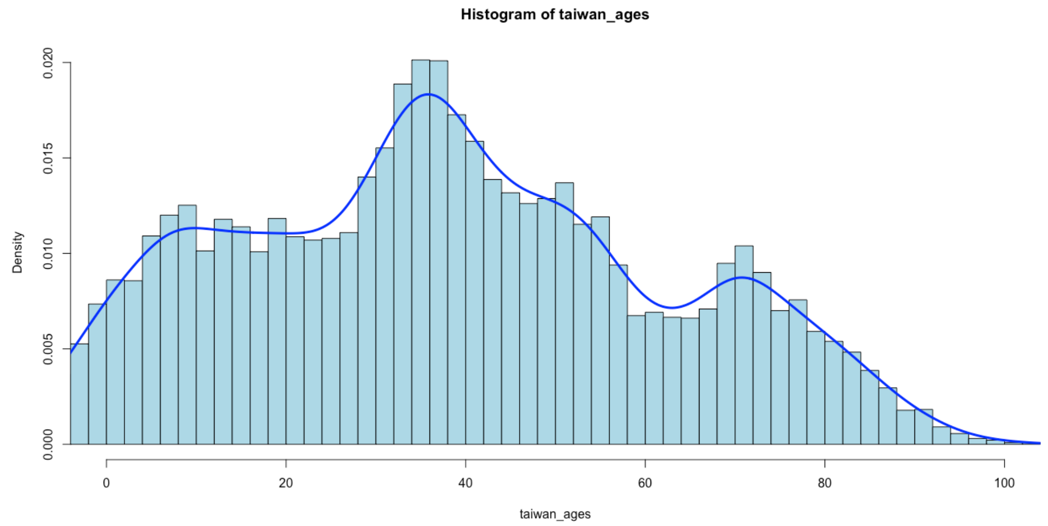
Let's create an artificial distribution that looks something like the Taiwan Age Distribution…




```r
# Individual datasets
a <- rnorm(2000, mean=5, sd=7)
b <- rnorm(2000, mean=20, sd=7)
c <- rnorm(2500, mean=35, sd=5)
d <- rnorm(3000, mean=50, sd=8)
e <- rnorm(1000, mean=70, sd=5)
f <- rnorm(1000, mean=80, sd=7)

# Creating a composite dataset
taiwan_ages <- c(a,b,c,d,e,f)
```




```r
hist(taiwan_ages, xlim=c(0,100), breaks=50, col="lightblue", probability=TRUE)

lines(density(taiwan_ages), col = "blue", lwd=3)
```

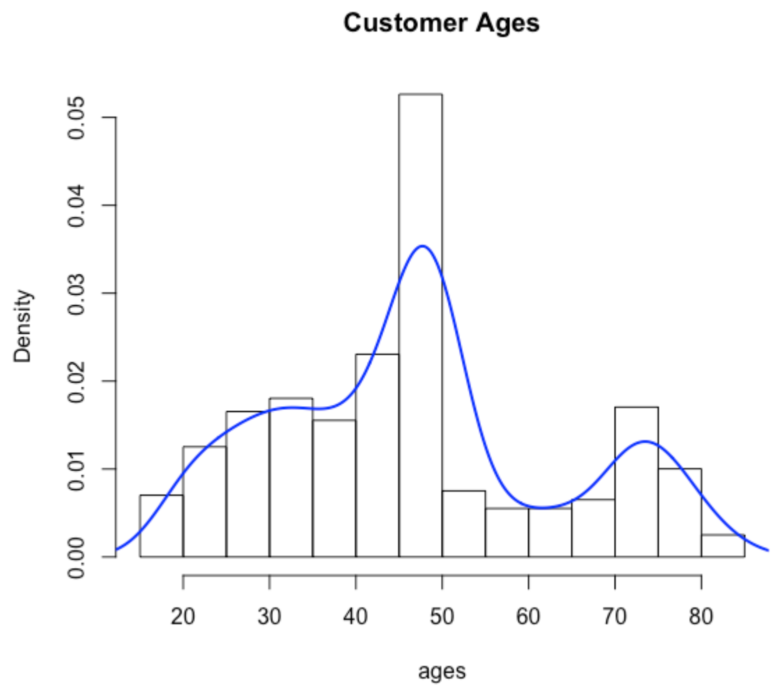


Descriptive Statistics

Let's look back at the customer ages dataset

```
customers <- read.table("customers.txt", header = TRUE)
ages <- customers$age


hist(ages, probability = TRUE, main = "Customer Ages", col="white")
lines(density(ages), col="blue", lwd=2)
```

`lines()`: *function that plots on top of existing plot*



Central Tendency

typical value, or average value, for a distribution of numbers

Dispersion

how spread, or stretched, out is a distribution of numbers

Measures of Central Tendency

```
sort(ages)
  [1] 18 19 19 19 19 19 19 19 19 20 20 20 20 20 21 21 21 21 21 21 21 22 22 23 23 23 23 23 23 24 24 24 25 25 25 25 25 25 25 26 26 26
 [43] 26 26 26 26 26 26 27 27 27 27 27 28 28 28 28 28 29 29 29 29 29 29 30 30 30 30 30 30 30 30 31 31 31 31 31 31 31 31 32 32 32 32
 [85] 32 32 32 32 33 33 33 33 33 34 34 34 34 34 34 34 34 34 35 35 35 35 35 35 36 36 36 36 36 37 37 37 37 37 37 37 37 38 38 38 38 38
[127] 38 39 39 39 39 39 40 40 40 40 40 40 40 41 41 41 41 41 41 42 42 42 42 42 42 42 42 43 43 43 43 43 43 44 44 44 44 45 45 45 45 45
[169] 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 47 47 47 47 47 47 47 47 47 47 47 47 47 47 47
[211] 47 47 47 48 48 48 48 48 48 48 48 48 48 48 48 48 48 48 48 48 48 48 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49 49
[253] 49 49 49 49 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 51 51 51 51
[295] 51 51 52 52 52 53 53 53 53 54 55 56 56 57 57 57 57 58 58 59 60 60 62 62 62 62 62 63 63 63 64 64 65 66 67 67 67 68 68 69 70 70
[337] 70 70 70 70 71 71 71 71 71 71 72 72 72 72 72 72 72 72 73 73 73 73 73 73 73 73 74 74 74 74 74 74 75 75 75 75 75 75 76 76 76 76
[379] 76 76 77 77 77 77 78 78 78 78 79 79 79 79 80 80 81 82 82 83 85
```

Mean : *the mean of the data you have*

$$\bar{x} = \frac{\sum x_i}{n}$$

```
n <- length(ages)
sum(ages) / n
[1] 46.80702

mean(ages)
[1] 46.80702
```

Our code looks like our statistical formulation!

The greek letter $\boldsymbol{\mu}$ *is pronounced 'mu'*

Median

```
median(ages)
[1] 47
```

```
median(c(1, 2))
[1] 1.5
```

For even numbered sets, median is the average of the middle two elements

Population Mode

```
# NOTE: no function to calculate mode in R!!!
```

Dispersion

Describing Dispersion

Range = maximum - minimum

```
summary(ages)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  18.00   34.00   47.00   46.81   52.50   85.00 

max(ages) - min(ages)       # 67
```

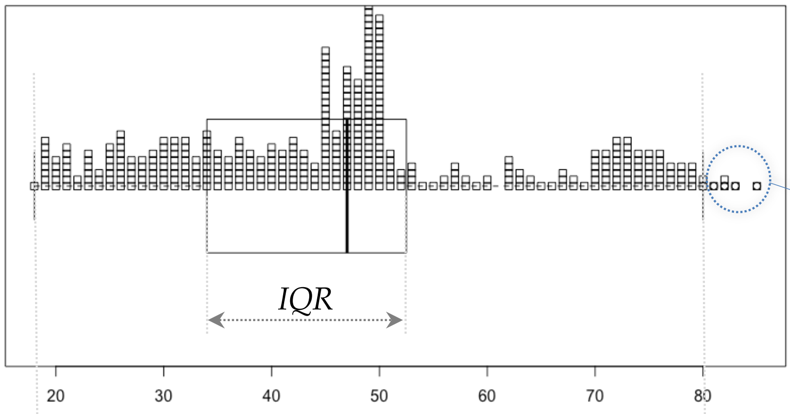
Interquartile range (IQR) $= Q_3 - Q_1$

```
quantile(ages, 3/4)
 75% 
52.5
```

use unname() *to remove name header from results*

```
Q1 <- quantile(ages, 1/4)   # 34.00
Q3 <- quantile(ages, 3/4)   # 52.50

iqr = unname(Q3 - Q1)             # 18.5

IQR(ages)                   # 18.5
```

```
visual <- boxplot(ages, horizontal = TRUE)
stripchart(ages, method="stack", add=T)
```



Q1 - 1.5*iqr

6.25

Q1 + 1.5*iqr

80.25

***Outliers**: more than 1.5 IQR away from edge of box*

```
visual$out
[1] 85 83 82 81 82
```

Detecting Outliers

Outliers are beyond:

```
lower:  Q1 - 1.5(IQR)
upper:  Q3 + 1.5(IQR)
```

```
lower_limit <- Q1 - 1.5*iqr
upper_limit <- Q3 + 1.5*iqr

ages[ ages < lower_limit | ages > upper_limit ]

  [1] 81 82 82 83 85
```

pipe symbol ' | *' means* <condition1> **OR** <condition2>

Read More: https://math.stackexchange.com/questions/966331/why-john-tukey-set-1-5-iqr-to-detect-outliers-instead-of-1-or-2

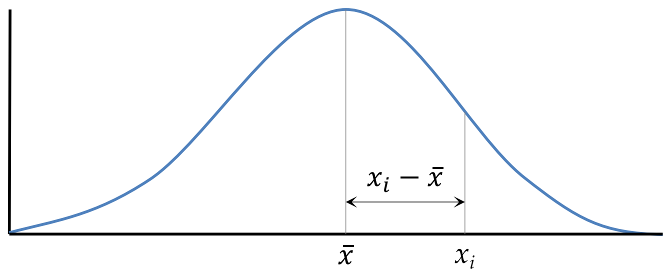
Elements of sample: x_i

```
ages
  [1] 49 69 41 73 45 71 50 43 70 32 47 77 64 50 50 45 49 47 62 50 47 72 47 63 21 49
       .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

[365] 23 74 31 20 50 30 82 70 43 20 50 48 18 45 62 41 71 19 73 26 75 41 46 49 49 23
[391] 74 53 23 51 71 50 50 67 74
```

Deviation of x_i

$$x_i - \bar{x}$$

```
ages - mean(ages)
  [1]   2.19  22.19  -5.81  26.19  -1.81  24.19   3.19  -3.81  23.19 -14.81   0.19
         .      .      .      .      .      .      .      .      .      .      .
[386]  -5.81  -0.81   2.19   2.19 -23.81  27.19   6.19 -23.81   4.19  24.19   3.19
[397]   3.19  20.19  27.19
```



Absolute Deviation of x_i

$$|x_i - \bar{x}|$$

```
abs(ages - mean(ages))
  [1]  2.19 22.19  5.81 26.19  1.81 24.19  3.19  3.81 23.19 14.81  0.19 30.19 17.19
        .     .     .     .     .     .     .     .     .     .     .     .     .
[378]  1.81 15.19  5.81 24.19 27.81 26.19 20.81 28.19  5.81  0.81  2.19  2.19 23.81
[391] 27.19  6.19 23.81  4.19 24.19  3.19  3.19 20.19 27.19
```

Mean Absolute Deviation (MAD)

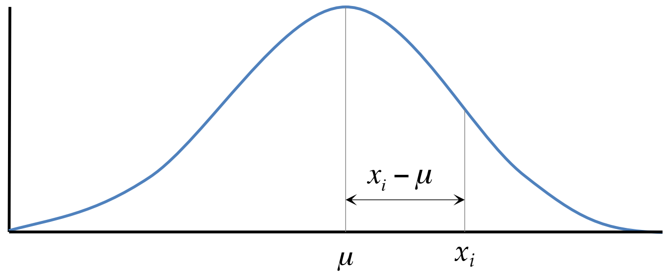
$$\frac{\sum|x_i - \bar{x}|}{n}$$

```
sum(abs(ages - mean(ages))) / length(ages)
[1] 12.66948

mean(abs(ages - mean(ages)))
[1] 12.66948
```

Median Absolute Deviation (MAD)

```
median(abs(ages - mean(ages)))
[1] 12.66948

mean(abs(ages - mean(ages)))
[1] 12.66948
```

Recall: *What is the average "difference between each age and the mean age"?*



Variability *(sum of squared deviations):*

sum of squares (SS)

$$\sum (x_i - \bar{x})^2$$

```
(ages - mean(ages))^2
  [1]    4.81  492.53   33.72  686.07    3.27  585.30   10.20   14.49  537.91
         .      .       .       .      .      .      .      .      .
[388]    4.81    4.81  566.77  739.46   38.35  566.77   17.58  585.30   10.20
[397]   10.20  407.76  739.46
```

```
sum((ages - mean(ages))^2)
[1] 106652.1
```

Variance

mean sum of squares (MSS)

$$s^2 = \frac{\sum (x_i - \bar{x})^2}{n-1}$$

$$s^2 = \frac{SS}{df}$$

```
variance = sum((ages - mean(ages))^2) / (length(ages) - 1)
[1] 267.9702
```

Note: population variance uses population mean (μ) and size (N)

$$\sigma^2 = \frac{\sum (x_i - \mu)^2}{\mathrm{N}}$$

*What are the **units** of variability, variance, and standard deviation?*

Standard Deviation

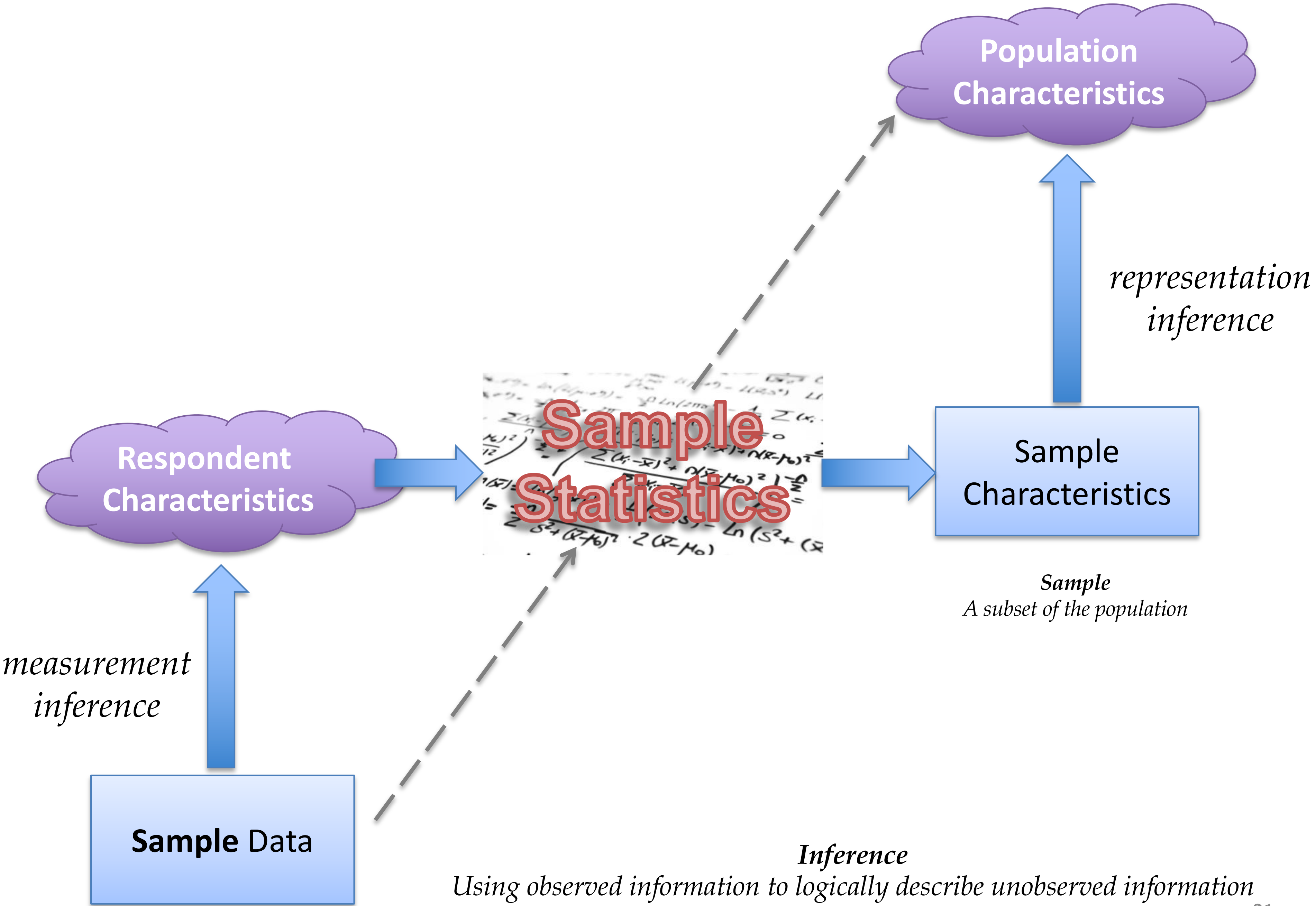
$$s = \sqrt{s^2}$$

```
sqrt(variance)
[1] 16.3698
```

```
sd(ages)
[1] 16.3698
```

*If we have **Standard Deviation**, why do we need any other measures of dispersion?*

Statistical Inference: from ***sample*** *to* ***population***



Sample
A subset of the population

Inference
Using observed information to logically describe unobserved information
Using imperfect information to describe larger, more abstract ideas

Simulation: Population vs. Sample

```
# Population of people who have seen a movie (10,000,000 people)
population_movie_ratings <- round(rnorm(mean=50, sd=9, n=10000000))
summary(population_movie_ratings)
```

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0      44      50      50      56     100
```

rnorm(…): Draws data randomly from normal distribution

```
# Sample of people we have asked (450)
sample_movie_ratings <- sample(population_movie_ratings, size=450)
summary(sample_movie_ratings)
```

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
25.00   43.00   49.00   48.99   55.00   76.00
```

sample(…): Randomly selects elements from data,
Each element has equal chance of selection

```
# Visualize population vs. sample
plot(density(population_movie_ratings, adjust=2),
     col="blue", lwd=2,
     main="population versus sample")

lines(density(sample_movie_ratings), lty="dashed", lwd=1)
```

```
boxplot(population_movie_ratings,
        sample_movie_ratings,
        horizontal=TRUE)
```