Introduction to Statistics in R
by Maggie Matsui and datacamp

descriptive statistics > describe and summarize data
inferential statistics > use a sample of data to make inferences about a larger population

Most common types of data
numeric (quantitative) > numeric values
- contiunuous (measured) > ie airplane speed or time waiting in line
- discrete (counted) > ex number of pets a person has or number of packages shipped

categorical (qualitative) > valuest that belong to distinct groups
- nominal (unordered) > ie married/unmarried or county of residence
- oridinal (ordered) > ex survey questionaire with answer range from strongly disagree to strongly agree

*categorical data can be given numbers as placeholders

Mean with R
often called the 'average'
add up all the numbers of interest and divide by the total number of data points
how to get using R > ex mean of hours slept by a group of mammals
mean(msleep$sleep_total)

Median
value where 50% of the data is lower than it, and 50% of the data is higher
with R:
#sort all the data in this case from least to greatest
sort(msleep$sleep_total)
#find the middle > in this example it is index 42
sort(msleep$sleep_total)[42]

Mode
the most frequent value in the dataset
in R:
#count how many occurrences and sort in descending order
msleep %>% count(sleep_total, sort = TRUE)
*mode is often used for categorical variables since they are often unordered and have no inherent numerical representation
#for another variable
msleep %>% count(vore, sort = TRUE)

example code
```
#pull dataset
msleep %>%
#pull all the insectivores from the dataset
       filter(vore == ‘insecti’) %>%
#get the mean and median of this group
       summarize(mean_sleep = mean(sleep_total),
                           median_sleep = median(sleep_total))
```

*remember mean is more sensitive to extreme values
mean is more sensitive to extreme values so works best with symmetrical datasets
can visualize the symmetry of the data by creating a histogram

non-symmetrical data is called skewed data
left-skewed > data is piled on the right
right-skewed > data is piled on the left
for this kind of data median is a better source of center measurement
*mean is pulled in the direction of the skew
lower than the median on the left-skewed data
higher than the median on the right-skewed data

Example
```
# Filter for Belgium
belgium_consumption <- food_consumption %>%
  filter(country == "Belgium")

# Filter for USA
usa_consumption <- food_consumption %>%
  filter(country == "USA")

# Calculate mean and median consumption in Belgium
mean(belgium_consumption$consumption)
median(belgium_consumption$consumption)

# Calculate mean and median consumption in USA
mean(usa_consumption$consumption)
median(usa_consumption$consumption)

food_consumption %>%
  # Filter for Belgium and USA
  filter(country %in% c("Belgium", "USA")) %>%
  # Group by country
```

```
  group_by(country) %>%
  # Get mean_consumption and median_consumption
  summarize(mean_consumption = mean(consumption),
            median_consumption = median(consumption))

food_consumption %>%
  # Filter for rice food category
  filter(food_category == "rice") %>%
  # Create histogram of co2_emission
  ggplot(aes(co2_emission)) +
    geom_histogram()

food_consumption %>%
  # Filter for rice food category
  filter(food_category == "rice") %>%
  # Get mean_co2 and median_co2
  summarize(mean_co2 = mean(co2_emission),
            median_co2 = median(co2_emission))
```

Variance
measures the average distance from each data point to the data's mean
calculating with R:

```
#calculate the distances between each point and the mean to get one number for every data point
dists <- msleep$sleep_total - mean(msleep$sleep_total)
#print this calculation
dists
#then square each distance and add them together
squared_dists <- (dists)^2
sum_sq_dists <- sum(squared_dists)
sum_sq_dists
#divide the sum of squared distances by the number of data points minus 1 (our example 83 points)
sum_sq_dists/82
```

**remember the units of variance are squared
*the higher the variance, the larger the spread

We can calculate all of this using the 'var' function

```
var(msleep$sleep_total)
```

Standard deviation is calculated by taking the square root of the variance
in R:

sqrt(var(msleep$sleep_total))

Can also calculate using the ‘sd’ function
sd(msleep$sleep_total)
*standard deviation units are not squared
this is one of the reasons when measuring spread standard deviation is preferred

Mean absolute deviation
takes the absolute value of the distances to the mean, and then takes the mean of those differences
dists <- msleep_total - mean(msleep$sleep_total)
mean(abs(dists))

Standard deviation vs. mean absolute deviation
SD squares distances, penalizing longer distances more than shorter ones
MAD penalizes each distance equally

Quartiles
split up the data into four equal parts
using R:
quantile(msleep$sleep_total)

Boxplots use quartiles
ggplot(msleep, aes(y = sleep_total)) + geom_boxplot()

Percentiles
we can split the data into other equal parts besides quarters
we can do this using the ‘probs’ argument
the ‘probs’ argument takes in a vector of proportions
quantile(msleep$sleep_total, probs = c(0, 0.2, 0.4, 0.6, 0.8, 1))

A shortcut via the ‘seq’ function
takes in the lowest number, the highest number, and the number to jump by
seq(from, to, by)
example:
quantile(msleep$sleep_total, probs = seq(0, 1, 0.2))

Interquartile range (IQR)
*also the height of the box in the boxplot
quantile(msleep$sleep_total, 0.75) - quantile(msleep$sleep_total, 0.25)

Outlier
data point that is substantially different from the others

how to know what is substantially different?
rule of thumb:
data < Q1 - 1.5 x IQR
or
data > Q3 + 1.5 x IQR

Finding outliers using R:

```
iqr <- quantile(msleep$bodywt, 0.75) - quantile(msleep$bodywt, 0.25)
lower_threshold <- quantile(msleep$bodywt, 0.25) - 1.5 * iqr
upper_threshold <- quantile(msleep$bodywt, 0.75) + 1.5 * iqr
#now filter to find the outliers
msleep %>% filter(bodywt < lower_threshold | bodywt > upper_threshold) %>%
       select(name, vore, sleep_total, bodywt)
```

Example

```
# Calculate variance and sd of co2_emission for each food_category
food_consumption %>%
  group_by(food_category) %>%
  summarize(var_co2 = var(co2_emission),
          sd_co2 = sd(co2_emission))

# Create subgraphs for each food_category: histogram of co2_emission
ggplot(food_consumption, aes(co2_emission)) +
  # Create a histogram
  geom_histogram() +
  # Create a separate sub-graph for each food_category
  facet_wrap(~ food_category)

# Calculate total co2_emission per country: emissions_by_country
emissions_by_country <- food_consumption %>%
  group_by(country) %>%
  summarize(total_emission = sum(co2_emission))

# Compute the first and third quartiles and IQR of total_emission
q1 <- quantile(emissions_by_country$total_emission, 0.25)
q3 <- quantile(emissions_by_country$total_emission, 0.75)
iqr <- q3 - q1

# Calculate the lower and upper cutoffs for outliers
lower <- q1 - 1.5 * iqr
upper <- q3 + 1.5 * iqr

# Filter emissions_by_country to find outliers
```

```
emissions_by_country %>%
  filter(total_emission < lower | total_emission > upper)
```

Measuring Chance
What's the probability of an event?
P(event) = # ways event can happen / total # of possible outcomes
Probability is always between zero and 100 percent
if probability of something is zero than it is impossible
100% something will certainly happen

Sampling from a data frame using R:
#data frame is sales_counts
#'sample_n' function takes in a data frame and the number of rows we want to pull out
#sample_n chooses randomly
```
sales_counts %>% sample_n(1)
```

Set a random seed
using R (example seed 5 - can be any number):
```
set.seed(5)
```
to ensure we get the same results each time we run the script
seed is a number that R's random number generator uses as a starting point so that we will generate the same random value each time

Sampling without replacement
means that once an item is picked that item is not placed back into the proverbial bag to potentially get randomly picked again
above R code is sampling without replacement

Sampling with replacement
means that even if an item is picked that item is placed back into the proverbial bag to potentially get randomly picked again
with R:
```
sales_counts %>% sample_n(2, replace = TRUE)
```
or another example using 5 samples
```
sample(sales_team, 5, replace = TRUE)
```

*Independent events
two events are independent if the probaility of the second event isn't affected by the outcome the first event
in general sampling with replacement, each event is independent

*dependent events

two events are dependent if the probability of the second event is affected by the outcome of the first event
ie sampling without replacement

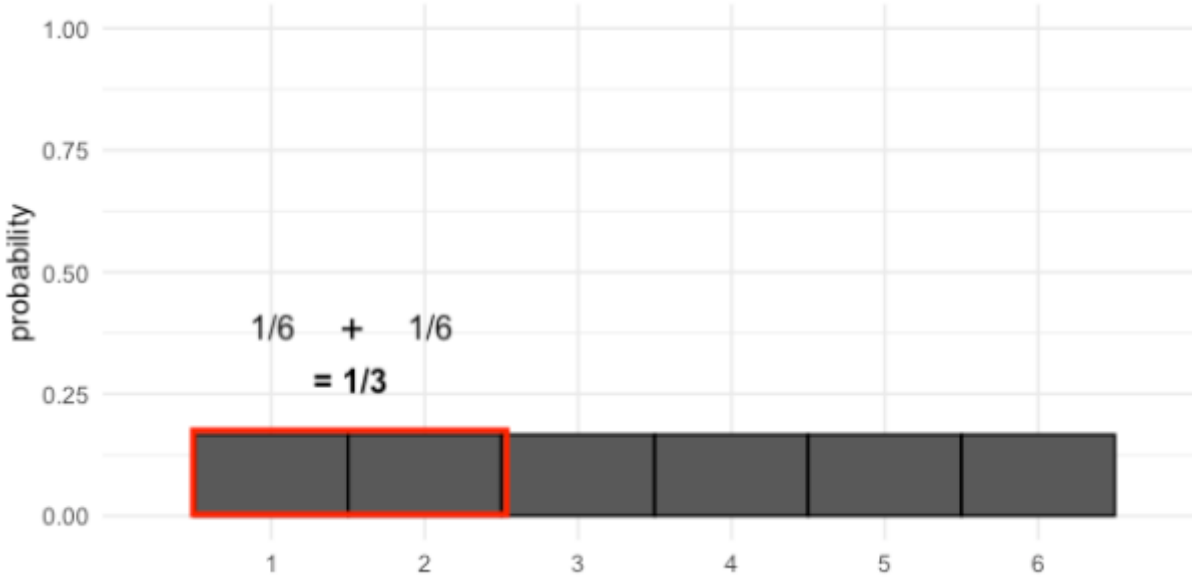
Example
```
# Calculate probability of picking a deal with each product
amir_deals %>%
  count(product) %>%
  mutate(prob = n/sum(n))
#mutate() creates a new column
#name of new column here is 'prob'
```

Probability distribution
describes the probability of each possible outcome in a scenario
'expected value' of a distribution is the mean of a probability distribution
calculate this by multiplying each value by its probability
example

Expected value of a fair die roll =

$$(1 \times \tfrac{1}{6}) + (2 \times \tfrac{1}{6}) + (3 \times \tfrac{1}{6}) + (4 \times \tfrac{1}{6}) + (5 \times \tfrac{1}{6}) + (6 \times \tfrac{1}{6}) = 3.5$$

*probabiliy = area
calculate probabilities of different outcomes by taking areas of the probability distribution
example (what is probability that a die roll is less than 2?)




this a discrete uniform distribution
all outcomes have the same probability

Sampling from discrete distributions using R
using data frame ‘die’

```
rolls_10 <- die %>%
      sample_n(10, replace = TRUE)
rolls_10
```

Visualize this example

```
ggplot(rolls_10, aes(n)) + geom_histogram(bins = 6)
```

*remember law of large numbers > as the size of you sample increases, the sample mean will approach the expected value

Example

```
# Create a histogram of group_size
ggplot(restaurant_groups, aes(group_size)) +
  geom_histogram(bins = 5)

# Create probability distribution
size_distribution <- restaurant_groups %>%
  count(group_size) %>%
  mutate(probability = n / sum(n))

# Calculate probability of picking group of 4 or more
size_distribution %>%
  # Filter for groups of 4 or larger
  filter(group_size >= 4) %>%
  # Calculate prob_4_or_more by taking sum of probabilities
  summarize(prob_4_or_more = sum(probability))
```

Continuous distributions
recall we use discrete distributions to model situations that involve discrete or countable variables
continuous uniform distribution can be modeled with a probability distribution
*the complexity here is that there are an infinite number of possibilities
we can use a continuous uniform distribution to represent equal opportunity for each of these infinite possibilities
example
waiting at a bus stop where the bus arrives in 12 minute intervals
can use a continuous uniform distribution
here probability still equates to area
say we want to know the probability of waiting between 4 and 7 minutes
P(4 < wait time < 7)

7-4=3 = width of our rectangle
1/12 = height of our rectangle
area = wxh
area = 3 x 1/12 = 1/4 = 25%

using R:
we can use the ‘punif’ function
used to calculate the cumulative distribution function (CDF) of a continuous uniform distribution
CDF gives you the probability that a random variable following a uniform distribution is less than or equal to a specific value
punif function arguments are:

- q which represents the quantile (ie the value for which you want to calculate the cumulative probability)
- min which represents the minimum value of the uniform distribution
- max which represents the maximum value of the uniform distribution
- lower.tail which is a logical value (True or False) indicating whether you want to calculate the probability for values less than or equal to q (True) or greater than q (False)

our example, still at the bus stop, we want to know wait time <7 minutes
punif(7, min=0, max=12)
output > 0.5833
for probability greater than 7 minutes we now need to use lower.tail argument
punif(7, min = 0, max = 12, lower.tail = False)
how about for our above example between 4 and 7 minutes
punif(7, min=0, max=12) - punif(4, min=0, max=12)

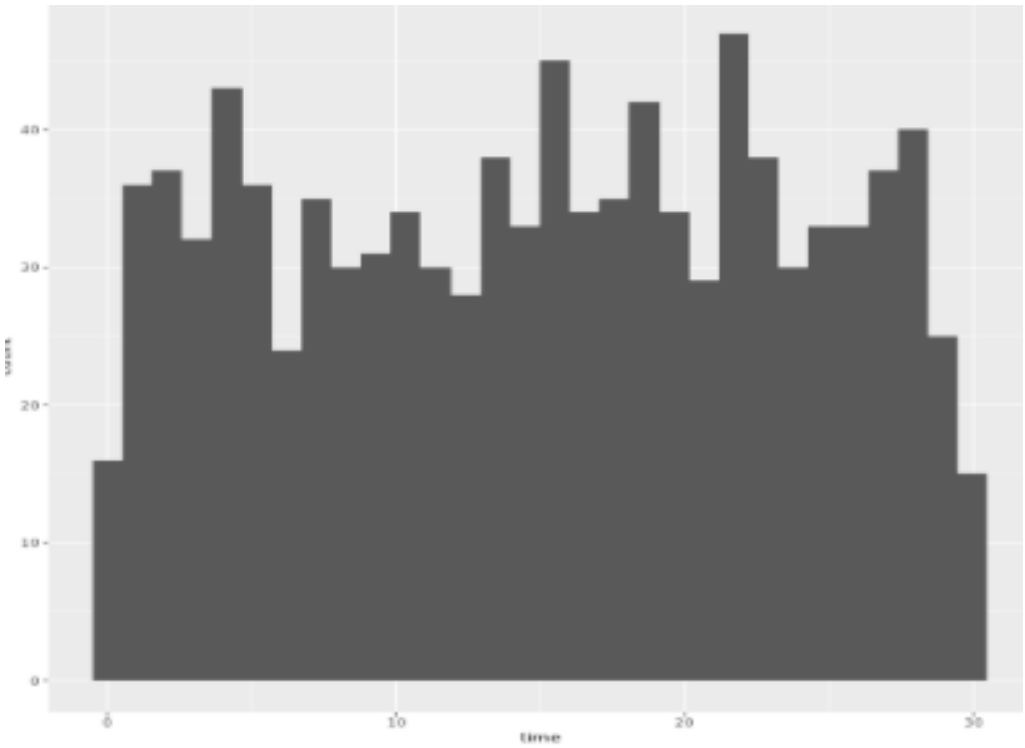
Continuous distributions do not have to be uniform
*no matter the shape the area beneath it must always equal 1

Example

```
# Set random seed to 334
set.seed(334)

# Generate 1000 wait times between 0 and 30 mins, save in time column
# use runif function to generate random numbers from a uniform distribution
wait_times %>%
  mutate(time = runif(1000, min = 0, max = 30)) %>%
  # Create a histogram of simulated times
  ggplot(aes(time)) +
    geom_histogram(bins=30)
```

output>




Binomial distribution
probability distribution of the number of successes in a sequence of independent trials
outcome has two possible values
yes/no, 0/1, success/failure, win/loss, etc
think coin flip (heads/tails)

using R:
rbinom function
arguments > (# of trials, # of coins, # probability of heads/success)
example flipping a coin
rbinom(1, 1, 0.5)
ie 1 trial, 1 coin, 50% probability for each value

another example
rbinom(10, 3, 0.5)
10 flips of 3 coins each with a prob of 50%
output > will be 10 values representing the total number of heads from each set of flips

binomial distribution can tell us the probability of getting some number of heads in a sequence of coin flips
*it is a discrete distribution since we are working with a countable outcome

described using two parameters
n: total number of performances (2nd rbinom argument)
p: probability of success (3rd rbinom argument)

Finding a specific set of successes from a certain amount of trials we use the probability mass function (PMF)
the PMF of the binomial distribution gives the probability of getting exactly ‘x’ successes in ‘size’ trials with a success probability of ‘prob’

- x is the number successes you want to calculate the probability for
- size is the number of trials or observations in each experiment
- prob is the probability of success in each trial

*we get PMH of binomial distribution in R using the dbinom(x, size, prob) function
example P(heads=7):
dbinom(7, 10, 0.5)
output > 0.117 (approx 12% chance that 7 of them will be heads)

What if we wanted to get the probability of getting a number of successes less than or equal to the first argument?
say in our example the probability of gettin 7 or fewer heads out of 10 coins
here we would use R’s pbinom function
pbinom(7, 10, 0.5)
pbinom function is used to calculate the cumulative probability of a binomial function
we call this the cumulative distribution function (CDF)
CDFs provide a way to understand how the probability of a random variable taking on a value less than or equal to a specific number changes across the range of possible values
F(x) = P(X <orequalto x)
where the probability of X takes on a value less than or equal to x
CDFs range from 0 to 1 as probabilities do
a non-decreasing function > as x increases, F(x) stays the same or increases

*can use the lower.tail argument to get the probability of a number of successes greater than the first argument
example
pbinom(7, 10, 0.5, lower.tail=FALSE)

Expected value of the binomial distribution can be calculated by multiplying n*p
EV = n x p
example, expected number of heads when we flip 10 coins is 10 x 0.5 = 5

*always remember in order for binomial distribution to apply, each trial must be independent

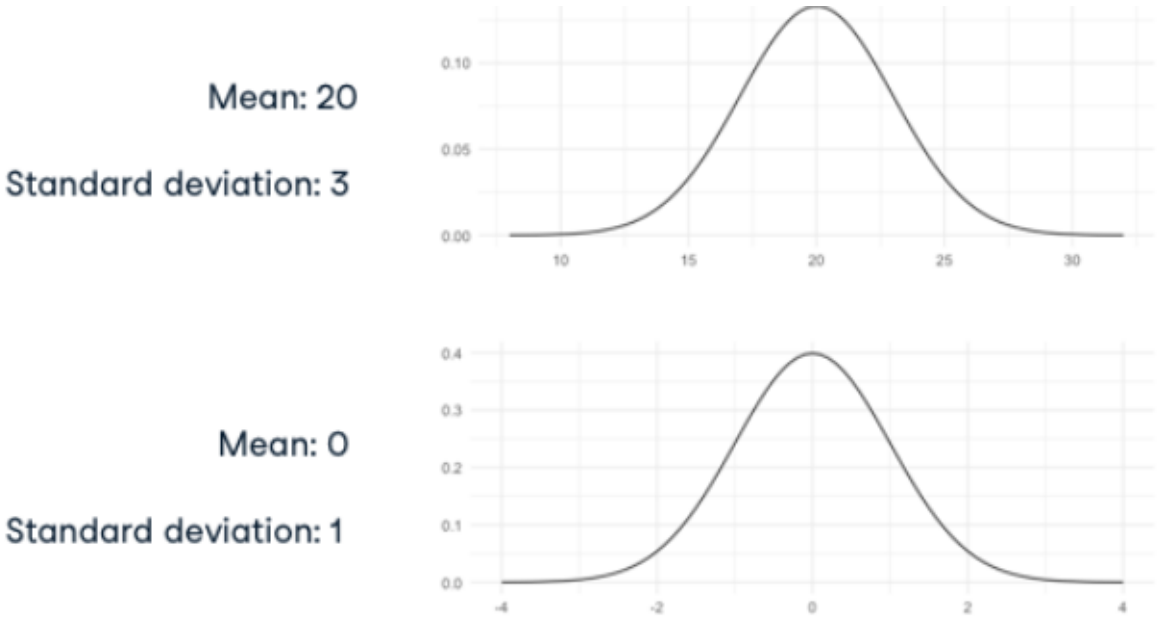
meaning one trial does not have an effect on the other

Example
# Set random seed to 10
set.seed(10)

# Simulate 52 weeks of 3 deals
deals <- rbinom(52, 3, 0.3)

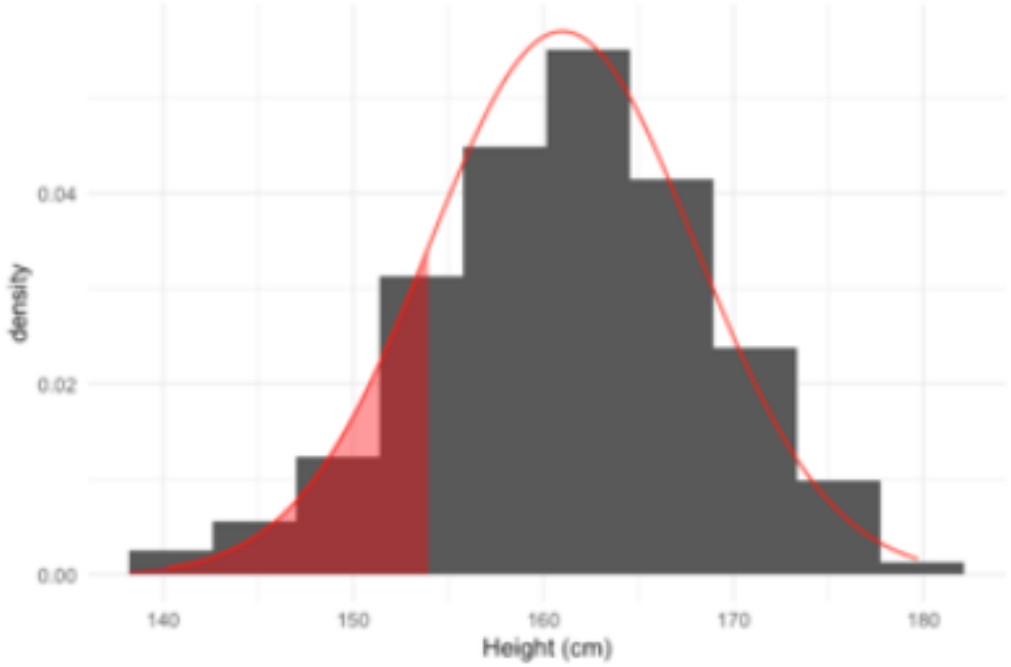
# Calculate mean deals won per week
mean(deals)

Normal distribution
‘bell curve’
symmetrical
left side is a mirror image of right side
area beneath the curve is 1
the probability never hits 0 > 0.006% of its area is contained beyond the edges of this graph
normal distribution is described by its mean and std
what does this mean? >




*when mean is 0 and a standard deviation of 1 on a normal distribution, it is called standard normal distribution
68% of area falls within 1 std of mean
95% of area falls within 2 std of mean
99.7 of area falls within 3 std of mean
this is referred to as the 68-95-99.7 rule

Alof of RWD falls naturally into the normal distribution
we can use this distribution to answer lots of questions
we can use the pnorm function
the pnorm function is used to calculate the CDF of the standard normal distribution
this is also referred to as the Z-distribution
pnorm allows you to calculate the probability that a random variable is less than or equal to a specified value (usually denoted as 'x')
pnorm returns the area under the standard normal curve to the left of the given value 'x'
exampe percent of woman shorter than 154cm
pnorm(154, mean = 161, sd = 7)
output > 0.159 (approx 16% of woman are shorter than 154cm)




for taller (ie the area to the right of x)
pnorm(154, mean=161, sd=7, lower.tail=FALSE)
output > 0.841
for between 154cm and 157cm
pnorm(157, mean=161, sd=7) - pnorm(154, mean=161, sd=7)

qnorm can be used to calculate percentiles of the standard normal distribution
this function allows you to find values of a random variable that correspond to specific probabilities or percentiles in the standard normal distribution
qnorm(0.9, mean=161, sd=7)
output > 169.97 (approx 90% of women are shorter than 170cm)

find the height where 90% of women are taller than
qnorm(0.9, mean=161, sd=7, lower.tail=FALSE)
ouput > 152.03 (approx 90% of women are taller than 152cm)

We can generate random numbers from a normal distribution using rnorm
rnorm(sample size, mean, std)
our example, generate 10 more random heights
rnorm(10, mean=161, sd=7)

The Central Limit Theorem (CLT)
this is the key to what makes the normal distribution so powerful
show in a simple example
we use the ‘c’ function to create vectors or combine multiple values into a single vector
‘c’ stands for combine or concatenate
example, creating a die and doing a sample roll of die 5 times
die <- c(1,2,3,4,5,6)
sample_of_5 <- sample(die, 5, replace = TRUE)
then take the mean
mean(sample_of_5)
now we want to repeat this process 10 times
use the replicate function
sample_means <- replicate(10, sample(die, 5, replace = TRUE) %>% mean())
if we were to continue to increase the sample size and plot we would see that the sampling distribution comes closer and closer to the normal distribution
This is the phenomenon referred to as the Central Limit Theorem
Official CLT definition > the sampling distribution of a statistic becomes closer to the normal distribution as the number of trials increases
*key factor > CLT only applies when samples ore taken randomly and are independent
CLT also applies to std
CLT also applies to distribution of the sample proportions
*since these sampling distributions are normal, we can take their mean to get an estimate of a distribution’s mean, std, or proportion
CLT comes in handy when you have a huge population and don’t have the resources to collect data on every piece of the population
with this distribution we can take a smaller sample and be confident in calculating a mean, std, or proportion that will equate to the entirety of the population

Example
# Set seed to 104
set.seed(104)

```
# Sample 20 num_users from amir_deals and take mean
sample(amir_deals$num_users, size = 20, replace = TRUE) %>%
  mean()

# Repeat the above 100 times
sample_means <- replicate(100, sample(amir_deals$num_users, size = 20, replace
= TRUE) %>% mean())

# Create data frame for plotting
samples <- data.frame(mean = sample_means)

# Histogram of sample means
ggplot(samples, aes(mean)) +
  geom_histogram(bins=10)
```

Example

```
# Set seed to 321
set.seed(321)

# Take 30 samples of 20 values of num_users, take mean of each sample
sample_means <- replicate(30, sample(all_deals$num_users, size=20) %>%
mean())

# Calculate mean of sample_means
mean(sample_means)

# Calculate mean of num_users in amir_deals
mean(amir_deals$num_users)
```

Poisson Distribution

a Poisson process is a process where events appear to happen at a certain rate but completely at random

examples

- number of animals adopted from an animal shelter
- number of people arriving at a restaurant per hour
- number of earthquakes in Californinia per year

Poisson distribution describes the probability of some number of events happening over a fixed period of time

Poisson is described by a value called lambda

lambda represents the average number of events per time interval

lambda also equates to the expected value of the distribution

*Poisson distribution is a discrete distribution since we're counting events

lambda changes the shape of the distribution

*however no matter what the distribution's peak is always at its lambda value

using R, question - P(# adoptions in a week = 5) if average adoptions per week is 8 (lambda)
dpois(5, lambda = 8)
output > 0.09 (approx 9% chance that there will be 5 adoptions in a week)

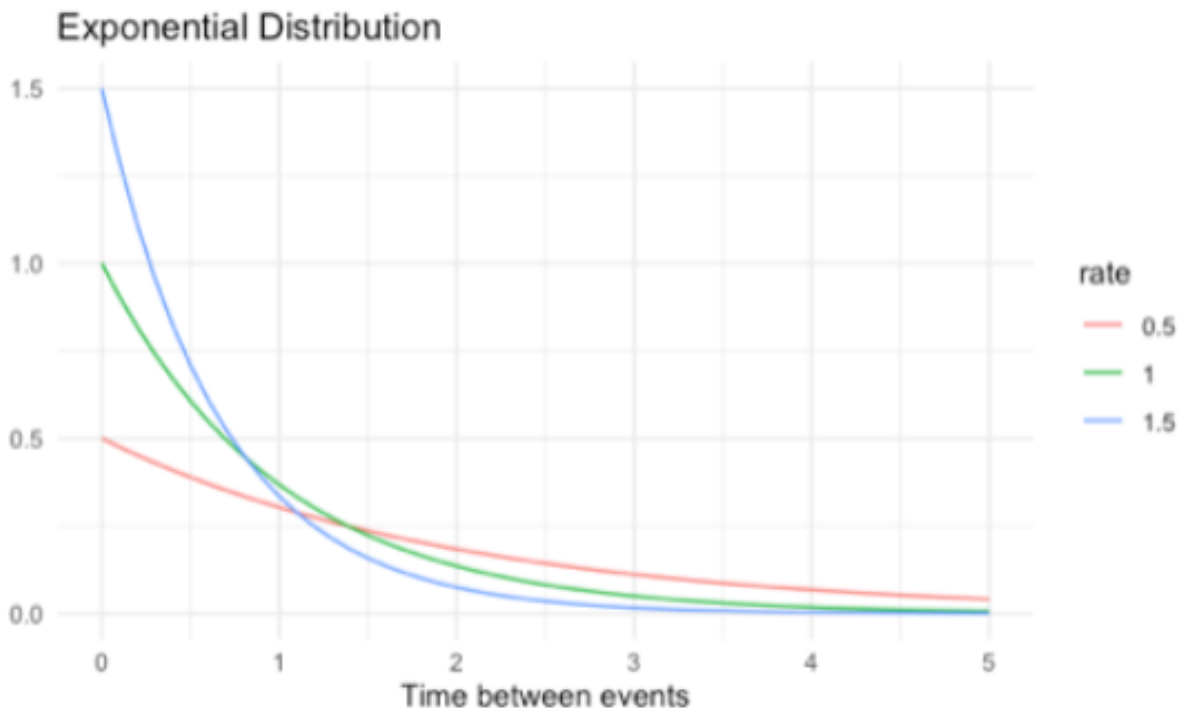
for value x or less we use the ppois function
ppois(5, lambda = 8)
output > 0.19

for value x or greater we use the ppois function with lower.tail argument set to FALSE
ppois(5, lambda = 8, lower.tail = FALSE)

generate random samples from a Poisson distribution
rpois(10, lambda=8)

*Just like other distributions, the sampling distribution of sample means of a Poisson distribution looks normal with a large number of samples

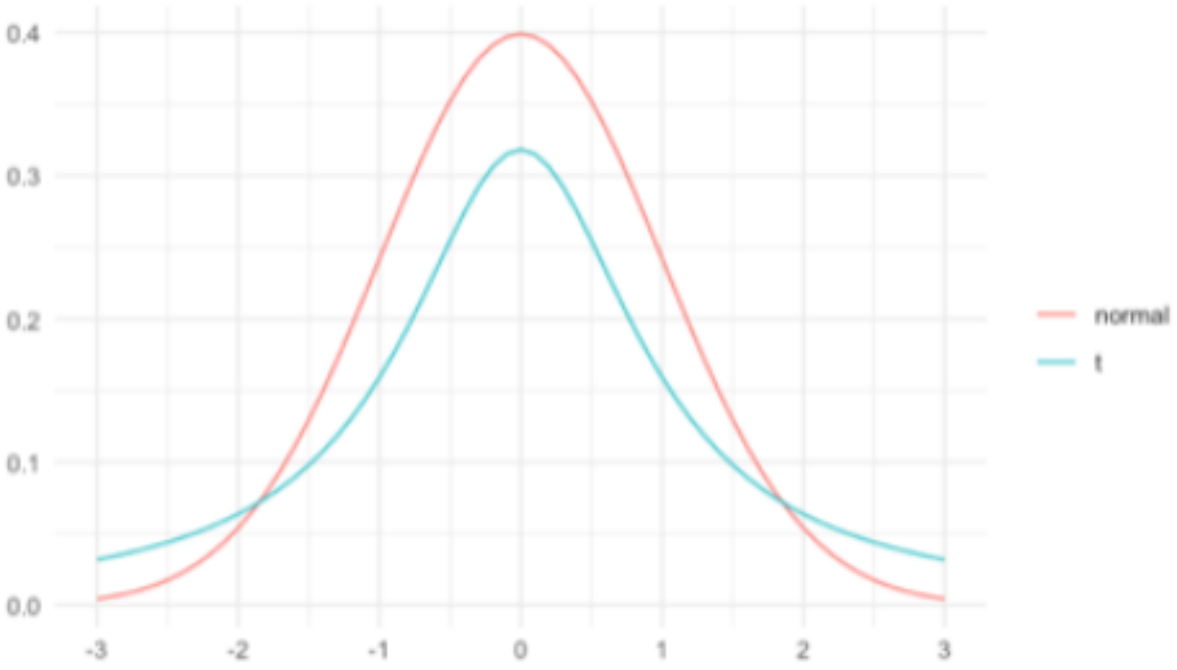
Exponential distribution
probability of time between Poisson events
continuous distribution
represented by lambda also in this context referred to as 'rate'
*rate affects the shape of the distribution and how steeply it declines



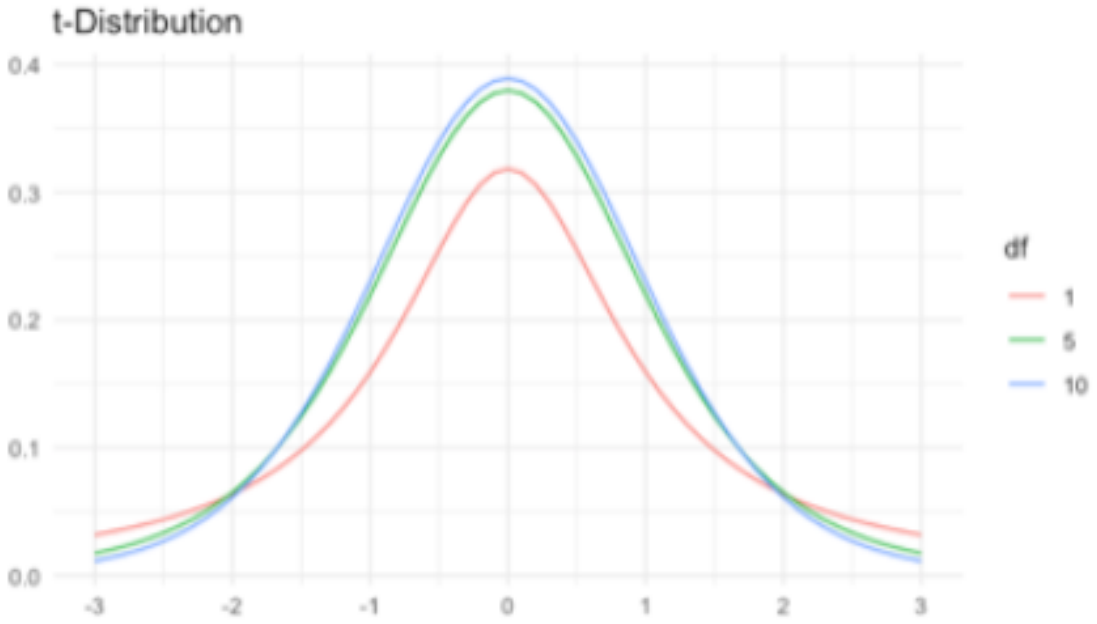

Using R:
examples
P(wait < 1min) = pexp(1, rate = 0.5)
P(wait > 4min) = pexp(4, rate = 0.5, lower.tail = FALSE)
P(1min < wait < 4min) = pexp(4, rate = 0.5) - pexp(1, rate = 0.5)

*remember that lambda is the expected value of the Poisson distribution (in terms of rate)
lambda measures frequency in terms of rate or number of events
exponential distribution (in terms of time)
expected value of the exponential distribution can be calculated by taking 1 divided by lambda

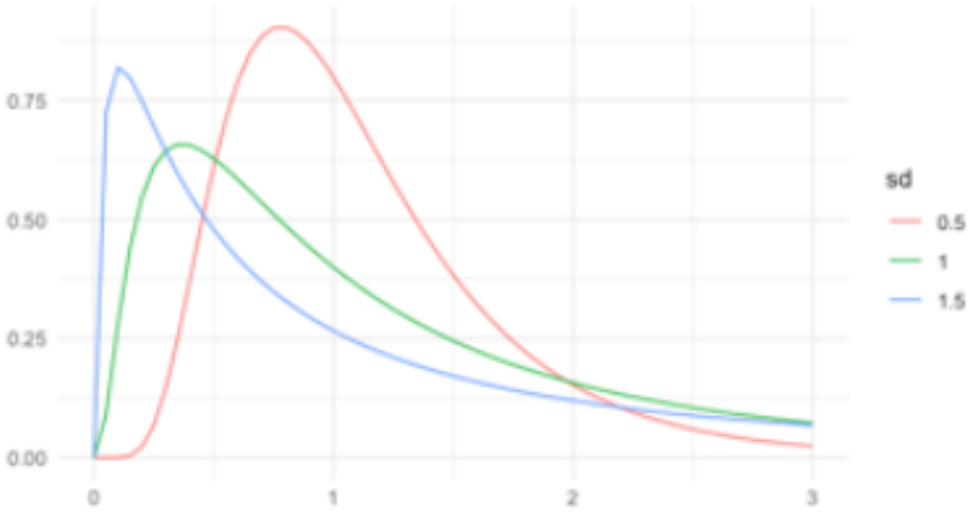
Student's t-distribution or t-distribution




similar to the normal distribution
the difference being that the t-distribution's tails are thicker
*this means the in a t-distribution observations are more likely to fall further from the mean
*t-distribution has a parameter called degrees of freedom > which affects the thickness of the distribution's tails
lower df = thicker tails and higher standard deviation



Log-normal distribution




variable whose logarithm is normally distributed
*results in distributions that are skewed
real world examples > length of chess games, blood pressure in adults

Correlation
assessing the relationship between two variables
x = explanatory/independent variable
y = response/dependent variable
correlation coefficient > quantifies the linear relationship between two variables
correlation coefficient is a number between -1 and 1
where the magnitude corresponds to the strength of the relationship between the variables
positive or negative corresponds to the direction of the relationship
coeff close to 0, x and y are considered to have no relationship and the scatterplot looks completely random
sign = direction
*positive means as x increases, y increases
*negative means as x increases, y decreases

Visualizing and adding a trendline using R

```
ggplot(df, aes(x, y)) +
      geom_point() +
      geom_smooth(method = 'lm', se = FALSE)
#scatterplot is formed using geom_point
#trendline is formed using geom_smooth
#'lm' inidicates that we want a linear trendline
#'se' determines error margins
```

Computing correlation using R
cor(df$x, df$y)
function takes in two numeric vectors
*doesnt' matter which order the vecotrs are passed into the function since the correlation between x and y is the same thing as the correlation between y and x
when calculating correlation, R will return missing values as NA
to ignore this need to set the 'use' argument within the cor function to 'pairwise.complete.obs'
many ways to calculate correlation but most common way is Pearson (denoted as 'r')
*essentially assessing the equality between the sample mean and the population mean
xbar = mean of x and ybar = mean of y

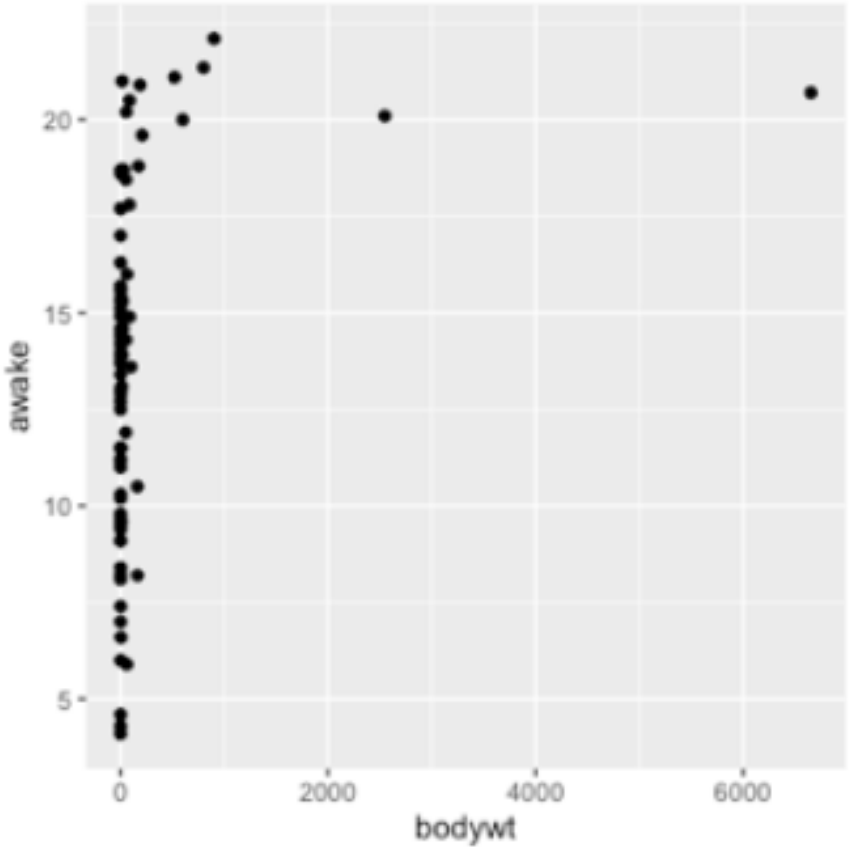
$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Example

```
# Add a linear trendline to scatterplot
ggplot(world_happiness, aes(life_exp, happiness_score)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)

# Correlation between life_exp and happiness_score
cor(world_happiness$life_exp, world_happiness$happiness_score)
```

Correlation caveats
non-linear relationships such as quadratic relationships
correlation coeff only measures the strength of linear relationships
when data is highly skewed we can apply log transformation
R example with a skewed dataset of mammal weight and mammal sleep
we go from this >




to this with a log transformation

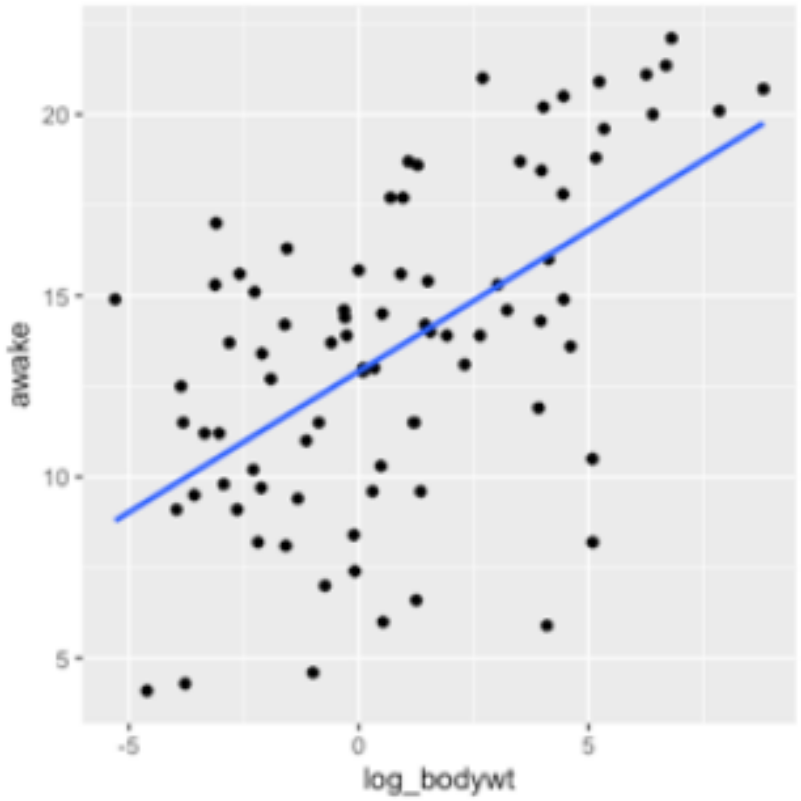
```
msleep %>%
        mutate(log_bodywt = log(bodywt)) %>%
        ggplot(aes(log_bodywt, awake)) +
        geom_point() +
        geom_smooth(method = 'lm', se = FALSE)
```

Example

```
# Create log_gdp_per_cap column
world_happiness <- world_happiness %>%
  mutate(log_gdp_per_cap = log(gdp_per_cap))

# Scatterplot of happiness_score vs. log_gdp_per_cap
ggplot(world_happiness, aes(log_gdp_per_cap, happiness_score)) +
  geom_point()

# Calculate correlation
cor(world_happiness$log_gdp_per_cap, world_happiness$happiness_score)
```



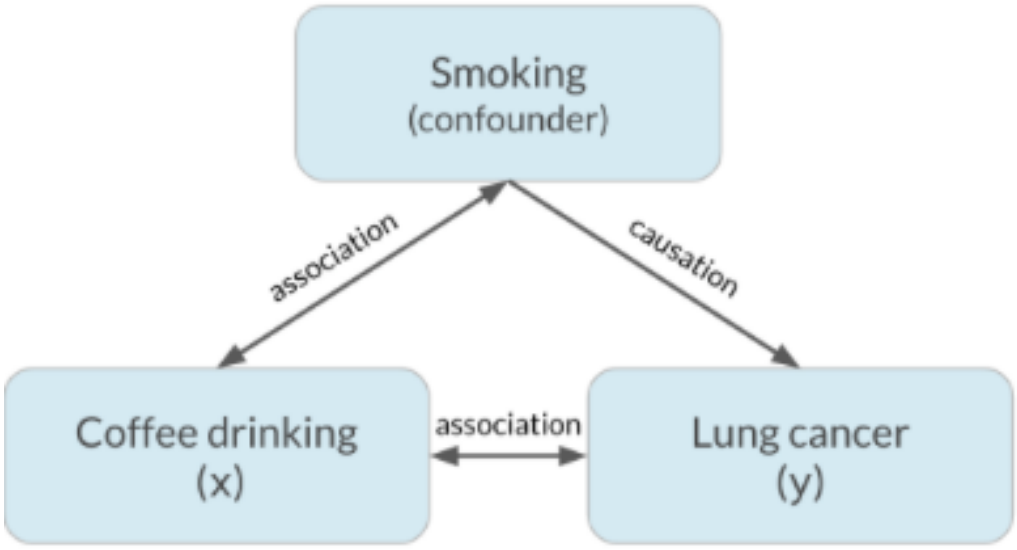

using the cor function prior to log transformation we get 0.3
after log transformation we get 0.57

Lots of transformations can be used to make a relationship more linear
- log(x)
- sqrt(x)
- reciprocal 1/x

or combo of the above, such as:
- log(x) and log(y)
- or sqrt(x) and 1/y

Why use a transformation? > certain statistical methods rely on variables having a linear relationship
such as linear regression

**Always remember that correlation does mean causation
always be on the lookout for 'spurious' correlations
example lung cancer and coffee drinking
this is a phenomenon called confounding which leads to spurious correlations




Design of experiments
Experiments generally aim to answer a question in the form:
What is the effect of the treatment on the response?
where treatment is the explanatory/independent variable
where response is the response/dependent variable

Gold standard of experiments will use:
Randomized controlled trial > participants are assigned to treatment/control randomly and not based on any other characteristics
this ensures that groups are comparable
Use of a placebo > this resembles treatment but has no effect
this ensures that participants will not know which group they're in
Double-blind > means that the person administering the experiment also doesn't know whether the treatment is real or a placebo
this helps prevent bials in the response and/or analysis of results

Observational studies
participants assign themselves (*usually based on pre-existing characteristics)
*with these studies you cannot establish causation
here you can only establish association
effects can be confounded by factors that got certain people into the control or treatment group

Longitudinal studies
particpants are followed over a period of time to examine effect of treatment on response
this type of study helps eliminate confounders

this type of study is difficult and expensive

Cross-sectional studies
data on participants is collected from a single snapshot in time
confounders are always likely in these studies
easier and cheaper to perform