Stats Demo

```r
# Set default chunk options

knitr::opts_chunk$set(
    fig.height = 6,
    fig.width = 7,
    message = TRUE,
    warning = FALSE,
    comment = ""
)

# Load necessary libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(psych)
```

```
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```r
library(stats)
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following objects are masked from 'package:psych':
##
##     alpha, rescale
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

Descriptive statistics

```
# Use built-in dataset 'starwars' for analysis
sw.desc <- starwars %>%
  select(height, mass) %>%
  drop_na() # as an alternative, include the `na.rm=T` argument in your measures functions

# Create a long version to use for dplyr and ggplot stuff later
sw.desc.long <- sw.desc %>%
  pivot_longer(c(height, mass), names_to = "measure")
```

The quickest way to see summary statistics for a numeric variable is the `summary()` function:

```
summary(sw.desc$height)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   66.0   170.0   180.0   174.4   192.0   234.0
```

```
summary(sw.desc$mass)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.00   55.60   79.00   97.31   84.50 1358.00
```

Or if you need to keep things a little more organied, create a summarized dataframe:

```
sw.desc.long %>%
  group_by(measure) %>%
  summarize(mean = mean(value),
         median = median(value),
         sd = sd(value),
         range = diff(range(value))
  )
```

```
# A tibble: 2 x 5
  measure  mean median    sd range
  <chr>   <dbl>  <dbl> <dbl> <dbl>
1 height  174.     180  35.5   168
2 mass     97.3     79 169.   1343
```

Distributions

Calculate measures of center

You can also calculate everything piece by piece.

"Measures of center" are different ways of talking about averages. Usually we think about "mean" as synonymous with "average", so calling these measures of center instead can be more precise.

Calculate mean and median with `mean()` and `median()`. There is no built-in mode function, but if you need one you can either write your own function or use the `modeest` library.

```
mean_height <- mean(sw.desc$height)
median_height <- median(sw.desc$height)

mean_mass <- mean(sw.desc$mass)
median_mass <- median(sw.desc$mass)
```

- The mean height is 174.356 cm and median height is 180 cm.
- The mean mass is 97.312 kg and median mass is 79 kg.

Calculate measures of spread

Measures of spread describe the distribution of continuous data around the center. Calculate standard deviation with `sd()`. Calculate range by getting a list of the minimum and maximum with `range()` and then using `diff()` to find the difference between the two.

```
sd_height <- sd(sw.desc$height) # standard deviation
range_height <- diff(range(sw.desc$height))

sd_mass <- sd(sw.desc$mass) # standard deviation
range_mass <- diff(range(sw.desc$mass))
```

- The standard deviation of height is 35.537 cm and the range is 168 cm.
- The standard deviation of mass is 169.457 kg and the range is 1343 kg.

Other common measures of spread include variance, quantiles, and interquartile range:

```
sw.desc.long %>%
  group_by(measure) %>%
  summarize(variance = var(value),
            median1 = median(value),
            median2 = quantile(value, probs=0.5),
            quartile = quantile(value, probs = c(.25, .75)),
            iqr1 = IQR(value)
  )
```

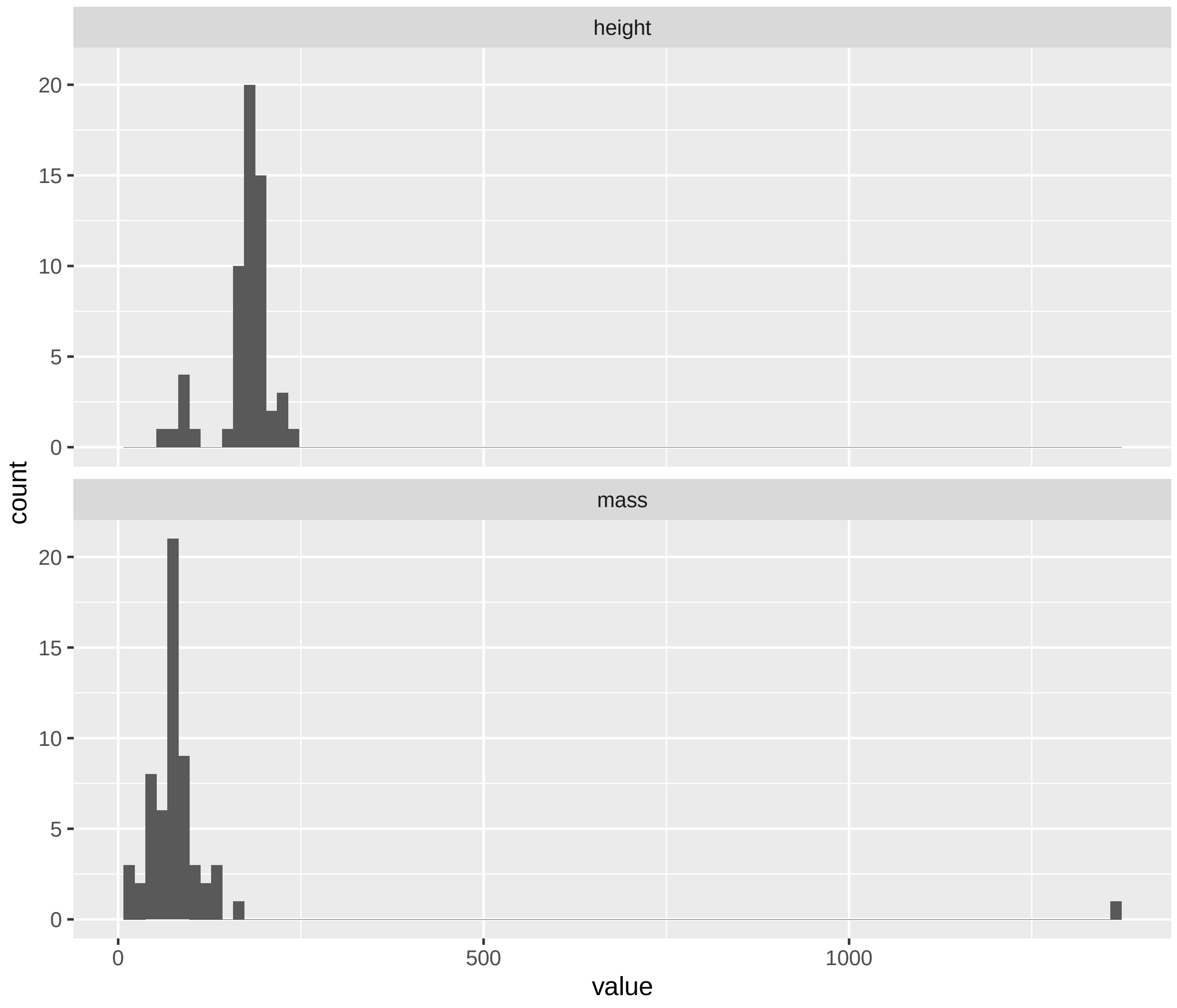
```
‘summarise()‘ has grouped output by ’measure’. You can override using the
‘.groups‘ argument.

# A tibble: 4 x 6
# Groups:   measure [2]
  measure variance median1 median2 quartile  iqr1
  <chr>      <dbl>   <dbl>   <dbl>    <dbl> <dbl>
1 height     1263.     180     180    170    22
2 height     1263.     180     180    192    22
3 mass      28716.      79      79     55.6  28.9
4 mass      28716.      79      79     84.5  28.9
```

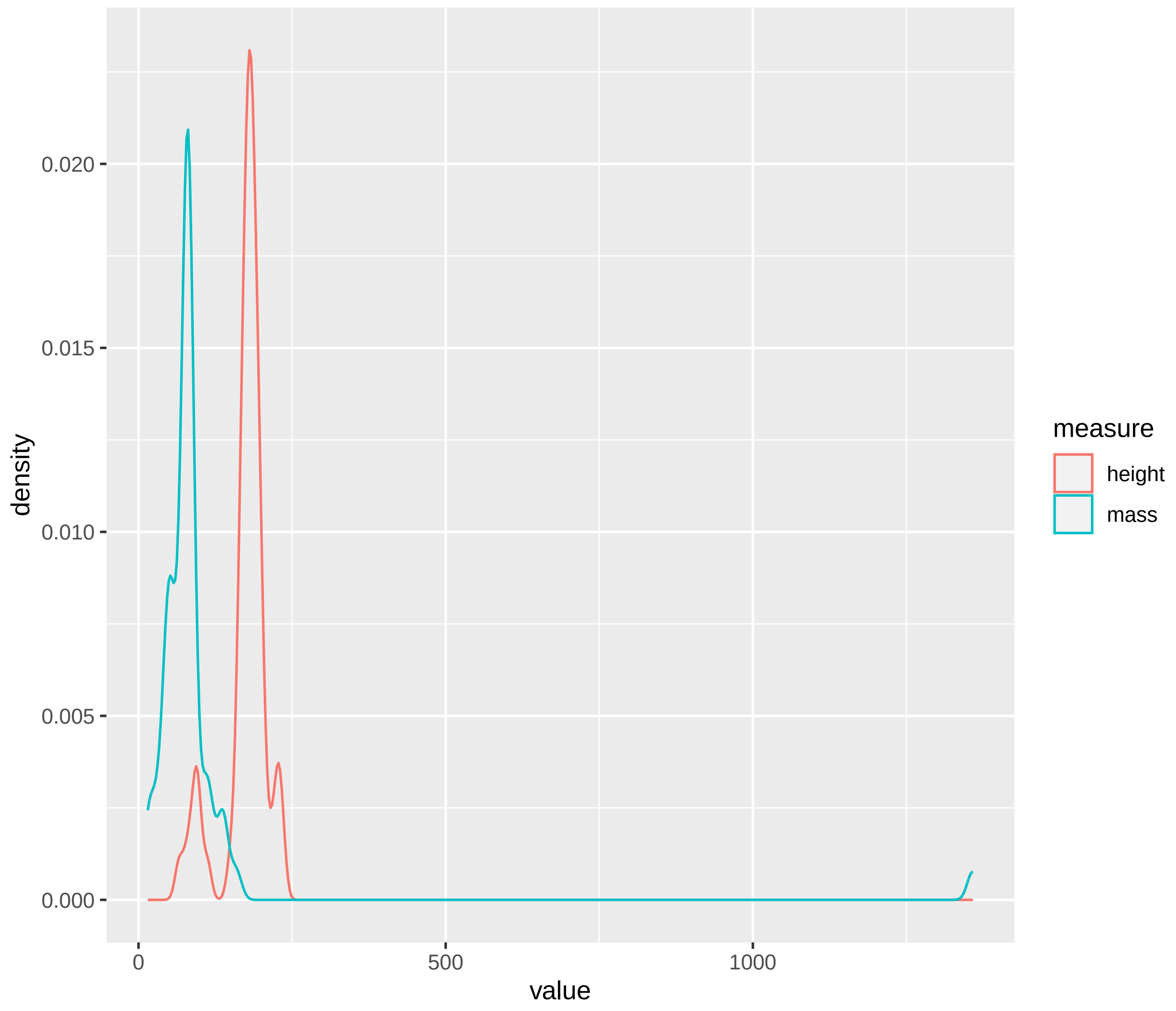
Visualize center and spread

Distribution plots visualize center and spread, for example boxplots, violin plots, histograms, and density plots.

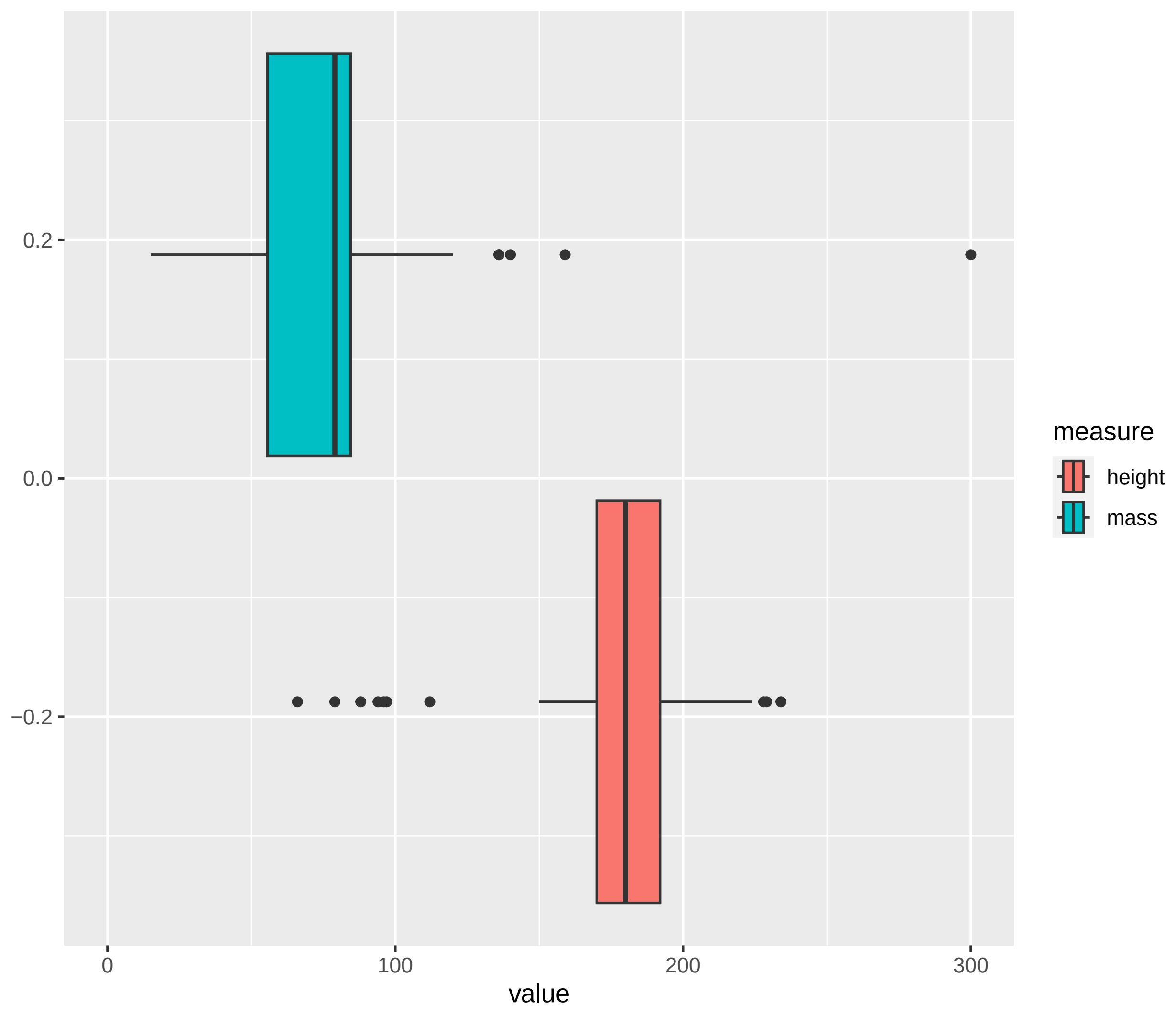
```r
ggplot(sw.desc.long) +
  geom_histogram(aes(value), binwidth=15) + facet_wrap(vars(measure), ncol=1)
```



```r
ggplot(sw.desc.long) +
  geom_density(aes(value, color=measure))
```

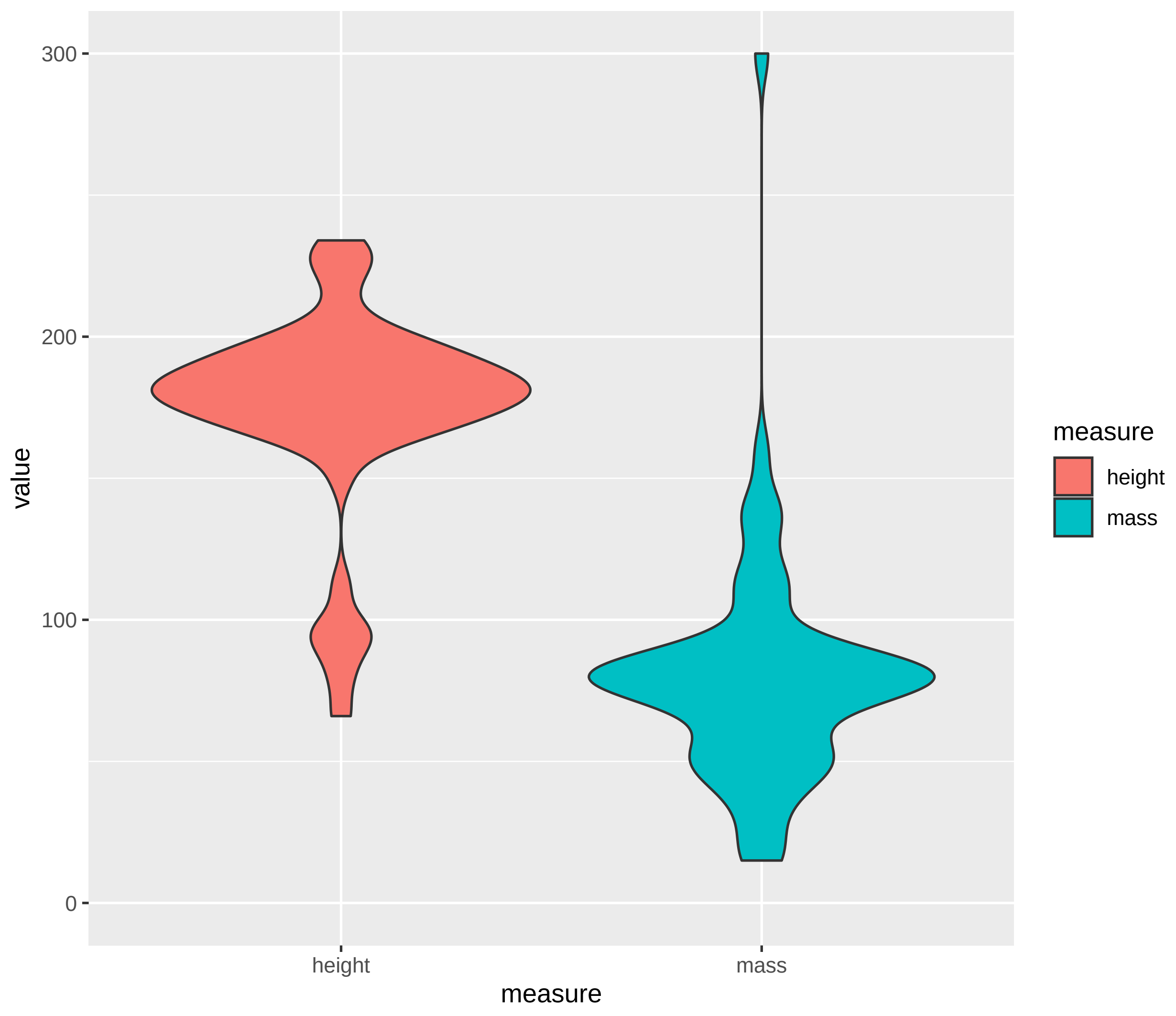


```r
ggplot(sw.desc.long) +
  geom_boxplot(aes(value, fill=measure)) +
  scale_x_continuous(limits=c(0,300), oob=squish)
```



```r
# you need the `scales` package for the oob argument to work as expected
# in this case the scale will "squish" values outside the limits
# to the nearest limit, so Jabba's huge mass will look like 300
```

```r
ggplot(sw.desc.long) +
  geom_violin(aes(x=measure, y=value, fill=measure)) +
  scale_y_continuous(limits=c(0,300), oob=squish)
```



```r
# you need the `scales` package for the oob argument to work as expected
# in this case the scale will "squish" values outside the limits
# to the nearest limit, so Jabba's huge mass will look like 300
```

Correlation

The `cor()` function creates a correlation matrix:

```r
mass_height_corr <- cor(sw.desc) # using a matrix-like object (e.g., df)
mass_height_corr2 <- cor(sw.desc[,1], sw.desc[,2]) # or vectors (e.g., df columns)
```

The correlation of height and mass is 0.131.

If you intend to use a correlation as a (quasi)hypothesis test, you'll need the `corr.test()` function in the `psych` package to give you *p*-values,

```r
mass_height_corr3 <- corr.test(sw.desc)
mass_height_corr4 <- corr.test(sw.desc[,1], sw.desc[,2])
```

```
mass_height_corr3
```

```
Call:corr.test(x = sw.desc)
Correlation matrix
       height mass
height   1.00 0.13
mass     0.13 1.00
Sample Size
[1] 59
Probability values (Entries above the diagonal are adjusted for multiple tests.)
       height mass
height   0.00 0.32
mass     0.32 0.00

 To see confidence intervals of the correlations, print with the short=FALSE option
```

```
mass_height_corr4
```

```
Call:corr.test(x = sw.desc[, 1], y = sw.desc[, 2])
Correlation matrix
       mass
height 0.13
Sample Size
[1] 59
These are the unadjusted probability values.
  The probability values  adjusted for multiple tests are in the p.adj object.
       mass
height 0.32

 To see confidence intervals of the correlations, print with the short=FALSE option
```

`View()` the object to see what the output contains and then extract elements like p-value:

```
mass_height_corr3$p
```

```
          height      mass
height 0.0000000 0.3232031
mass   0.3232031 0.0000000
```

```
mass_height_corr3$p.adj
```

```
[1] 0.3232031
```

```
mass_height_corr4$p
```

```
            mass
height 0.3232031
```

The default corr method is pearson, but you can change this. For example, Spearman rank correlation is useful for small samples:

```
mass_height_corr_Spearman <- corr.test(sw.desc, method = "spearman")
```

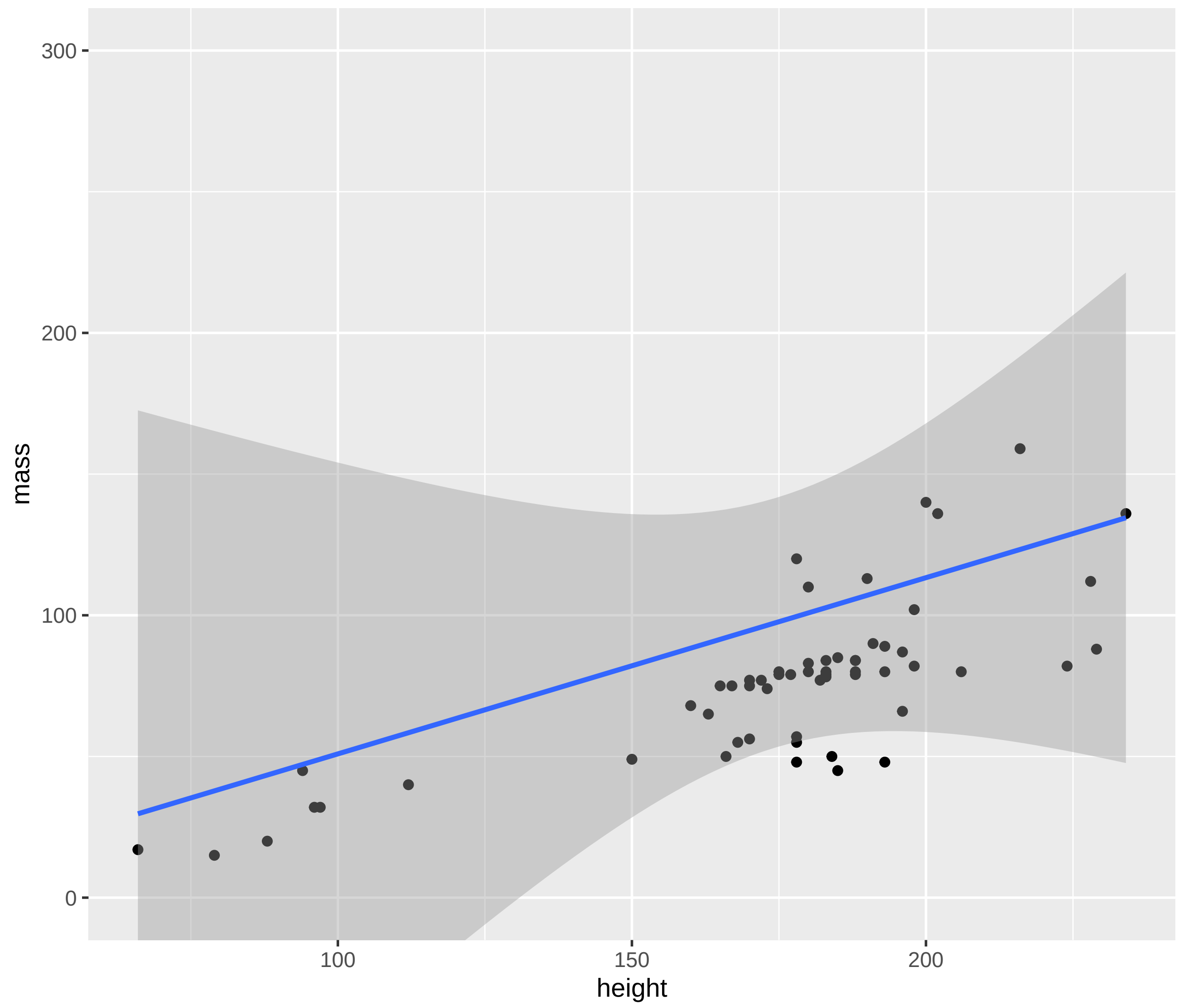
- Pearson $\rho = 0.131$ ($p = 0.323$)
- Spearman ranked $\rho = 0.719$ ($p = 0$)

Visualize correlation

Visualizing correlation is functionally the same as visualizing linear regression. Combine a scatter plot with a regression line (using `geom_smooth()`):

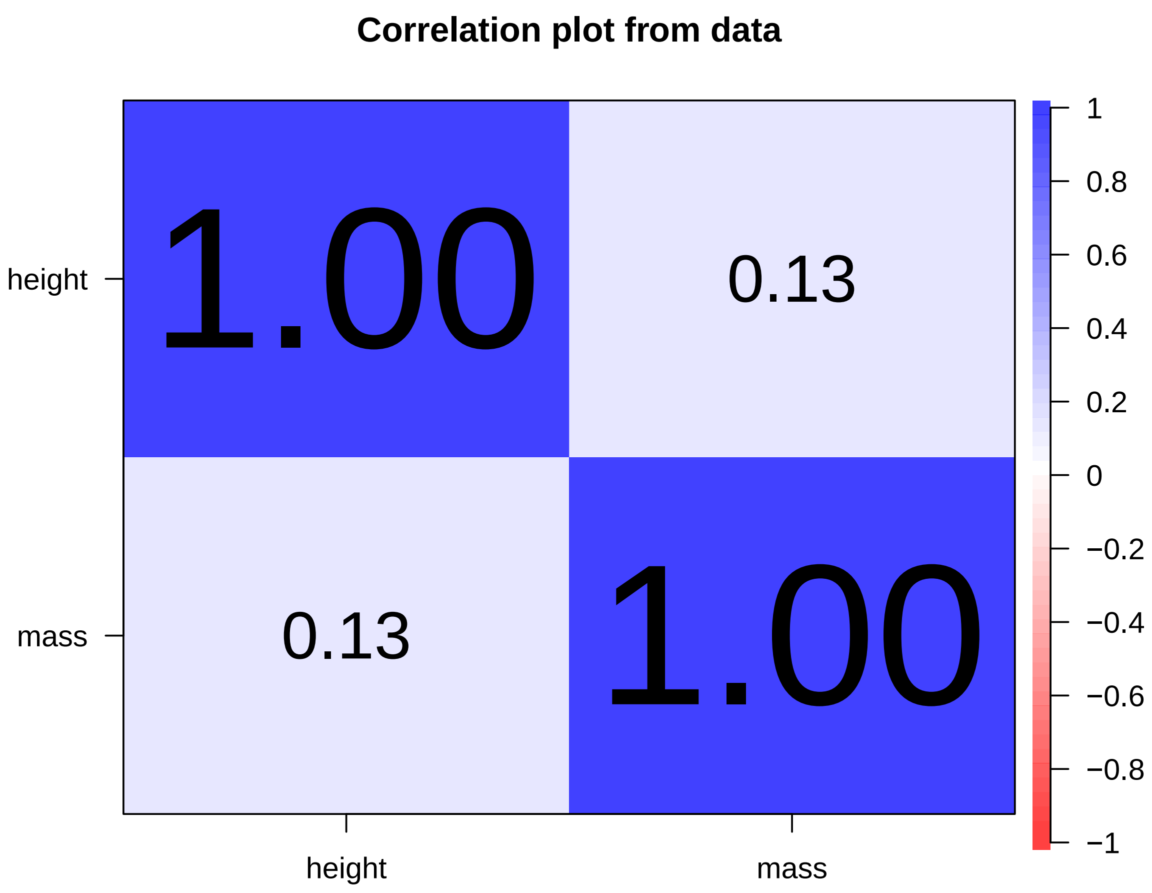
```
ggplot(sw.desc) +
  geom_point(aes(height,mass)) +
  geom_smooth(aes(height,mass), method="lm") +
  coord_cartesian(ylim = c(0,300))
```

```
'geom_smooth()' using formula = 'y ~ x'
```

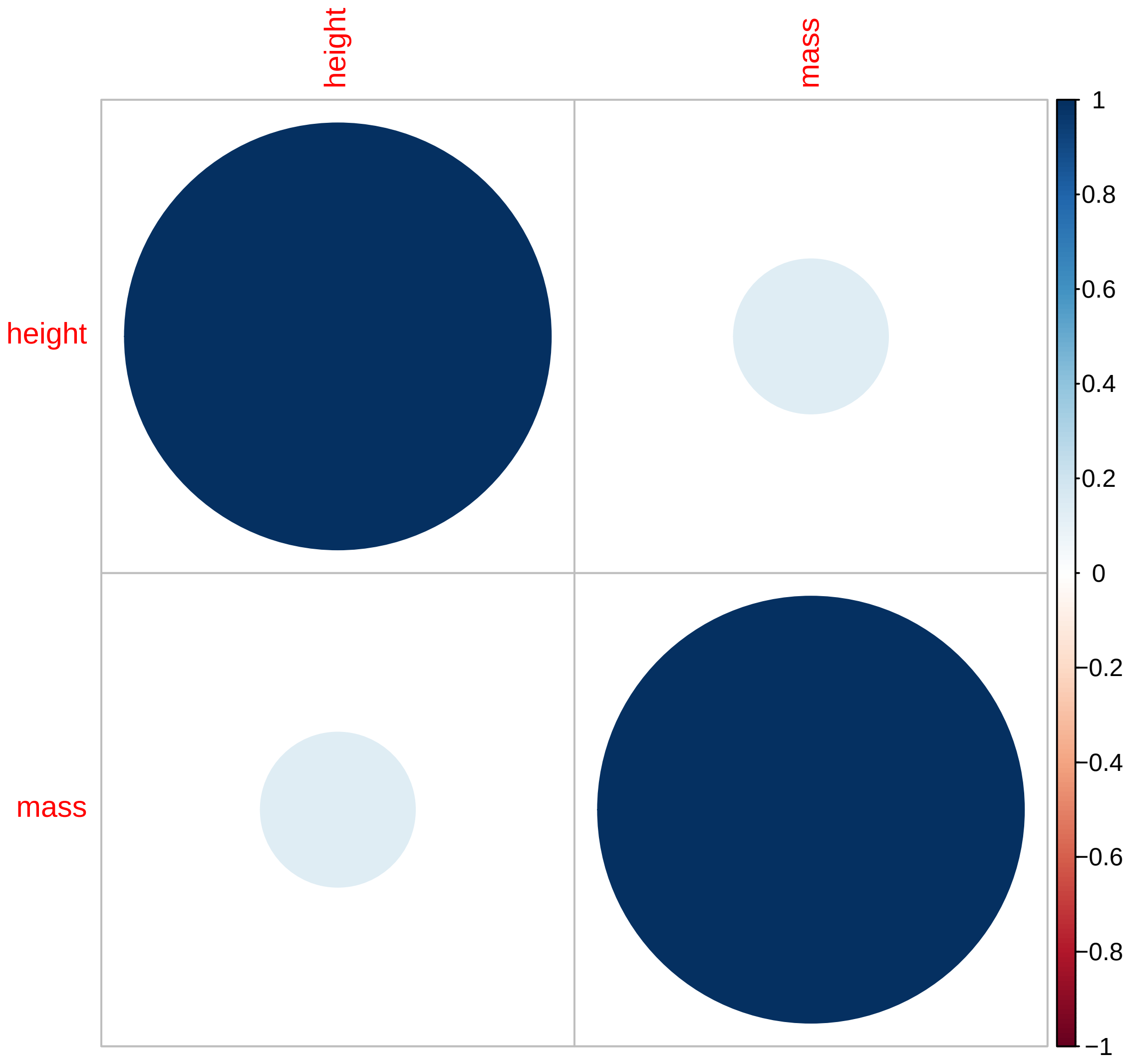


You can also visualize the correlation matrix with functions from other packages.

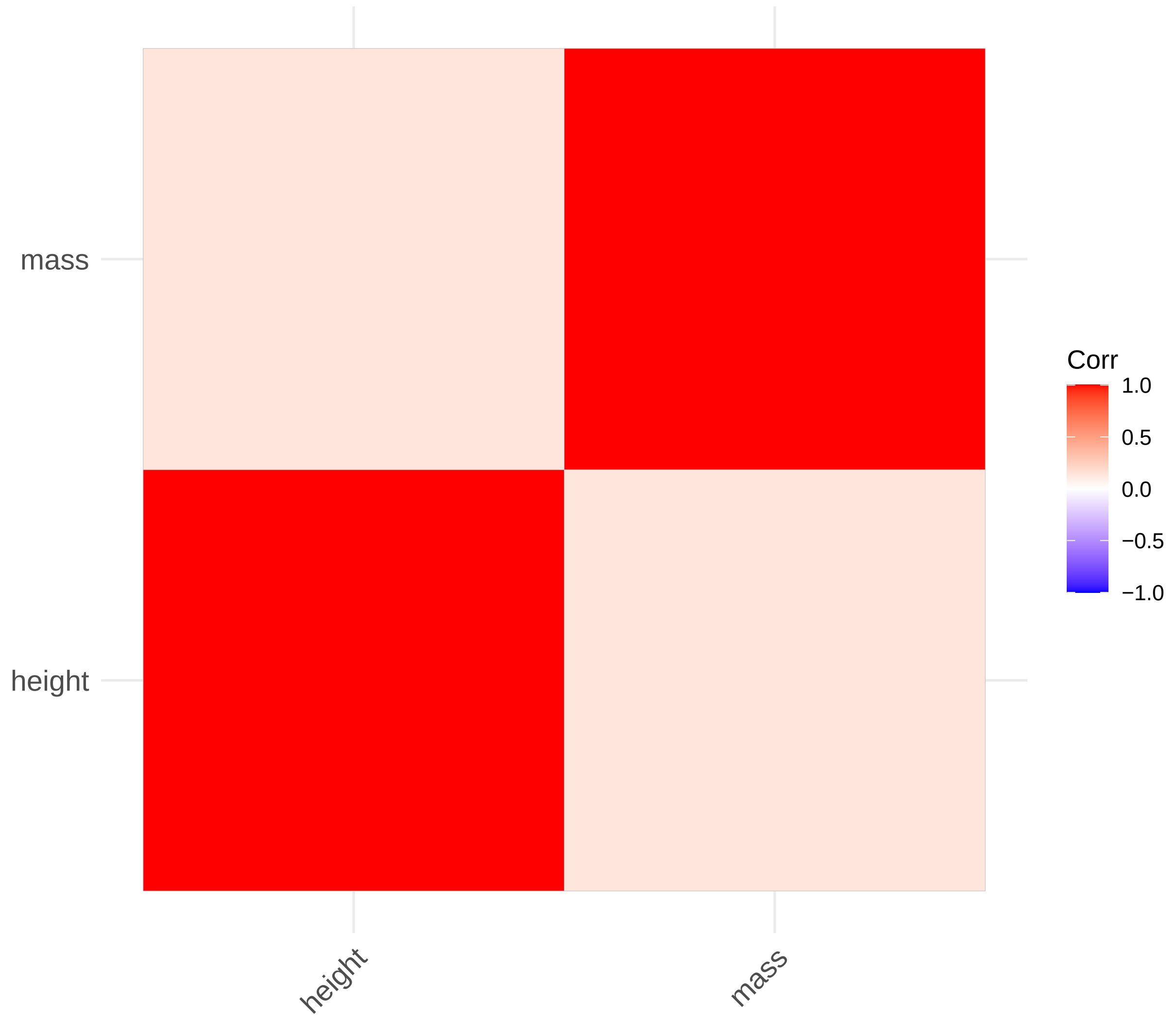
```r
psych::corPlot(sw.desc)
```



```r
corrplot::corrplot(mass_height_corr)
```



```r
ggcorrplot::ggcorrplot(mass_height_corr)
```

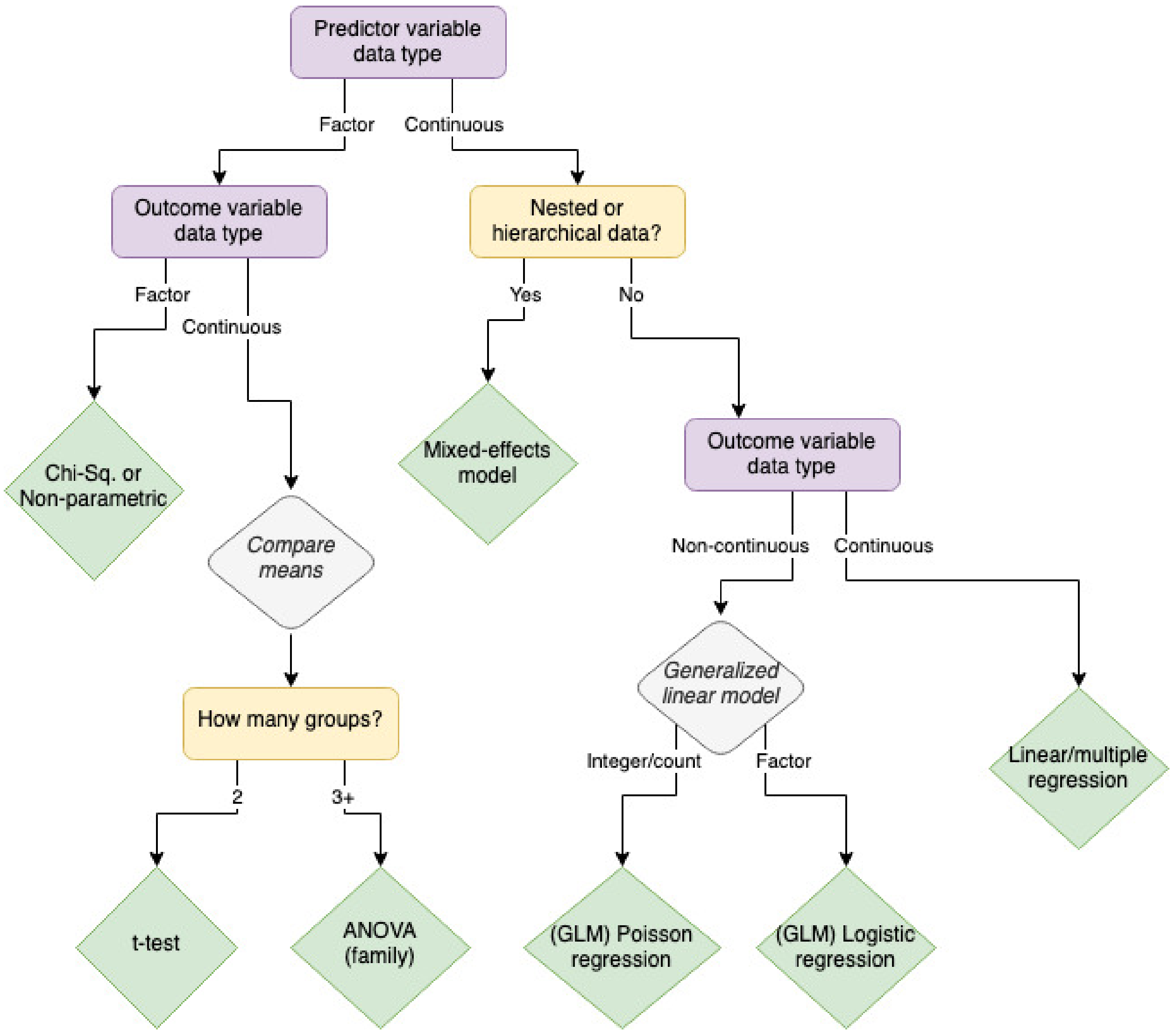


None of these are thrilling with just two variables, but they can be very useful when you're using a correlation matrix across many variables.

Hypothesis Testing

Hypothesis testing is anything you might usually think of as "results." Essentially: *do these data suggest some kind of non-random pattern?* The best hypothesis test to use will depend on a few factors, most significantly the data type of the independent (predictor) and dependent (outcome) variables.

This flowchart is a quick-and-dirty, imperfect cheatsheet:



Categorical Predictors

t-tests

1-sample A 1-sample t-test tells you the likelihood that the "true" mean of a value is not equal to 0 (or another reasonable, specific alternative).

For example, a 1-sample t-test on the `mass` variable should return significant results, rejecting the null hypothesis that the true mean is 0. Since mass is necessarily a positive value, it is impossible that the true mean would be 0.

```r
t.test(sw.desc$mass)
```

```

	One Sample t-test

data:  sw.desc$mass
t = 4.4109, df = 58, p-value = 4.525e-05
```

```
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
  53.15109 141.47264
sample estimates:
mean of x
 97.31186
```

We can alternatively specify the mean that the null hypothesis should assume. Let's assume that the `starwars` dataset contains the mass of literally every character in Star Wars. In that case, the true population mean mass for Star Wars characters is 97.312 kg. We can specify the null/true mean with the `mu` ($/mu$) argument.

Is the mean of this sample different than the true mean?

```r
t.test(sw.desc$mass, mu=mean(sw.desc$mass))
```

```

    One Sample t-test

data:  sw.desc$mass
t = 0, df = 58, p-value = 1
alternative hypothesis: true mean is not equal to 97.31186
95 percent confidence interval:
  53.15109 141.47264
sample estimates:
mean of x
 97.31186
```

Obviously not, since the "sample" is what the true mean was calculated on. But if we consider the full dataset the true, full population, we can compare a sample to that population.

```r
slice_sample(sw.desc, n=15) %>%
  t.test(sw.desc$mass,mu=mean(sw.desc$mass))
```

```

    Welch Two Sample t-test

data:  . and sw.desc$mass
t = -0.39385, df = 45.482, p-value = 0.6955
alternative hypothesis: true difference in means is not equal to 97.31186
95 percent confidence interval:
 -16.89122 174.14749
sample estimates:
mean of x mean of y
175.94000  97.31186
```

In nearly all cases (depending on your random seed) this will result in rejecting the null hypothesis. Essentially this is showing that there are some values (that of one mister The-Hutt, with a mass of 1358) of mass that is such an outlier it makes the mean of the full sample not actually representative of the "average". (This is a case where median might be a better measure measure of center than mean.) If we get rid of the extreme outlier and use that as the "true" mean, things might look different.

```r
sw.desc.nojabba <- sw.desc %>%
  filter(!(mass > 5*median(mass)))
```

Now randomly sampling from the dataframe will usually *not* be significantly different from that mean. Sometimes it will be though, just because of random variation. Sometimes is will be *extremely* significantly different. Why?

```r
# notice that only the mu argument uses the dataset without jabba
slice_sample(sw.desc, n=25) %>%
  t.test(sw.desc$mass, mu=mean(sw.desc.nojabba$mass))
```

```

    Welch Two Sample t-test

data:  . and sw.desc$mass
t = -2.0296, df = 75.744, p-value = 0.04591
alternative hypothesis: true difference in means is not equal to 75.57586
95 percent confidence interval:
 -19.97334  74.67761
sample estimates:
mean of x mean of y 
124.66400  97.31186 
```

```r


# similar df as before but now with some possible grouping vars
sw.desc2 <- starwars %>%
  select(sex, gender, species, homeworld, height, mass)
```

2-sample While a 1-sample t-test compares a sample mean against a static value (like 0), a 2-sample t-test compares two sample means against each other. The null hypothesis of a 2-sample t-test is that the true means of the group are not different.

Is the mass of male characters different from female characters?

```r
## overall, sig.
ttest2_overall <- t.test(
       filter(sw.desc2, sex == "male")$mass,
       filter(sw.desc2, sex == "female")$mass
       )

ttest2_overall
```

```

    Welch Two Sample t-test

data:  filter(sw.desc2, sex == "male")$mass and filter(sw.desc2, sex == "female")$mass
t = 4.8612, df = 43.298, p-value = 1.571e-05
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 14.94106 36.11926
```

```
sample estimates:
mean of x mean of y
 80.21905  54.68889
```

For all characters, yes. We can reject the null hypothesis that the means are the same ($t = 4.861$, $p < 0$).

What if we just look at the humans?

```r
ttest2_humans <- t.test(
  filter(sw.desc2, sex == "male", species == "Human")$mass,
  filter(sw.desc2, sex == "female", species == "Human")$mass
)
ttest2_humans
```

```

    Welch Two Sample t-test

data:  filter(sw.desc2, sex == "male", species == "Human")$mass and filter(sw.desc2, sex == "female", sp
t = 2.8744, df = 2.7808, p-value = 0.06986
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -4.648771 63.417398
sample estimates:
mean of x mean of y
 85.71765  56.33333
```

For humans, no. We cannot reject the null hypothesis that the means are the same ($t = 2.874$, $p < 0.07$).

Outside the Star Wars Cinematic Universe, we know that the mean mass of male humans is higher than that of female humans. Rather than looking for *any* difference in means, we have a theoretical reason to look for a difference in one particular direction. If we set `alternative = "greater"`, the null hypothesis is that the true difference in means (mean-of-males - mean-of-females) is less than or equal to 0.

```r
ttest2_humans_greater <- t.test(
  filter(sw.desc2, sex == "male", species == "Human")$mass,
  filter(sw.desc2, sex == "female", species == "Human")$mass,
  alternative = "greater"
)
ttest2_humans_greater
```

```

    Welch Two Sample t-test

data:  filter(sw.desc2, sex == "male", species == "Human")$mass and filter(sw.desc2, sex == "female", sp
t = 2.8744, df = 2.7808, p-value = 0.03493
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 4.533443      Inf
sample estimates:
mean of x mean of y
 85.71765  56.33333
```

Now we do see a significant effect. We can reject the null hypothesis that the mean mass of females is greater than or equal to that of males are the same ($t = 2.874$, $p < 0.035$).

Important optional arguments for t-tests:

- True mean (μ): `mu`
  - In a 1-sample test, the null hypothesis will compare the mean to 0 by default. You can change this to the "true mean".
- Alt hypothesis: `alternative = c("two.sided", "less", "greater")`
  - By default this tests that the 1-var mean is not equal to 0 (or μ) or that the 2-vars means are not equal to each other. If you are specifically looking to demonstrate that the mean is greater than or less than 0 (or μ) or that one particular group's mean is greater than the others (e.g., you expect the control group to have poorer outcomes than the treatment/intervention group), set this to `less` or `greater`.
- Paired: `paired = FALSE`
  - If the observations are related in some way, you can use a paired t-test. For example if you want to compare growth between pre-test and post-test, you're more interested in the change for each individual rather than either mean test score *per se*.
- Confidence level: `conf.level = 0.95`
  - Set an alternative confidence interval when comparing means. This is rarely changed; 95% is almost always the expectation here.

ANOVA

Think of an Analysis of Variance (ANOVA) as an extension of the t-test. With a t-test you can compare the mean of 1 group to a static value or the means of 2 groups to each other. The basic functionality of ANOVA is to allow you compare three or more groups.

ANOVA is a whole family of analyses, but we'll focus on just 1-way ANOVA and 2-way ANOVA. One-way ANOVA is appropriate when there is one categorical independent variable with multiple levels, while two-way ANOVA is used when there are two categorical independent variables and their interaction effect needs to be examined.

```r
sw.hyp <- starwars %>%
  select(name, height, mass, hair_color, skin_color, eye_color, sex, gender, homeworld, species) %>%
  mutate(hair_color = str_remove(hair_color, "[,].*$"),
         eye_color = str_remove(eye_color, "[,].*$"),
         skin_color = str_remove(skin_color, "[,].*$")) %>%
  mutate(sex3cat = case_when(sex %in% c("male", "female") ~ sex,
                             TRUE ~ "other"),
         hair4cat = case_when(hair_color %in% c("white", "grey") ~ "light",
                              hair_color %in% c("blond", "blonde") ~ "blond",
                              hair_color == "none" ~ "none",
                              TRUE ~ "dark"))
```

1-Way ANOVA **Example:** A psychologist wants to compare the effectiveness of three different stress reduction techniques (e.g., mindfulness meditation, progressive muscle relaxation, and deep breathing exercises) on reducing anxiety levels among participants.

One-way ANOVA can be used to test for significant differences in anxiety levels (dependent variable, continuous) across the three stress reduction techniques (independent variable, factor).

If the p-value from the ANOVA test is significant, post-hoc tests (e.g., Tukey's HSD) can be conducted to determine which techniques differ significantly from each other.

```r
anova_sex3 <- aov(height ~ sex3cat, data=sw.hyp)
summary(anova_sex3)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
sex3cat      2   5917    2958   2.541 0.0853 .
Residuals   78  90822    1164
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
6 observations deleted due to missingness
```

```r
TukeyHSD(anova_sex3)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = height ~ sex3cat, data = sw.hyp)

$sex3cat
                   diff       lwr       upr     p adj
male-female    7.551378 -16.76734 31.870094 0.7393848
other-female -18.471429 -52.22770 15.284846 0.3953726
other-male   -26.022807 -53.97481  1.929192 0.0733375
```

Here there is a trending but non-significant difference in height across the 3 sex categories $F() = 2.541$, $p = 0.085$.

Using Tukey post-hoc adjustment we can see this difference is primarily driven by the difference in height between those in the "male" and "other" category.

2-Way ANOVA **Example:** A psychologist conducts a study to investigate the effects of both gender (male vs. female) and stress level (low vs. high) on performance in a cognitive task.

In this scenario, there are two independent variables: gender (with two levels: male and female) and stress level (with two levels: low and high). The dependent variable is performance in the cognitive task. Two-way ANOVA would be used to assess the main effects of gender and stress level, as well as their interaction effect on performance. The interaction effect indicates whether the effect of one independent variable depends on the level of the other independent variable.

```r
anova_sex3_hair4 <- aov(height ~ sex3cat + hair4cat, data=sw.hyp)
summary(anova_sex3_hair4)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
sex3cat      2   5917  2958.4   2.520 0.0873 .
hair4cat     3   2771   923.6   0.787 0.5051
Residuals   75  88052  1174.0
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
6 observations deleted due to missingness
```

```r
TukeyHSD(anova_sex3_hair4)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = height ~ sex3cat + hair4cat, data = sw.hyp)

$sex3cat
                   diff       lwr       upr     p adj
male-female    7.551378 -16.88666 31.989414 0.7412438
other-female -18.471429 -52.39333 15.450471 0.3985203
other-male   -26.022807 -54.11195  2.066339 0.0751112

$hair4cat
                   diff       lwr      upr     p adj
dark-blond    0.8780075 -46.57273 48.32875 0.9999583
light-blond -16.0775689 -76.47250 44.31736 0.8969627
none-blond    7.5086536 -39.94208 54.95939 0.9756109
light-dark  -16.9555764 -59.92405 26.01289 0.7284309
none-dark     6.6306461 -14.58997 27.85126 0.8443509
none-light   23.5862225 -19.38225 66.55469 0.4773422
```

```r
anova_sex3_gender <- aov(height ~ sex3cat + gender, data=sw.hyp)
summary(anova_sex3_gender)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
sex3cat      2   9114    4557   3.914 0.0243 *
gender       1   2167    2167   1.862 0.1766
Residuals   73  85000    1164
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
10 observations deleted due to missingness
```

```r
TukeyHSD(anova_sex3_gender)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = height ~ sex3cat + gender, data = sw.hyp)

$sex3cat
                   diff       lwr       upr     p adj
male-female    7.551378 -16.79951 31.902268 0.7394416
other-female -33.071429 -72.90621  6.763355 0.1228312
other-male   -40.622807 -75.66121 -5.584402 0.0190413

$gender
                       diff       lwr      upr    p adj
masculine-feminine 3.518817 -16.04968 23.08732 0.721092
```

Aside from being used as a hypothesis test itself, another important use for ANOVA is comparing model fit. For example, you create 3 possible regressions to test whether household income and/or proximity to grocery stores affects stress level using one variable, both variables, or both and an interaction effect. Passing these models to the `anova()` function can tell you which model best explains a predictive effect, so you can move forward just using that model.

We can use the `mtcars` dataset to show a simple example: Does horsepower and/or weight predict a car's fuel consumption?

```r
# Use the mtcars dataset

# Model 1: Predicting mpg (miles per gallon) using horsepower
model_hp <- lm(mpg ~ hp, data = mtcars)
summary(model_hp)
```

```
Call:
lm(formula = mpg ~ hp, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-5.7121 -2.1122 -0.8854  1.5819  8.2360 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 30.09886    1.63392  18.421  < 2e-16 ***
hp          -0.06823    0.01012  -6.742 1.79e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.863 on 30 degrees of freedom
Multiple R-squared:  0.6024,    Adjusted R-squared:  0.5892 
F-statistic: 45.46 on 1 and 30 DF,  p-value: 1.788e-07
```

```r
# Model 2: Predicting mpg using horsepower and weight
model_hpwt <- lm(mpg ~ hp + wt, data = mtcars)
summary(model_hpwt)
```

```
Call:
lm(formula = mpg ~ hp + wt, data = mtcars)

Residuals:
   Min     1Q Median     3Q    Max 
-3.941 -1.600 -0.182  1.050  5.854 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
hp          -0.03177    0.00903  -3.519  0.00145 ** 
wt          -3.87783    0.63273  -6.129 1.12e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.593 on 29 degrees of freedom
Multiple R-squared:  0.8268,    Adjusted R-squared:  0.8148 
F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12
```

```r
# Model 3: Predicting mpg using horsepower, weight, with an interaction effect
model_int <- lm(mpg ~ hp*wt, data = mtcars)
summary(model_int)
```

```
Call:
lm(formula = mpg ~ hp * wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.0632 -1.6491 -0.7362  1.4211  4.5513 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 49.80842    3.60516  13.816 5.01e-14 ***
hp          -0.12010    0.02470  -4.863 4.04e-05 ***
wt          -8.21662    1.26971  -6.471 5.20e-07 ***
hp:wt        0.02785    0.00742   3.753 0.000811 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.153 on 28 degrees of freedom
Multiple R-squared:  0.8848,	Adjusted R-squared:  0.8724 
F-statistic: 71.66 on 3 and 28 DF,  p-value: 2.981e-13
```

All three models show a significant effect of the predictor variable(s). The question becomes which of these to use for the rest of the analyses and in the interpretation of our results. Comparing these models in an ANOVA tells us which model (if any) has a significantly better predictive fit.

```r
# Compare model fit using ANOVA; anova() function is in the stats package
anova_result <- anova(model_hp, model_hpwt, model_int)

# View the ANOVA table
anova_result
```

```
Analysis of Variance Table

Model 1: mpg ~ hp
Model 2: mpg ~ hp + wt
Model 3: mpg ~ hp * wt
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)    
1     30 447.67                                  
2     29 195.05  1   252.627 54.512 4.856e-08 ***
3     28 129.76  1    65.286 14.088 0.0008108 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The *p*-values here indicate whether there is a significant difference in fit between one model and the model that came before it. Assuming significant difference, the best model fit is the one with the lowest residual sum of squares (RSS).

Note that depending on the type of models you're comparing, you might need to find the lowest value of something else. For example with mixed-effects models you'll (typically) look for the lowest BIC.