Research Methods and Data Analysis (IAWEL)

Jill R D MacKay

Invalid Date

Table of contents

1 Research Methods and Data Analysis (IAWEL)

Preface

This book accompanies the Research Methods and Data Analysis course on the International Animal Welfare Ethics and Law MSc at the Royal (Dick) School of Veterinary Studies.

It is a companion document to the course, and not core to the materials.

Throughout the RMDA Lectures, you will see a number of statistical tests, data visualisations, data manipulation, text mining, and simple calculations. Almost inevitably, each one of these steps will have been performed in R.

Your R textbook is R@R(D)SVS, and that textbook will explain how to download and install R, how to run simple commands in R, and more. This RMDA textbook is like an accompanying document to your lecture materials, and is a place to help you move your R and statistical knowledge along.

Packages in this textbook

There are a range of packages used in this book, including Tidyverse (Wickham et al. 2019), effsize (Torchiano 2020), ggstatsplot (Patil 2021), vcd (Zeileis, Meyer, and Hornik 2007), wordcloud (Fellows 2018), easystats (Lüdecke et al. 2022), rstan (Stan Development Team 2023), rstanarm (Brilleman et al. 2018)

You may need to download and install a package or load a package for some of these commands to work.

Licensing

for the benefit of the public at large and to the detriment of our heirs and successors. We intend this dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

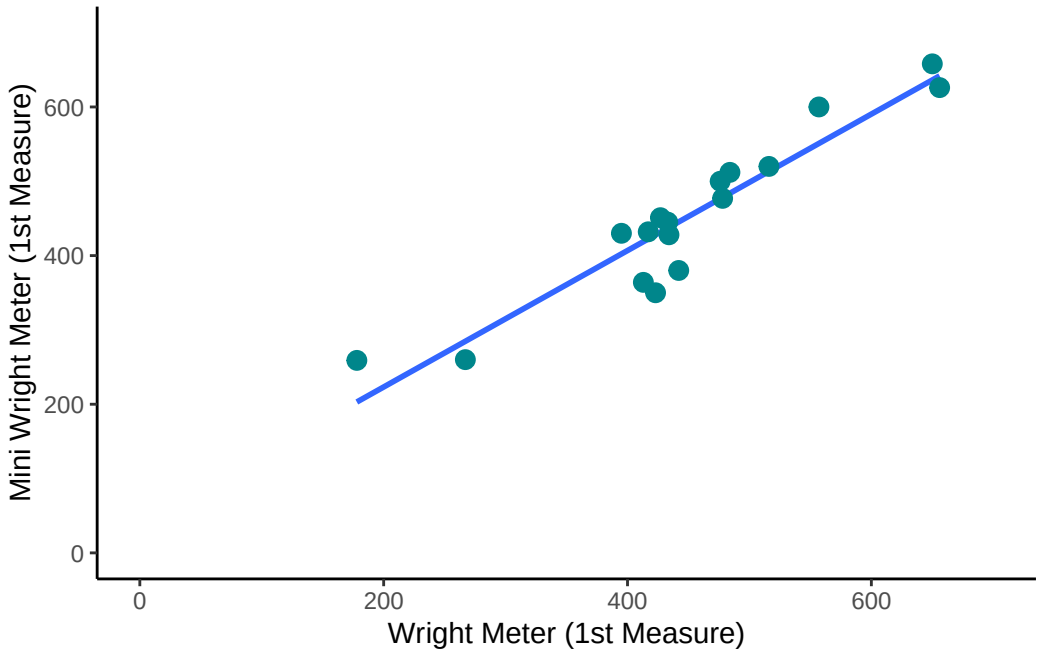
For more information, please refer to https://unlicense.org

2 Week 1: The Philosophy of Science

Lecture 3: The Replication Crisis

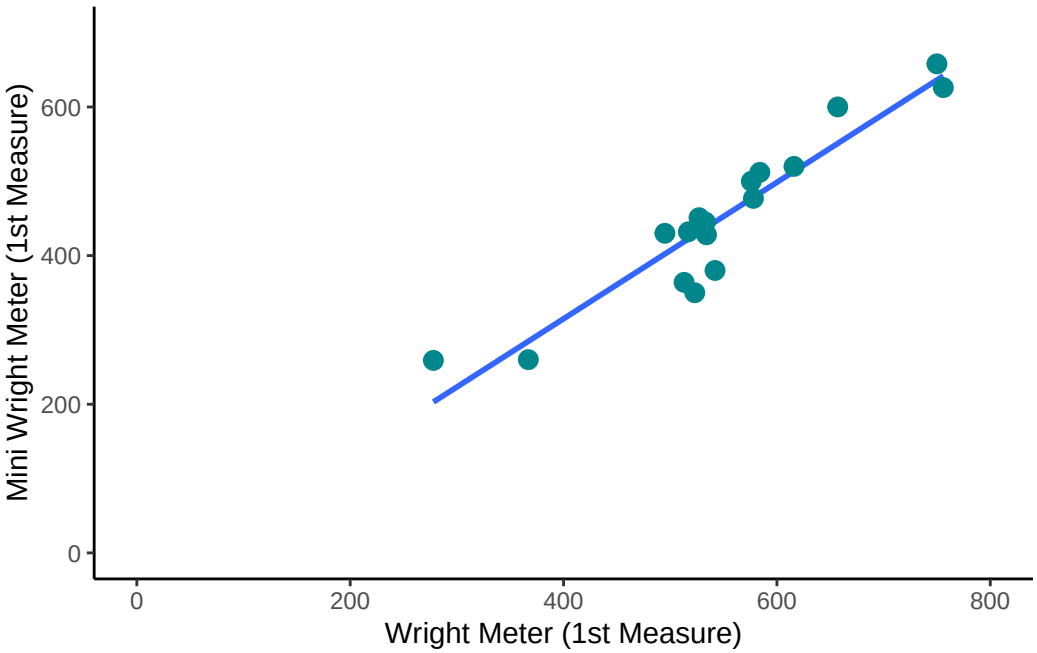
Bland-Altmann Plots are generated with the following code.

```
library(tidyverse)

bland <-  tibble(
  subject = c(1:17),
  Wright1 = c(484,395,516,434,476,557,413,442,650,433,417,656,267,478,178,423,427),
  Wright2 = c(490,397,512,401,470,611,415,431,638,429,420,633,275,492,165,372,421),
  Mini1 = c(512,430,520,428,500,600,364,380,658,445,432,626,260,477,259,350,451),
  Mini2 = c(525,415,508,444,500,625,460,390,642,432,420,605,227,467,268,370,443)
)

bland |>
  ggplot(aes(x = Wright1, y = Mini1)) +
  stat_smooth(method="lm", se=FALSE) +
  geom_point(colour = "turquoise4", size = 3) +
  scale_x_continuous(limits = c(0,700)) +
  scale_y_continuous(limits = c(0,700)) +
  theme_classic() +
  labs(x = "Wright Meter (1st Measure)", y = "Mini Wright Meter (1st Measure)")
```



And then if we add 100 to each measure, we see a very similar plot:

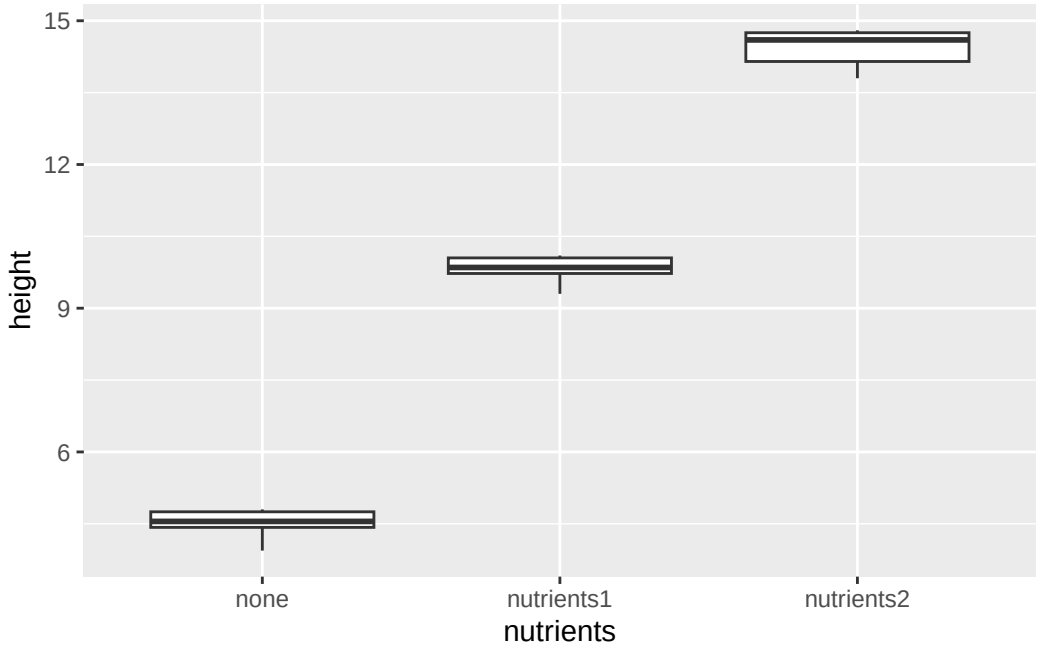
```
bland |>
  mutate (Wright1 = (Wright1+100)) %>%
  ggplot(aes(x = Wright1, y = Mini1)) +
  stat_smooth(method="lm", se=FALSE) +
  geom_point(colour = "turquoise4", size = 3) +
  scale_x_continuous(limits = c(0,800)) +
  scale_y_continuous(limits = c(0,700)) +
  theme_classic() +
  labs(x = "Wright Meter (1st Measure)", y = "Mini Wright Meter (1st Measure)")
```


Mini Wright Meter (1st Measure)
600
400
200
0
0
200
400
600
800
Wright Meter (1st Measure)

Lecture 5: Introduction to Research Methods

2.1 Create data and plot

```
library(tidyverse)

plants <- tibble(none = c(4.8, 4.8, 3.94, 4.4,4.5,4.6),
                 nutrients1  = c( 10.1, 9.7, 9.8, 9.9, 9.3, 10.1),
                 nutrients2 = c(14.8, 14.6, 14.8, 14, 13.8, 14.6))

plants |>
  pivot_longer(cols = c(none, nutrients1,nutrients2),
               names_to = "nutrients",
               values_to = "height") |>
  ggplot(aes(x = nutrients, y = height)) +
  geom_boxplot()
```



2.2 Run an ANOVA on Plant data

```
longplants <- plants |>
  pivot_longer(cols = c(none, nutrients1,nutrients2),
                              names_to = "nutrients",
                              values_to = "height")

plant_model <- aov(height ~ nutrients, data = longplants)

summary(plant_model)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
nutrients    2 296.10  148.05    1184 <2e-16 ***
Residuals   15   1.88    0.13
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.3 Read and Run Crude Chicken Correlations

```
crudechicks <- tibble(year = c("2000", "2001", "2002", "2003",
                               "2004", "2005", "2006", "2007",
                               "2008", "2009"),
                      chicken = c(54.2, 54, 56.8, 57.5, 59.3, 60.5, 60.9,
                                  59.9, 58.7, 56),
                      crude = c(3311, 3405, 3336, 3521, 3674, 3670, 3685,
                                3656, 3571, 3307))


cor.test(crudechicks$chicken, crudechicks$crude, method = "spearman")
```

```
    Spearman's rank correlation rho

data:  crudechicks$chicken and crudechicks$crude
S = 20, p-value = 0.001977
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.8787879
```

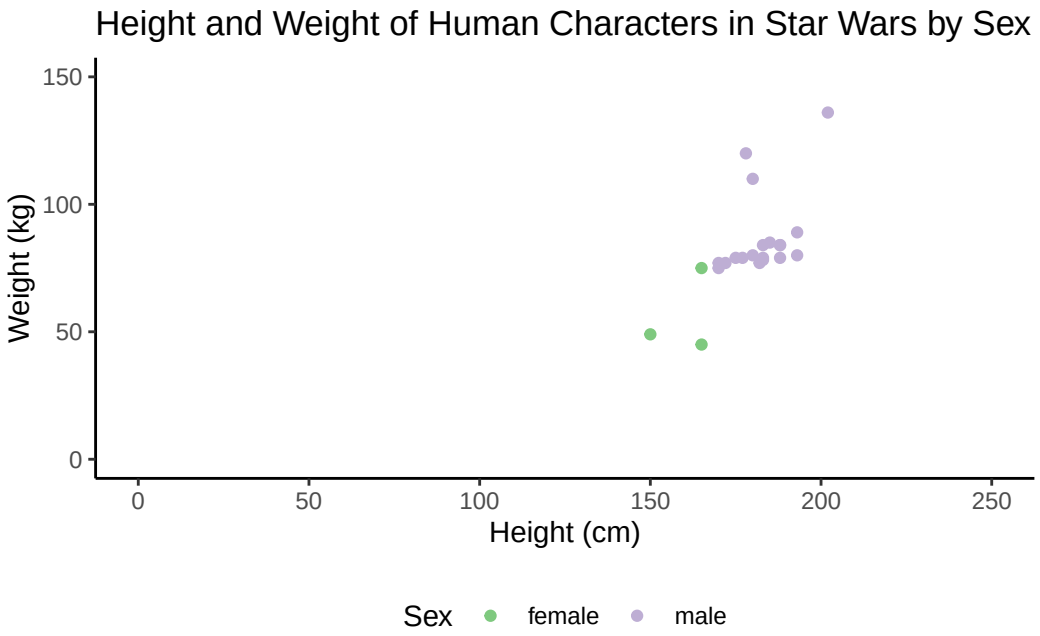
3 Week 2: The Use and Abuse of Data

Lecture 2: Data Visualisation

This code will help you replicate the charts in Lecture 2

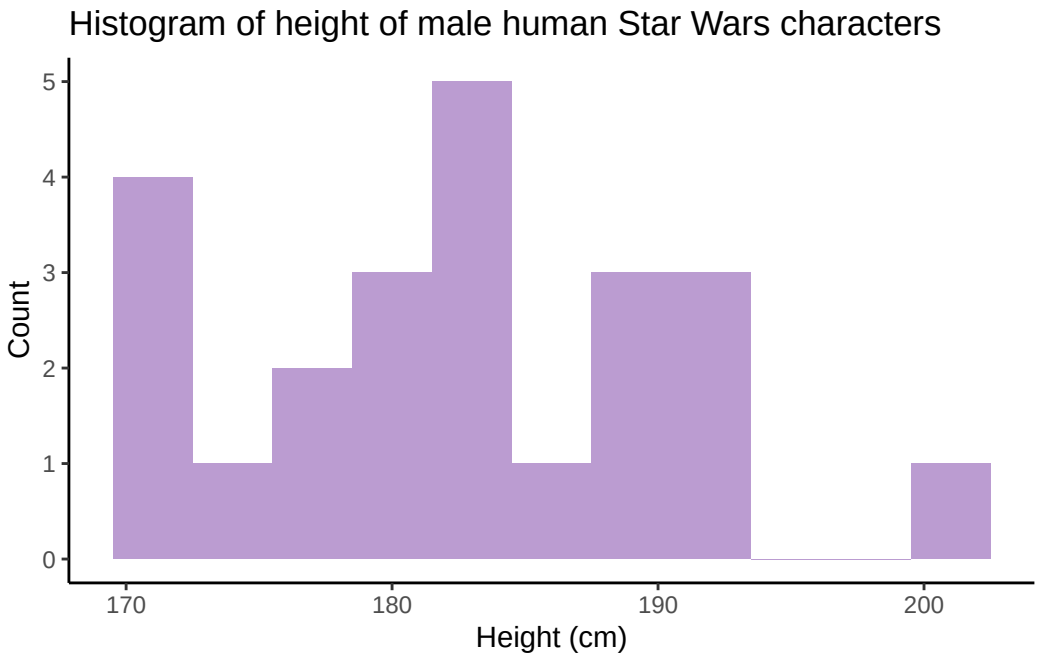
Height vs Weight by Sex

```
starwars |>
  filter(species == "Human") |>
  ggplot(aes(x = height, y = mass, colour = sex)) +
  geom_point() +
  theme_classic() +
  scale_x_continuous(limits = c(0,250)) +
  scale_y_continuous(limits = c(0,150)) +
  scale_colour_brewer(palette = "Accent", name = "Sex") +
  theme(legend.position = "bottom") +
  labs(x = "Height (cm)",
       y = "Weight (kg)",
       title = "Height and Weight of Human Characters in Star Wars by Sex")
```



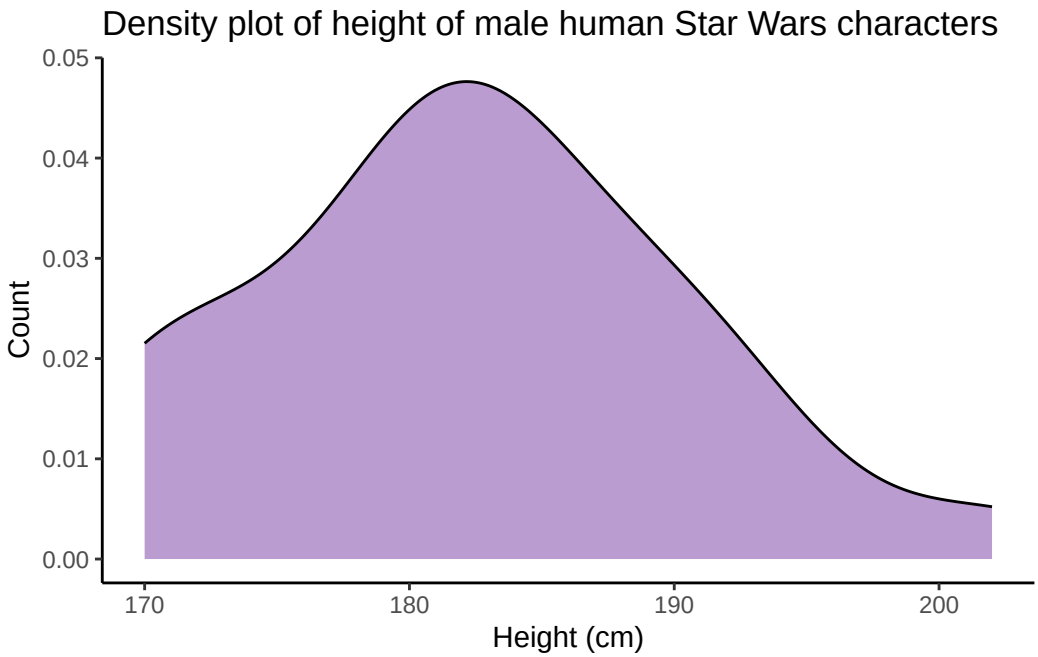
Histogram of male height

```
starwars |>
  filter(species == "Human",
         sex == "male") |>
  ggplot(aes(x = height)) +
  geom_histogram(binwidth = 3, fill = "#bb9cd1") +
  theme_classic() +
  labs(x = "Height (cm)",
       y = "Count",
       title = "Histogram of height of male human Star Wars characters")
```



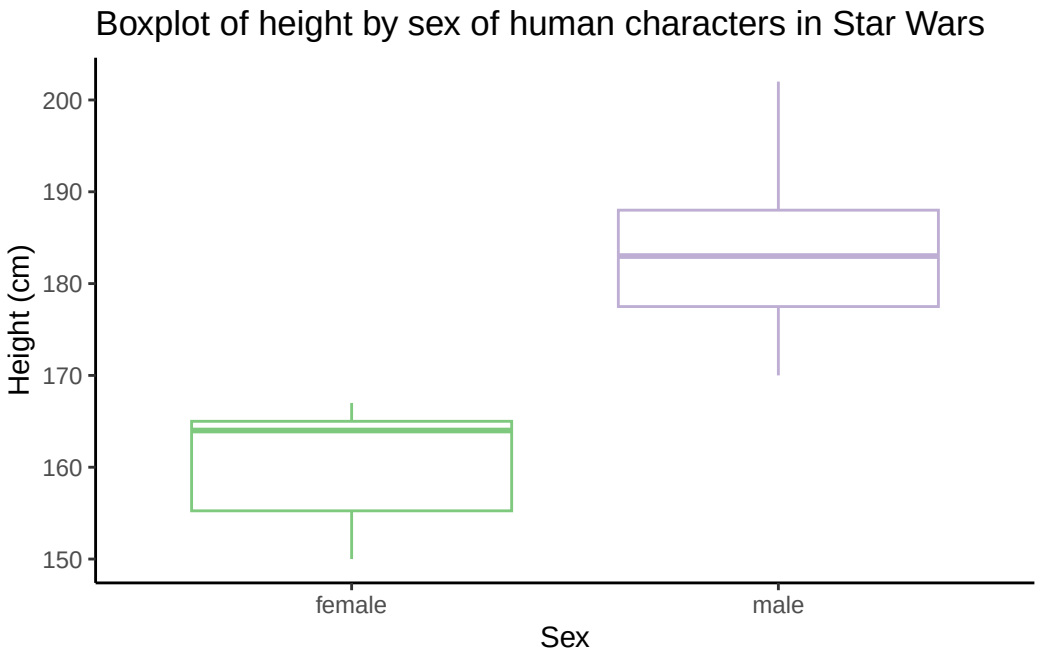
Density plot of male height

```
starwars |>
  filter(species == "Human",
         sex == "male") |>
  ggplot(aes(x = height)) +
  geom_density(fill = "#bb9cd1") +
  theme_classic() +
  labs(x = "Height (cm)",
       y = "Count",
       title = "Density plot of height of male human Star Wars characters")
```



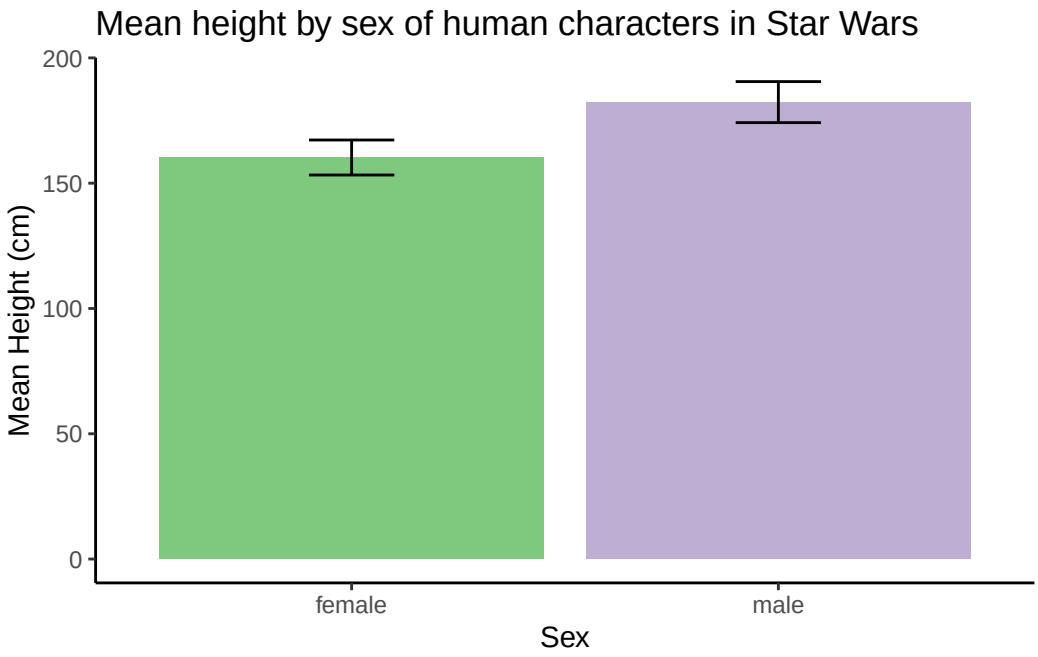
Boxplot of height

```
starwars |>
  filter(species == "Human") |>
  ggplot(aes(y = height, x = sex, colour = sex)) +
  geom_boxplot() +
  theme_classic() +
  scale_colour_brewer(palette = "Accent", name = "Sex") +
  labs(y = "Height (cm)",
       x = "Sex",
       title = "Boxplot of height by sex of human characters in Star Wars") +
  theme(legend.position ="none")
```



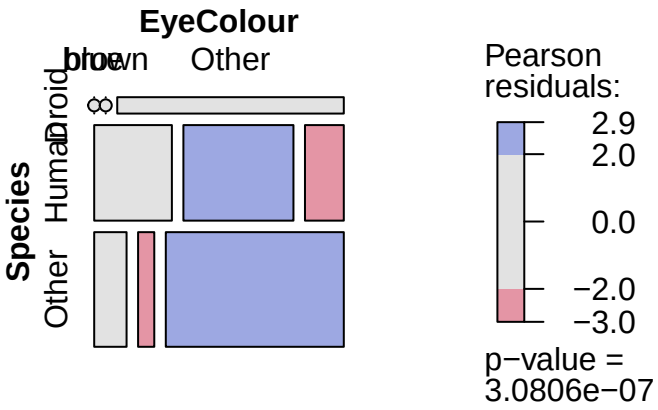
Mean height (bar chart)

```
starwars |>
  filter(species == "Human") |>
  group_by(sex) |>
  summarise(ht = mean(height, na.rm = TRUE),
            sd = sd(height, na.rm = TRUE)) |>
  ggplot(aes(x = sex, y = ht, fill = sex)) +
  geom_bar(stat = "identity") +
  geom_errorbar(aes(ymin = ht-sd, ymax = ht+sd), width = 0.2)+
  scale_fill_brewer(palette = "Accent") +
  labs(y = "Mean Height (cm)",
       x = "Sex",
       title = "Mean height by sex of human characters in Star Wars") +
  theme_classic() +
  theme(legend.position = "none")
```



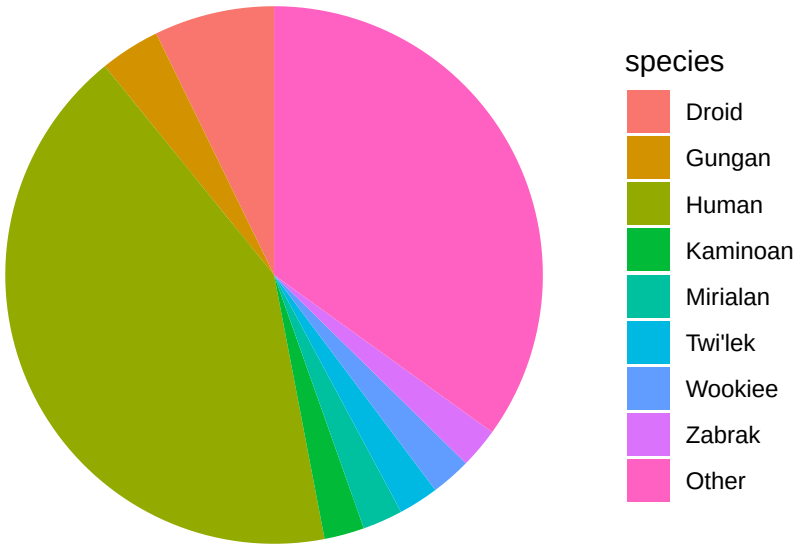
Mosaic Plot

```
library(vcd)


startbl <-  starwars |>
  mutate(Species = fct_lump_n(species, 2),
         EyeColour = fct_lump_n(eye_color,2))


mosaic(~ Species + EyeColour, data = startbl,shade = TRUE, legend = TRUE)
```

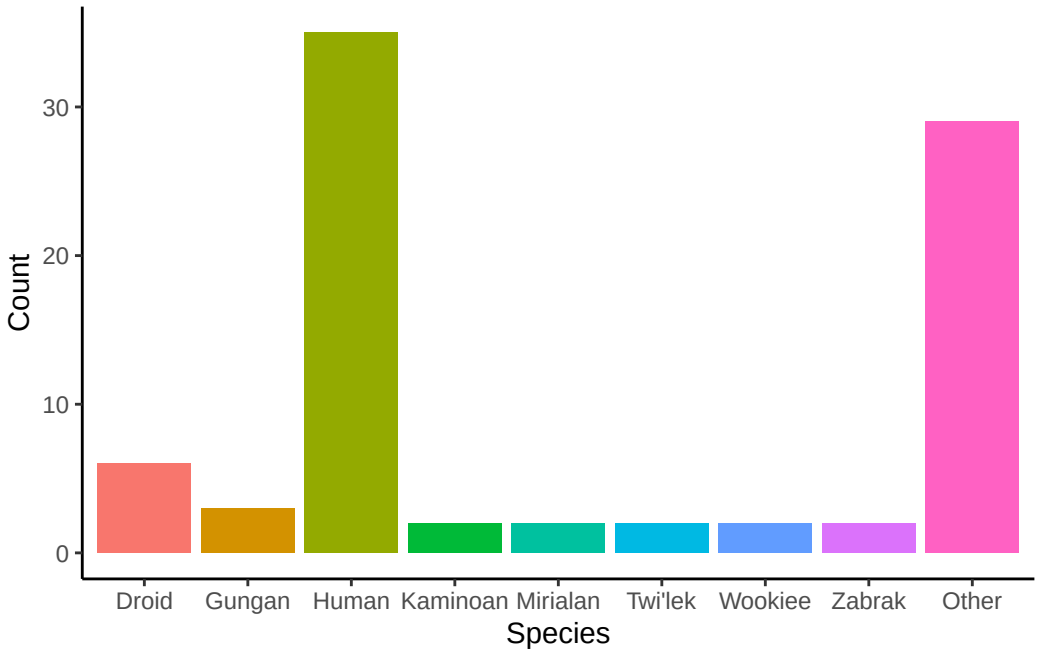


Pie Charts are Just Bad Bar Charts

```
starwars |>
  mutate(species = fct_lump_n(species, 4)) |>
  group_by(species) |>
  filter(!is.na(species)) |>
  tally() |>
  ggplot(aes(x = "", fill = species, y = n)) +
  geom_bar(stat = "identity", width = 1) +
  theme_void() +
  coord_polar("y", start = 0)
```




```
starwars |>
  mutate(species = fct_lump_n(species, 4)) |>
  group_by(species) |>
  filter(!is.na(species)) |>
  tally() |>
  ggplot(aes(x = species, fill = species, y = n)) +
  geom_bar(stat = "identity") +
  theme_classic() +
  labs(x = "Species", y = "Count") +
  theme(legend.position = "none")
```

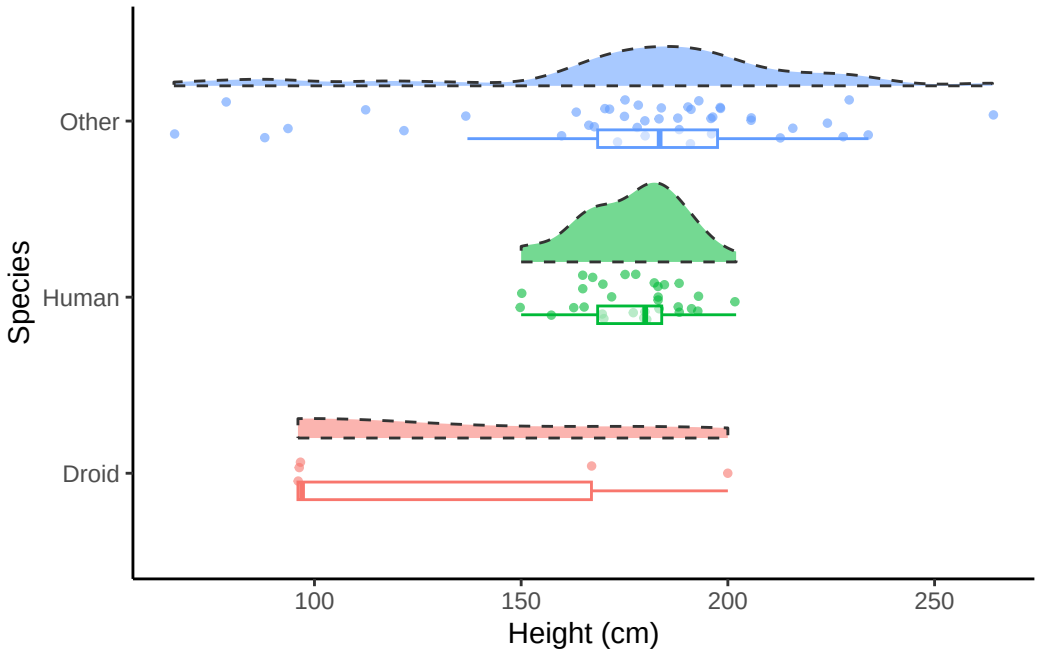


Wordclouds

```
library(wordcloud)
starwars |>
  count(homeworld) |>
  with(wordcloud(words = homeworld, freq = n, min.freq=1, random.order = FALSE, rot.per =
                 colors = brewer.pal(6, "Accent"), use.r.layout = FALSE))
```

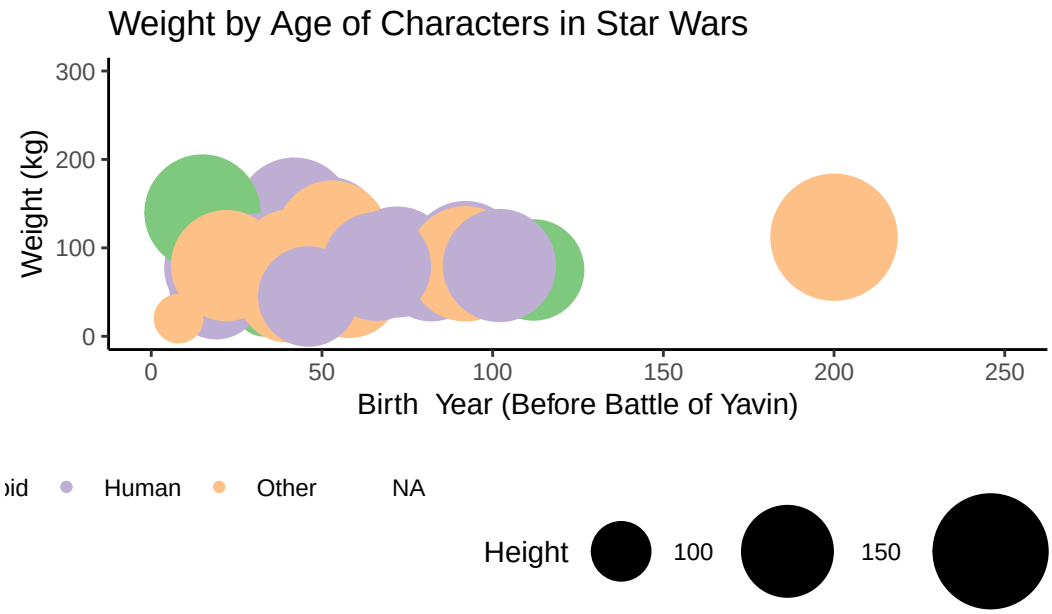
Raincloud Plots

```
starwars |>
  mutate(species = fct_lump_n(species,2)) |>
  filter(!is.na(species)) |>
  ggplot(aes(x = species)) +
  geom_point(aes(y = height, colour = species), position = position_jitter(width = .13), s
  see::geom_violinhalf(aes(y = height, alpha= 0.3, fill = species), linetype = "dashed", p
  geom_boxplot(aes(y = height, alpha = 0.3, colour = species), position = position_nudge(x
  theme_classic() +
  labs(x = "Species", y = "Height (cm)") +
  theme(legend.position = "none") +
  coord_flip()
```



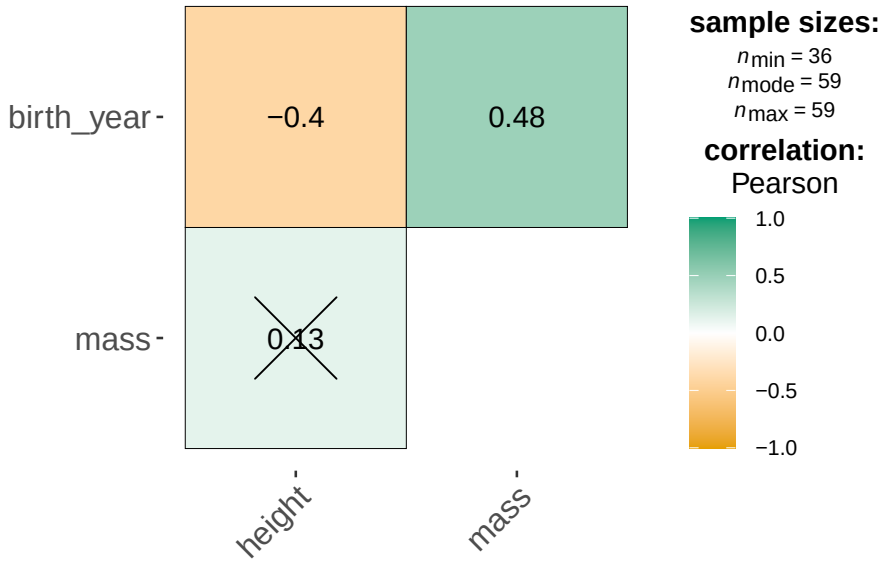
Bubble plots

```
starwars |>
  mutate(col = fct_lump_n(species, 2)) |>
  ggplot(aes(x = birth_year, y = mass, size = height, colour = col)) +
  geom_point() +
  scale_size(range = c(.1, 24), name="Height") +
  theme_classic() +
  scale_x_continuous(limits = c(0,250)) +
  scale_y_continuous(limits = c(0,300)) +
  scale_colour_brewer(palette = "Accent", name = "Species") +
  theme(legend.position = "bottom") +
  labs(x = "Birth  Year (Before Battle of Yavin)",
       y = "Weight (kg)",
       title = "Weight by Age of Characters in Star Wars")
```



Correlation plots

```
starwars |>
  select(height, mass, birth_year) |>
  ggcorrmat()
```



X = non−significant at p < 0.05 (Adjustment: Holm)

Lecture 3: The Mean as a Basic Model

Data and custom function for this lecture

```
library(tidyverse)

heifers <- tibble(heifers = c(211.3, 200.4, 220.1, 200.8, 222.0, 209.3,
             195.8, 220.4, 226.2, 218.7, 193.7, 209.7))



wage <- readxl::read_excel("assets/UKWageData2023ONS.xlsx",
                           skip = 5)

find_mode <- function(x) {
  ux <- unique(x)
  tab <- tabulate(match(x, ux))
  ux[tab == max(tab)]
}
```

Finding central tendency

```
heifers |>
  summarise(mean = mean(heifers),
            median = median(heifers),
            min = min(heifers),
            max = max(heifers),
            mode = find_mode(round(heifers, 0)))
```

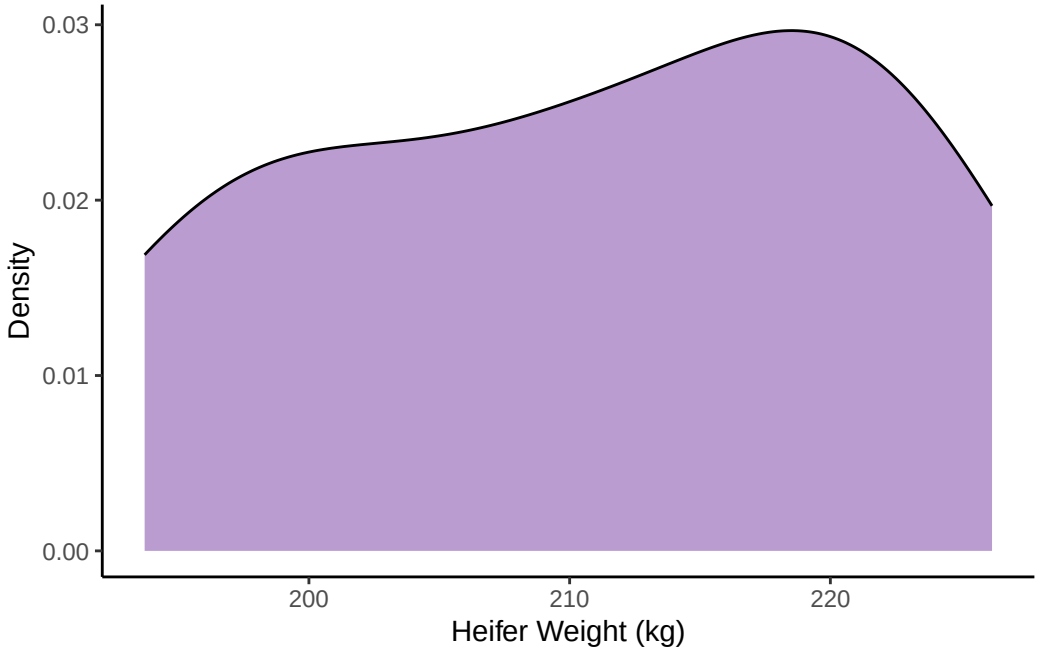
```
# A tibble: 1 x 5
   mean median   min   max  mode
  <dbl>  <dbl> <dbl> <dbl> <dbl>
1  211.   210.  194.  226.   220
```

```r
wage |>
  summarise(mean = mean(Median),
            median = median(Median),
            min = min(Median),
            max = max(Median),
            mode = find_mode(round(Median,0)))
```

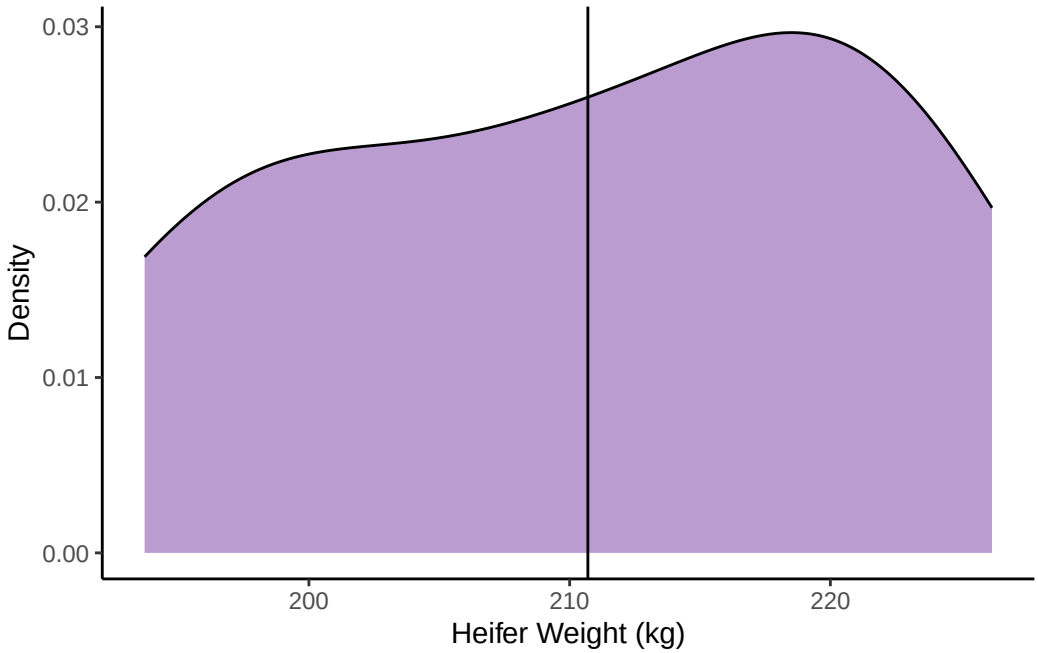
```
# A tibble: 4 x 5
    mean median   min   max  mode
   <dbl>  <dbl> <dbl> <dbl> <dbl>
1 34475.  31988 17859 84131 28216
2 34475.  31988 17859 84131 35248
3 34475.  31988 17859 84131 26000
4 34475.  31988 17859 84131 25000
```

The Mean and Outliers

```r
heifers |>
  ggplot(aes(x = heifers)) +
  geom_density(fill = "#bb9cd1") +
  theme_classic() +
  labs(x = "Heifer Weight (kg)",
       y = "Density")
```

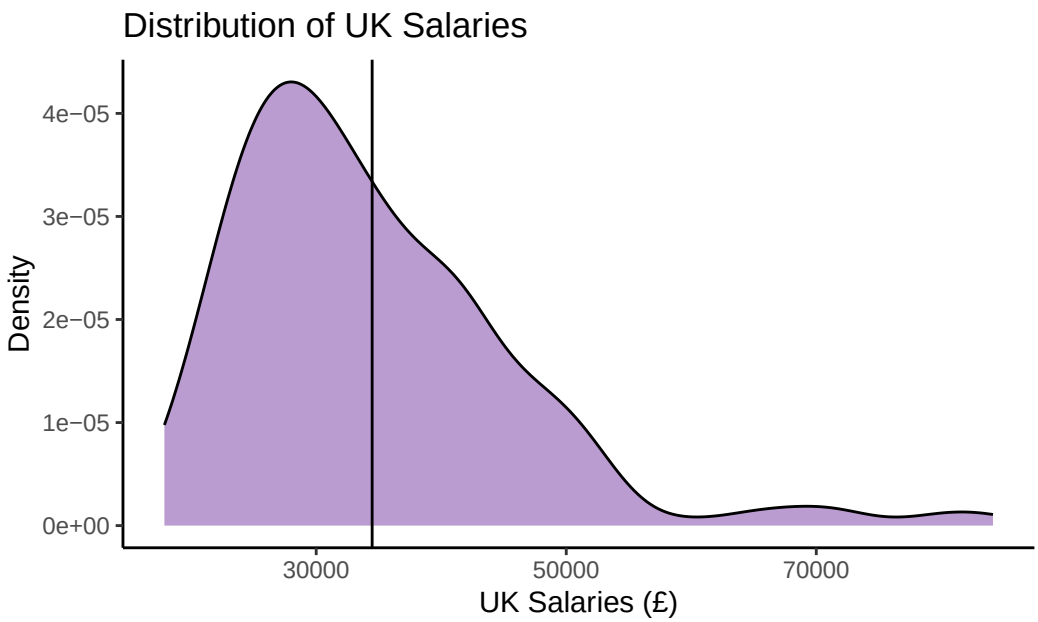



```
heifers |>
  ggplot(aes(x = heifers)) +
  geom_density(fill = "#bb9cd1") +
  geom_vline(aes(xintercept = 210.7)) +
  theme_classic() +
  labs(x = "Heifer Weight (kg)",
       y = "Density")
```



Mean UK Salary

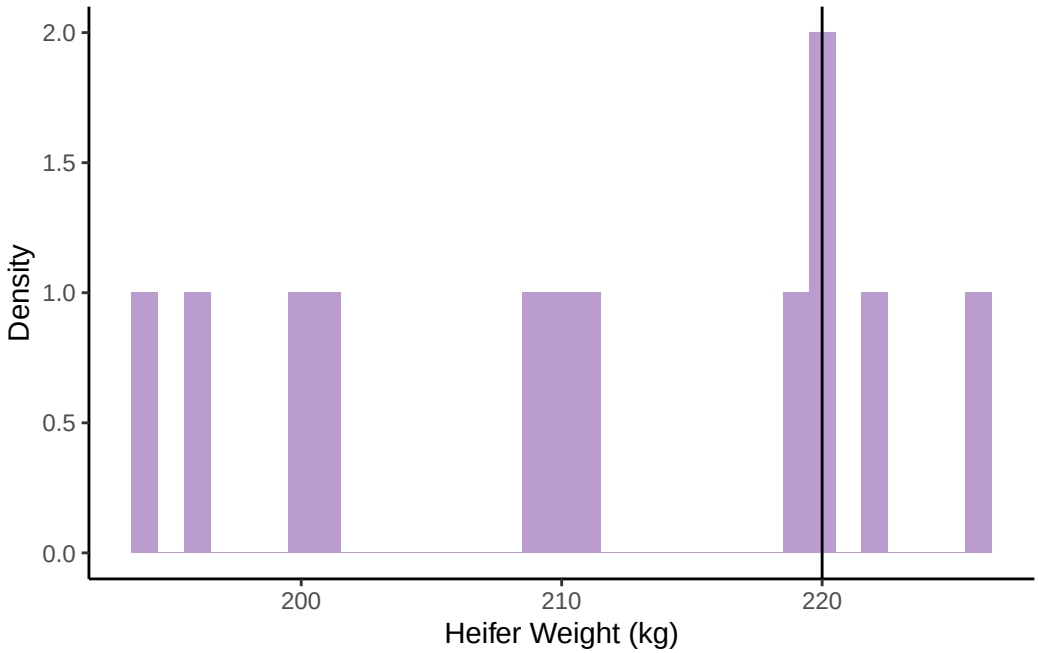
```
wage |>
  ggplot(aes(x = Median)) +
  geom_density(fill = "#bb9cd1") +
  theme_classic() +
  geom_vline(aes(xintercept = 34475)) +
  labs(x = "UK Salaries (£)",
       y = "Density",
       title = "Distribution of UK Salaries",
       caption = "Data taken from ONS 2023 Median Salaries by Field, n = 329 fields")
```



Data taken from ONS 2023 Median Salaries by Field, n = 329 fields

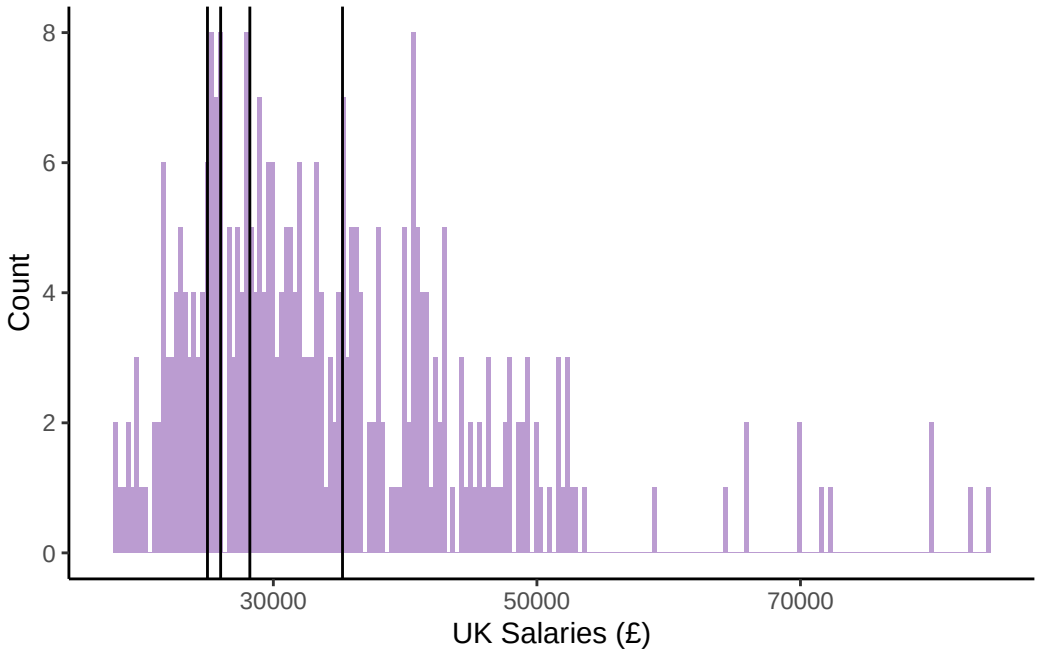
The Mode

```
heifers |>
  ggplot(aes(x = heifers)) +
  geom_histogram(fill = "#bb9cd1", binwidth = 1) +
  geom_vline(aes(xintercept = 220)) +
  theme_classic() +
  labs(x = "Heifer Weight (kg)",
       y = "Density")
```



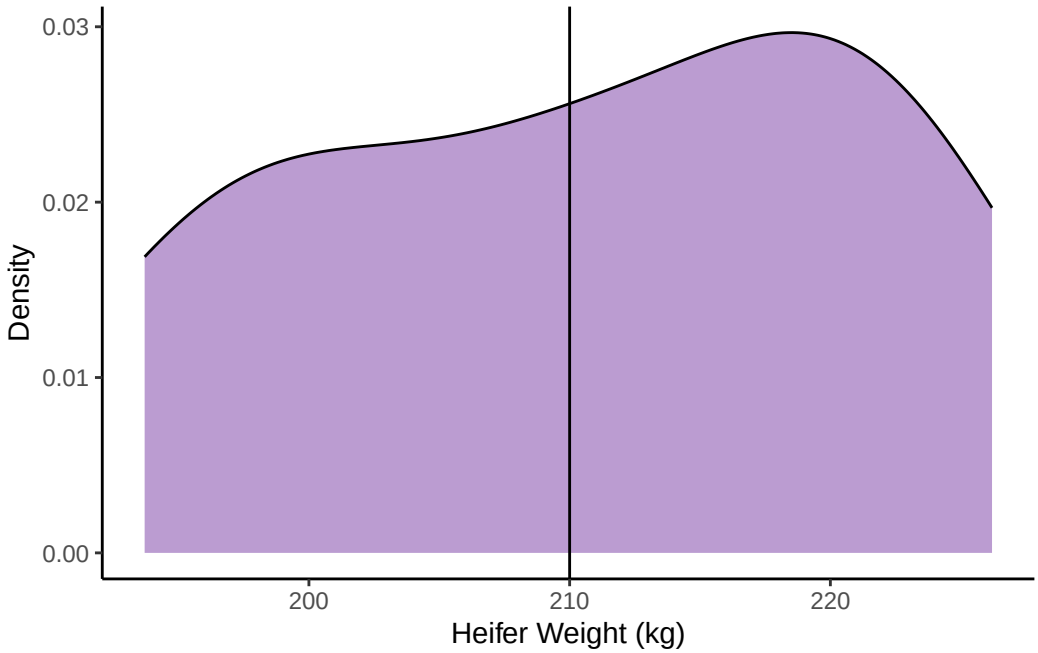
Multiple Modes

```
wage |>
  ggplot(aes(x = Median)) +
  geom_histogram(fill = "#bb9cd1", bins = 200) +
  geom_vline(aes(xintercept = 25000)) +
  geom_vline(aes(xintercept = 26000)) +
  geom_vline(aes(xintercept = 28216)) +
  geom_vline(aes(xintercept = 35248)) +
  theme_classic() +
  labs(x = "UK Salaries (£)",
       y = "Count")
```



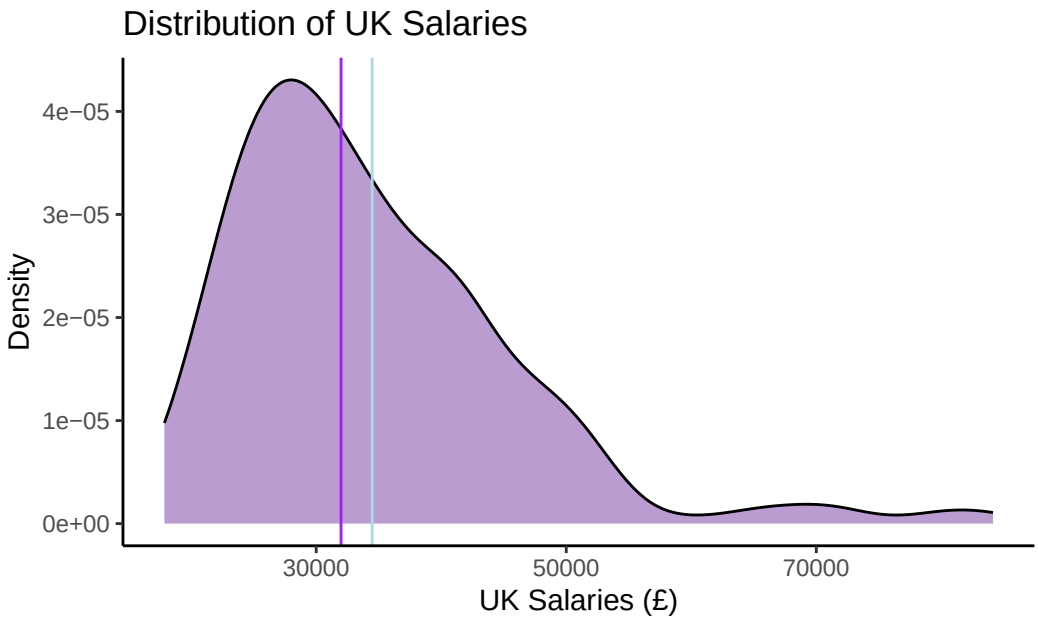
The Median

```
heifers |>
  ggplot(aes(x = heifers)) +
  geom_density(fill = "#bb9cd1") +
  geom_vline(aes(xintercept = 210)) +
  theme_classic() +
  labs(x = "Heifer Weight (kg)",
       y = "Density")
```



Median UK Salary

```
wage |>
  ggplot(aes(x = Median)) +
  geom_density(fill = "#bb9cd1") +
  theme_classic() +
  geom_vline(aes(xintercept = 34475), colour = "lightblue") +
  geom_vline(aes(xintercept = 31988),colour = "purple") +
  labs(x = "UK Salaries (£)",
       y = "Density",
       title = "Distribution of UK Salaries",
       caption = "Data taken from ONS 2023 Median Salaries by Field, n = 329 fields")
```



Data taken from ONS 2023 Median Salaries by Field, n = 329 fields

4 Week 3: Introduction to Analyses

4.1 Introduction to statistics

4.1.1 Set up your environment and packages

```
library(tidyverse)
library(easystats)
library(rstan)
library(rstanarm)

cat_weights <- tibble(avg_daily_snacks  = c(3, 2, 4, 2, 3, 1, 1, 0, 1, 0, 2, 3, 1, 2, 1, 3
                      weight = c(3.8, 3.9, 5, 3.7,  4.1, 3.6, 3.7, 3.6, 3.8, 4.1, 4.3, 3.9
                      environ = c("Indoor", "Indoor", "Outdoor", "Indoor",
                                  "Outdoor", "Indoor", "Outdoor", "Indoor",
                                  "Indoor", "Indoor", "Outdoor", "Indoor",
                                  "Outdoor", "Indoor", "Indoor", "Outdoor"))
```

4.1.2 Example data

```
cat_weights |>
  summarise("Mean Weight (kg)" = mean(weight),
            "SD Weight (kg)" = sd(weight),
            "Mean Daily Snacks" = mean (avg_daily_snacks),
            )
```

```
# A tibble: 1 x 3
  `Mean Weight (kg)` `SD Weight (kg)` `Mean Daily Snacks`
               <dbl>            <dbl>               <dbl>
1               3.92            0.373                1.81
```

4.1.3 Visualise

```
cat_weights |>
  ggplot(aes(x = avg_daily_snacks, y = weight)) +
  geom_point() +
  labs(x = "Average Daily Snacks", y = "Cat Weight") +
  theme_classic() +
  scale_y_continuous(limits = c(0,5))
```

4.1.4 A Linear Model

```
model_fcat <- lm(weight ~ avg_daily_snacks, data = cat_weights)
summary(model_fcat)
report::report(model_fcat)
parameters(model_fcat)
plot(model_parameters(model_fcat), show_intercept = TRUE)
plot(model_parameters(model_fcat))

cat_weights |>
  ggplot(aes(x = avg_daily_snacks, y = weight)) +
  geom_point() +
  labs(x = "Average Daily Snacks", y = "Cat Weight",
       caption = "Weight ~ Average Daily Snacks shown") +
  theme_classic() +
  scale_y_continuous(limits = c(0,5)) +
  geom_abline(slope = 0.20, intercept = 3.55)
```

4.1.5 A Bayesian Model

```
model_bcat <- stan_glm(weight ~ avg_daily_snacks, data = cat_weights)
summary(model_bcat)
describe_posterior(model_bcat)
report::report(model_bcat)

posteriors <- get_parameters(model_bcat)

posteriors |>
  ggplot(aes(x = avg_daily_snacks)) +
```

```
  geom_density(fill = "lightblue") +
  theme_classic() +
  labs(x = "Posterior Coefficient Estimates for Average Daily Snacks",
       y = "Density",
       caption = "Median Estimate Shown") +
  geom_vline(xintercept = 0.21, color = "darkblue", linewidth = 1)
```

4.1.6 A Linear model with a factor

```
model_fcat2 <- lm(weight ~ avg_daily_snacks + environ, data = cat_weights)
summary(model_fcat2)
report::report(model_fcat2)
parameters(model_fcat2)
plot(model_parameters(model_fcat2), show_intercept = TRUE)
plot(model_parameters(model_fcat2))


cat_weights |>
  ggplot(aes(x = avg_daily_snacks, y = weight, colour = environ)) +
  geom_point() +
  labs(x = "Average Daily Snacks", y = "Cat Weight",
       caption = "Weight ~ Average Daily Snacks shown") +
  theme_classic() +
  scale_y_continuous(limits = c(0,5)) +
  geom_smooth()
```

4.1.7 Bayesian Framework

```
model_bcat2 <- stan_glm(weight ~ avg_daily_snacks + environ, data = cat_weights)
summary(model_bcat2)
describe_posterior(model_bcat2)
report::report(model_bcat2)

posteriors2 <- get_parameters(model_bcat2)


posteriors2 |>
  pivot_longer(cols = c(avg_daily_snacks, environOutdoor),
```

```
              names_to = "Parameter",
              values_to="estimate") |>
  ggplot() +
  geom_density(aes(x = estimate, fill = Parameter)) +
  theme_classic() +
  labs(x = "Posterior Coefficient Estimates",
       y = "Density") +
  facet_wrap(facets = ~Parameter, ncol = 1) +
  theme(legend.position = "none")
```

4.2 Meta Analyses

Calculate rs from R2

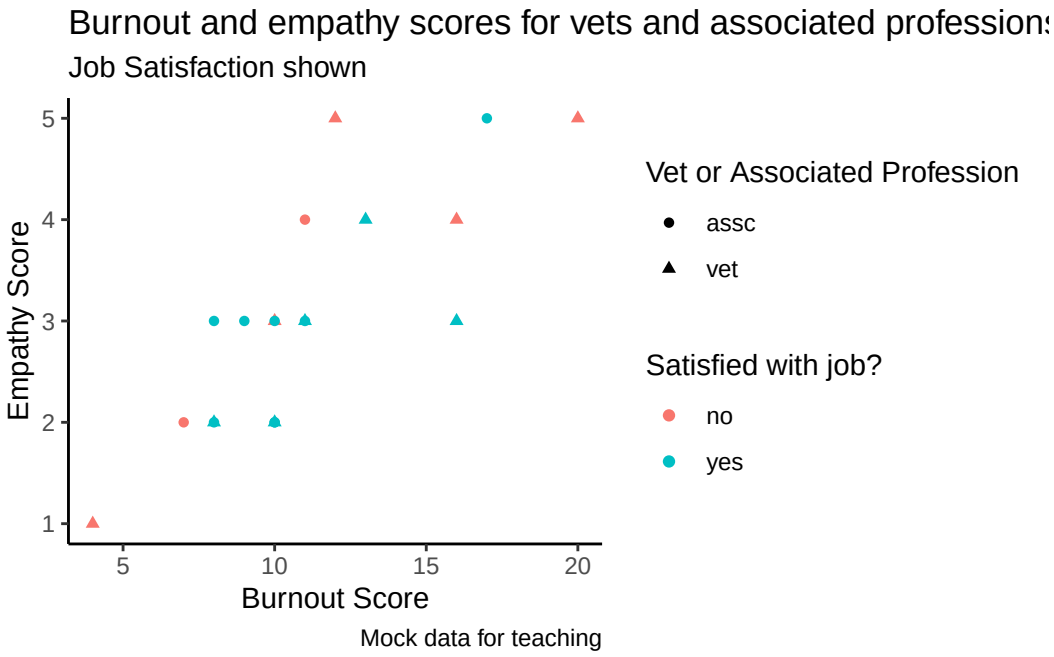
```
sqrt(0.11)
```

4.3 Effect Sizes

Mock Data and visualisation

```
job_dat <- tibble(job = c("vet", "vet", "vet","vet", "vet", "vet", "vet", "vet", "vet", "v
                          "assc", "assc", "assc", "assc", "assc", "assc", "assc", "assc",
                  burnout = c(13, 12, 4, 16, 16, 20, 8, 10, 11, 10,
                              10, 11, 8, 7, 8, 10, 9, 11, 17, 10),
                  empathy = c(4, 5, 1, 4,3, 5, 2, 3,3,2,
                              2, 3, 3, 2, 2, 3, 3, 4, 5, 2),
                  satisfaction = c("yes", "no", "no", "no", "yes", "no", "yes", "no", "yes
                                   "yes", "yes", "yes", "no", "yes", "yes", "yes","no", "y


job_dat |>
  ggplot(aes(x = burnout, y = empathy, shape = job, colour = satisfaction)) +
  geom_point() +
  theme_classic() +
  labs(title = "Burnout and empathy scores for vets and associated professions",
       subtitle = "Job Satisfaction shown",
       caption = "Mock data for teaching",
       x = "Burnout Score",
```

```
      y = "Empathy Score") +
  scale_shape_discrete(name = "Vet or Associated Profession") +
  scale_color_discrete(name = "Satisfied with job?")
```



4.3.0.1 Calculcate Cohen's d

```
library(effsize)

cohen.d(d = job_dat$burnout, f = job_dat$job)
```

```
Cohen's d

d estimate: -0.5048995 (medium)
95 percent confidence interval:
     lower      upper
-1.4593128  0.4495138
```

5 Week 4: Considerations for Collecting Data

6 Lecture 2 - Why do we model

This code will help you replicate the stats in Lecture 2

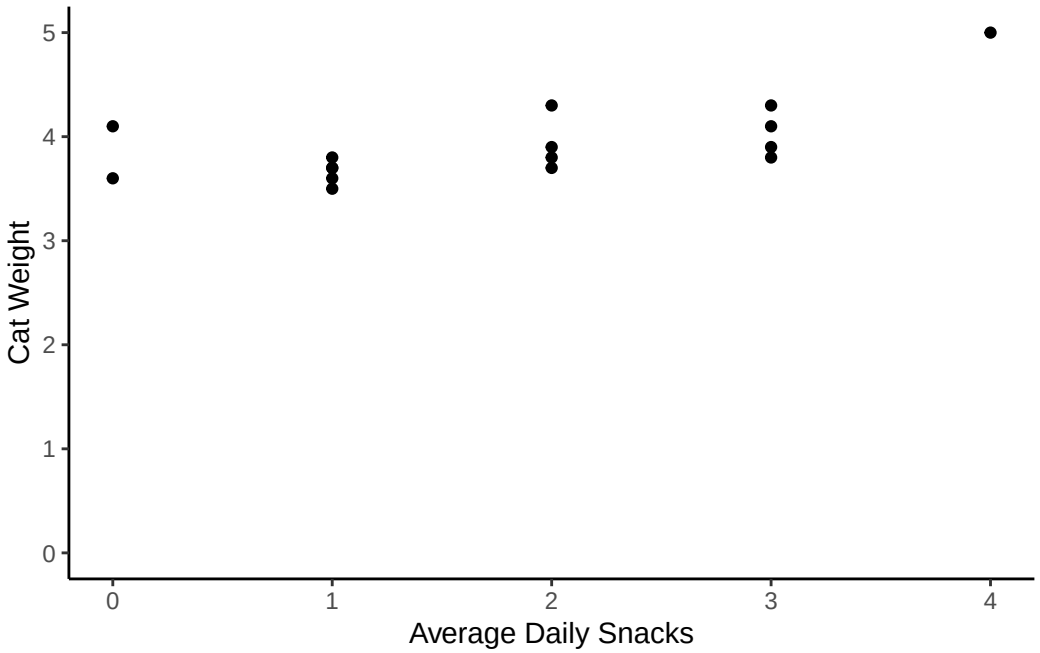
```
library(tidyverse)
library(report)
```

```
cat_weights <- tibble(avg_daily_snacks  = c(3, 2, 4, 2, 3, 1, 1, 0, 1, 0, 2, 3, 1, 2, 1, 3
                      weight = c(3.8, 3.9, 5, 3.7,  4.1, 3.6, 3.7, 3.6, 3.8, 4.1, 4.3, 3.9
```

Create the first plot

Note the changes to the y scale

```
cat_weights |>
  ggplot(aes(x = avg_daily_snacks, y = weight)) +
  geom_point() +
  labs(x = "Average Daily Snacks", y = "Cat Weight") +
  theme_classic() +
  scale_y_continuous(limits = c(0,5))
```


Cat Weight
5
4
3
2
1
0
0
1
2
3
4
Average Daily Snacks

7 Week 5: Sources of Data

This week has no content yet, please check back later!

8 Week 6: Analysing Qualitative Data

This week has no content yet, please check back later!

9 Week 7: Analysing Quantitative Data

This week has no content yet, please check back later!

10 Weeks 8 & 9: Analytical Softwares

This week has no content yet, please check back later!

11 Week 10: Project Proposals

This week has no content yet, please check back later!

12 References

References

The cover image duck comes from Pixabay, as a Creative Commons 0 image by Clker-Free-Vector-Images-3736

Brilleman, SL, MJ Crowther, M Moreno-Betancur, J Buros Novik, and R Wolfe. 2018. "Joint Longitudinal and Time-to-Event Models via Stan." https://github.com/stan-dev/stancon_talks/.

Fellows, Ian. 2018. *Wordcloud: Word Clouds*. https://CRAN.R-project.org/package=wordcloud.

Lüdecke, Daniel, Mattan S. Ben-Shachar, Indrajeet Patil, Brenton M. Wiernik, and Dominique Makowski. 2022. "Easystats: Framework for Easy Statistical Modeling, Visualization, and Reporting." *CRAN*. https://easystats.github.io/easystats/.

Patil, Indrajeet. 2021. "Visualizations with statistical details: The 'ggstatsplot' approach." *Journal of Open Source Software* 6 (61): 3167. https://doi.org/10.21105/joss.03167.

Stan Development Team. 2023. "RStan: The R Interface to Stan." https://mc-stan.org/.

Torchiano, Marco. 2020. *Effsize: Efficient Effect Size Computation*. https://doi.org/10.5281/zenodo.1480624.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Zeileis, Achim, David Meyer, and Kurt Hornik. 2007. "Residual-Based Shadings for Visualizing (Conditional) Independence." *Journal of Computational and Graphical Statistics* 16 (3): 507–25. https://doi.org/10.1198/106186007X237856.