Homework 2

Brady Miller

Table of contents

. 2
Question 1 2
Question 2 8
Question 3 13

Appendix **17**

Link to the Github repository

❗ Due: Tue, Feb 14, 2023 @ 11:59pm

Please read the instructions carefully before submitting your assignment.

1. This assignment requires you to only upload a `PDF` file on Canvas
2. Don't collapse any code cells before submitting.
3. Remember to make sure all your code output is rendered properly before uploading your submission.

Please add your name to the author information in the frontmatter before submitting your assignment

For this assignment, we will be using the Abalone dataset from the UCI Machine Learning Repository. The dataset consists of physical measurements of abalone (a type of marine snail) and includes information on the age, sex, and size of the abalone.

We will be using the following libraries:

```r
library(readr)
library(tidyr)
library(ggplot2)
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(purrr)
library(cowplot)
```

Question 1

> 30 points
>
> EDA using `readr`, `tidyr` and `ggplot2`

1.1 (5 points)

Load the "Abalone" dataset as a tibble called `abalone` using the URL provided below. The `abalone_col_names` variable contains a vector of the column names for this dataset (to be consistent with the R naming pattern). Make sure you read the dataset with the provided column names.

```r
library(readr)
url <- "http://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data"

abalone_col_names <- c(
  "sex",
  "length",
  "diameter",
  "height",
  "whole_weight",
  "shucked_weight",
  "viscera_weight",
  "shell_weight",
  "rings"
)

# reading in the url to create 'abalone' tibble
abalone <- read.csv(url)

# renaming columns in 'abalone' to col names provided
colnames(abalone) = abalone_col_names
```

1.2 (5 points)

Remove missing values and `NA`s from the dataset and store the cleaned data in a tibble called `df`. How many rows were dropped?

```
df <- abalone %>%
  na.omit()
```

No rows were dropped. The number of rows in 'df' is the same as 'abalone'. Also, per the website, the abalone has no missing data, so thus, no rows should have been dropped.

1.3 (5 points)

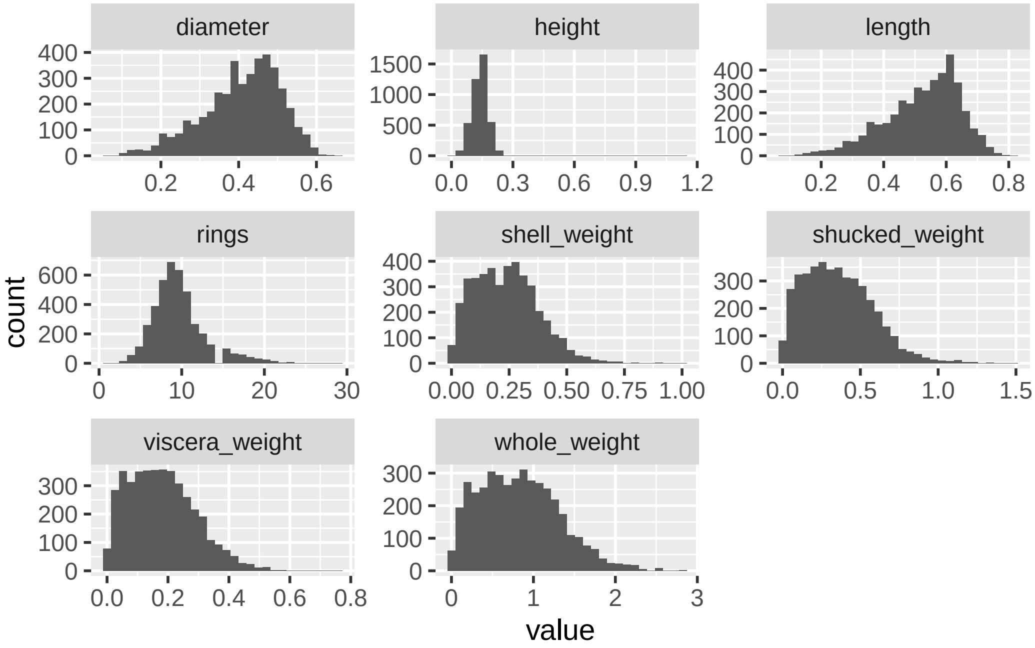
Plot histograms of all the quantitative variables in a **single plot** [1]

```
# using gather to put column names into one column and the values into another
df %>%
  select(!sex) %>%
  gather(cols, value) %>%
  ggplot() +
```

[1] You can use the `facet_wrap()` function for this. Have a look at its documentation using the help console in R

```
    geom_histogram(aes(value)) +
    facet_wrap(~ cols, scales = 'free')
```

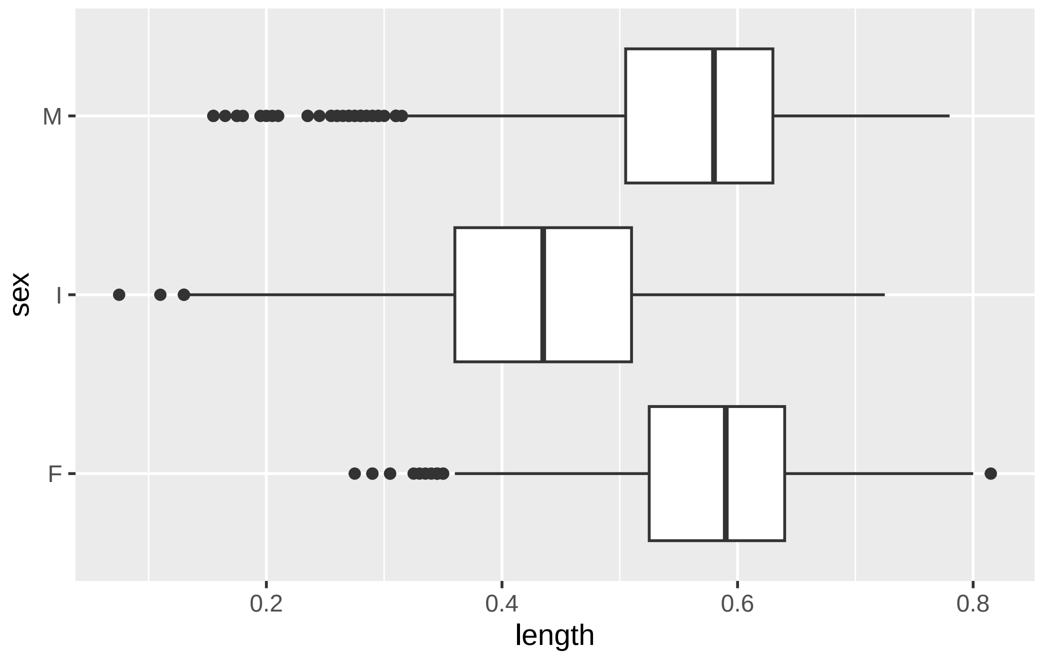
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



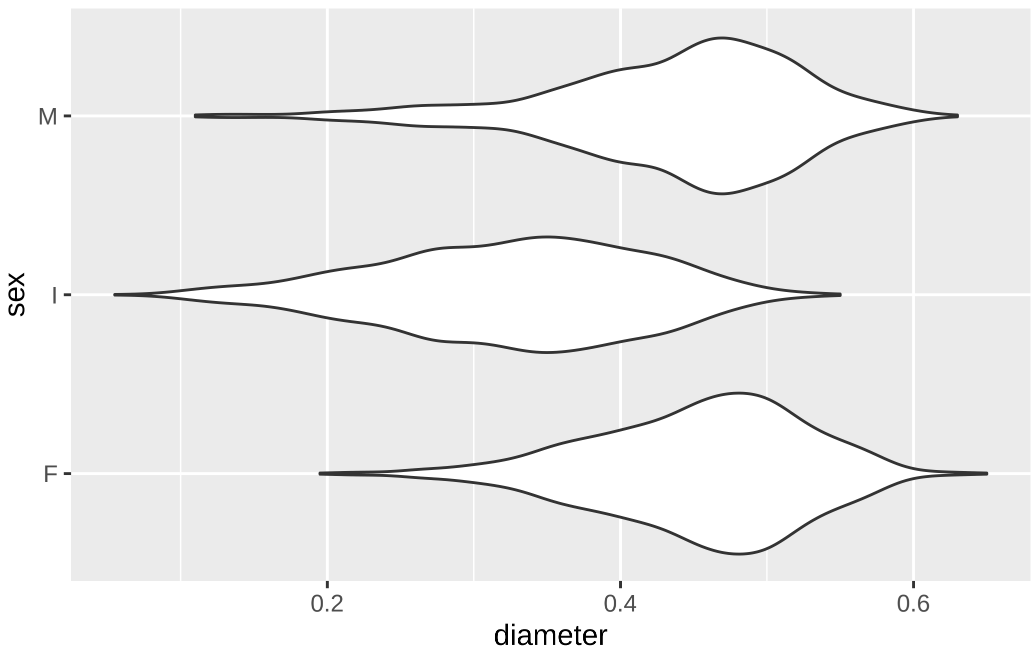
1.4 (5 points)

Create a boxplot of `length` for each `sex` and create a violin-plot of of `diameter` for each `sex`. Are there any notable differences in the physical appearences of abalones based on your analysis here?

```
df %>%
  ggplot() +
  geom_boxplot(aes(x = length, y = sex))
```



```
df %>%
  ggplot() +
  geom_violin(aes(x = diameter, y = sex))
```

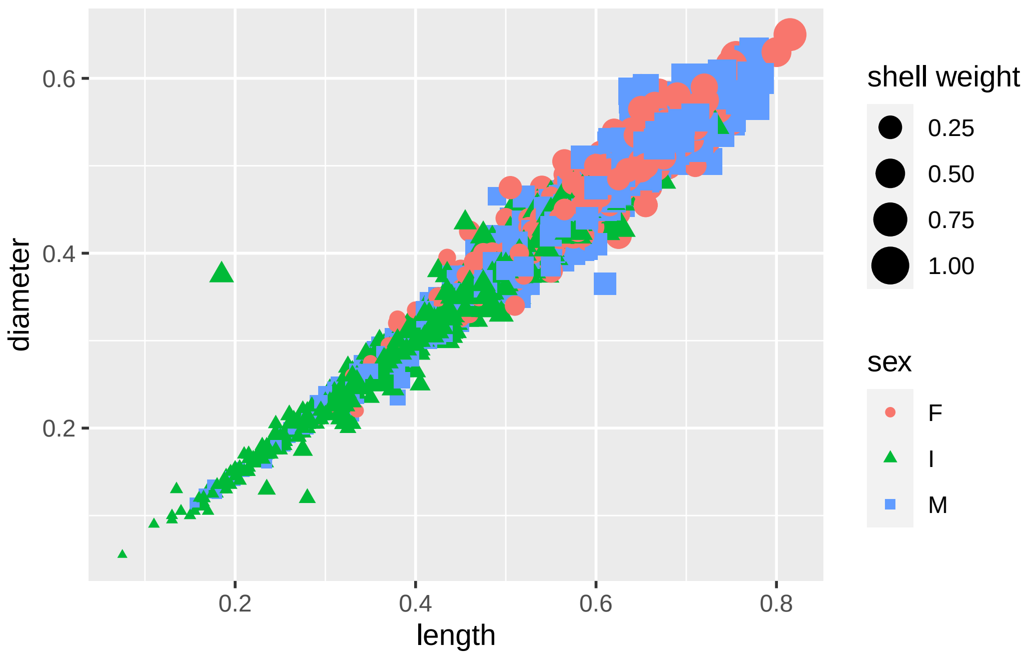


Based on the boxplot and violin plot, the male and female abalones are very similar in physical appearance as their distributions for length and diameter were nearly identical. They had both had an average of about 0.6 mm for their length and had the greatest density in diameter at about 0.49 mm. However, the miniumum and outlier lengths and diameters of the female abalones were much smaller than that of the male abalones. For the infant abalones, as expected, their lengths and diameters are much smaller than the males and females. I was surprised with how close the maximum length for the infant abalones was to the maximum lengths for the female and male abalones.

1.5 (5 points)

Create a scatter plot of `length` and `diameter`, and modify the shape and color of the points based on the `sex` variable. Change the size of each point based on the `shell_wight` value for each observation. Are there any notable anomalies in the dataset?

```
df %>%
  ggplot() +
  geom_point(aes(x = length, y = diameter, shape = sex, color = sex, size =
             shell_weight)) +
  labs(size = "shell weight")
```



There is one very noticeable anomaly in the data set. This occurs with an infant abalone, in which its diameter is nearly two times its length, which is very abnormal. Typically, infant abalones with a length of about 0.2 mm, have a diameter of about 0.15 mm and a shell_weight that's very small, in this case we will say 0.1 g. For this anomaly, it has a length of about 0.2 mm, a diameter of about 0.38 mm and shell weight around 0.25 g. —

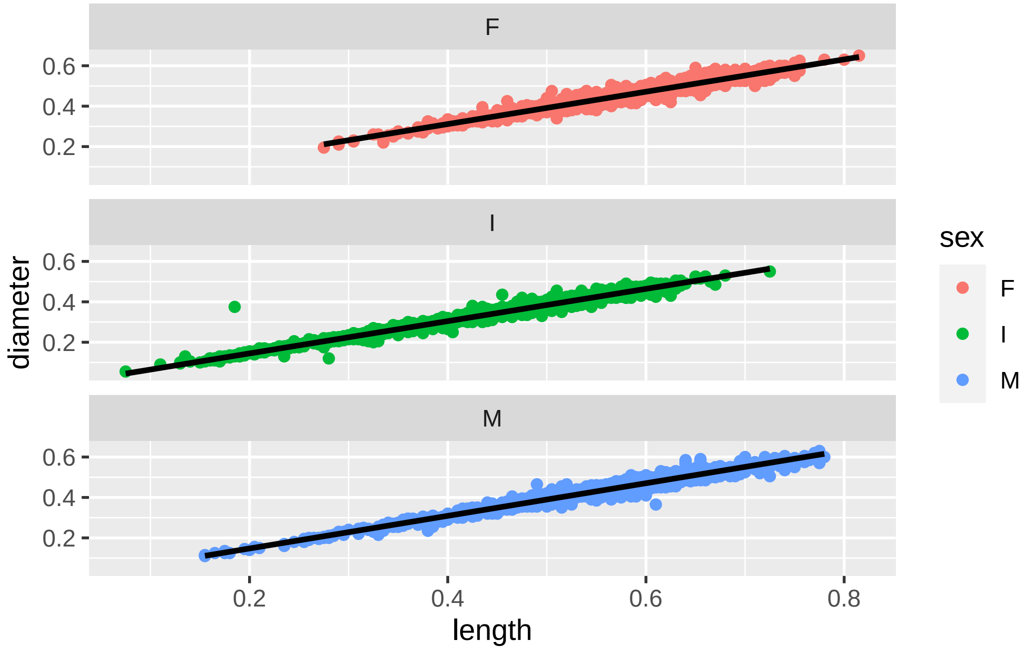
1.6 (5 points)

For each `sex`, create separate scatter plots of `length` and `diameter`. For each plot, also add a **linear** trendline to illustrate the relationship between the variables. Use the `facet_wrap()` function in R for this, and ensure that the plots are vertically stacked **not** horizontally. You should end up with a plot that looks like this: [2]

```
df %>%
  ggplot() +
  geom_point(aes(x = length, y = diameter, color = sex)) +
  geom_smooth(aes(x = length, y = diameter), method = lm, color = 'black') +
  facet_wrap(~sex, nrow = 3)
```

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

[2] Plot example for 1.6



—

Question 2

40 points

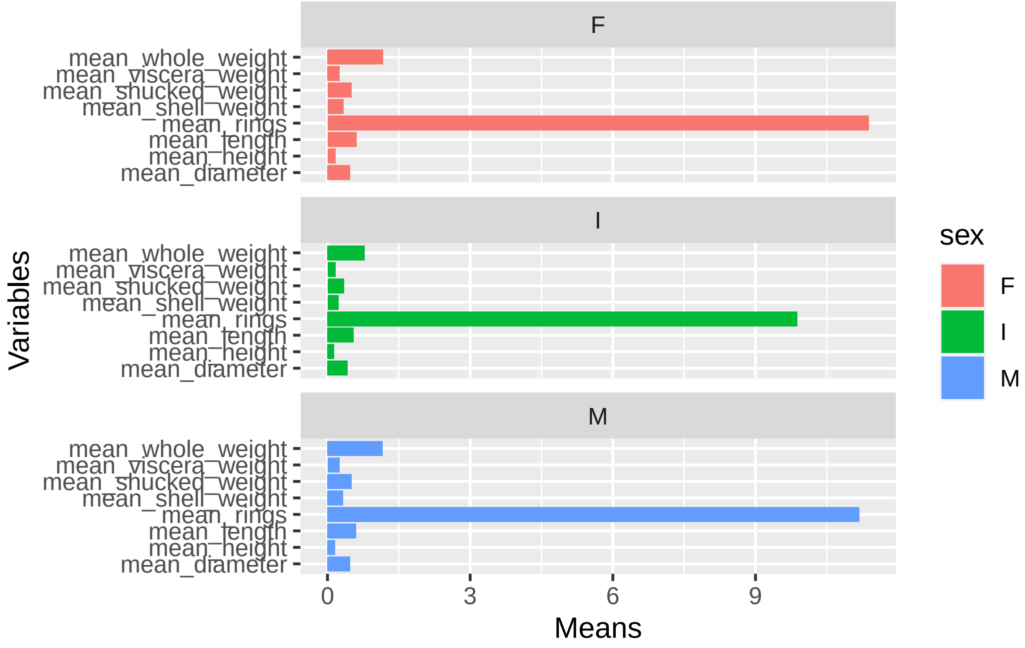
More advanced analyses using `dplyr`, `purrrr` and `ggplot2`

2.1 (10 points)

Filter the data to only include abalone with a length of at least 0.5 meters. Group the data by `sex` and calculate the mean of each variable for each group. Create a bar plot to visualize the mean values for each variable by `sex`.

```
# creating dataset that contains all the mean values each variable
# pivot_longer to put the column names into one column and the means into another
df2 <- df %>%
  filter(length >= 0.5) %>%
  group_by(sex) %>%
  summarise(mean_length = mean(length),
            mean_diameter = mean(diameter),
            mean_height = mean(height),
            mean_whole_weight = mean(whole_weight),
            mean_shucked_weight = mean(shucked_weight),
            mean_viscera_weight = mean(viscera_weight),
            mean_shell_weight = mean(shell_weight),
            mean_rings = mean(rings)) %>%
    pivot_longer(!sex,
                 names_to = 'Vars',
                 values_to = 'Means')

# plot the new dataset, facet wrapping by sex and stacking bar charts vertically
df2 %>%
  ggplot() +
  geom_bar(aes(x = Means, y = Vars, fill = sex), stat = 'identity') +
  facet_wrap(vars(sex), dir = 'v') +
  ylab("Variables")
```



2.2 (15 points)

Implement the following in a **single command**:

1. Temporarily create a new variable called `num_rings` which takes a value of:

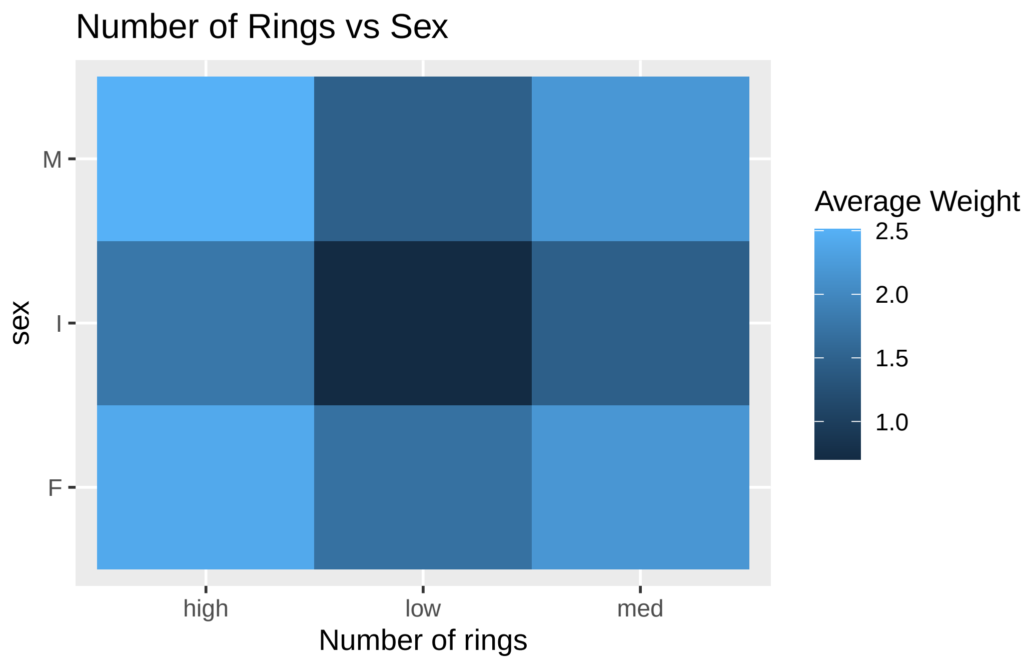
- `"low"` if `rings < 10`
- `"high"` if `rings > 20`, and
- `"med"` otherwise

2. Group `df` by this new variable and `sex` and compute `avg_weight` as the average of the `whole_weight + shucked_weight + viscera_weight + shell_weight` for each combination of `num_rings` and `sex`.

3. Use the `geom_tile()` function to create a tile plot of `num_rings` vs `sex` with the color indicating of each tile indicating the `avg_weight` value.

```r
# adding temp variable using if statements to assign new val based on cur value
# summarize the total weight of each abalone
df %>%
  mutate(num_rings=ifelse(rings>20, 'high', ifelse(rings<10, 'low', 'med'))) %>%
  group_by(num_rings, sex) %>%
  summarise(avg_weight = mean(whole_weight + shucked_weight + viscera_weight +
                             shell_weight)) %>%
  ggplot() +
  geom_tile(aes(x = num_rings, y = sex, fill = avg_weight)) +
  xlab("Number of rings") +
  labs(fill = 'Average Weight') +
  ggtitle('Number of Rings vs Sex')
```

```
`summarise()` has grouped output by 'num_rings'. You can override using the
`.groups` argument.
```



2.3 (5 points)

Make a table of the pairwise correlations between all the numeric variables rounded to 2 decimal points. Your final answer should look like this [3]

```
# selecting all columns in table except for the sex column
df2 <- df %>%
  select(length, diameter, height, whole_weight, shucked_weight, viscera_weight,
         shell_weight, rings)

# getting the correlation values for each pair of quantitative variables
data.cor <- round(cor(df2),2)
data.cor
```

```
               length diameter height whole_weight shucked_weight
length           1.00     0.99   0.83         0.93           0.90
diameter         0.99     1.00   0.83         0.93           0.89
height           0.83     0.83   1.00         0.82           0.77
whole_weight     0.93     0.93   0.82         1.00           0.97
```

[3] Table for 2.3

```
shucked_weight   0.90     0.89   0.77         0.97           1.00
viscera_weight   0.90     0.90   0.80         0.97           0.93
shell_weight     0.90     0.91   0.82         0.96           0.88
rings            0.56     0.58   0.56         0.54           0.42
               viscera_weight shell_weight rings
length                   0.90         0.90  0.56
diameter                 0.90         0.91  0.58
height                   0.80         0.82  0.56
whole_weight             0.97         0.96  0.54
shucked_weight           0.93         0.88  0.42
viscera_weight           1.00         0.91  0.50
shell_weight             0.91         1.00  0.63
rings                    0.50         0.63  1.00
```

2.4 (10 points)

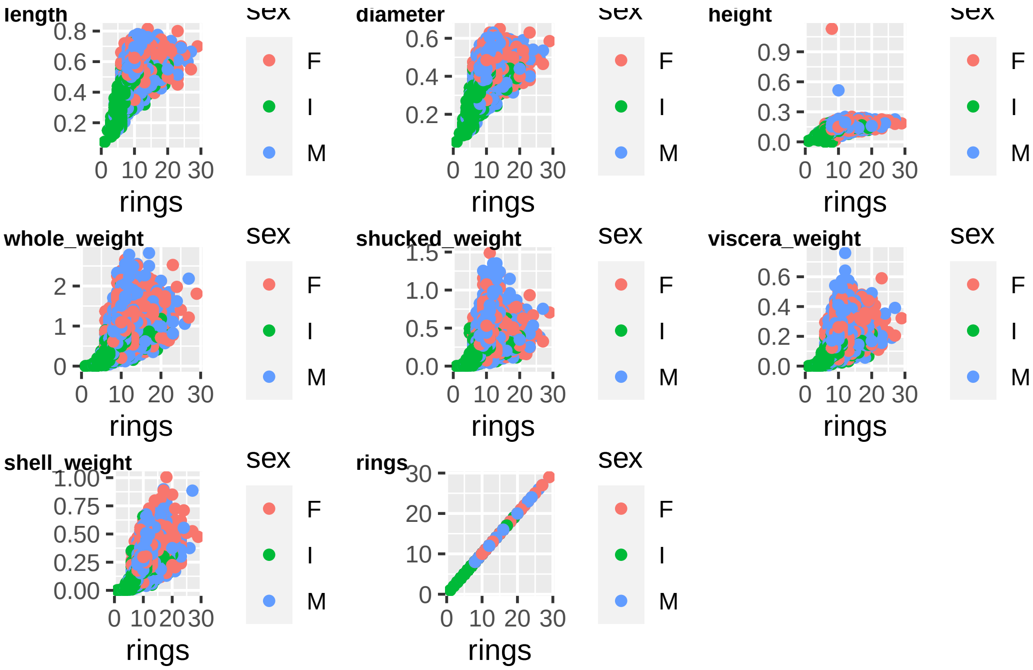
Use the `map2()` function from the `purrr` package to create a scatter plot for each *quantitative* variable against the number of `rings` variable. Color the points based on the `sex` of each abalone. You can use the `cowplot::plot_grid()` function to finally make the following grid of plots.

```r
# creating 2 new datasets that will be used for the map2() function
dfrings <- df %>%
  select(rings)

dfquant <- df %>%
  select(!sex)

# specifying the 2 datasets that are to be plotted against each other
# using ggplot to indicate we want rings to be plotted against other variables
plot <- map2(dfquant, dfrings, ~ ggplot(df) + geom_point(aes(x = rings, y=.x,
             color=sex)) + ylab(' '))

# using cowplot::plot_grid() to make the grid of plots
cowplot::plot_grid(plotlist = plot, labels = colnames(dfquant), ncol = 3,
                   vjust = 0.85, hjust = 0, label_size = 8)
```



Question 3

30 points

Linear regression using `lm`

3.1 (10 points)

Perform a simple linear regression with `diameter` as the covariate and `height` as the response. Interpret the model coefficients and their significance values.

```
x <- df$diameter
y <- df$height

model <- lm(y ~ x, df)
summary(model)
```

```
Call:
lm(formula = y ~ x, data = df)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.15513 -0.01044 -0.00148  0.00852  1.00906 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.003784   0.001512  -2.502   0.0124 *  
x            0.351346   0.003602  97.540   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0231 on 4174 degrees of freedom
Multiple R-squared:  0.6951,    Adjusted R-squared:  0.695 
F-statistic:  9514 on 1 and 4174 DF,  p-value: < 2.2e-16
```

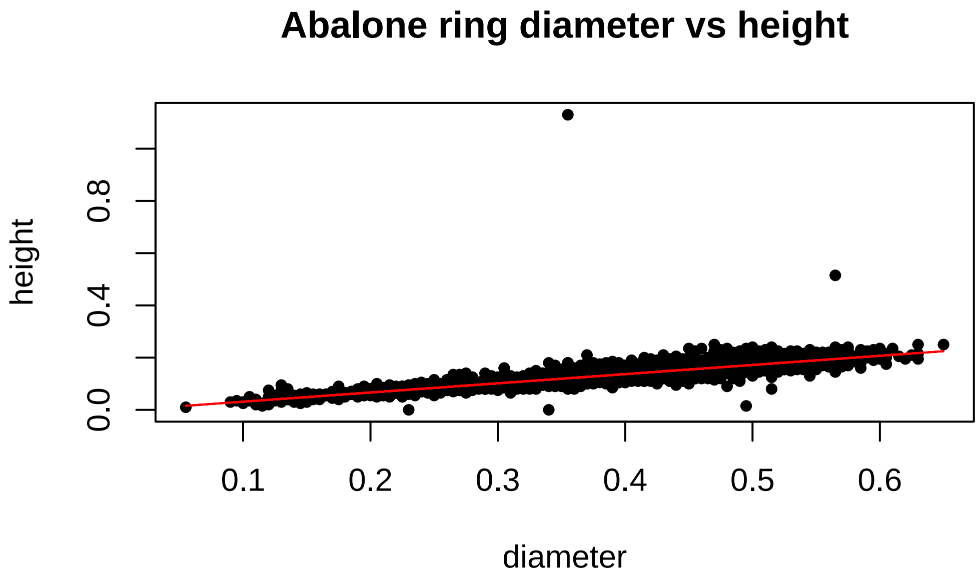
The intercept coefficient is -0.003784. This means that a 'hypothetical' abalone with diameter of 0 mm will have a length of -0.00378 mm. Obviously this isn't possible, but it shows that the intercept of the regression line is -0.003784 mm. The diameter coefficient is the slope and is equal to 0.351346, which means that for every increase in diameter by 1 mm, then the length will increase by 0.351346 mm. The significance value for the intercept (β_0) is somewhat high at 0.0124, so that *p*-value has some significant. The diameter *p*-value is very small, indicating that it is very significant. This tells us to accept the alternate hypothesis against the null hypothesis, meaning that diameter is a good predictor of length.

3.2 (10 points)

Make a scatterplot of `height` vs `diameter` and plot the regression line in `color="red"`. You can use the base `plot()` function in R for this. Is the linear model an appropriate fit for this relationship? Explain.

```r
x <- df$diameter
y <- df$height

plot(x, y, xlab = 'diameter', ylab = 'height', pch=20,
     main = 'Abalone ring diameter vs height')
lines(x, fitted(lm(y ~x)), col = 'red')
```



This linear model seems to be an pretty appropriate fit for this relationship. The regression line does a good job reducing potential error from the points by passing through the middle section of the chunks of points. There are few outliers in the data set that may impact the R^2 value because of how far from the regression line they are. Despite this, the linear model is an appropriate representation and predictor of the data for this relationship.

3.3 (10 points)

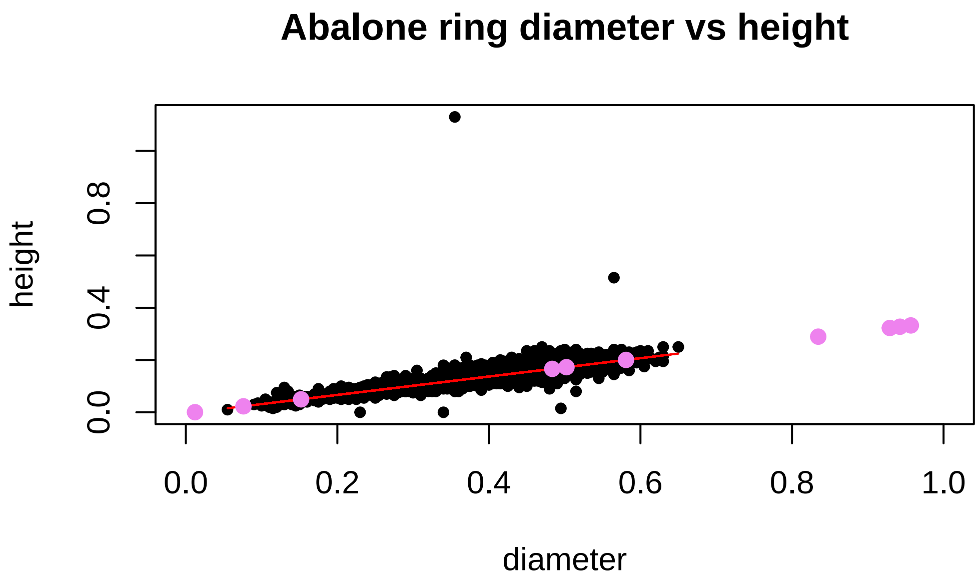
Suppose we have collected observations for "new" abalones with `new_diameter` values given below. What is the expected value of their `height` based on your model above? Plot these new observations along with your predictions in your plot from earlier using `color="violet"`

```
new_diameters <- c(
  0.15218946,
  0.48361548,
  0.58095513,
  0.07603687,
  0.50234599,
  0.83462092,
  0.95681938,
  0.92906875,
  0.94245437,
  0.01209518
)

new_x <- data.frame(x = new_diameters)

new_height <- predict(model, new_x)


plot(x, y, xlab = 'diameter', ylab = 'height', pch=20,
     main = 'Abalone ring diameter vs height', xlim = c(0,1))
lines(x, fitted(lm(y~x)), col = 'red')
points(new_diameters %>% unlist(), new_height, col = 'violet', pch = 19)
```



Appendix

Session Information

Print your R session information using the following command

```
sessionInfo()
```

```
R version 4.2.2 (2022-10-31 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 22000)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

attached base packages:
[1] stats     graphics  grDevices datasets  utils     methods   base

other attached packages:
[1] cowplot_1.1.1 purrr_1.0.1   dplyr_1.0.10  ggplot2_3.4.0 tidyr_1.2.1
[6] readr_2.1.3

loaded via a namespace (and not attached):
 [1] pillar_1.8.1     compiler_4.2.2   tools_4.2.2      digest_0.6.31
 [5] lattice_0.20-45  nlme_3.1-160     gtable_0.3.1     jsonlite_1.8.4
 [9] evaluate_0.20    lifecycle_1.0.3  tibble_3.1.8     mgcv_1.8-41
[13] pkgconfig_2.0.3  rlang_1.0.6      Matrix_1.5-1     cli_3.6.0
[17] DBI_1.1.3        rstudioapi_0.14  yaml_2.3.6       xfun_0.36
[21] fastmap_1.1.0    withr_2.5.0      stringr_1.5.0    knitr_1.41
[25] generics_0.1.3   vctrs_0.5.1      hms_1.1.2        grid_4.2.2
[29] tidyselect_1.2.0 glue_1.6.2       R6_2.5.1         fansi_1.0.3
[33] rmarkdown_2.20   farver_2.1.1     tzdb_0.3.0       magrittr_2.0.3
```

```
[37] splines_4.2.2    scales_1.2.1     ellipsis_0.3.2   htmltools_0.5.4
[41] assertthat_0.2.1 colorspace_2.0-3 renv_0.16.0-53   labeling_0.4.2
[45] utf8_1.2.2       stringi_1.7.12   munsell_0.5.0
```