Homework 2

Kesong Lin

Table of contents

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",
```

```r
    "diameter",
    "height",
    "whole_weight",
    "shucked_weight",
    "viscera_weight",
    "shell_weight",
    "rings"
  )

  abalone <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/abalone/ab
    as_tibble()
  abalone
```

```
# A tibble: 4,177 x 9
   sex   length diameter height whole_weight shucked_wei~1 visce~2 shell~3 rings
   <chr>  <dbl>    <dbl>  <dbl>        <dbl>         <dbl>   <dbl>   <dbl> <int>
 1 M      0.455    0.365  0.095        0.514        0.224   0.101    0.15     15
 2 M      0.35     0.265  0.09         0.226        0.0995  0.0485   0.07      7
 3 F      0.53     0.42   0.135        0.677        0.256   0.142    0.21      9
 4 M      0.44     0.365  0.125        0.516        0.216   0.114    0.155    10
 5 I      0.33     0.255  0.08         0.205        0.0895  0.0395   0.055     7
 6 I      0.425    0.3    0.095        0.352        0.141   0.0775   0.12      8
 7 F      0.53     0.415  0.15         0.778        0.237   0.142    0.33     20
 8 F      0.545    0.425  0.125        0.768        0.294   0.150    0.26     16
 9 M      0.475    0.37   0.125        0.509        0.216   0.112    0.165     9
10 F      0.55     0.44   0.15         0.894        0.314   0.151    0.32     19
# ... with 4,167 more rows, and abbreviated variable names 1: shucked_weight,
#   2: viscera_weight, 3: shell_weight
```

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?

```r
  df <- abalone %>% drop_na()

  rows_dropped <- nrow(abalone) - nrow(df)

  df
```

```
# A tibble: 4,177 x 9
   sex   length diameter height whole_weight shucked_wei~1 visce~2 shell~3 rings
   <chr>  <dbl>    <dbl>  <dbl>        <dbl>         <dbl>   <dbl>   <dbl> <int>
 1 M      0.455    0.365  0.095        0.514        0.224   0.101    0.15     15
 2 M      0.35     0.265  0.09         0.226        0.0995  0.0485   0.07      7
 3 F      0.53     0.42   0.135        0.677        0.256   0.142    0.21      9
 4 M      0.44     0.365  0.125        0.516        0.216   0.114    0.155    10
 5 I      0.33     0.255  0.08         0.205        0.0895  0.0395   0.055     7
 6 I      0.425    0.3    0.095        0.352        0.141   0.0775   0.12      8
 7 F      0.53     0.415  0.15         0.778        0.237   0.142    0.33     20
 8 F      0.545    0.425  0.125        0.768        0.294   0.150    0.26     16
 9 M      0.475    0.37   0.125        0.509        0.216   0.112    0.165     9
10 F      0.55     0.44   0.15         0.894        0.314   0.151    0.32     19
# ... with 4,167 more rows, and abbreviated variable names 1: shucked_weight,
#   2: viscera_weight, 3: shell_weight
```

```r
rows_dropped
```

```
[1] 0
```

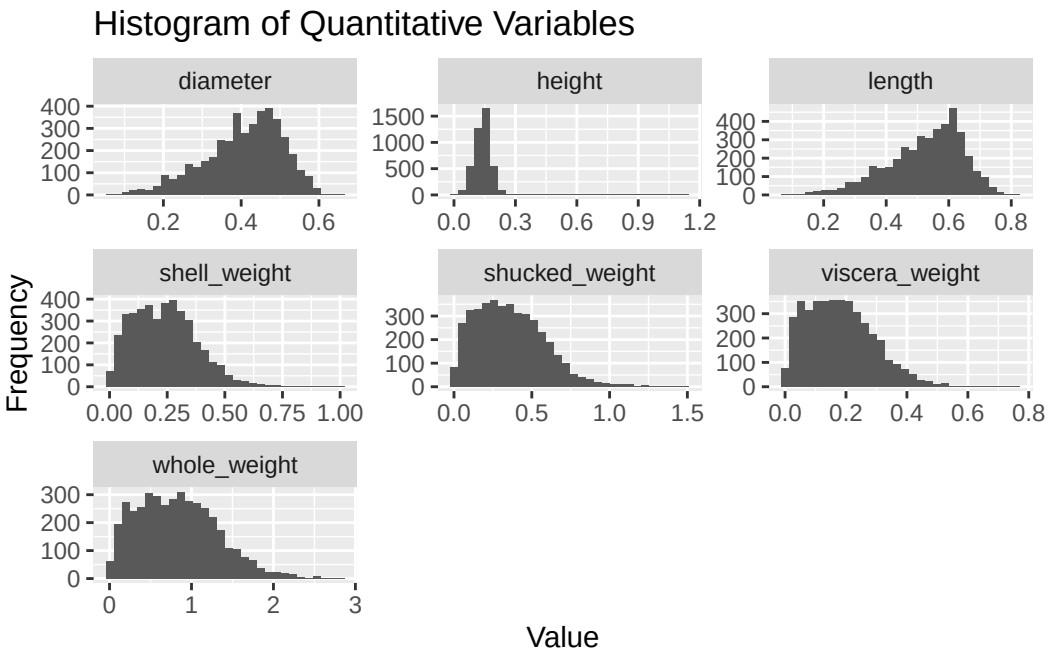
```
0 roll droped
```

1.3 (5 points)

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

```r
df_long <- tidyr::pivot_longer(df, cols = starts_with(c("len", "diam", "heig", "w", "sh",


ggplot(df_long, aes(x = value)) +
  geom_histogram(bins = 30) +
  facet_wrap(~variable, scales = "free") +
  xlab("Value") +
  ylab("Frequency") +
  ggtitle("Histogram of Quantitative Variables")
```

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



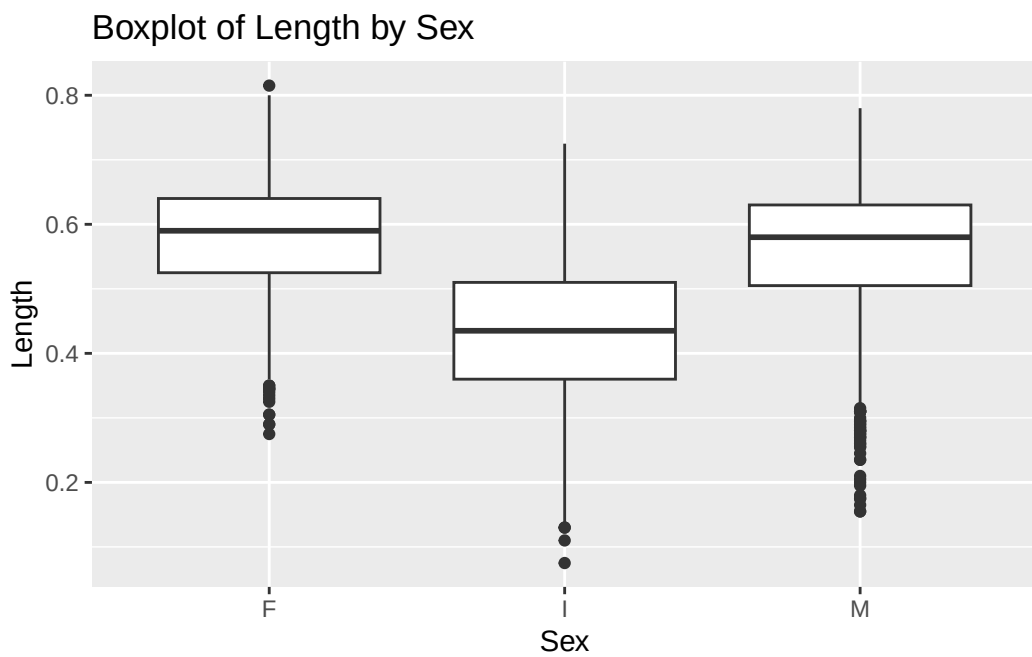
1.4 (5 points)

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?

```
ggplot(df, aes(x = df$sex, y = df$length)) +
  geom_boxplot() +
  ggtitle("Boxplot of Length by Sex") +
  xlab("Sex") +
  ylab("Length")
```

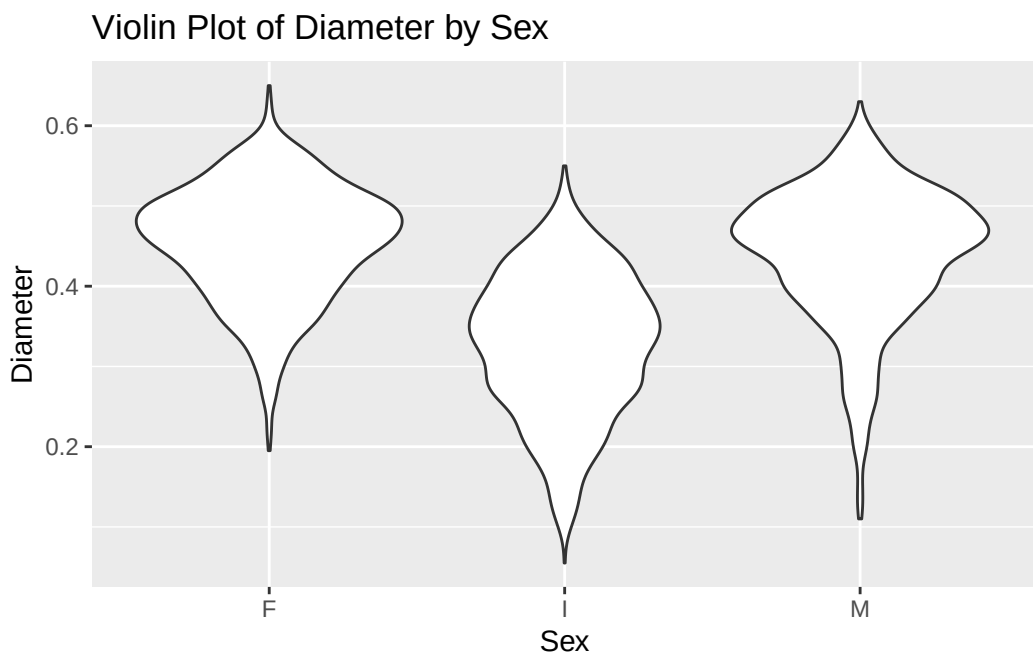
```
Warning: Use of `df$sex` is discouraged.
i Use `sex` instead.
```

```
Warning: Use of `df$length` is discouraged.
i Use `length` instead.
```




```r
ggplot(df, aes(x = df$sex, y = diameter)) +
  geom_violin() +
  ggtitle("Violin Plot of Diameter by Sex") +
  xlab("Sex") +
  ylab("Diameter")
```

```
Warning: Use of `df$sex` is discouraged.
i Use `sex` instead.
```

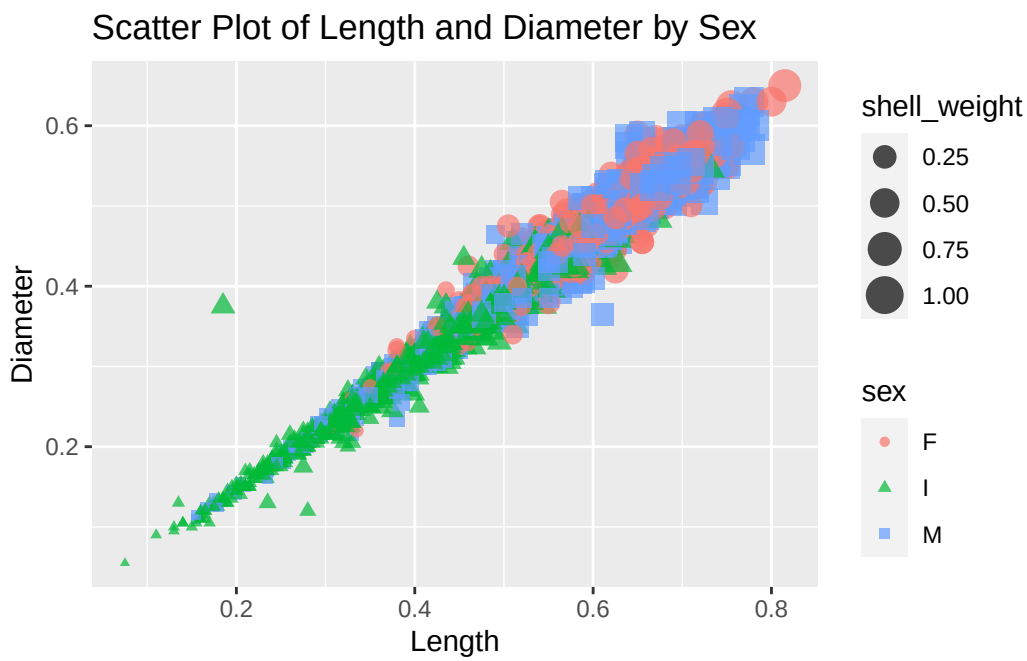



```
Based on graph, we can easily see abalones of different sexes show marked differences in body
```

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?

```
ggplot(df, aes(x = length, y = diameter, color = sex, shape = sex, size = shell_weight)) +
  geom_point(alpha = 0.7) +
  ggtitle("Scatter Plot of Length and Diameter by Sex") +
  xlab("Length") +
  ylab("Diameter")
```




```
From the graph, i dont think there is any notable anomalies, but there has couple outliers i
```

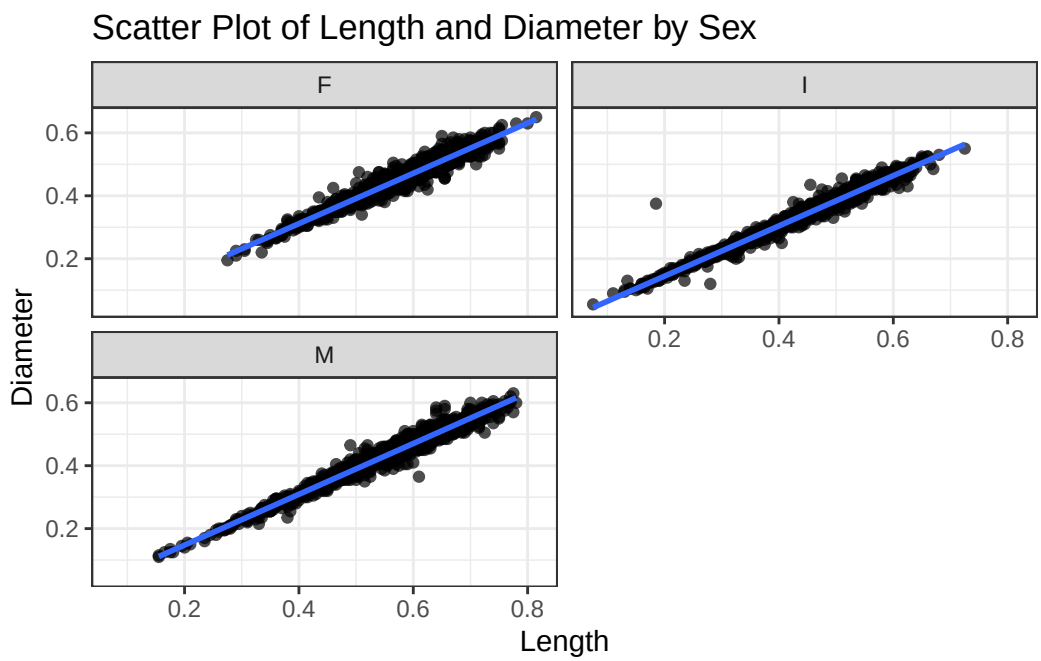
1.6 (5 points)

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]

```r
ggplot(df, aes(x = length, y = diameter)) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "lm", se = FALSE) +
  ggtitle("Scatter Plot of Length and Diameter by Sex") +
  xlab("Length") +
  ylab("Diameter") +
  facet_wrap(~ sex, nrow = 2) +
  theme_bw()
```

```
`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)

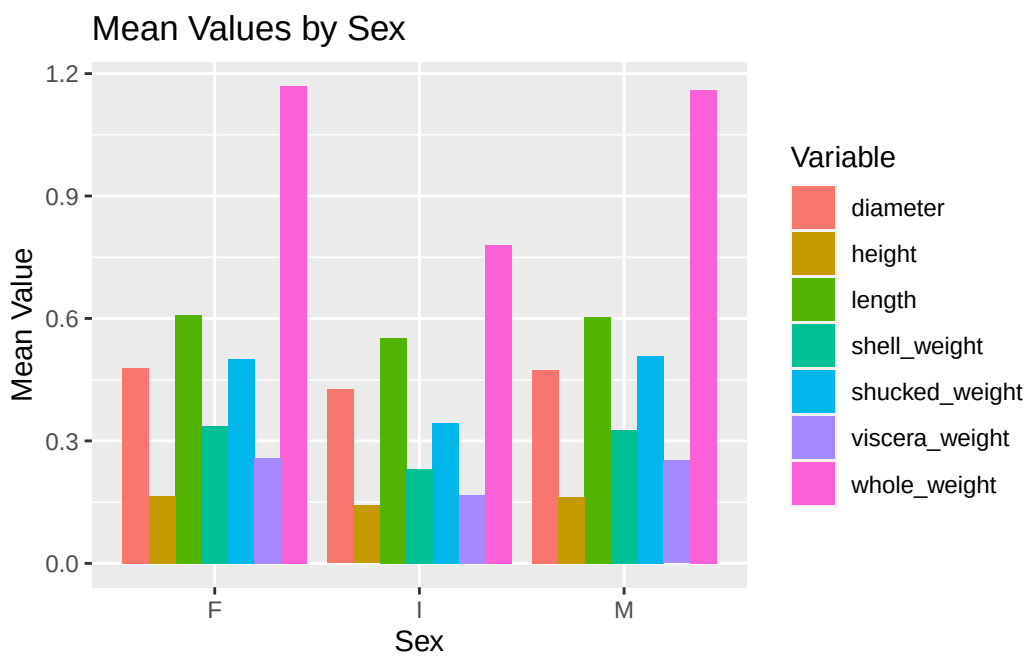
Filter the data to only include abalone with a length of at least 0.5 meters. Group the data by `sex` and calculate the mean of each variable for each group. Create a bar plot to visualize the mean values for each variable by `sex`.

```
df_mean <- df %>%
  filter(length >= 0.5) %>%
  group_by(sex) %>%
  summarize_all(mean)
```

```r
df_mean
```

```
# A tibble: 3 x 9
  sex   length diameter height whole_weight shucked_weight visce~1 shell~2 rings
  <chr>  <dbl>    <dbl>  <dbl>        <dbl>          <dbl>   <dbl>   <dbl> <dbl>
1 F      0.608    0.478  0.165        1.17           0.501   0.258   0.336 11.4
2 I      0.551    0.426  0.142        0.780          0.343   0.167   0.231  9.88
3 M      0.604    0.474  0.163        1.16           0.509   0.252   0.327 11.2
# ... with abbreviated variable names 1: viscera_weight, 2: shell_weight
```

```r
df_mean <- tidyr::pivot_longer(df_mean, cols = starts_with(c("len", "diam", "heig", "w", "

ggplot(df_mean, aes(x = sex, y = value, fill = variable)) +
  geom_col(position = "dodge") +
  labs(x = "Sex", y = "Mean Value", fill = "Variable") +
  ggtitle("Mean Values by Sex")
```



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:

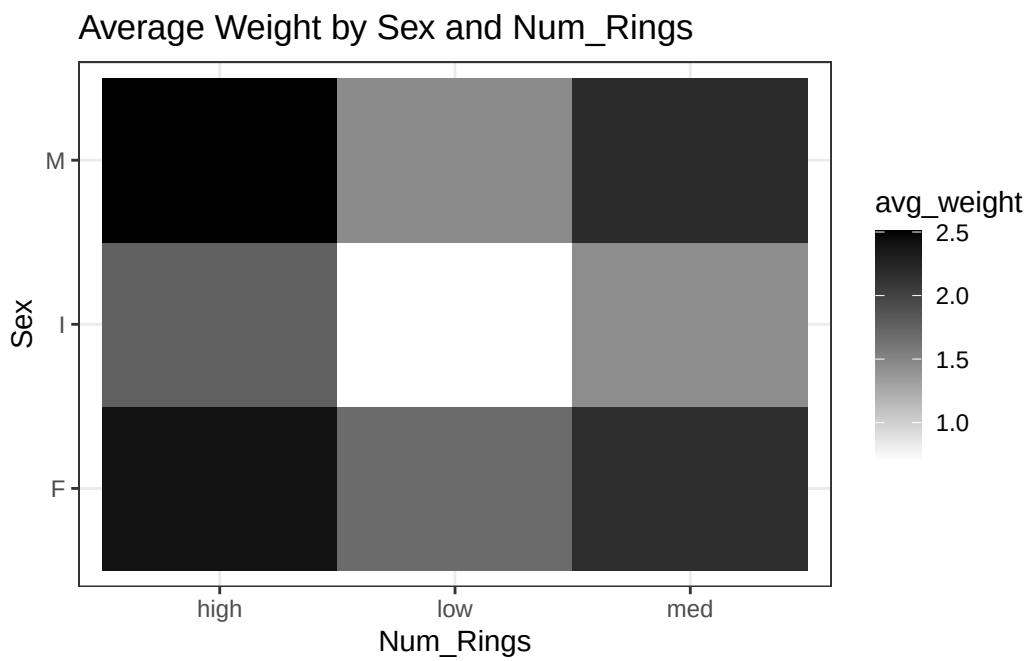
- `"low"` if `rings < 10`
- `"high"` if `rings > 20`, and
- `"med"` otherwise

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.

```
df %>%
  mutate(num_rings = case_when(
    rings < 10 ~ "low",
    rings > 20 ~ "high",
    TRUE ~ "med"
  )) %>%
  group_by(num_rings, sex) %>%
  summarize(avg_weight = mean(whole_weight + shucked_weight + viscera_weight + shell_weigh
  ggplot(aes(x = num_rings, y = sex)) +
  geom_tile(aes(fill = avg_weight)) +
  scale_fill_gradient(low = "white", high = "black") +
  ggtitle("Average Weight by Sex and Num_Rings") +
  xlab("Num_Rings") +
  ylab("Sex") +
  theme_bw()
```

```
`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]

```
df_table <- df
  df_table %>%
  select_if(is.numeric) %>%
  round(2) %>%
  cor() %>%
  as.data.frame() %>%
  tibble::rownames_to_column(var = "variable") %>%
  gather(key = "variable2", value = "correlation", -variable) %>%
  mutate(correlation = round(correlation, 2)) %>%
  arrange(desc(abs(correlation)))
```

```
         variable      variable2 correlation
1          length         length        1.00
```

[3] Table for 2.3

```
2        diameter       diameter        1.00
3          height         height        1.00
4    whole_weight   whole_weight        1.00
5  shucked_weight shucked_weight        1.00
6  viscera_weight viscera_weight        1.00
7    shell_weight   shell_weight        1.00
8           rings          rings        1.00
9        diameter         length        0.99
10         length       diameter        0.99
11 shucked_weight   whole_weight        0.97
12 viscera_weight   whole_weight        0.97
13   whole_weight shucked_weight        0.97
14   whole_weight viscera_weight        0.97
15   shell_weight   whole_weight        0.96
16   whole_weight   shell_weight        0.96
17 viscera_weight shucked_weight        0.93
18 shucked_weight viscera_weight        0.93
19   whole_weight         length        0.92
20   whole_weight       diameter        0.92
21         length   whole_weight        0.92
22       diameter   whole_weight        0.92
23   shell_weight viscera_weight        0.91
24 viscera_weight   shell_weight        0.91
25 shucked_weight         length        0.90
26 viscera_weight         length        0.90
27   shell_weight         length        0.90
28 viscera_weight       diameter        0.90
29   shell_weight       diameter        0.90
30         length shucked_weight        0.90
31         length viscera_weight        0.90
32       diameter viscera_weight        0.90
33         length   shell_weight        0.90
34       diameter   shell_weight        0.90
35 shucked_weight       diameter        0.89
36       diameter shucked_weight        0.89
37   shell_weight shucked_weight        0.88
38 shucked_weight   shell_weight        0.88
39         height       diameter        0.83
40       diameter         height        0.83
41         height         length        0.82
42         length         height        0.82
43   whole_weight         height        0.81
44   shell_weight         height        0.81
```

```
45         height   whole_weight        0.81
46         height   shell_weight        0.81
47 viscera_weight         height        0.79
48         height viscera_weight        0.79
49 shucked_weight         height        0.77
50         height shucked_weight        0.77
51          rings   shell_weight        0.63
52   shell_weight          rings        0.63
53          rings       diameter        0.57
54       diameter          rings        0.57
55          rings         length        0.56
56         length          rings        0.56
57          rings         height        0.55
58         height          rings        0.55
59          rings   whole_weight        0.54
60   whole_weight          rings        0.54
61          rings viscera_weight        0.50
62 viscera_weight          rings        0.50
63          rings shucked_weight        0.42
64 shucked_weight          rings        0.42
```

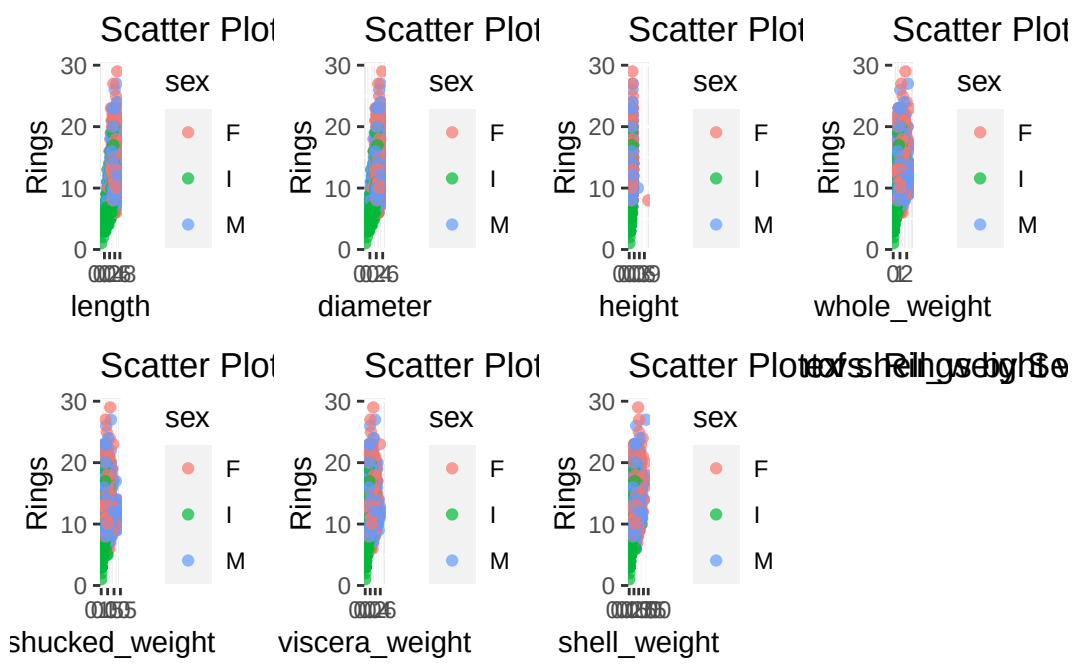
```r
df_table
```

```
# A tibble: 4,177 x 9
   sex   length diameter height whole_weight shucked_wei~1 visce~2 shell~3 rings
   <chr>  <dbl>    <dbl>  <dbl>        <dbl>         <dbl>   <dbl>   <dbl> <int>
 1 M      0.455    0.365  0.095        0.514        0.224   0.101    0.15     15
 2 M      0.35     0.265  0.09         0.226        0.0995  0.0485   0.07      7
 3 F      0.53     0.42   0.135        0.677        0.256   0.142    0.21      9
 4 M      0.44     0.365  0.125        0.516        0.216   0.114    0.155    10
 5 I      0.33     0.255  0.08         0.205        0.0895  0.0395   0.055     7
 6 I      0.425    0.3    0.095        0.352        0.141   0.0775   0.12      8
 7 F      0.53     0.415  0.15         0.778        0.237   0.142    0.33     20
 8 F      0.545    0.425  0.125        0.768        0.294   0.150    0.26     16
 9 M      0.475    0.37   0.125        0.509        0.216   0.112    0.165     9
10 F      0.55     0.44   0.15         0.894        0.314   0.151    0.32     19
# ... with 4,167 more rows, and abbreviated variable names 1: shucked_weight,
#   2: viscera_weight, 3: shell_weight
```

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.

```
quant_vars <- c("length", "diameter", "height", "whole_weight", "shucked_weight", "viscera

scatterplot <- function(data, x, y) {
  ggplot(data, aes(x = !!sym(x), y = rings, color = sex)) +
    geom_point(alpha = 0.7) +
    ggtitle(paste0("Scatter Plot of ", x, " vs. Rings by Sex")) +
    xlab(x) +
    ylab("Rings")
}

scatterplots <- map2(quant_vars, quant_vars, scatterplot, data = df)

plot_grid(plotlist = scatterplots, ncol = 4)
```



—

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.

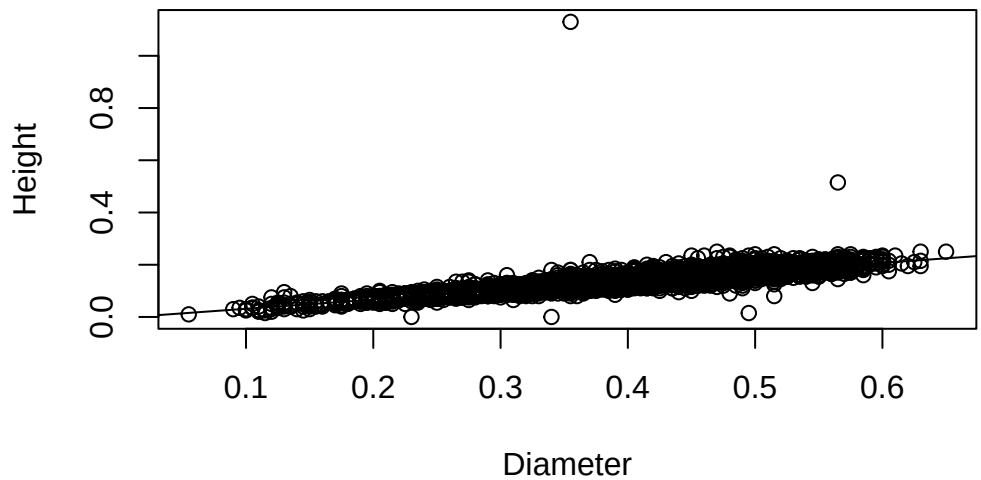
```r
lm_height_diameter <- lm(height ~ diameter,  df)
summary(lm_height_diameter)
```

```
Call:
lm(formula = height ~ diameter, data = df)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.15513 -0.01053 -0.00147  0.00852  1.00906 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.003803   0.001512  -2.515   0.0119 *  
diameter     0.351376   0.003602  97.544   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0231 on 4175 degrees of freedom
Multiple R-squared:  0.695, Adjusted R-squared:  0.695 
F-statistic:  9515 on 1 and 4175 DF,  p-value: < 2.2e-16
```

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
plot(df$diameter, df$height, xlab = "Diameter", ylab = "Height")
abline(lm_height_diameter)
```



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"`

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

```r
)

new_abalones <- data.frame(diameter = new_diameters)

predict(lm_height_diameter, newdata = new_abalones)
```

```
           1            2            3            4            5            6
0.0496723682 0.1661276096 0.2003304536 0.0229141546 0.1727090665 0.2894625947
           7            8            9           10
0.3324002348 0.3226493217 0.3273527111 0.0004465615
```

—

Appendix

Session Information

Print your R session information using the following command

```r
sessionInfo()
```

```
R version 4.2.2 (2022-10-31)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Ventura 13.2

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

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
```