title: “Homework 2” author: “Arjun Laxman” toc: true title-block-banner: true title-block-style: default format: pdf # format: pdf

Link to the Github repository

> **Due: Feb 9, 2024 @ 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:

```
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
```

```
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.

```
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"
)
#reads the dataset
abalone <- read_csv(url, col_names = abalone_col_names) # Insert your code here
```

```
Rows: 4177 Columns: 9
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (1): sex
dbl (8): length, diameter, height, whole_weight, shucked_weight, viscera_wei...

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(abalone,10)
```

```
# A tibble: 10 × 9
   sex   length diameter height whole_weight shucked_weight viscera_weight
   <chr>  <dbl>    <dbl>  <dbl>        <dbl>          <dbl>          <dbl>
 1 M      0.455    0.365  0.095        0.514         0.224          0.101
 2 M      0.35     0.265  0.09         0.226         0.0995         0.0485
 3 F      0.53     0.42   0.135        0.677         0.256          0.142
 4 M      0.44     0.365  0.125        0.516         0.216          0.114
 5 I      0.33     0.255  0.08         0.205         0.0895         0.0395
 6 I      0.425    0.3    0.095        0.352         0.141          0.0775
 7 F      0.53     0.415  0.15         0.778         0.237          0.142
 8 F      0.545    0.425  0.125        0.768         0.294          0.150
 9 M      0.475    0.37   0.125        0.509         0.216          0.112
10 F      0.55     0.44   0.15         0.894         0.314          0.151
# ℹ 2 more variables: shell_weight <dbl>, rings <dbl>
```

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?

```
library(dplyr)


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

```
[1] 0
```

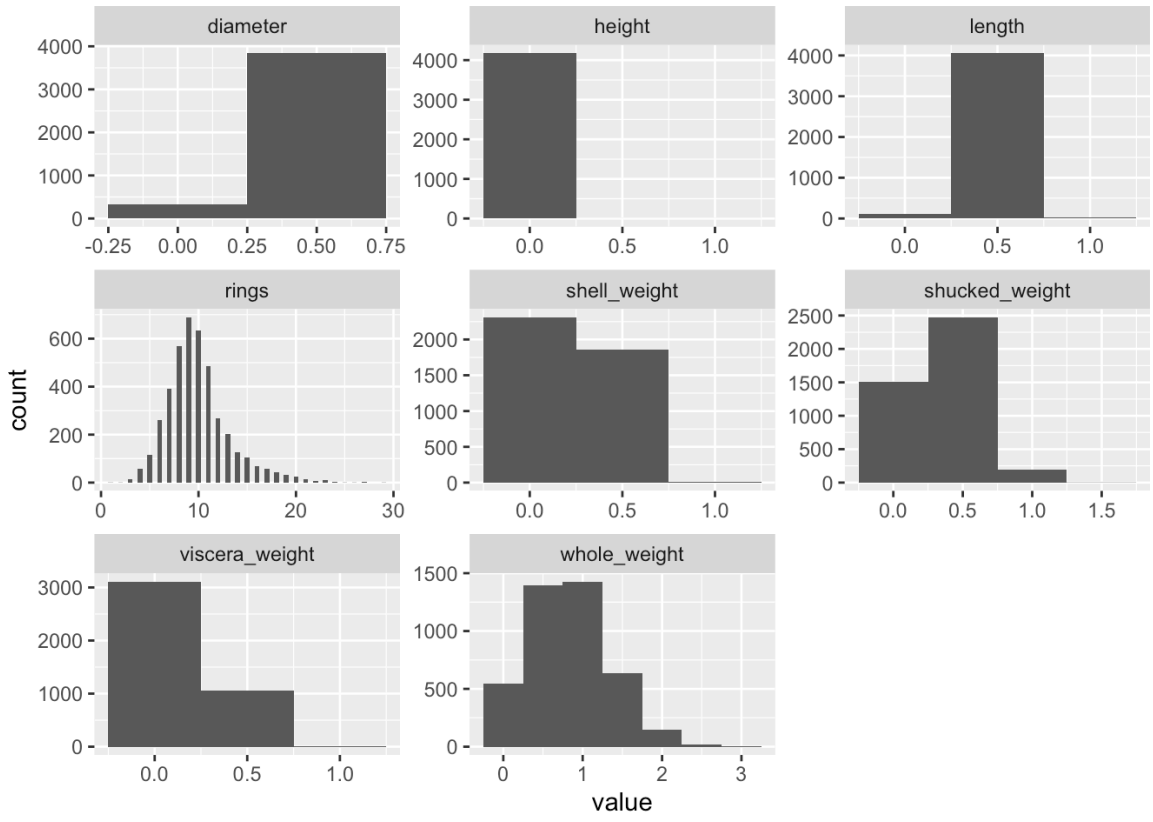
---

##### 1.3 (5 points)

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

::: {.cell}

```{.r .cell-code}
df_long <- df %>%
  pivot_longer(cols = -sex, names_to = "variable", values_to = "value")

# Plot histograms for each quantitative variable

ggplot(df_long, aes(x = value)) +
  geom_histogram(binwidth = 0.5) +
  # Set bin width for histograms
  facet_wrap(~variable, scales = "free")
```




```
# Separate panel for each variable
```

:::

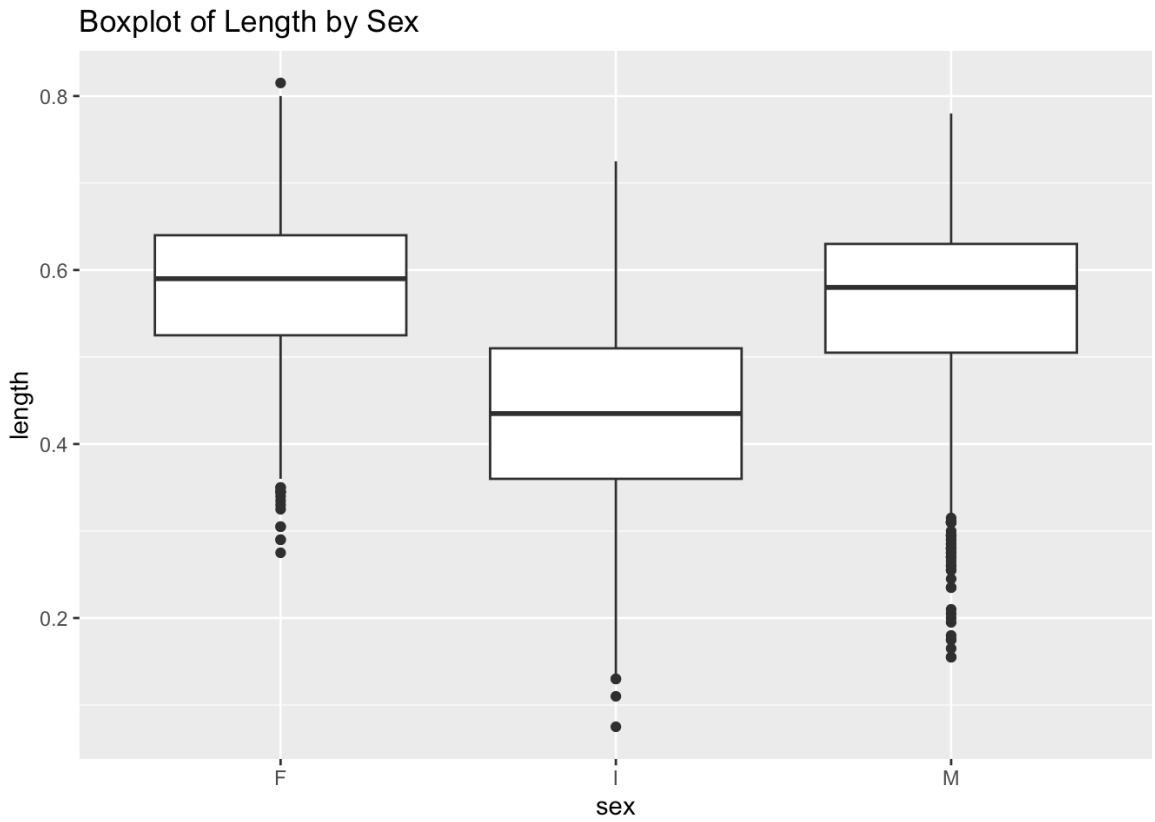
---

##### 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?

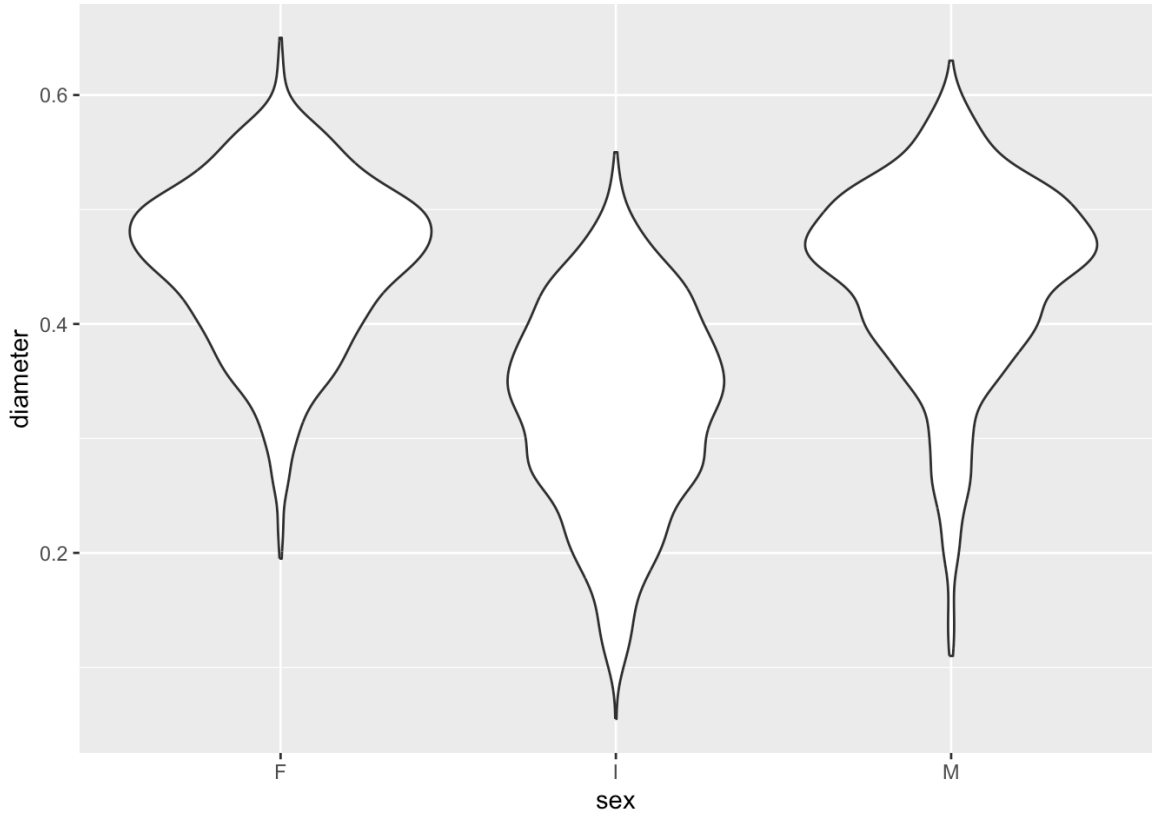
::: {.cell}

```{.r .cell-code}
ggplot(df, aes(x = sex, y = length)) +
  geom_boxplot() +
  ggtitle("Boxplot of Length by Sex")
```




:::

```r
# Violin plot for diameter by sex

violinplot_diameter <- ggplot(df, aes(x = sex, y = diameter)) +
  geom_violin()
violinplot_diameter
```

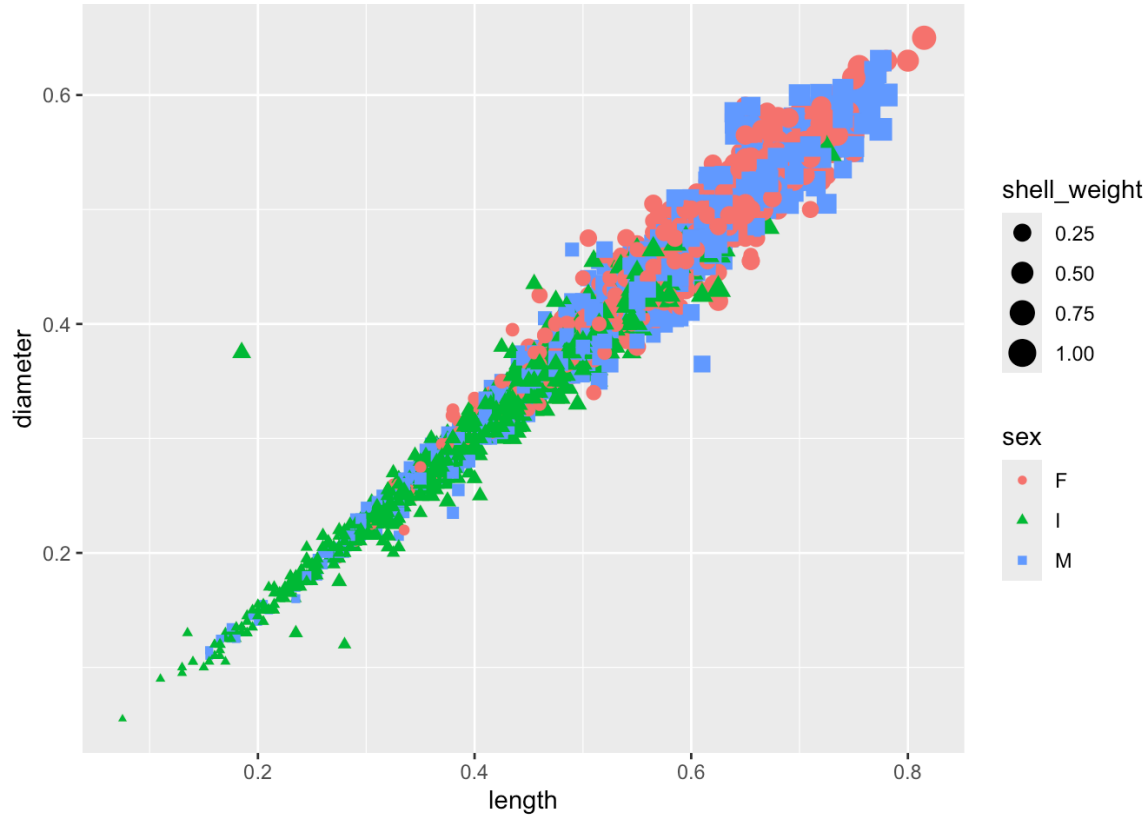



```
# Display the violin plot
```

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?

```
scatter_plot <- ggplot(df, aes(x = length, y = diameter, shape = sex, color = sex, size = shell_weight))
  geom_point() +  # Add points to the plot
  scale_size_continuous(range = c(1, 5))  # Adjust the size scale for points to enhance visibility

# Display the scatter plot
scatter_plot
```

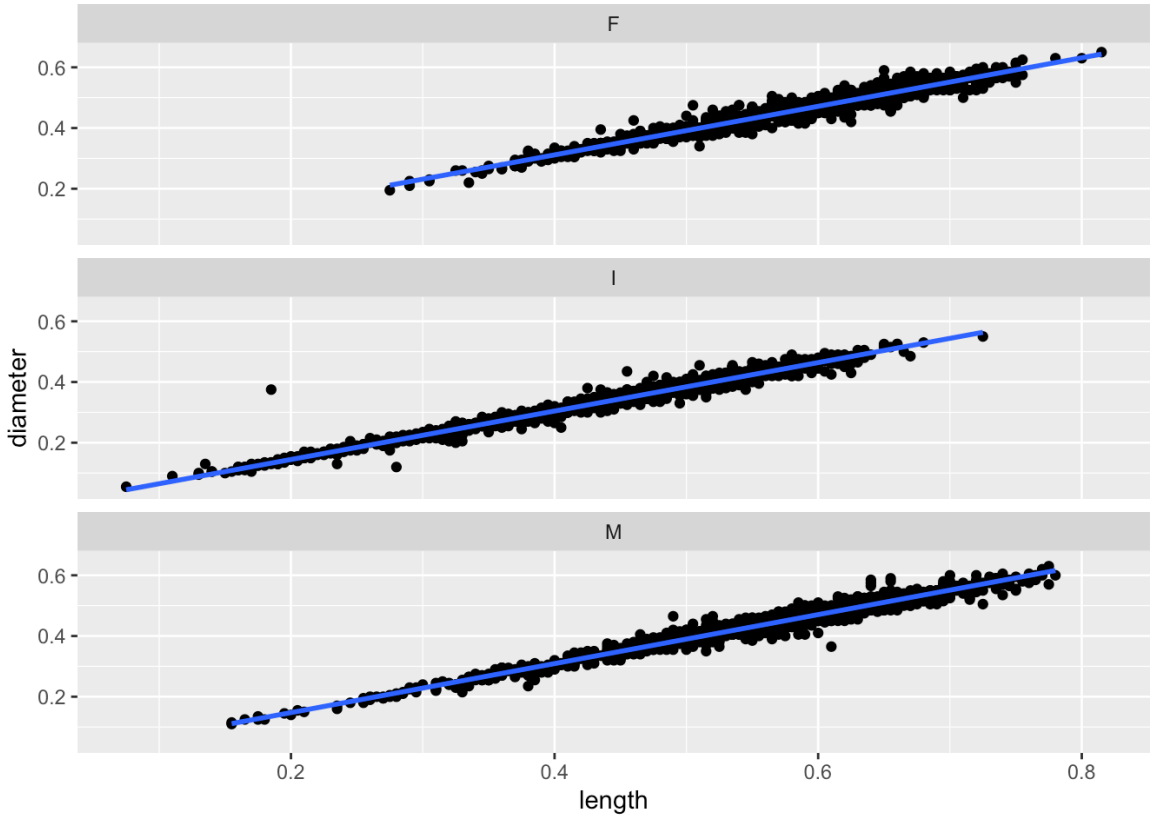



1.6 (5 points)

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: [1]

```
#compare patterns across groups by 'sex;'
length_diameter_plots <- ggplot(df, aes(x = length, y = diameter)) +
  geom_point() +  # Plot points for each observation
  geom_smooth(method = "lm", se = FALSE) +  # Add linear regression line without confidence envelope
  facet_wrap(~sex, ncol = 1)  # Organize plots into separate panels vertically

# Display the combined scatter plots
length_diameter_plots
```

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




## Question 2

**40 points**

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

2.1 (10 points)

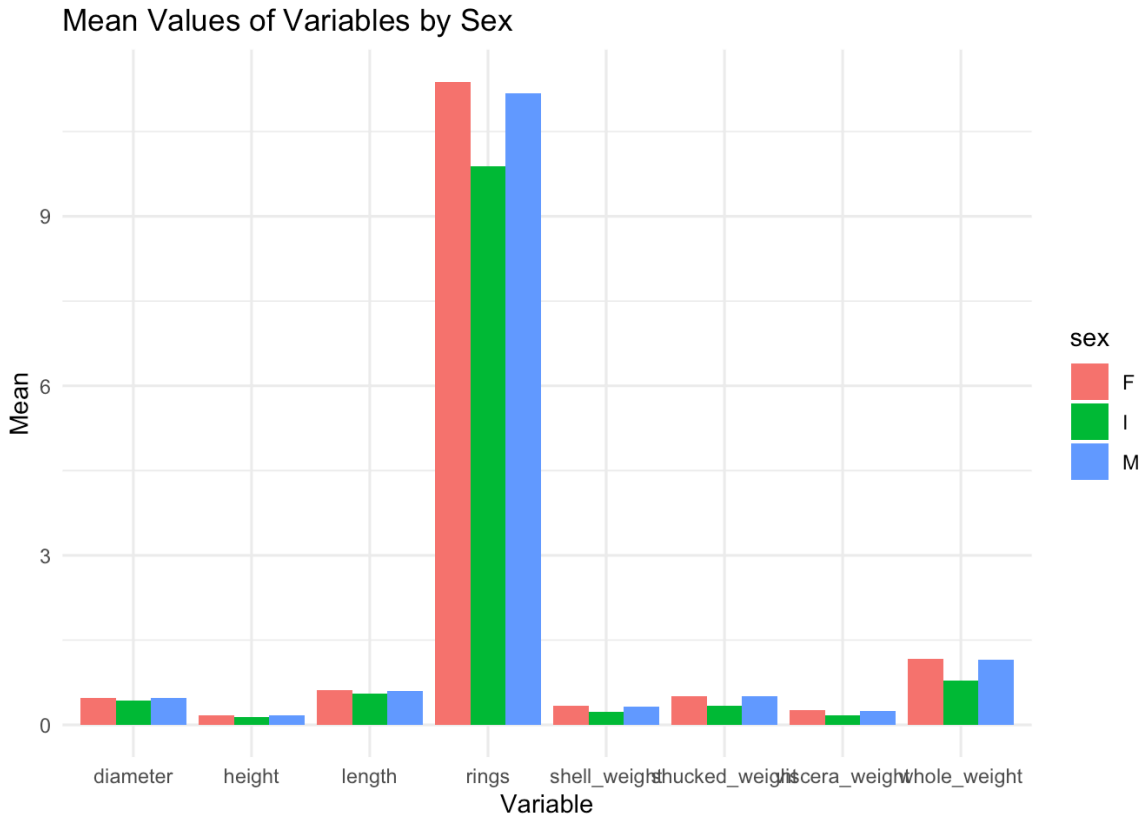
Filter the data to only include abalone with a length of at least $0.5$ meters. Group the data by `sex` and calculate the mean of each variable for each group. Create a bar plot to visualize the mean values for each variable by `sex`.

```
df %>% ... # Insert your code here
```

```
# Filter the data, group by 'sex', calculate means, and create a bar plot
df %>%
  filter(length >= 0.5) %>%  # Filter abalone with length at least 0.5 meters
  group_by(sex) %>%          # Group by sex
  summarise(across(.cols = where(is.numeric), mean, na.rm = TRUE)) %>%  # Calculate means
  pivot_longer(-sex, names_to = "variable", values_to = "mean_value") %>%  # Reshape for plotting
  ggplot(aes(x = variable, y = mean_value, fill = sex)) +  # Prepare plot
    geom_bar(stat = "identity", position = "dodge") +  # Use bars to show mean values
    theme_minimal() +  # Minimal theme
    labs(title = "Mean Values of Variables by Sex", x = "Variable", y = "Mean")
```

```
Warning: There was 1 warning in `summarise()`.
ℹ In argument: `across(.cols = where(is.numeric), mean, na.rm = TRUE)`.
ℹ In group 1: `sex = "F"`.
Caused by warning:
! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
```

```
Supply arguments directly to `.fns` through an anonymous function instead.

  # Previously
  across(a:b, mean, na.rm = TRUE)

  # Now
  across(a:b, \(x) mean(x, na.rm = TRUE))
```




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

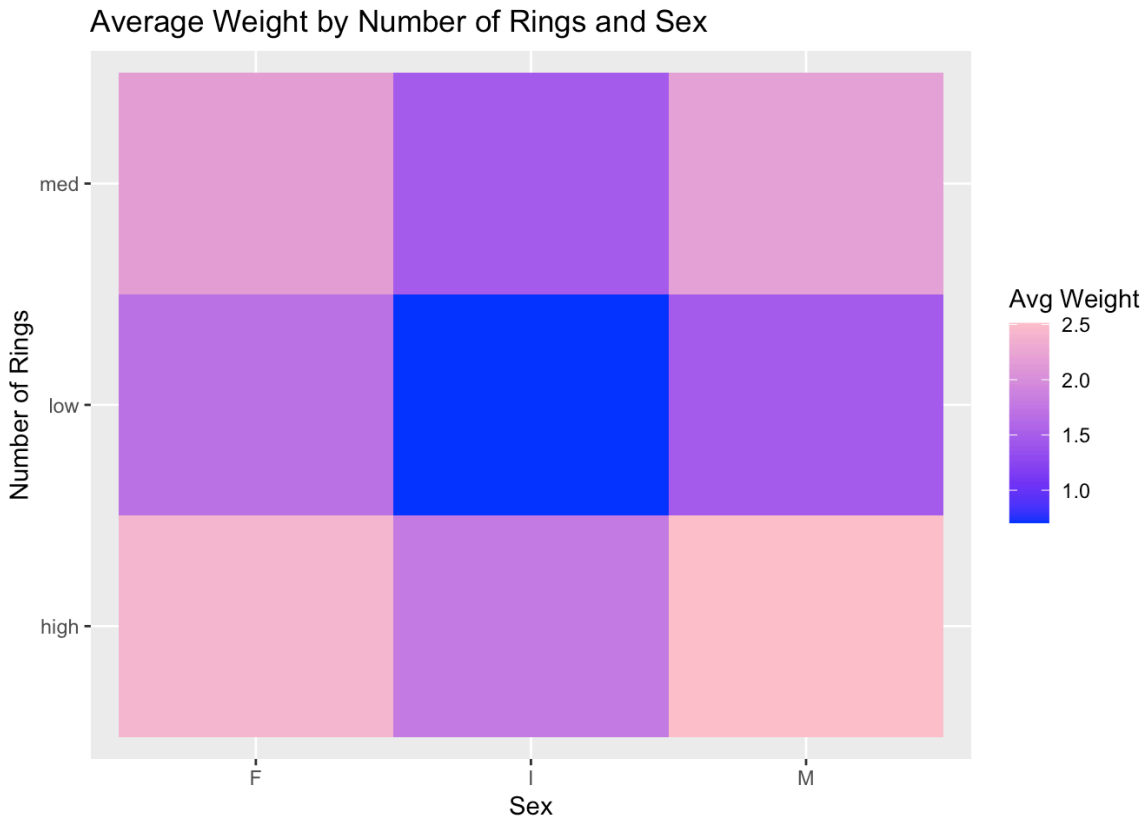
2. Group `df` by this new variable and `sex` and compute `avg_weight` as the average of the `whole_weight + shucked_weight + viscera_weight + shell_weight` for each combination of `num_rings` and `sex`.

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) %>%
  summarise(avg_weight = mean(whole_weight + shucked_weight + viscera_weight + shell_weight, na.rm = TRU
  ggplot(aes(x = sex, y = num_rings, fill = avg_weight)) +
```

```
  geom_tile() +
  scale_fill_gradient(low = "blue", high = "pink") +
  labs(title = "Average Weight by Number of Rings and Sex", x = "Sex", y = "Number of Rings", fill = "Av
```

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



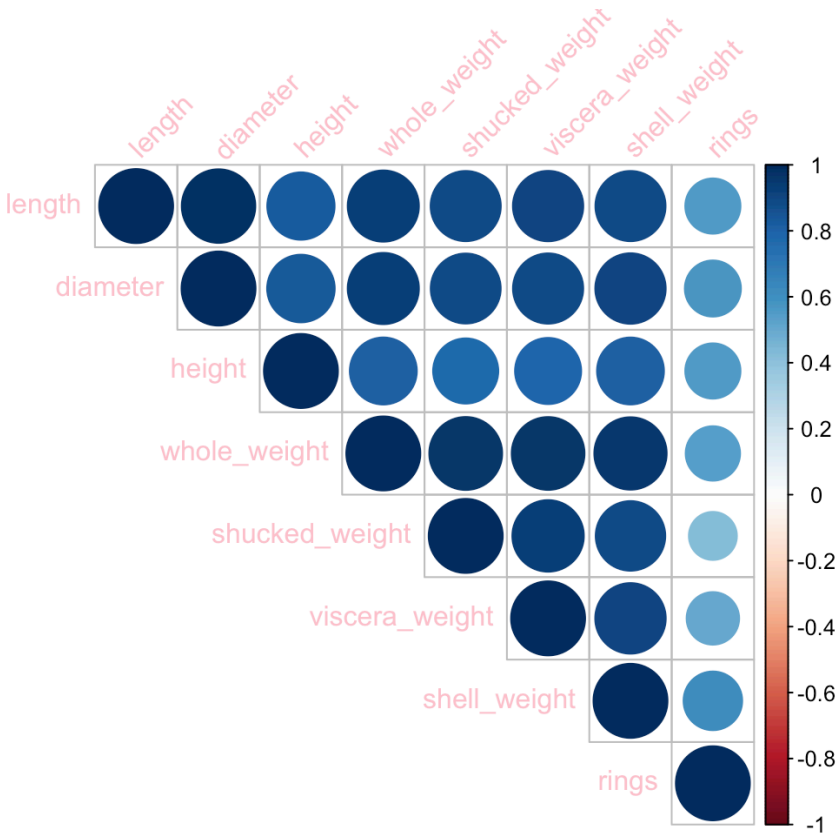

### 2.3 (5 points)

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

```
library(corrplot)
```

```
corrplot 0.92 loaded
```

```
numeric_data <- df %>% select(where(is.numeric))
cor_matrix <- cor(numeric_data)

corrplot(cor_matrix, method = "circle", type = "upper", tl.col = "pink", tl.srt = 45)
```



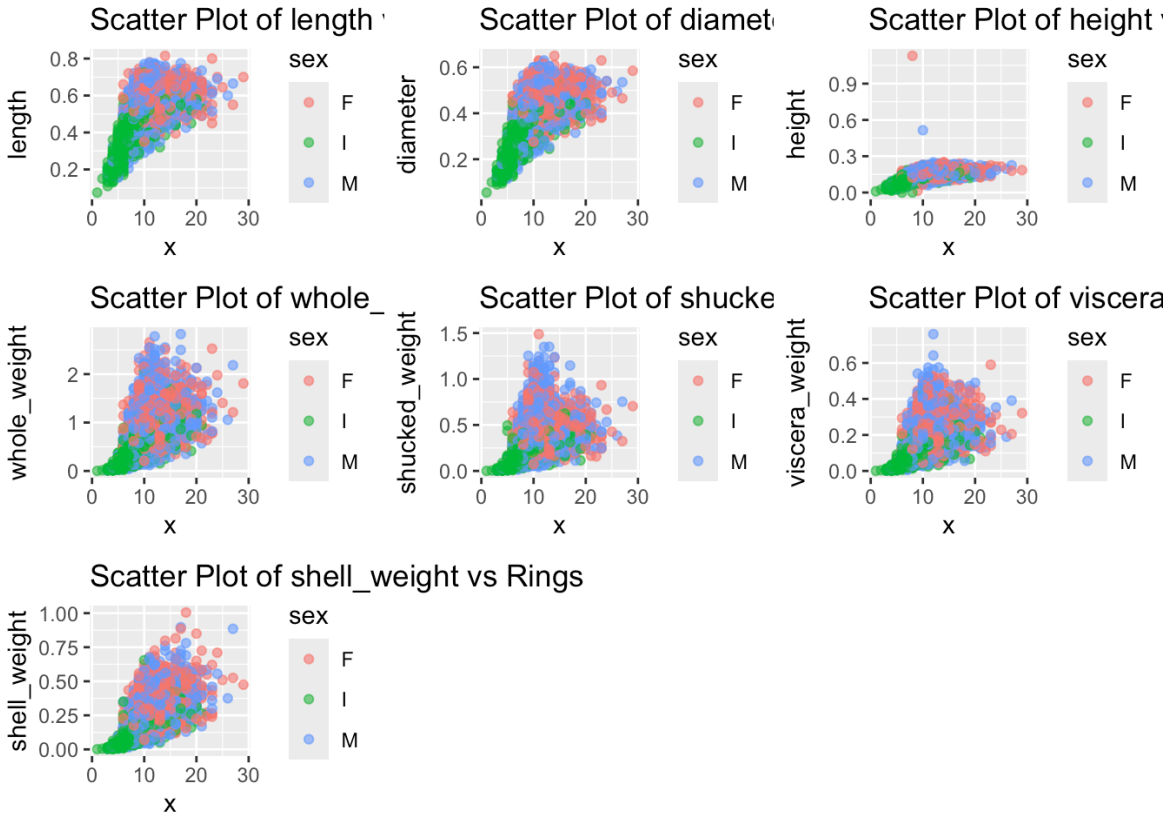

2.4 (10 points)

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.

```
library(purrr)
library(cowplot)

# Filter out non-numeric columns except 'rings' and 'sex'
numeric_vars <- select(df, where(is.numeric), -rings)

# Generate scatter plots
plot_list <- map2(
  names(numeric_vars),
  rep(list(df$rings), length(numeric_vars)),
  ~ {
    ggplot(df, aes_string(x = .y, y = .x, color = "sex")) +
      geom_point(alpha = 0.6) +
      ggtitle(paste("Scatter Plot of", .x, "vs Rings"))
  }
)
```

```
Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
ℹ Please use tidy evaluation idioms with `aes()`.
ℹ See also `vignette("ggplot2-in-packages")` for more information.
```

```
# Combine plots into a grid
plot_grid(plotlist = plot_list, align = "v")
```







## 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.

```
library(dplyr)

# Simple lin. reg. with D as the covariate and H as the response
model <- lm(height ~ diameter, data = df)

# Output the summary of the reg model
summary(model)
```

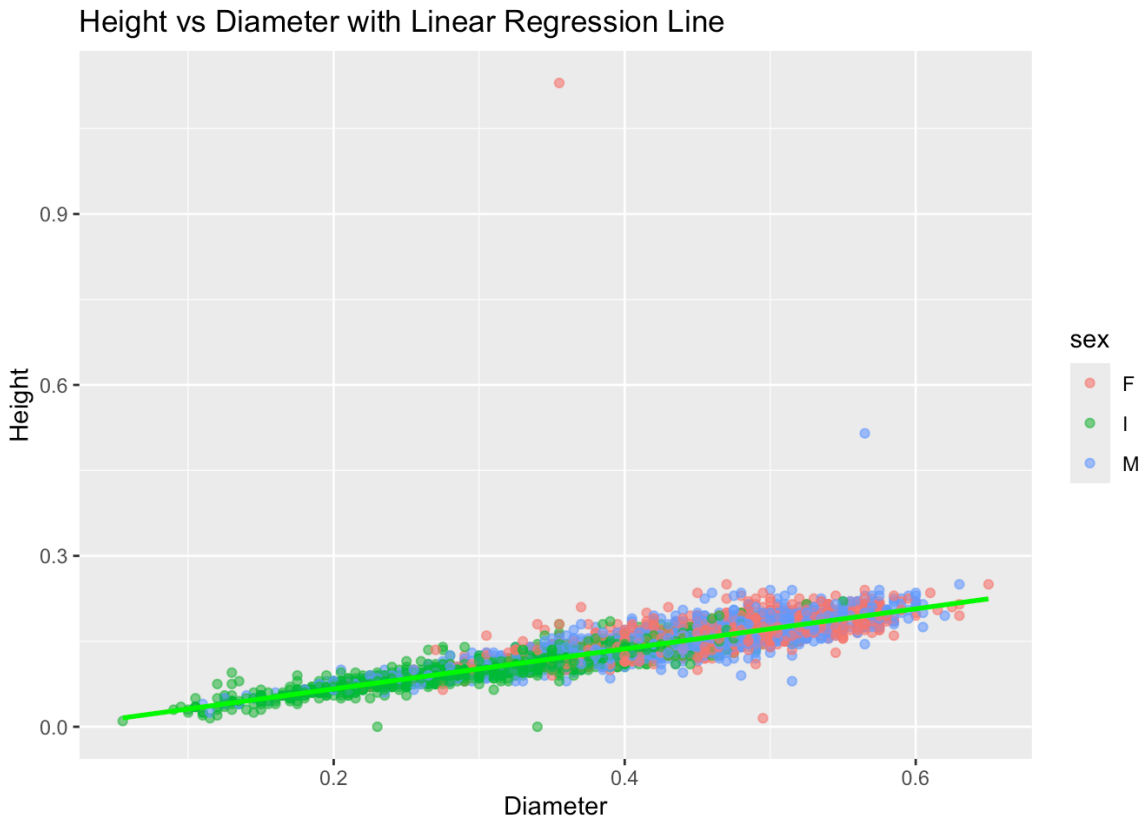
```
Call:
lm(formula = height ~ diameter, data = df)

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.

```
library(ggplot2)

# Scatter plot with regression line
ggplot(df, aes(x = diameter, y = height)) +
  geom_point(aes(color = sex), alpha = 0.6) +  # Color points by sex
  geom_smooth(method = "lm", se = FALSE, color = "green") +  # Add linear regression line
  labs(title = "Height vs Diameter with Linear Regression Line",
       x = "Diameter", y = "Height")
```

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



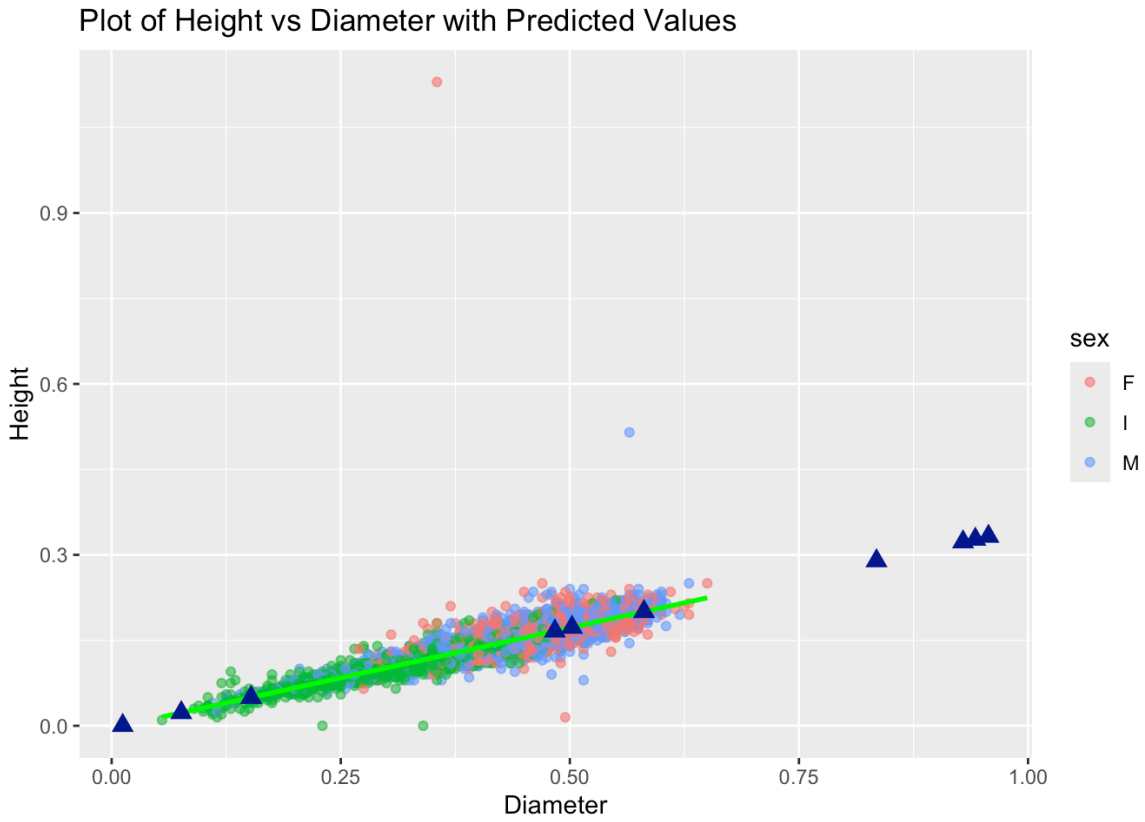

3.3 (10 points)

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
)


# Create a new data frame for prediction
new_data <- data.frame(diameter = new_diameters)

# Use the model to predict heights for these new diameters
predicted_heights <- predict(model, newdata = new_data)

# Add these predictions back to the new data frame for plotting
new_data$height <- predicted_heights

# Original scatter plot of height vs diameter with the regression line
ggplot(df, aes(x = diameter, y = height)) +
  geom_point(aes(color = sex), alpha = 0.6) +
```

```
  geom_smooth(method = "lm", se = FALSE, color = "green") +
  labs(title = "Height vs Diameter with Linear Regression Line", x = "Diameter", y = "Height") +
  # Add new predicted data points in violet for visibility
  geom_point(data = new_data, aes(x = diameter, y = height), color = "darkblue", size = 3, shape = 17) +
  ggtitle("Plot of Height vs Diameter with Predicted Values")
```

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




# Appendix

### Session Information

Print your `R` session information using the following command

```
sessionInfo()
```

```
R version 4.3.3 (2024-02-29)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Sonoma 14.4.1

Matrix products: default
```

```
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK version
3.11.0

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

time zone: America/New_York
tzcode source: internal

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

other attached packages:
[1] corrplot_0.92 cowplot_1.1.3 purrr_1.0.2   dplyr_1.1.4   ggplot2_3.5.1
[6] tidyr_1.3.1   readr_2.1.5

loaded via a namespace (and not attached):
 [1] Matrix_1.6-5      bit_4.0.5         gtable_0.3.5      jsonlite_1.8.8
 [5] crayon_1.5.2      compiler_4.3.3    tidyselect_1.2.1  parallel_4.3.3
 [9] splines_4.3.3     scales_1.3.0      fastmap_1.1.1     lattice_0.22-5
[13] R6_2.5.1          labeling_0.4.3    generics_0.1.3    curl_5.2.1
[17] knitr_1.45        tibble_3.2.1      munsell_0.5.1     pillar_1.9.0
[21] tzdb_0.4.0        rlang_1.1.3       utf8_1.2.4        xfun_0.43
[25] bit64_4.0.5       cli_3.6.2         mgcv_1.9-1        withr_3.0.0
[29] magrittr_2.0.3    digest_0.6.35     grid_4.3.3        vroom_1.6.5
[33] rstudioapi_0.16.0 hms_1.1.3         nlme_3.1-164      lifecycle_1.0.4
[37] vctrs_0.6.5       evaluate_0.23     glue_1.7.0        farver_2.1.1
[41] fansi_1.0.6       colorspace_2.1-0  rmarkdown_2.26    tools_4.3.3
[45] pkgconfig_2.0.3   htmltools_0.5.8.1
```

| length | diameter | height | whole_weight | shucked_weight | viscera_weight | shell_weight | rings | |
|---|---|---|---|---|---|---|---|---|
| length | 1.00 | 0.99 | 0.83 | 0.93 | 0.90 | 0.90 | 0.90 | 0.56 |
| diameter | 0.99 | 1.00 | 0.83 | 0.93 | 0.89 | 0.90 | 0.91 | 0.57 |
| height | 0.83 | 0.83 | 1.00 | 0.82 | 0.77 | 0.80 | 0.82 | 0.56 |
| whole_weight | 0.93 | 0.93 | 0.82 | 1.00 | 0.97 | 0.97 | 0.96 | 0.54 |
| shucked_weight | 0.90 | 0.89 | 0.77 | 0.97 | 1.00 | 0.93 | 0.88 | 0.42 |
| viscera_weight | 0.90 | 0.90 | 0.80 | 0.97 | 0.93 | 1.00 | 0.91 | 0.50 |
| shell_weight | 0.90 | 0.91 | 0.82 | 0.96 | 0.88 | 0.91 | 1.00 | 0.63 |
| rings | 0.56 | 0.57 | 0.56 | 0.54 | 0.42 | 0.50 | 0.63 | 1.00 |

### Footnotes

1. Plot example for 1.6 ↩
2. Table for 2.3 ↩