Final Project 3

Annie Glenning

2024-08-19

```
setwd("/Users/annieglenning/Documents/Dartmouth/SU24/QBS_103/Data") # set my working directory to where
original_gene_data <- read.table("QBS103_GSE157103_genes.csv", header = TRUE, sep = ",") # reading in t
series_data <- read.table("QBS103_GSE157103_series_matrix.csv", header = TRUE, sep = ",") # reading in
```

```
# transposing the gene data
original_gene_data <- as.data.frame(original_gene_data)
if (is.character(original_gene_data[1, 1])) {
  gene_data <- t(original_gene_data)
  colnames(gene_data) <- gene_data[1, ]
  gene_data <- gene_data[-1, ]
  gene_data <- as.data.frame(gene_data)
}
```

Identify one gene, one continuous covariate, and two categorical covariates

```
AAGAB <- as.numeric(gene_data$AAGAB) # one gene
```

Info for Table of Summary Statistics

```
# Install table1 package if not already installed
#install.packages("table1")

# Load required packages
library(table1)
```

```
##
## Attaching package: 'table1'

## The following objects are masked from 'package:base':
##
##     units, units<-
```

```
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
```

```
# Example dataset preparation (you should replace this with your actual data)
# Assuming `data` is your data frame and contains the relevant variables

# Convert categorical variables to factors
#table_data <- series_data %>%
#  mutate(sex = factor(sex),
#         disease_status = factor(disease_status),
#         icu_status = factor(icu_status))  # Add another categorical variable here

# Example of selecting 3 continuous variables and 3 categorical variables
#continuous_vars <- c("age", "ferritin.ng.ml.", "procalcitonin.ng.ml..")
#categorical_vars <- c("sex", "disease_status", "icu_status")
```

```
# Define a custom label for your variables if desired
#labels <- list(
#  age = "Age (years)",
#  ferritin.ng.ml. = "Ferritin (ng/ml)",
#  procalcitonin.ng.ml.. = "Procalcitonin (ng/ml)",
#  sex = "Sex",
#  disease_status = "Disease Status",
#  icu_status = "ICU Status"
#)

# Create the table stratified by a categorical variable, e.g., `disease_status`
#table <- table1(~ age + ferritin.ng.ml. + procalcitonin.ng.ml.. + sex + icu_status | disease_status,
#                data = table_data,
#                render.categorical = function(x) #sprintf("%d (%0.1f%%)", sum(x), mean(x) * 100),
#                render.continuous = function(x) {
#                    if (length(x) <= 5) {
#                        sprintf("%0.1f [%0.1f, %0.1f]", median(x), quantile(x, 0.25), quantile(x, 0.75
#                    } else {
#                        sprintf("%0.1f (%.1f)", mean(x), sd(x))
#                    }
#                },
#                overall = FALSE,  # Set to TRUE if you want overall summary stats
#                topclass = "Rtable1-zebra")
```

sex

```
# percent
#length(series_data$age[series_data$sex == ' male'])/length(series_data$age)*100
# percent
#length(series_data$age[series_data$sex == ' female'])/length(series_data$age)*100
#length(series_data$age[series_data$sex == ' unknown'])/length(series_data$age)*100
```

```
#mean(as.numeric(series_data$age[series_data$sex == ' male']), na.rm = TRUE) # note getting rid of >89
#sd(as.numeric(series_data$age[series_data$sex == ' male']), na.rm = TRUE)
#mean(as.numeric(series_data$age[series_data$sex == ' female']), na.rm = TRUE)
#sd(as.numeric(series_data$age[series_data$sex == ' female']), na.rm = TRUE)
#mean(as.numeric(series_data$age[series_data$sex == ' unknown']), na.rm = TRUE)
#sd(as.numeric(series_data$age[series_data$sex == ' unknown']), na.rm = TRUE)
```

```
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$sex == ' male']), na.rm = TRUE) # 11 unknown
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$sex == ' male']), na.rm = TRUE)
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$sex == ' female']), na.rm = TRUE) # 4 unknown
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$sex == ' female']), na.rm = TRUE)
#series_data$ferritin.ng.ml.[series_data$sex == ' unknown'] # 1 unknown
```

```
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$sex == ' male']), na.rm = TRUE) # 14 unk
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$sex == ' male']), na.rm = TRUE)
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$sex == ' female']), na.rm = TRUE) # 9 un
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$sex == ' female']), na.rm = TRUE)
#series_data$procalcitonin.ng.ml..[series_data$sex == ' unknown'] # 1 unknown
```

icu_status

```
#length(series_data$age[series_data$icu_status == ' yes'])/length(series_data$age) *100
#length(series_data$age[series_data$icu_status == ' no'])/length(series_data$age) *100
```

```
#mean(as.numeric(series_data$age[series_data$icu_status == ' yes']), na.rm = TRUE)
#sd(as.numeric(series_data$age[series_data$icu_status == ' yes']), na.rm = TRUE)
#mean(as.numeric(series_data$age[series_data$icu_status == ' no'] ), na.rm = TRUE) # gets rid of " :" a
#sd(as.numeric(series_data$age[series_data$icu_status == ' no']), na.rm = TRUE)
```

```
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$icu_status == ' yes']), na.rm = TRUE)  # 7 unk
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$icu_status == ' yes']), na.rm = TRUE)
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$icu_status == ' no']), na.rm = TRUE)  # 9 unkn
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$icu_status == ' no']), na.rm = TRUE)
```

```
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$icu_status == ' yes']), na.rm = TRUE)  #
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$icu_status == ' yes']), na.rm = TRUE)
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$icu_status == ' no']), na.rm = TRUE)  #
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$icu_status == ' no']), na.rm = TRUE)
```

disease_status

```
#length(series_data$age[series_data$disease_status == "disease state: COVID-19"])/length(series_data$ag
#length(series_data$age[series_data$disease_status == "disease state: non-COVID-19"])/length(series_dat
```

```
#mean(as.numeric(series_data$age[series_data$disease_status == "disease state: COVID-19"]), na.rm = TRU
#sd(as.numeric(series_data$age[series_data$disease_status == "disease state: COVID-19"]), na.rm = TRUE)
#mean(as.numeric(series_data$age[series_data$disease_status == "disease state: non-COVID-19"]), na.rm =
#sd(as.numeric(series_data$age[series_data$disease_status == "disease state: non-COVID-19"]), na.rm = T
```

```
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: COVID-19"]),
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: COVID-19"]), n
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: non-COVID-19
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: non-COVID-19"]
```

```
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$disease_status == "disease state: COVID-
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$disease_status == "disease state: COVID-19
#mean(as.numeric(series_data$procalcitonin.ng.ml..[series_data$disease_status == "disease state: non-CO
#sd(as.numeric(series_data$procalcitonin.ng.ml..[series_data$disease_status == "disease state: non-COVI
```

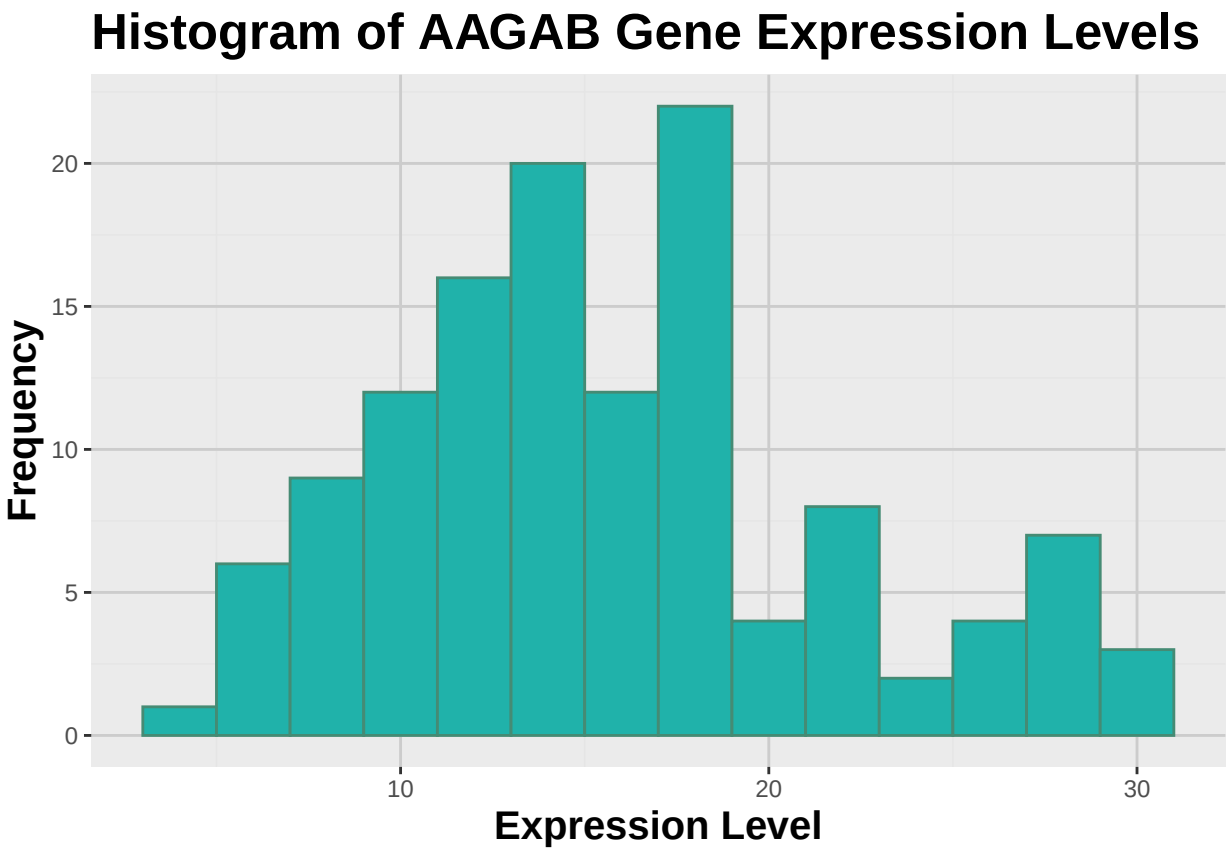
Finalizing Histogram

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v ggplot2   3.5.1     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## -- 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 errors
```

```
library(ggplot2)
```

```
ggplot(gene_data, aes(x = as.numeric(AAGAB))) +
  geom_histogram(binwidth = 2, fill = "lightseagreen", color = "aquamarine4") +
  labs( # labeling the title and axis
    title = "Histogram of AAGAB Gene Expression Levels",
    x = "Expression Level",
    y = "Frequency"
  ) +
  theme(
    plot.title = element_text(size = 19, face = "bold"), # title
    axis.title = element_text(size = 15, face = "bold"), # axis
    panel.grid.major = element_line(color = "grey80"), # background
    panel.grid.minor = element_line(color = "grey90")
    )
```

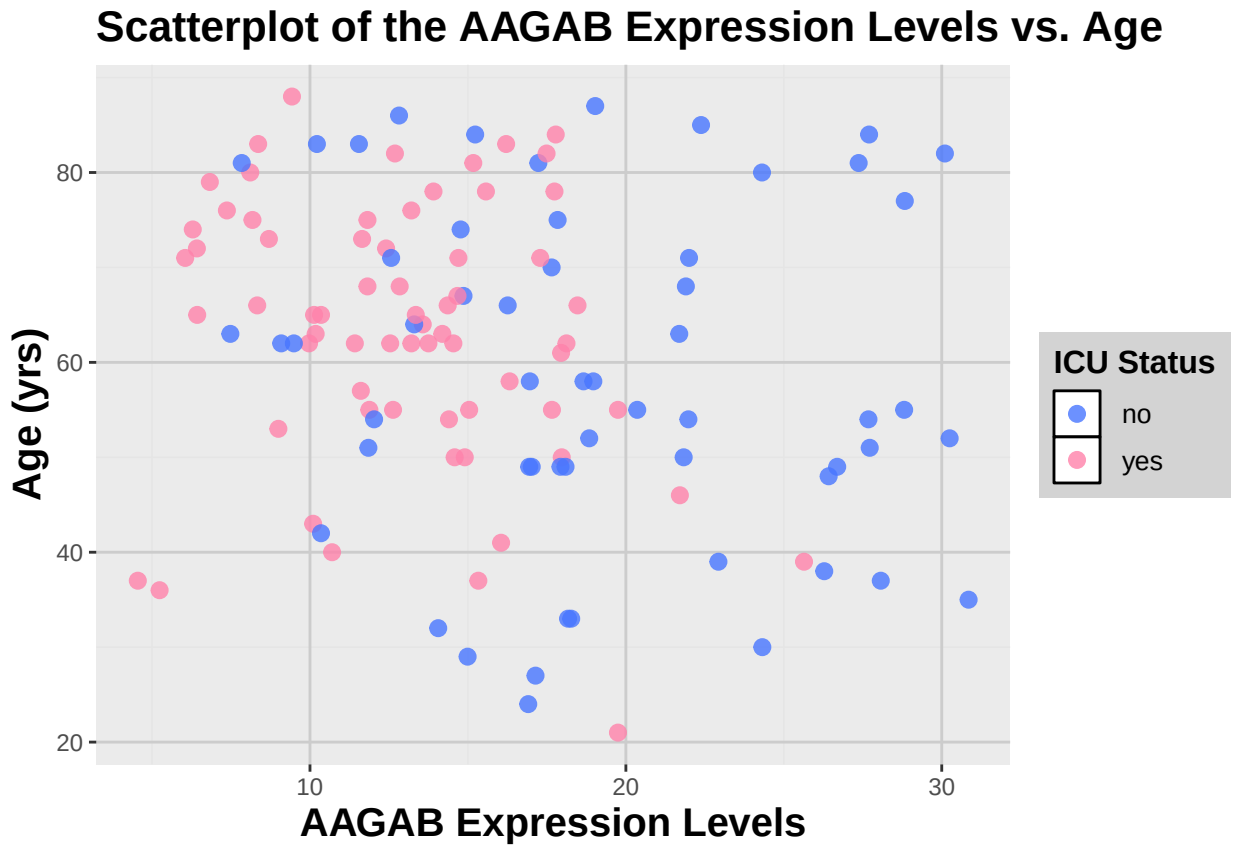


Finalizing Scatterplot

```
ggplot(series_data, aes(x = as.numeric(AAGAB), y = as.numeric(age),
                        color = factor(icu_status))) + # seperarting the points by icu status
  geom_point(size = 2.5, alpha = 0.8) + # size and transparency of the point
  labs( # labeling the title and axis
    title = "Scatterplot of the AAGAB Expression Levels vs. Age",
    x = "AAGAB Expression Levels",
    y = "Age (yrs)",
    color = "ICU Status" # legend title
    ) +
  scale_color_manual(values = c(" yes" = "palevioletred1", " no" = "royalblue1")) +  # setting the colo
  theme(
    plot.title = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 15, face = "bold"),
    legend.title = element_text(size = 12, face = "bold"),  # editing the legend
    legend.text = element_text(size = 10),
    legend.background = element_rect(fill = "lightgray", color = NA),
    legend.key = element_rect(fill = "white", color = "black"),
    panel.grid.major = element_line(color = "grey80"), # background
    panel.grid.minor = element_line(color = "grey90")
    )
```

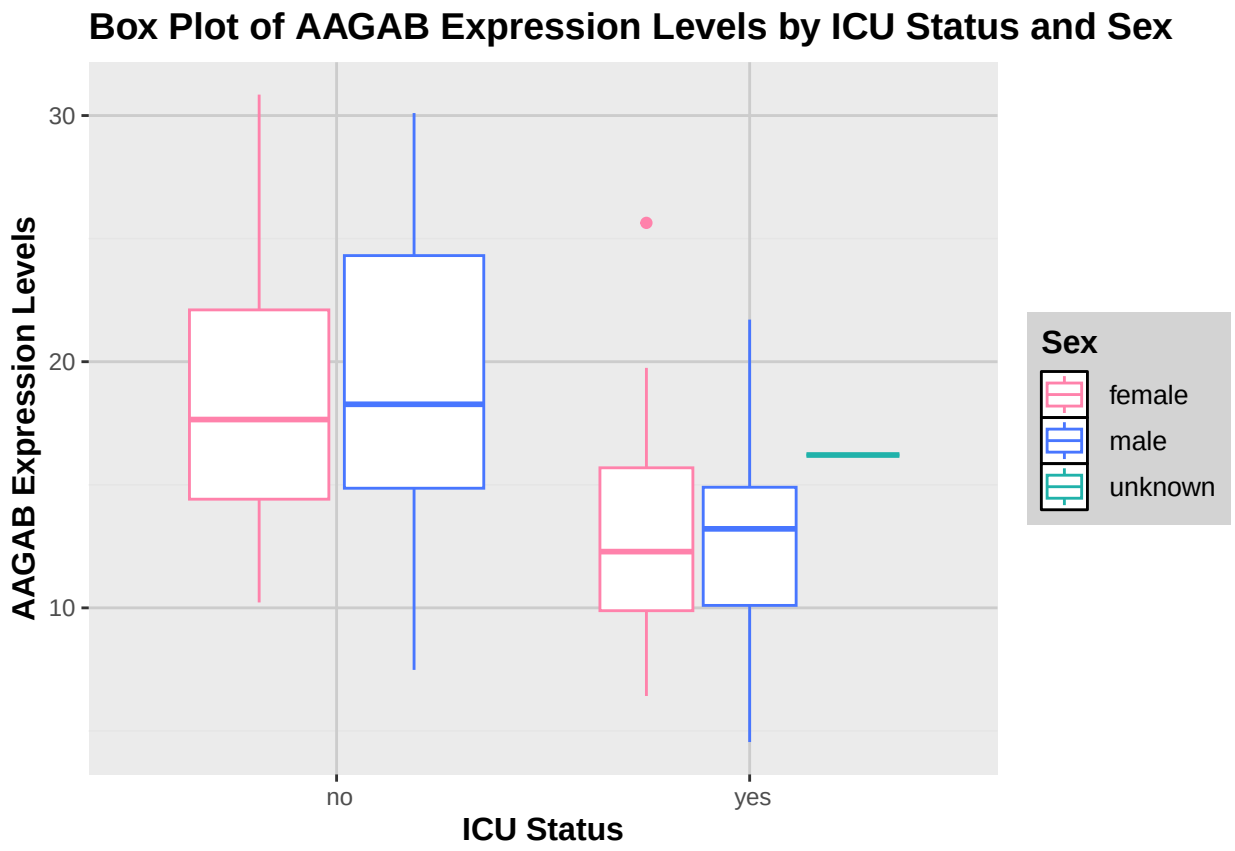
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



Finalizing Box Plot

```
ggplot(series_data,aes(x = icu_status, y = as.numeric(AAGAB), color = sex)) +
  geom_boxplot() +
  labs( # labeling the title and axis
    title = "Box Plot of AAGAB Expression Levels by ICU Status and Sex",
    x = "ICU Status",
    y = "AAGAB Expression Levels",
    color = "Sex"
  ) +
  scale_color_manual(values = c(" female" = "palevioletred1", " male" = "royalblue1", " unknown" = "ligh
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    axis.title = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 12, face = "bold"), # editing the legend
    legend.text = element_text(size = 10),
    legend.background = element_rect(fill = "lightgray", color = NA),
    legend.key = element_rect(fill = "white", color = "black"),
    panel.grid.major = element_line(color = "grey80"), # background
    panel.grid.minor = element_line(color = "grey90")
  )
```




```
#ggplot(series_data,aes(x = disease_status, y = as.numeric(AAGAB), color = sex)) +
#  geom_boxplot() +
#  labs( # labeling the title and axis
#    title = "Box Plot of AAGAB Expression Levels by Disease Status and Sex",
#    x = "Disease Status",
#    y = "AAGAB Expression Levels",
#    color = "Sex"
#  ) +
#  scale_color_manual(values = c(" female" = "palevioletred1", " male" = "royalblue1", " unknown" = "li
#  theme(
#    plot.title = element_text(size = 13, face = "bold"),
#    axis.title = element_text(size = 12, face = "bold"),
#    legend.title = element_text(size = 12, face = "bold"), # editing the legend
#    legend.text = element_text(size = 10),
#    legend.background = element_rect(fill = "lightgray", color = NA),
#    legend.key = element_rect(fill = "white", color = "black"),
#    panel.grid.major = element_line(color = "grey80"), # background
#    panel.grid.minor = element_line(color = "grey90")
#  )
```

New Plot Type

```
#series_data$sex_disease_status <- paste(series_data$sex, series_data$disease_status, sep = " & ")
```

```r
# Density plots with semi-transparent fill
#ggplot(series_data, aes(x=as.numeric(age), fill=icu_status)) + geom_density(alpha=.3)
#ggplot(series_data, aes(x=as.numeric(ferritin), fill=disease_status)) + geom_density(alpha=.3)
#ggplot(series_data, aes(x=as.numeric(ferritin), fill=sex)) + geom_density(alpha=.3)
#ggplot(series_data, aes(x=as.numeric(ferritin), fill=sex_disease_status)) + geom_density(alpha=.3)
```

```r
mu <- series_data %>%
  group_by(disease_status) %>%
  summarize(grp.mean = mean(ferritin.ng.ml., na.rm = TRUE))
```

```
## Warning: There were 2 warnings in `summarize()`.
## The first warning was:
## i In argument: `grp.mean = mean(ferritin.ng.ml., na.rm = TRUE)`.
## i In group 1: `disease_status = "disease state: COVID-19"`.
## Caused by warning in `mean.default()`:
## ! argument is not numeric or logical: returning NA
## i Run `dplyr::last_dplyr_warnings()` to see the 1 remaining warning.
```

```r
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: COVID-19"]),
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: COVID-19"]), n
#mean(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: non-COVID-19
#sd(as.numeric(series_data$ferritin.ng.ml.[series_data$disease_status == "disease state: non-COVID-19"]
```

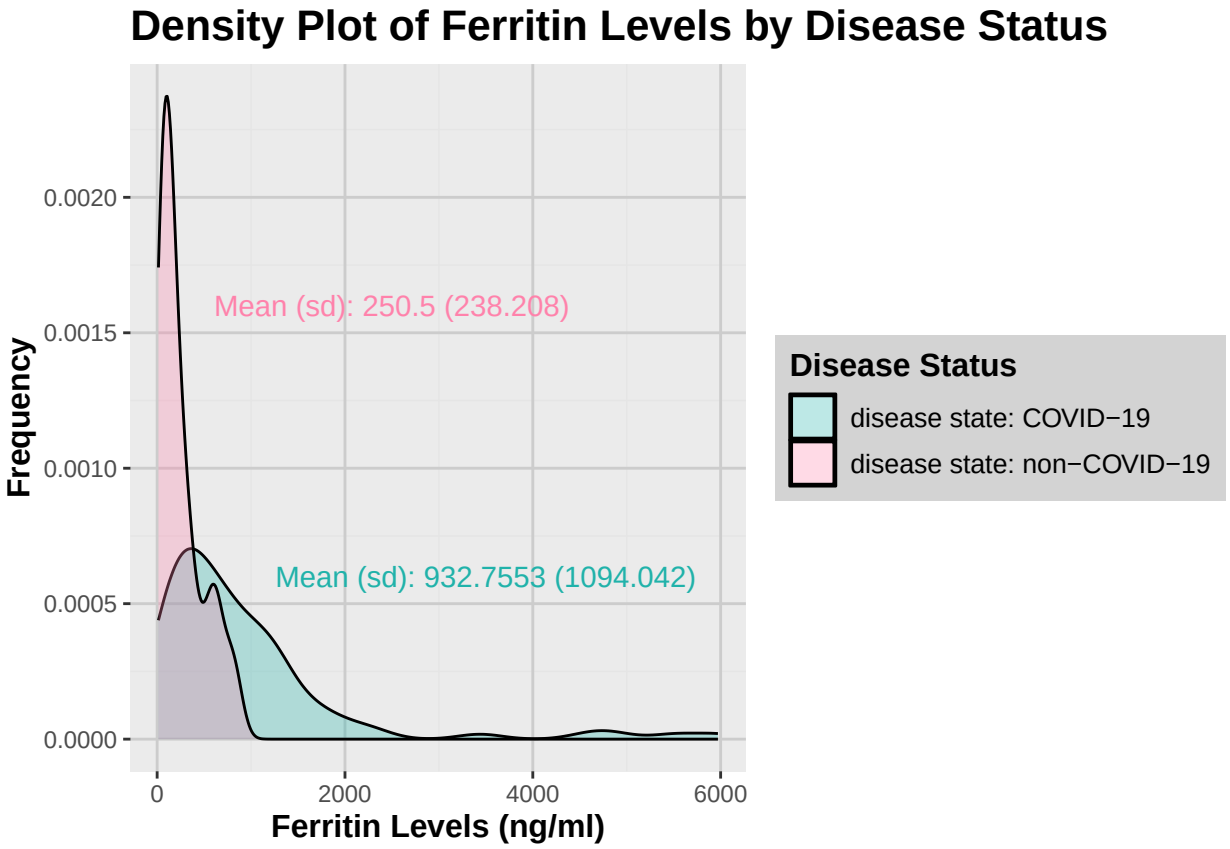
```r
# Density plots with semi-transparent fill
ggplot(series_data, aes(x=as.numeric(series_data$ferritin.ng.ml.), fill=series_data$disease_status)) + g
  labs( # labeling the title and axis
    title = "Density Plot of Ferritin Levels by Disease Status",
    x = "Ferritin Levels (ng/ml)",
    y = "Frequency",
    fill = "Disease Status"
  ) +
  scale_fill_manual(values = c("disease state: COVID-19" = "lightseagreen", "disease state: non-COVID-19
  annotate(geom = 'text',x = 2500,y = 0.0016,label = 'Mean (sd): 250.5 (238.208)',color = 'palevioletred
  annotate(geom = 'text',x = 3500,y = 0.0006,label = 'Mean (sd): 932.7553 (1094.042)',color = 'lightseag
  theme(
    plot.title = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 12, face = "bold"), # editing the legend
    legend.text = element_text(size = 10),
    legend.background = element_rect(fill = "lightgray", color = NA),
    legend.key = element_rect(fill = "white", color = "black"),
    panel.grid.major = element_line(color = "grey80"), # background
    panel.grid.minor = element_line(color = "grey90")
  )
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
## Warning: Use of `series_data$ferritin.ng.ml.` is discouraged.
## i Use `ferritin.ng.ml.` instead.
```

```
## Warning: Use of `series_data$disease_status` is discouraged.
## i Use `disease_status` instead.
```

```
## Warning: Removed 16 rows containing non-finite outside the scale range
## (‘stat_density()‘).
```



Heatmap

```
#install.packages('pheatmap')
library(pheatmap)
```

```
heatmap_gene_data <- gene_data %>%
  mutate(across(c(AAGAB, ABI1, ABHD5, ABHD2, AAAS, AAMDC, AAMP, AAR2, AARS1, AARSD1, AASDHPPT, AATF, AA

# convert into a matrix
heatmap_matrix <- as.matrix(heatmap_gene_data[, c("AAGAB", "ABI1", "ABHD5", "ABHD2", "AAAS", "AAMDC", "

# adding tracking bars
tracking <- data.frame(
  ICU_Status = factor(series_data$icu_status),
  Disease_Status = factor(series_data$disease_status)
)
rownames(tracking) <- rownames(heatmap_gene_data)

# plotting heatmap
pheatmap(
```

```
  heatmap_matrix,
  cluster_rows = TRUE,
  cluster_cols = TRUE,
  annotation_row = tracking,
  show_rownames = TRUE,
  show_colnames = TRUE,
  fontsize_row = 1,
  #height = 10,
  #cellheight = 1
)
```