QBS103_FinalSubmission

Hannah Bahram Pour

2024-08-20

```
# installing needed packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- 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 error
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(pheatmap)
library(table1)
```

```
##
## Attaching package: 'table1'
##
## The following objects are masked from 'package:base':
##
##     units, units<-
```

```
# most of this code is the exact same as Submission 1 and 2

# setting my working directory
setwd("/Users/hannahbahrampour/Desktop")

# checking to see where I am
getwd()
```

```
## [1] "/Users/hannahbahrampour/Desktop"
```

```r
# using read.csv to read in both of the files and assign them to shorter variable names

genes <- read.csv(file = "QBS103_GSE157103_genes.csv")

series_matrix <- read.csv(file = "QBS103_GSE157103_series_matrix.csv")

# melting the genes data into the long format
# Jaini helped me understand the concept of melting and why it is
# necessary in this case
gene_long <- genes %>% tidyr::gather(key = "ParticipantID", value =
                                        "Expression", -X)

# rename a column in the series_matrix to match with genes_long
series_matrix <- series_matrix %>%
  rename(ParticipantID = participant_id)

# merge the data together
data_merged <- merge(gene_long, series_matrix, by = "ParticipantID")
```

```r
#install.packages('xtable')
library(table1)
library(xtable)
```

```
##
## Attaching package: 'xtable'

## The following objects are masked from 'package:table1':
##
##     label, label<-
```

```r
# manual list of labels for my table
labels_list <- list(
  disease_status = "Disease Status",
  sex = "Sex",
  ferritin = "Ferritin (ng/ml)",
  lactate = "Lactate (mmol/l)",
  sofa = "Sofa",
  icu_status = "ICU Status"
)

# making copy of merged data
datatable <- data_merged

# changing all the unknown values to NA
datatable[datatable == "unknown" | datatable == " unknown" |
                datatable == " :" | datatable == " >89"] <- NA

datatable <- na.omit(datatable) # omitting the NAs

levels(datatable$disease_status) # creating levels for disease status
```

```
## NULL
```

```r
# function to get the median's of my variables
# I got help from Antara to troubleshoot my function when it wasn't working
mtable <- function(x, name, ...){
  if (!is.numeric(x)) {
    return(render.categorical.default(x))
  }
  # laying out all my statistical calculations for variables
  calc <- switch(name,
                 ferritin.ng.ml. = "Median [Min, Max]",
                 lactate.mmol.l. = "Median [Min, Max]",
                 sofa = "Median [Min, Max]",
                 "Mean (SD)")

  # doing the actual calculations
  parse.abbrev.render.code(c("", calc))(x)
}

# ensuring all values into the table are numeric
datatable$ferritin.ng.ml. <- as.numeric(datatable$ferritin.ng.ml.)
datatable$lactate.mmol.l. <- as.numeric(datatable$lactate.mmol.l.)
datatable$sofa <- as.numeric(datatable$sofa)

# actually making my table
table1(~ icu_status + sex + ferritin.ng.ml. + lactate.mmol.l. + sofa |
              disease_status, data = datatable,
              render = mtable, overall = "Overall")
```

```
## Get nicer `table1` LaTeX output by simply installing the `kableExtra` package
```

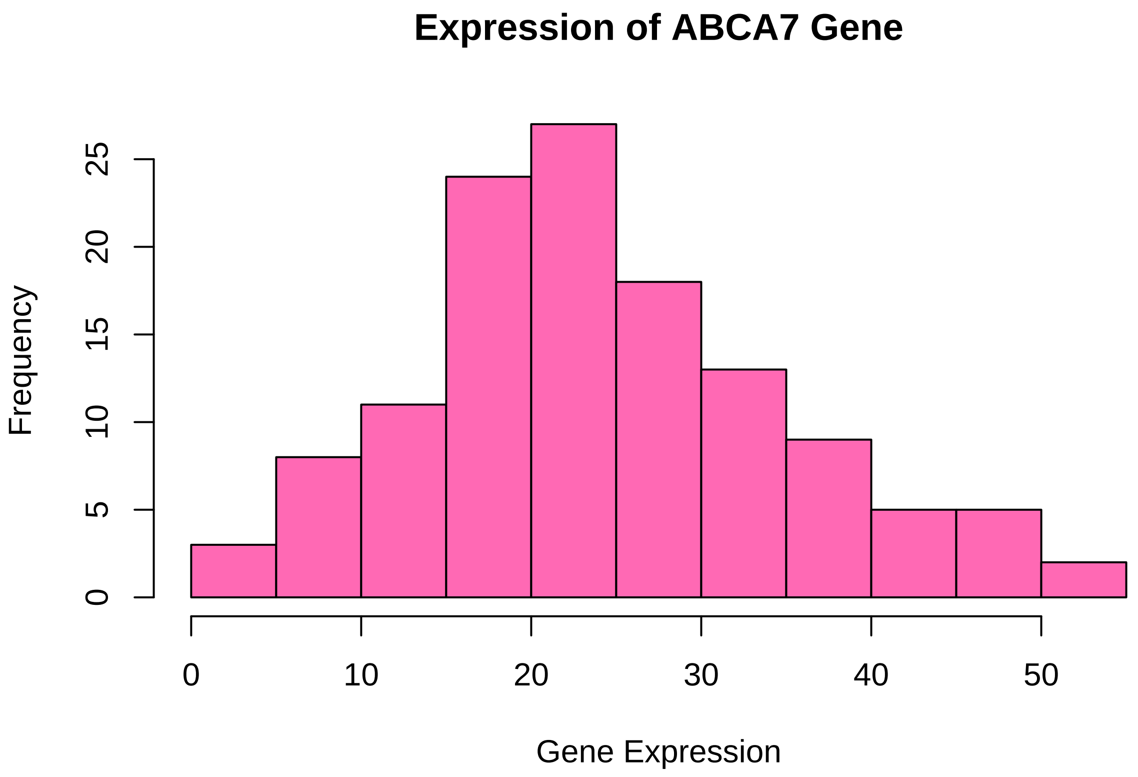
	disease state: COVID-19	disease state: non-COVID-19	Overall
	(N=4000)	(N=500)	(N=4500)
icu_status			
no	300 (7.5%)	0 (0%)	300 (6.7%)
yes	3700 (92.5%)	500 (100%)	4200 (93.3%)
sex			
female	800 (20.0%)	300 (60.0%)	1100 (24.4%)
male	3200 (80.0%)	200 (40.0%)	3400 (75.6%)
ferritin.ng.ml.			
Median [Min, Max]	811 [77.0, 5510]	211 [46.0, 297]	735 [46.0, 5510]
lactate.mmol.l.			
Median [Min, Max]	1.20 [0.500, 2.85]	3.68 [0.950, 9.91]	1.22 [0.500, 9.91]
sofa			
Median [Min, Max]	7.50 [2.00, 18.0]	9.00 [3.00, 12.0]	8.00 [2.00, 18.0]

```r
# I used the link below to help me use table1 and format my table
# https://cran.r-project.org/web/packages/table1/vignettes/table1-examples.html
```

```r
# this code is almost identical to my code from submission 1
# with the exception of code improvements I made based off feedback

# selecting my gene and filtering for it
# assign this clean selected gene data to a variable
clean_data <- data_merged %>%
  filter(X == "ABCA7") %>% # gene selection
  select(X, ParticipantID, Expression, age, sex, icu_status)

# ensure values are numeric if not already
clean_data$Expression <- as.numeric(clean_data$Expression)

# added this based off submission 1 feedback
# ensuring that age is numeric
clean_data$age <- as.numeric(clean_data$age)
```

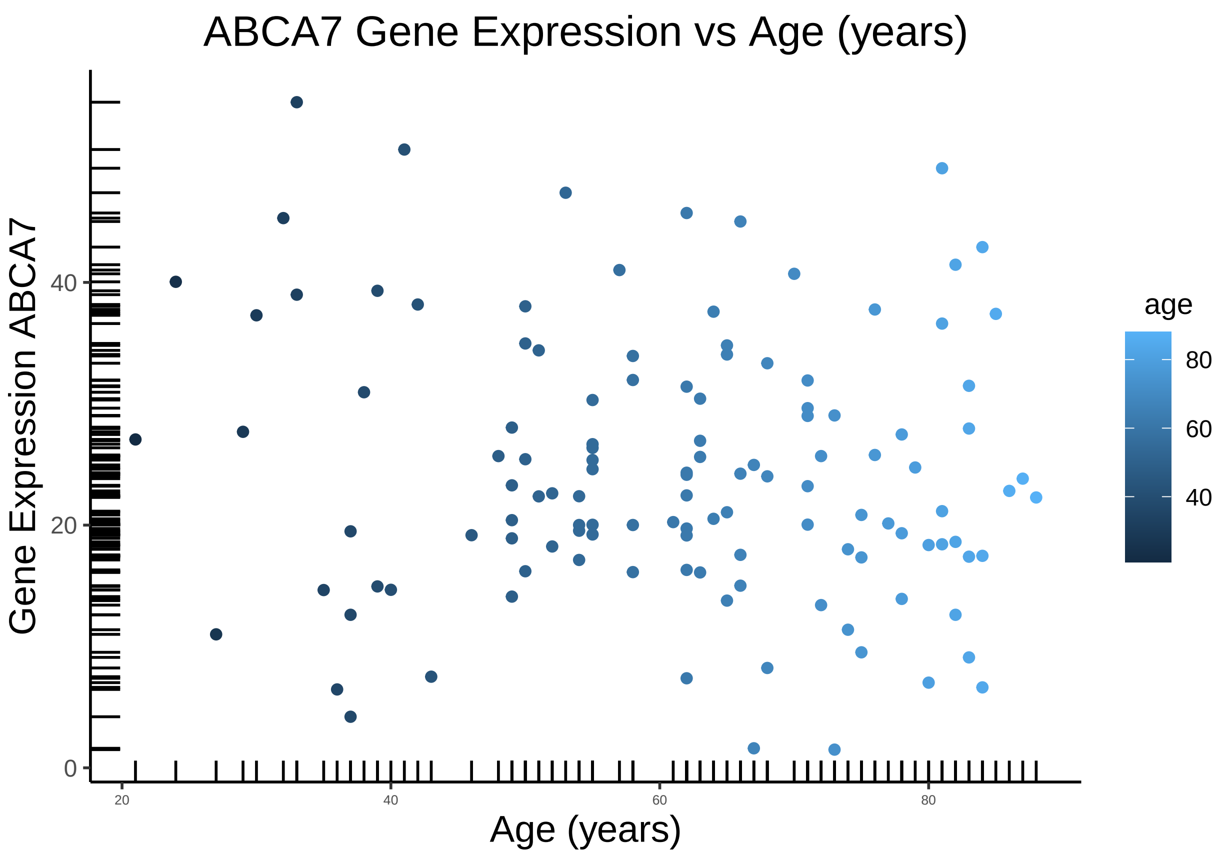
```
## Warning: NAs introduced by coercion
```

```r
# CREATING HISTOGRAM
# making the histogram hot pink and labeling it
hist(clean_data$Expression, main = paste("Expression of ABCA7 Gene"),
breaks=10, col = "hotpink", xlab = "Gene Expression")
```

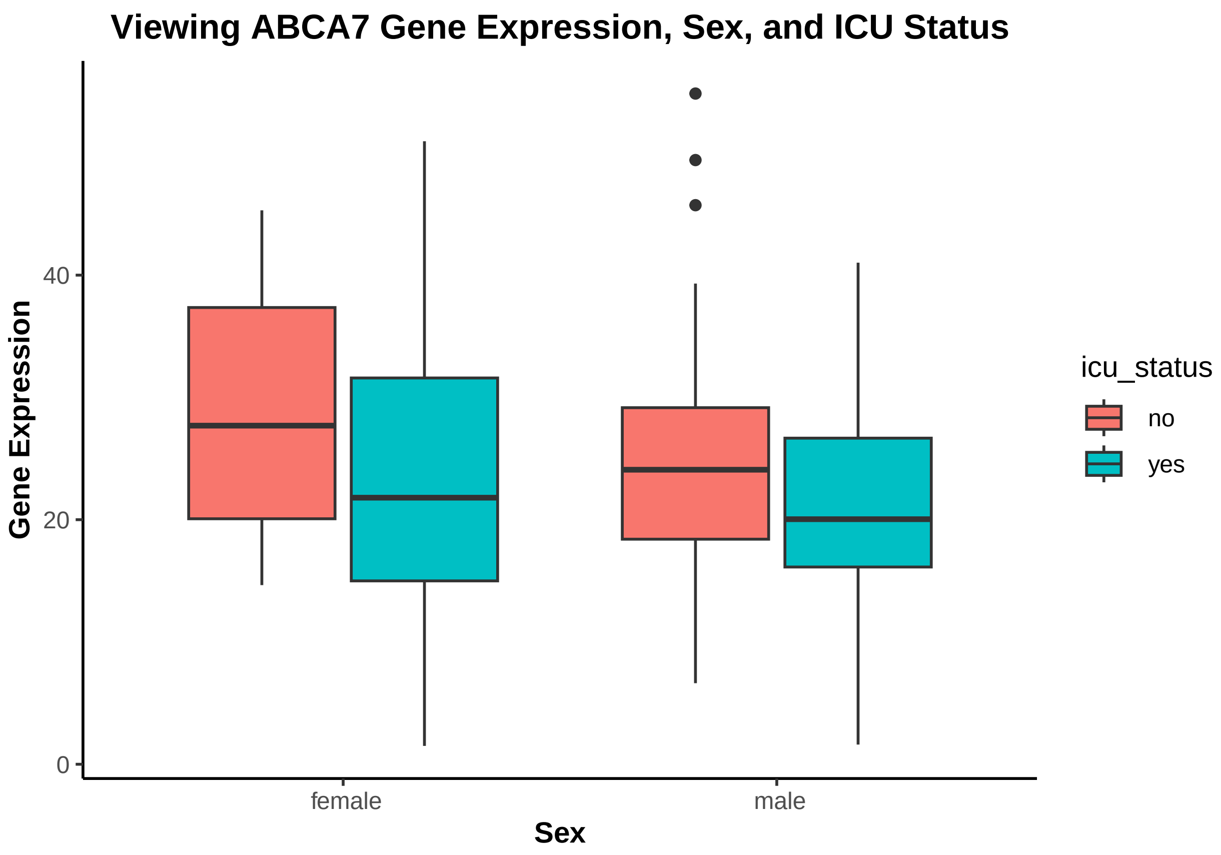


```r
# CREATING SCATTERPLOT
# for gene expression and continuous covariate (age)
ggplot(clean_data, aes(x = age, y = Expression)) +
  geom_point(aes(color = age)) + # adding points and color dependent on age
  geom_rug() + # adding rug
  labs(title = "ABCA7 Gene Expression vs Age (years)",
       x = "Age (years)", y = "Gene Expression ABCA7") + # label the scatterplot
  theme_classic() + # getting rid of the background grid
  theme( # adjusting text sizes
    plot.title = element_text(hjust = 0.5, size = 16),
    axis.title = element_text(size = 14),
    legend.title = element_text(hjust = 0.5),
    axis.text.x = element_text(size = 5) # trying to make the x-axis more readable
  )
```

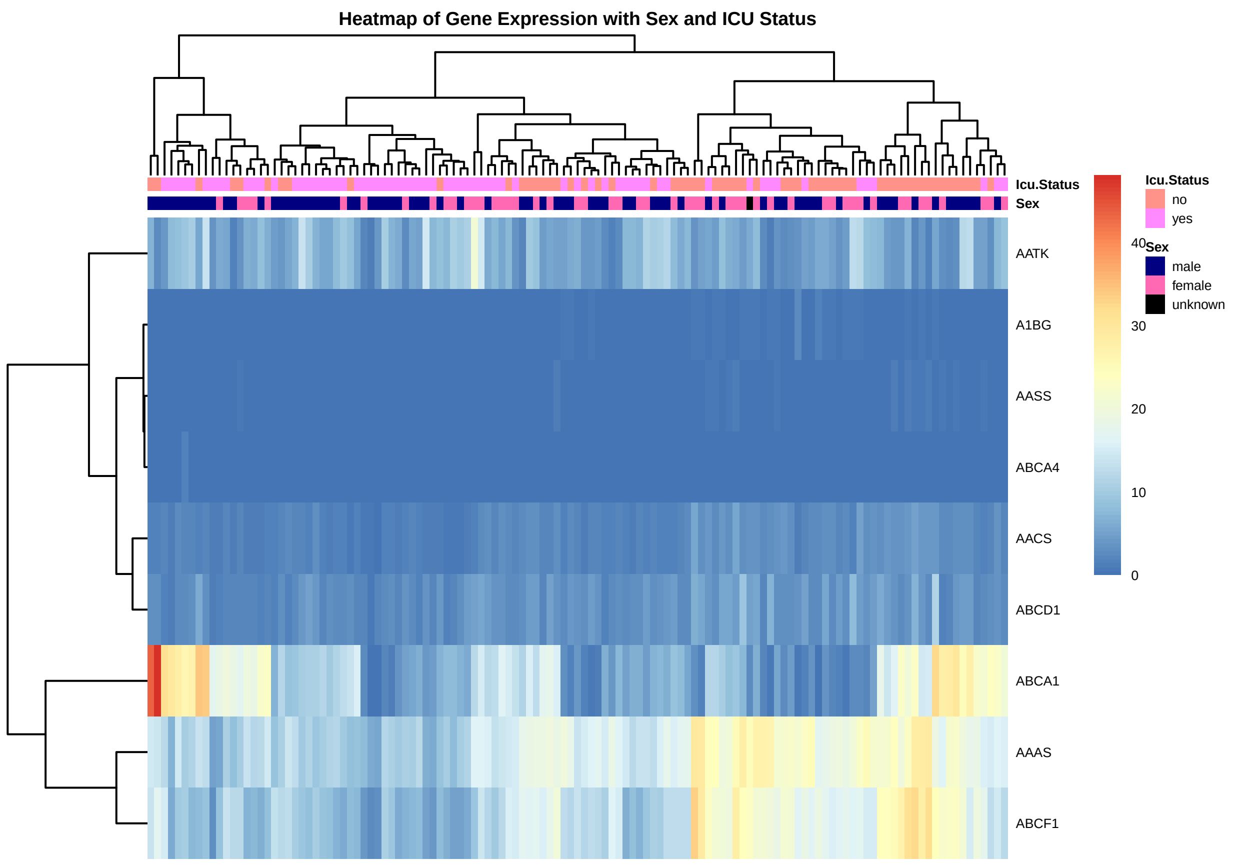
```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



```
# cleaning out data for unknown sex
new_clean_data <- clean_data %>%
  filter(sex == " male" | sex == " female")

# CREATING BOXPLOT
# gene expression separated by two categorical covariates (sex and ICU status)
ggplot(new_clean_data, aes(x = sex, y = Expression, fill = icu_status)) +
  geom_boxplot() + # adding in the boxplot
  labs(title = "Viewing ABCA7 Gene Expression, Sex, and ICU Status",
       x = "Sex", y = "Gene Expression") + # labeling things
  scale_alpha_manual(name = "ICU Status") +
  theme_classic() + # getting rid of background grid
  theme( # adjusting the title and axis title
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold")
  )
```

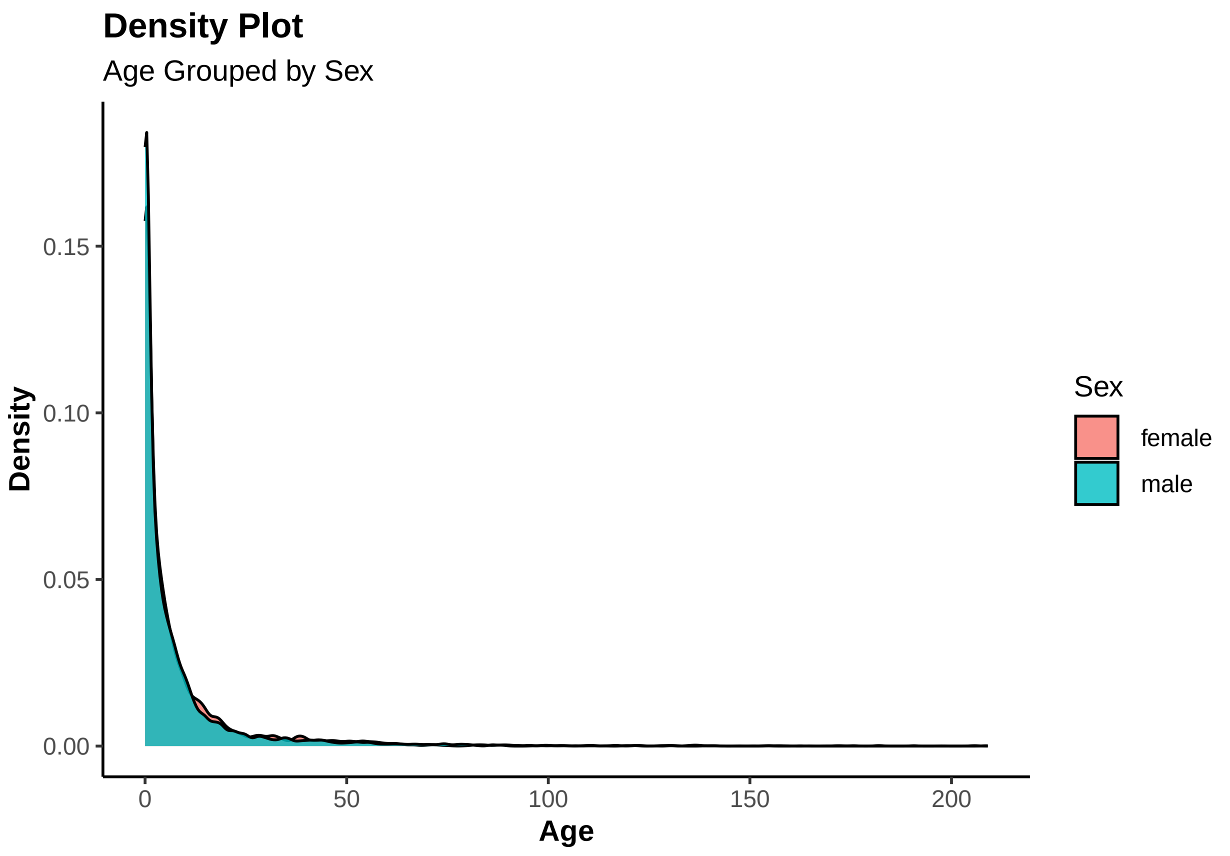


```r
# GENERATING HEATMAP

# Jaini helped me troubleshoot my heatmap!
# She figured out why my data wasn't showing

# filtering out data that I want and ten genes
heatmap_data <- data_merged %>%
  dplyr::filter(X %in% c("A1BG", "AASS", "AATK", "ABCA1", "AASS", "AAAS",
                         "AACS", "ABCD1", "ABCF1", "ABCA4")) %>%
  tidyr::pivot_wider(names_from = X, values_from = Expression)

# getting gene data into matrix
heatmap_matrix <- heatmap_data %>%
  select(all_of(c("A1BG", "AASS", "AATK", "ABCA1", "AASS", "AAAS",
                         "AACS", "ABCD1", "ABCF1", "ABCA4"))) %>% as.matrix()

row.names(heatmap_matrix) <- heatmap_data$ParticipantID

# converting to data frame
heatmap_matrix <- as.data.frame(t(heatmap_matrix))

annotationData <- data.frame(row.names = colnames(heatmap_matrix),
                         'Sex' = heatmap_data$sex,
                         'Icu Status' = heatmap_data$icu_status)

# setting colors for annotations
annotationColors <- list(
  Sex = c(' male' = 'navyblue', ' female' = 'hotpink', ' unknown' = 'black'),
  icu_status = c('yes' = 'darkgreen', 'no' = 'red'))

# generating heatmap
pheatmap(heatmap_matrix,
         annotation_col = annotationData,
         annotation_colors = annotationColors,
         clustering_distance_rows = 'euclidean',
         clustering_distance_cols = 'euclidean',
         show_rownames = TRUE,
         show_colnames = FALSE,
         fontsize = 5,
         annotation_legend = TRUE,
         main = 'Heatmap of Gene Expression with Sex and ICU Status')
```



```r
# GENERATING NEW PLOT TYPE
# Selected plot type: density plot

# selecting columns and data of interest for density plot
density_data <- data_merged %>%
  select(X, ParticipantID, Expression, age, sex, icu_status)

# cleaning out all the sex's that are identified as unknown
density_data <- density_data %>%
  filter(sex == " male" | sex == " female")
```

```r
# creating density plot
density_plot <- ggplot(density_data, aes(x = Expression))
density_plot + geom_density(aes(fill = sex), alpha = 0.8) +
  labs(title = "Density Plot", y = "Density", subtitle = "Age Grouped by Sex",
       x = "Age", fill = "Sex") +
  theme_classic() +
  theme(
    plot.title = element_text(face = "bold"),
    axis.title = element_text(hjust = 0.5, face = "bold")
  )
```



```r
# learned about density plot's and how to program them in R at this website:
# http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html
```