JFAITH_QBS103_final_project

2025-07-12

Comparing Biomarkers to Covid-19

Data Wrangling

Read in files

```r
gene_data <- read.csv("data/QBS103_GSE157103_genes.csv")
meta_data <- read.csv('data/QBS103_GSE157103_series_matrix-1.csv')
```

Clean Metadata

```r
# Cleaning disease status names
cleaned_state <- function(status) {
  if (status == "disease state: COVID-19") {
    return("Diseased")
  } else {
    return("Healthy")
  }
}

meta_data$covid_status <- sapply(meta_data$disease_status, cleaned_state)

# shortening subject ids but preserving uniqueness
shorten_subject_names <- function(name) {
  split_name <- strsplit(name, "_")[[1]]   # split into pieces

  if (split_name[1] == "COVID") {
    return(paste0("Subject_C", split_name[2]))

  } else {
    return(paste0("Subject_N", split_name[2]))
  }
}

meta_data$subject_id <- sapply(meta_data$participant_id, shorten_subject_names)
#making sure shortened ID are all unique (i.e. still same number of unique ids as before)
length(unique(meta_data$participant_id)) == length(unique(meta_data$subject_id))
```

```
## [1] TRUE
```

Gene isolation

I chose to look at the ABCF1 gene becaue of it's role in regulating immune responses. Source: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0175918

```r
# Select ABCF1 gene
gene <- gene_data[which(gene_data$X=='ABCF1'), ]

# Get all column names except X
column_names <- names(gene)[-1]

#Melt dataframe
#referenced: https://tidyr.tidyverse.org/reference/pivot_longer.html
gene_long <- pivot_longer(gene, cols = all_of(column_names), names_to = "participant_id", values_to = "
```

Merge metadata & gene data

```r
gene_dataset <- inner_join(gene_long, meta_data, by="participant_id")
# referenced: https://datascienceplus.com/merging-datasets-with-tidyverse/


# Check to see if every record in the gene data matched to the metadata table since inner joining
if (nrow(gene_long) == nrow(meta_data)) {
  print('All records matched and none were dropped through the merge')
} else (
  print('missing records')
)
```

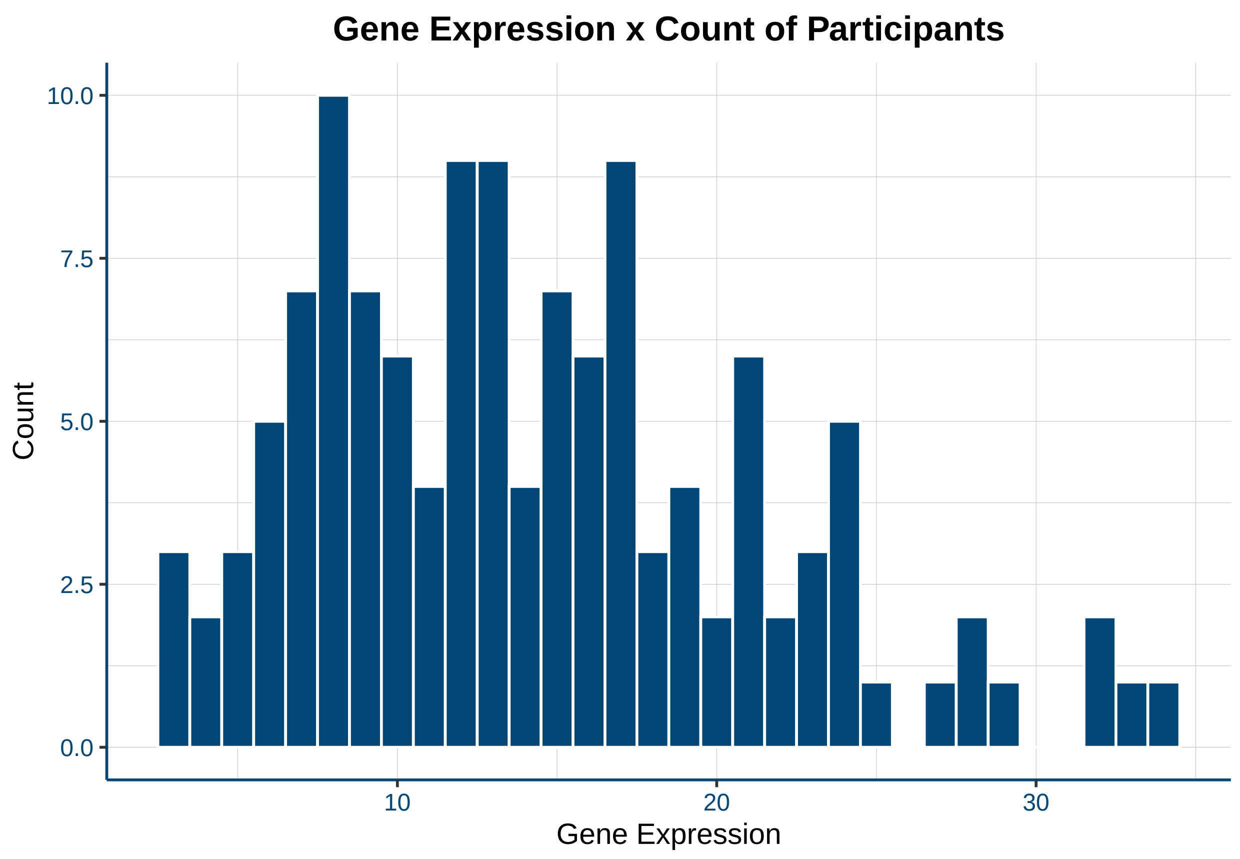
```
## [1] "All records matched and none were dropped through the merge"
```

Visualizations

Histogram for gene expression

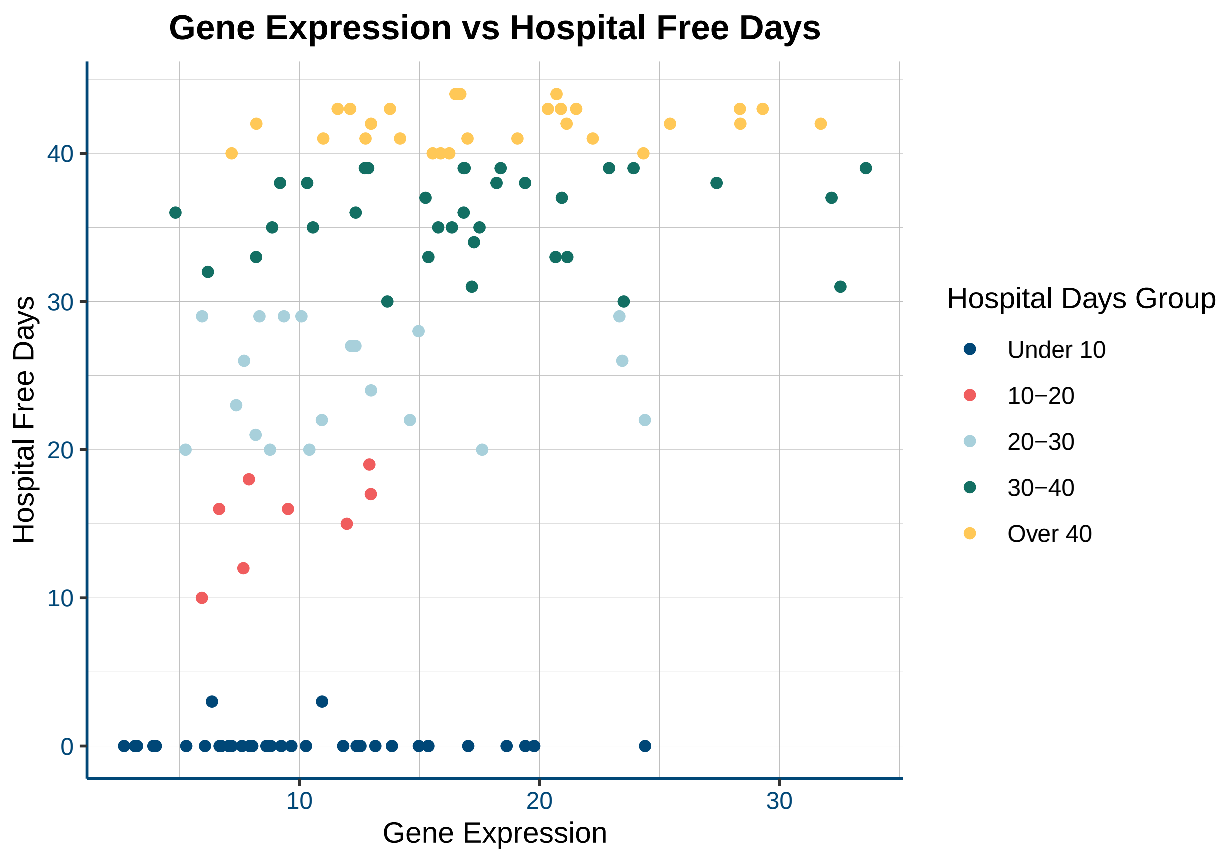
```r
# Adding custom color category by days in the hospital
gene_dataset$HospitalDaysGroup <- cut(
  gene_dataset$hospital.free_days_post_45_day_followup,
  breaks = c(0, 10, 20, 30, 40, 50),
  labels = c('Under 10', '10-20', '20-30', '30-40', 'Over 40'),
  right = FALSE
)
```

```r
#Custom Color Palette
colorPalette <- c('#004777', '#F05D5E', '#A8D0DB', '#136F63', '#FFC857', '#9DD9D2')

myTheme <- theme(
        panel.border = element_blank(),
        panel.grid.major = element_line(colour="grey", linewidth = rel(.1)),
        panel.grid.minor= element_line(colour="grey", linewidth = rel(.1)),
        #panel.grid.minor = element_blank(),
        # Define my axis
        plot.title = element_text(colour = "black", hjust = .5, face='bold'),
        axis.line = element_line(colour = '#004777', linewidth = rel(1)),
        axis.title.x = element_text(color='black'),
        #axis.title.y = element_text(color='black'),
        axis.text = element_text(color='#004777'),
        # Set plot background
        panel.background = element_blank()
        )


ggplot(gene_dataset, aes(x = expression)) +
  geom_histogram(binwidth=1, color="white", fill='#004777') +
  labs(title="Gene Expression x Count of Participants", x = 'Gene Expression',y = 'Count') +
  myTheme
```



Scatterplot for gene expression and continuous covariate

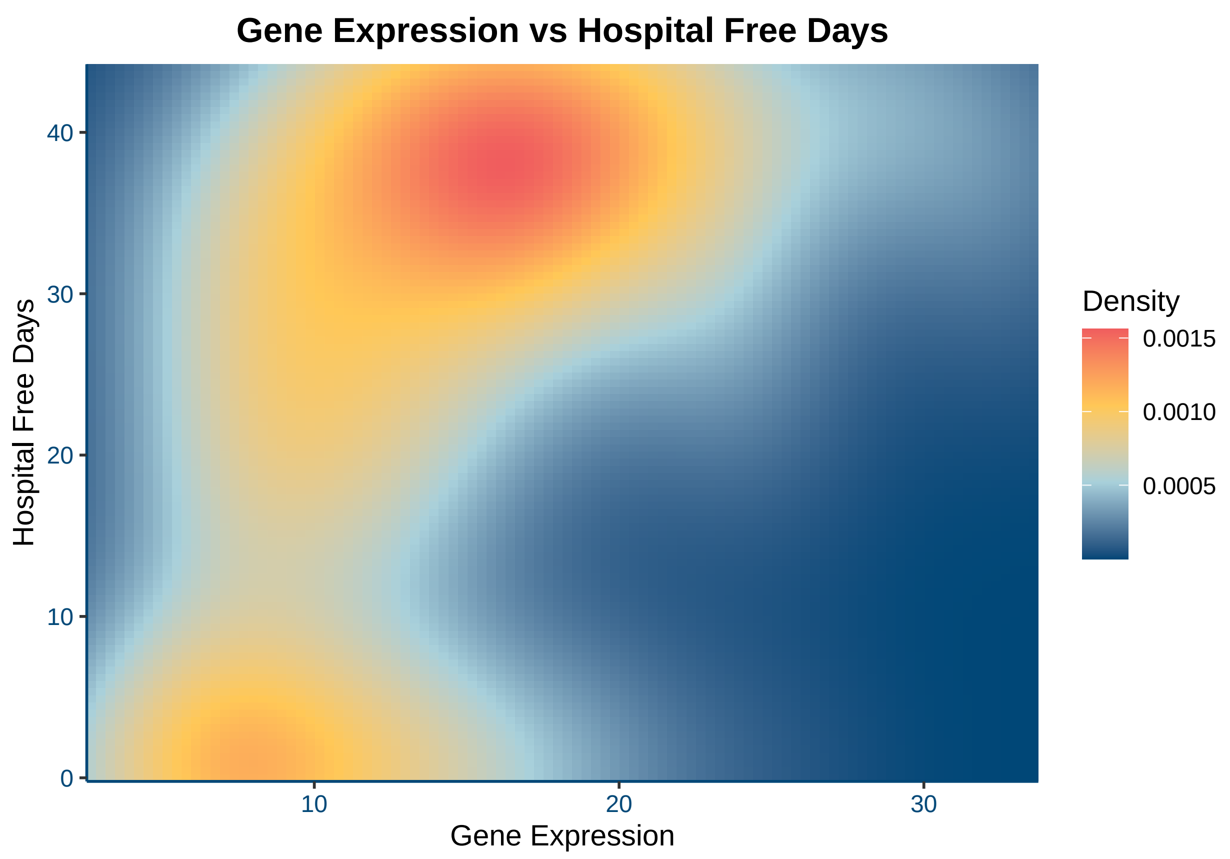
```
ggplot(gene_dataset, aes(x = expression, y = hospital.free_days_post_45_day_followup, color = HospitalDa
  geom_point() +
  labs(title="Gene Expression vs Hospital Free Days",
       x = 'Gene Expression',
      y = 'Hospital Free Days',
      color = "Hospital Days Group")+
  scale_color_manual(values = colorPalette)+
  myTheme
```



Densite Chart

```
ggplot(gene_dataset, aes(x = expression, y = hospital.free_days_post_45_day_followup)) +
    stat_density_2d(aes(fill = ..density..), geom = "raster", contour = FALSE) +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0)) +
    scale_fill_gradientn(colors = c('#004777', '#A8D0DB', '#FFC857', '#F05D5E'))+
    labs(title="Gene Expression vs Hospital Free Days",
         x = 'Gene Expression',
         y = 'Hospital Free Days',
         fill = "Density")+
    myTheme
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Boxplot of gene expression separated by both categorical covariates

```r
ggplot(gene_dataset, aes(x = covid_status, y = expression, color=sex)) +
  geom_boxplot() +
  labs(title="Gene Expression vs Covid-19 Status",
       x = 'Covid Status',
       y = 'Gene Expression',
       color = 'Sex')+
  scale_color_manual(values = colorPalette, labels=str_to_title) +
  myTheme
```

Heatmap

```r
# Making X the row name
gene_data <- gene_data[order(gene_data$X), ]
rownames(gene_data) <- gene_data$X

# Removing the extra X row
gene_data = subset(gene_data, select = -c(X) )

# Shortening subject ids just for display purposes
colnames(gene_data) <- sapply(colnames(gene_data), shorten_subject_names)
```

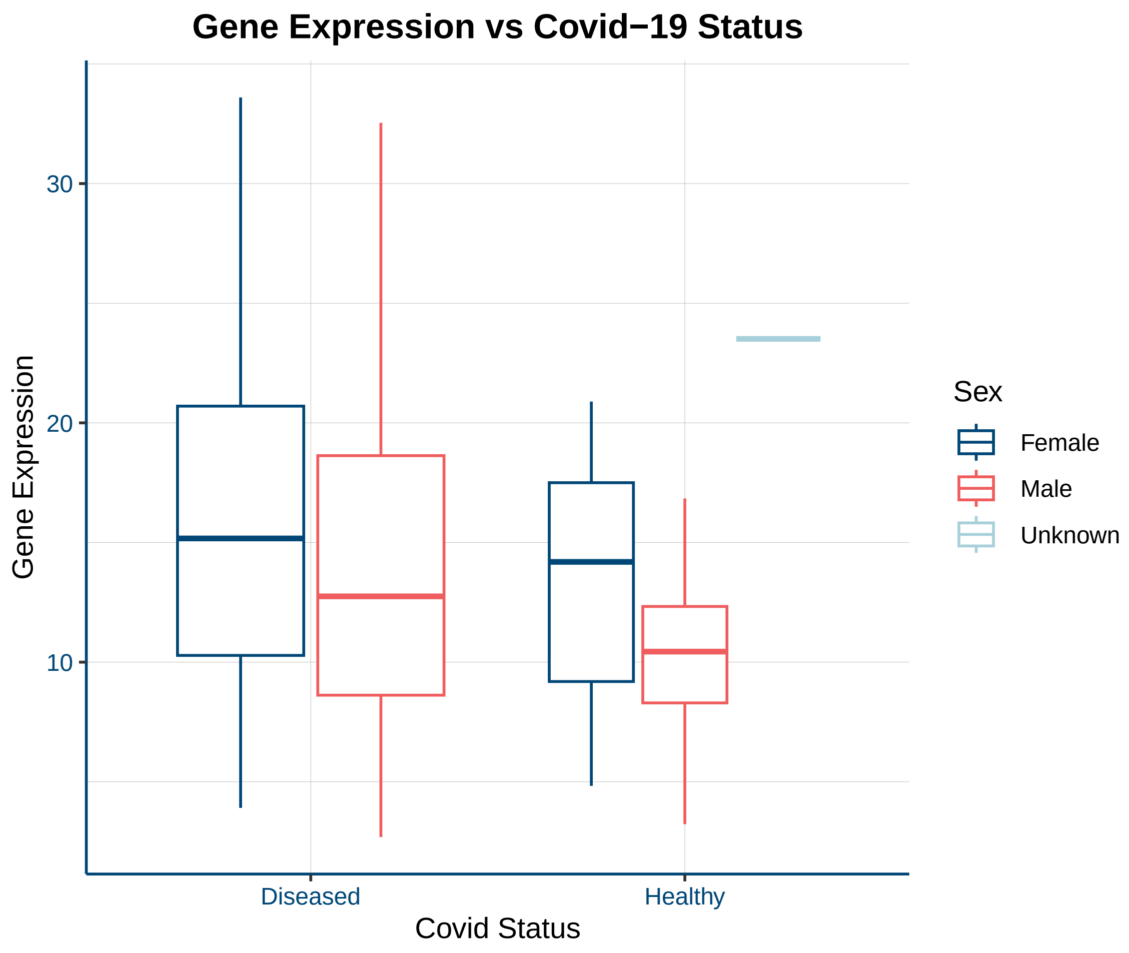
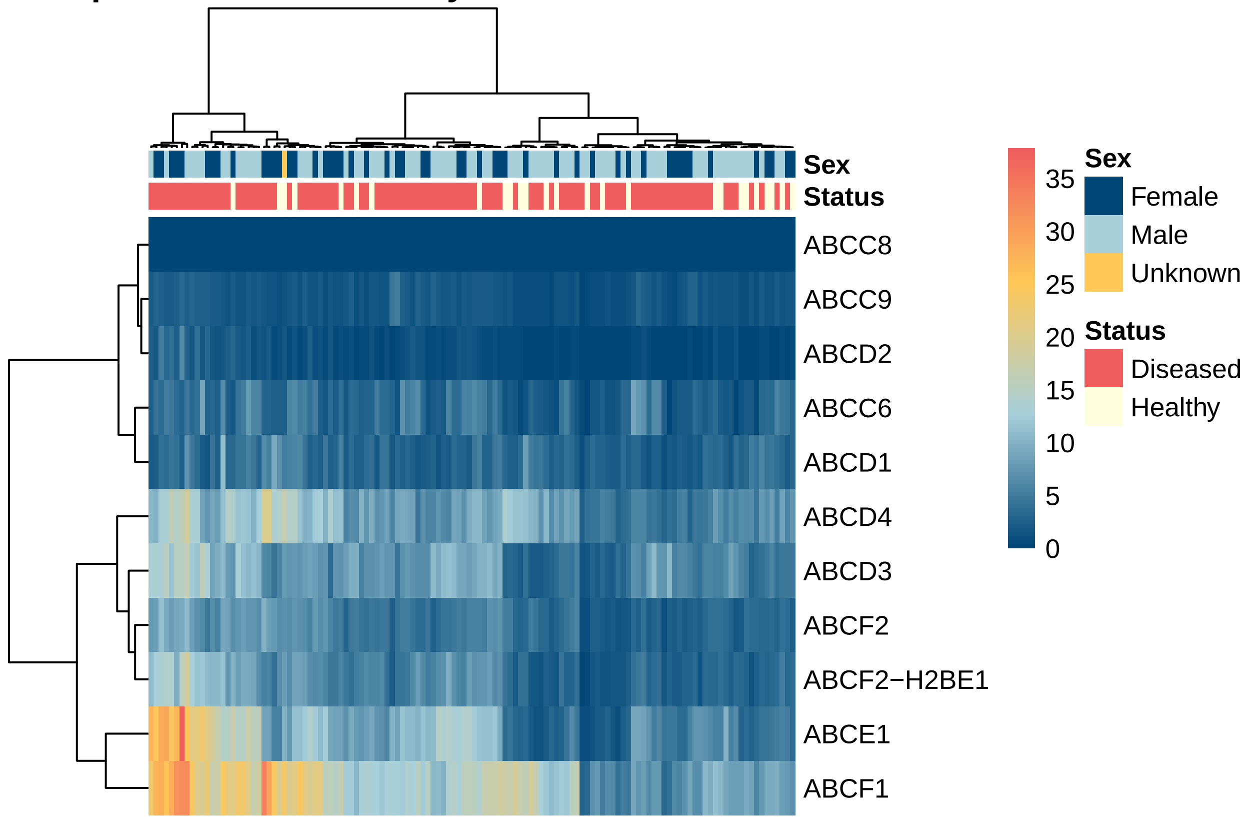


Figure 1: Gene Expression vs Hospital Free Days

```r
annotationData <- data.frame(
  row.names = meta_data$subject_id,
  Status = factor(meta_data$covid_status,
                  levels = c("Diseased", "Healthy"),
                  labels = c("Diseased", "Healthy")),
  Sex=factor(meta_data$sex,
             levels = c(" female", " male", " unknown"),
             labels = c("Female", "Male", "Unknown"))
  # I used ChatGPT here ^^ to figure out how to change the labels to capitals
  # I was struggling to find documentation elsewhere
)

annotationColors <- list(
  Status = c("Diseased" = "#F05D5E", "Healthy" = "lightyellow"),
  Sex    = c("Female" = "#004777","Male" = "#A8D0DB","Unknown" = "#FFC857")
)

# Source: https://davetang.org/muse/2018/05/15/making-a-heatmap-in-r-with-the-pheatmap-package/

# Generate heatmap with clustering
pheatmap(gene_data[60:70, ],
         cluster_rows = T,
         cluster_cols = T,
         annotation_col = annotationData,
         annotation_colors = annotationColors,
         color = colorRampPalette(c('#004777', '#A8D0DB', '#FFC857', '#F05D5E'))(100),
         cluster_distance_cols='euclidean',
         cluster_distance_rows='euclidean',
         clustering_method = 'ward.D',
         show_colnames = F,
         main = "Gene Expression Annotated by COVID Disease Status and Sex "
         )
```

ne Expression Annotated by COVID Disease Status and Sex



```r
# Make sure all relevant columns are numeric since I'm getting NA errors
gene_dataset$age <- as.numeric(gene_dataset$age)
```

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

```r
gene_dataset$hospital.free_days_post_45_day_followup <- as.numeric(gene_dataset$hospital.free_days_post_
gene_dataset$expression <- as.numeric(gene_dataset$expression)

# Variables by type
continuous_variables <- c('expression', 'age', 'hospital.free_days_post_45_day_followup')
covariates <- c('sex', 'icu_status')

## function that creates the vector for me with the summary statistics
## general ideas is it walks through the continuous variables and the
## covariate and appends each items summary stats to a vector
summarize_function <- function(df) {
  summary_vector <- c()

  # Continuous variables
  # Append mean (sd) to vectro
  for (var in continuous_variables) {
    m <- mean(df[[var]], na.rm = TRUE)
    s <- sd(df[[var]], na.rm = TRUE)
    summary_vector <- c(summary_vector, paste0(round(m, 1), " (", round(s, 1), ")"))
  }
  # Sex
  summary_vector <- c(summary_vector, "")
  for (sex_cat in c(" female", " male", " unknown")) {
    count <- sum(df$sex == sex_cat, na.rm = TRUE)
    percent <- round((count / nrow(df)) * 100, 1)
    summary_vector <- c(summary_vector, paste0(count, " (", percent, ")"))
  }

  # ICU status
  summary_vector <- c(summary_vector, "")
  for (status in c(" no", " yes")) {
    count <- sum(df$icu_status == status, na.rm = TRUE)
    percent <- round((count / nrow(df)) * 100, 1)
    summary_vector <- c(summary_vector, paste0(count, " (", percent, ")"))
  }

  return(summary_vector)
}
```

```r
healthy_gd  <- gene_dataset[gene_dataset$covid_status == "Healthy", ]
diseased_gd <- gene_dataset[gene_dataset$covid_status == "Diseased", ]

healthy_n <- nrow(healthy_gd)
diseased_n <- nrow(diseased_gd)

healthy  <- summarize_function(healthy_gd)
diseased <- summarize_function(diseased_gd)

# Source:
# https://vivdas.medium.com/create-latex-and-ms-word-tables-in-r-6ac919204247
#Define Table
table1 <- data.frame(
  Variable = c('Gene ABCF1 Expression mean (sd)',
               'Age mean (sd)',
               'Hospital Free Days mean (sd)',
               'Sex n (%)',
               'Female',
               'Male',
               'Unknown',
               'ICU Status (%)',
               'No',
               'Yes'
  ),
  Healthy = healthy,
  Diseased = diseased
  )

table1 <- kable(
  table1,
  format = "latex",
  booktabs = TRUE,
  col.names = c("Variable",
                paste0("Healthy (n=", healthy_n, ')'),
                paste0("Diseased (n=", diseased_n, ')')
  ),
  caption = "Summary Table",
  align = c("l","r","r"),
  escape = TRUE
)%>%
  add_indent(c(5, 6, 7, 9,10)) %>%
  kable_classic(full_width = FALSE)

# Write LaTeX table code into a file
writeLines(table1, "summary_table.tex")
```