lucas_analysis

Lucas Childs

2025-11-03

1. What is the reason for log transforming protein levels in `biomarker-raw.csv`?

```r
set.seed(1234)
library(here)
```

```
## here() starts at /Users/lucaschilds/PSTAT197A/module-1-biomarker-data-table13
```

```r
rawdata <- read.csv(here("data", "biomarker-raw.csv"))

# random sample of 4 proteins to look at distributions of their levels
rand_indices <- sample(3:ncol(rawdata), 4)

prot1d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[1]])
prot1 <- prot1d[!is.na(prot1d)]

prot2d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[2]])
prot2 <- prot2d[!is.na(prot2d)]

prot3d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[3]])
prot3 <- prot3d[!is.na(prot3d)]

prot4d <- as.numeric(rawdata[2:nrow(rawdata), rand_indices[4]])
prot4 <- prot4d[!is.na(prot4d)]

summary(prot1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    2568    4649    5169    5243    5668    7435
```

```r
summary(prot2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    3317    8790   12179   17164   18800  122168
```

```r
summary(prot3)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   283.1   378.7   414.2   457.8   496.9  1894.7
```

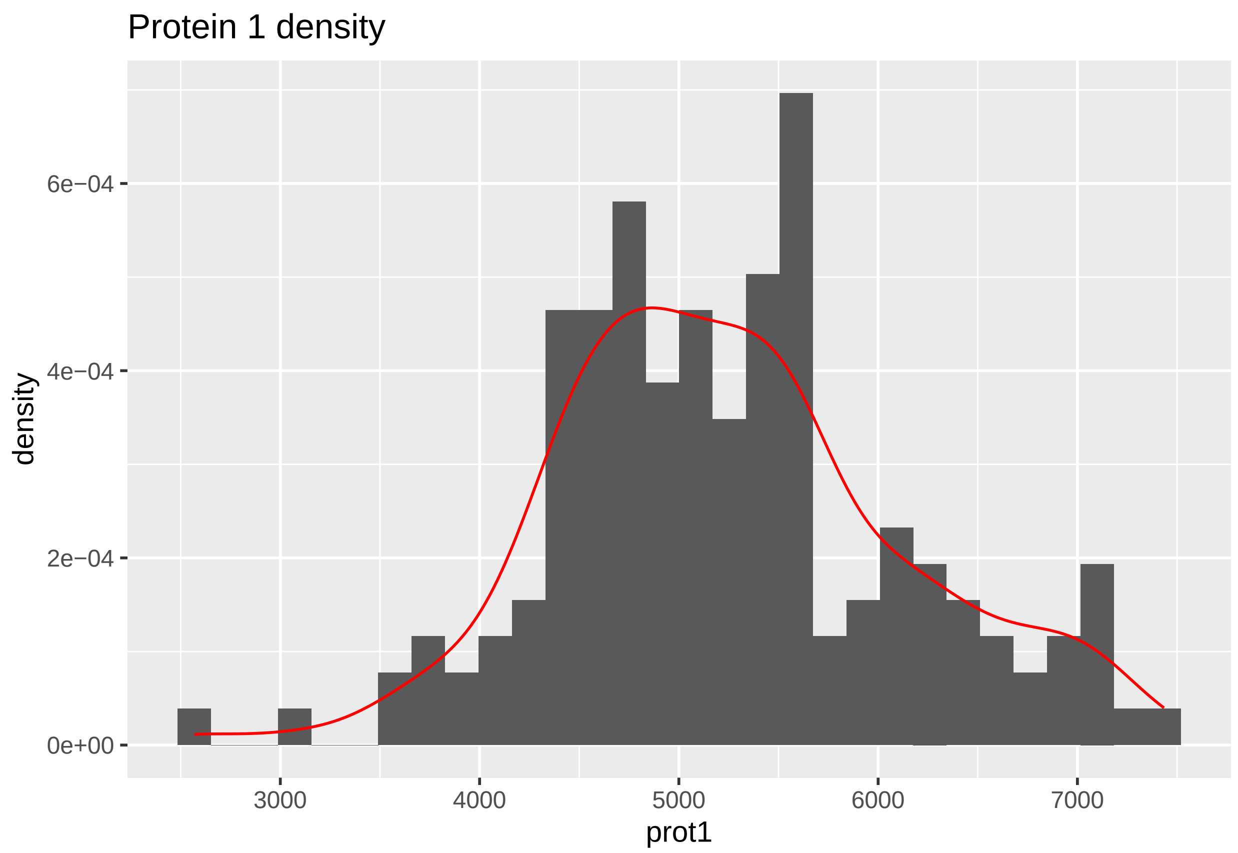
```r
summary(prot4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   954.9  5434.7  6539.2  6551.7  7719.1 10917.8
```

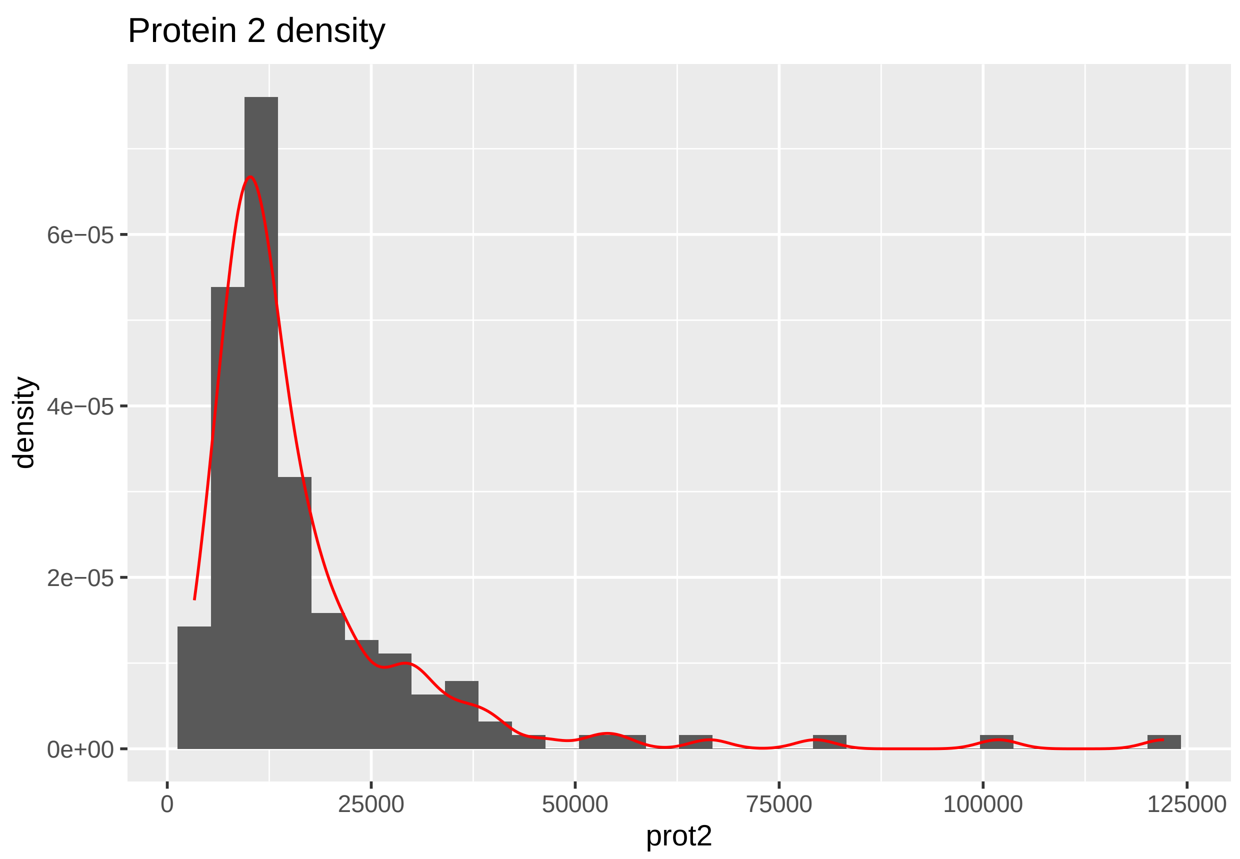
Looking at the summary statistics for 4 randomly selected proteins, the mean values differ significantly, with `prot1`'s mean being 17,164 and `prot3`'s mean being ≈ 458. `prot2`'s mean is roughly 5000 units greater than its median as well, indicating a right skew.

The log transformation of the protein levels helps compress the scale of protein levels since we have a wide range of positive values, where some are very large. Additionally, the logarithm helps with skewed data, and we have evidence that some of the data is skewed, since `prot2` has a mean much larger than its median.

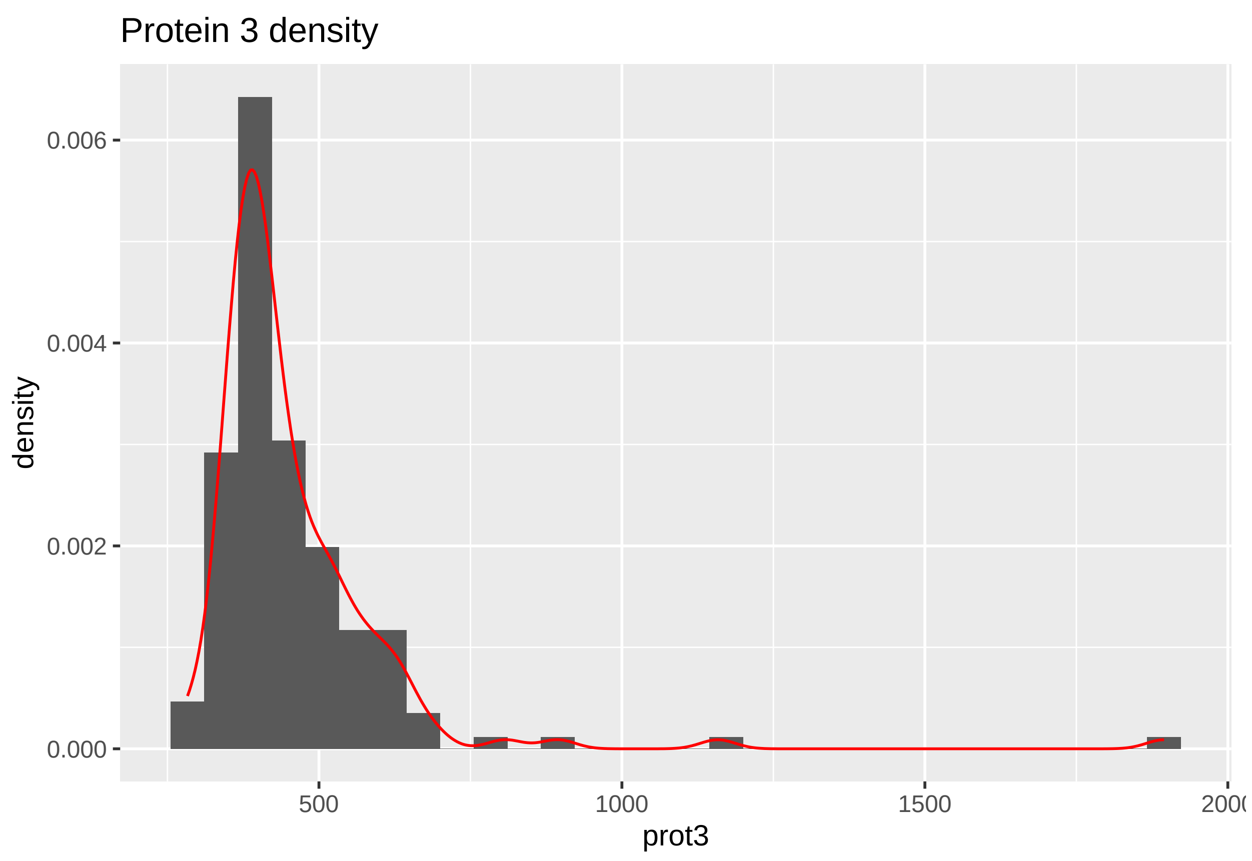
```r
library(ggplot2)
ggplot(as.data.frame(prot1), aes(x = prot1)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 1 density')
```



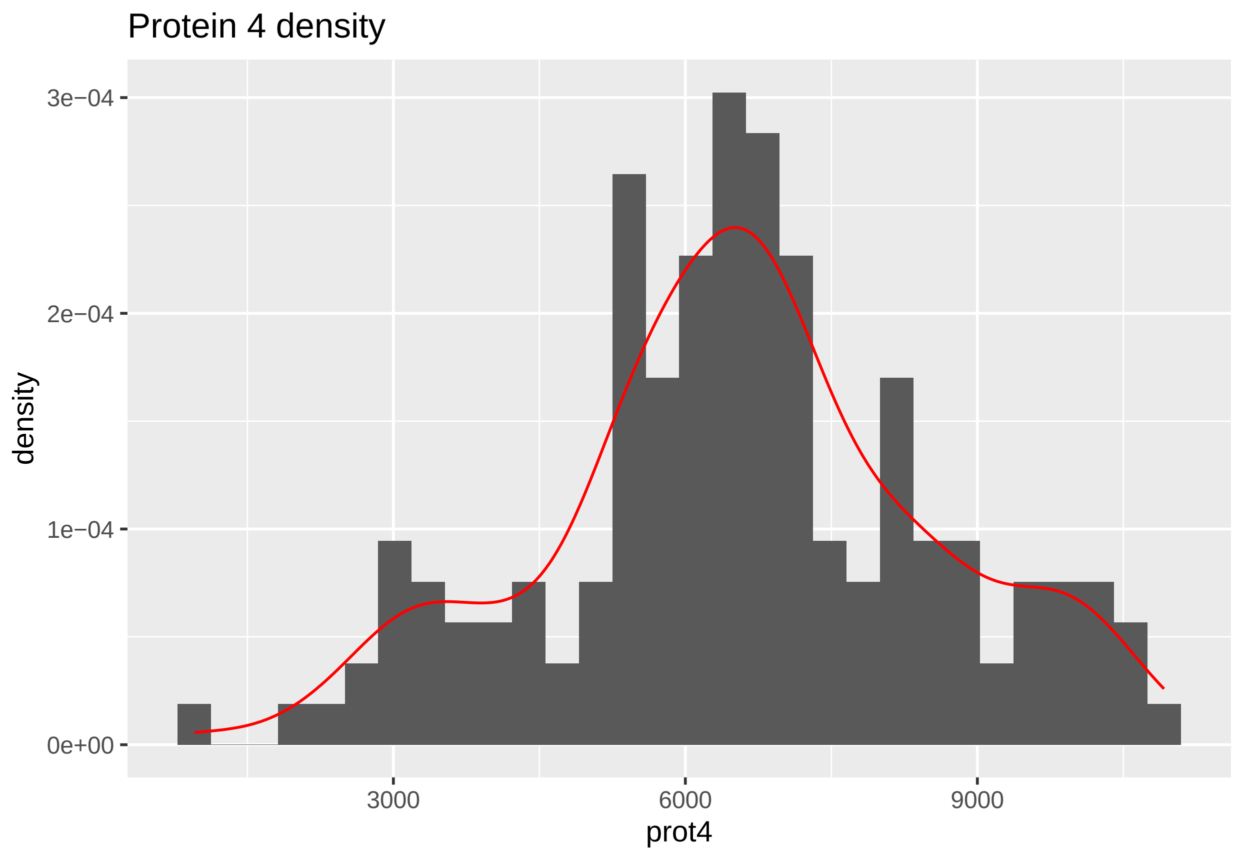
```r
ggplot(as.data.frame(prot2), aes(x = prot2)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 2 density')
```



```r
ggplot(as.data.frame(prot3), aes(x = prot3)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 3 density')
```



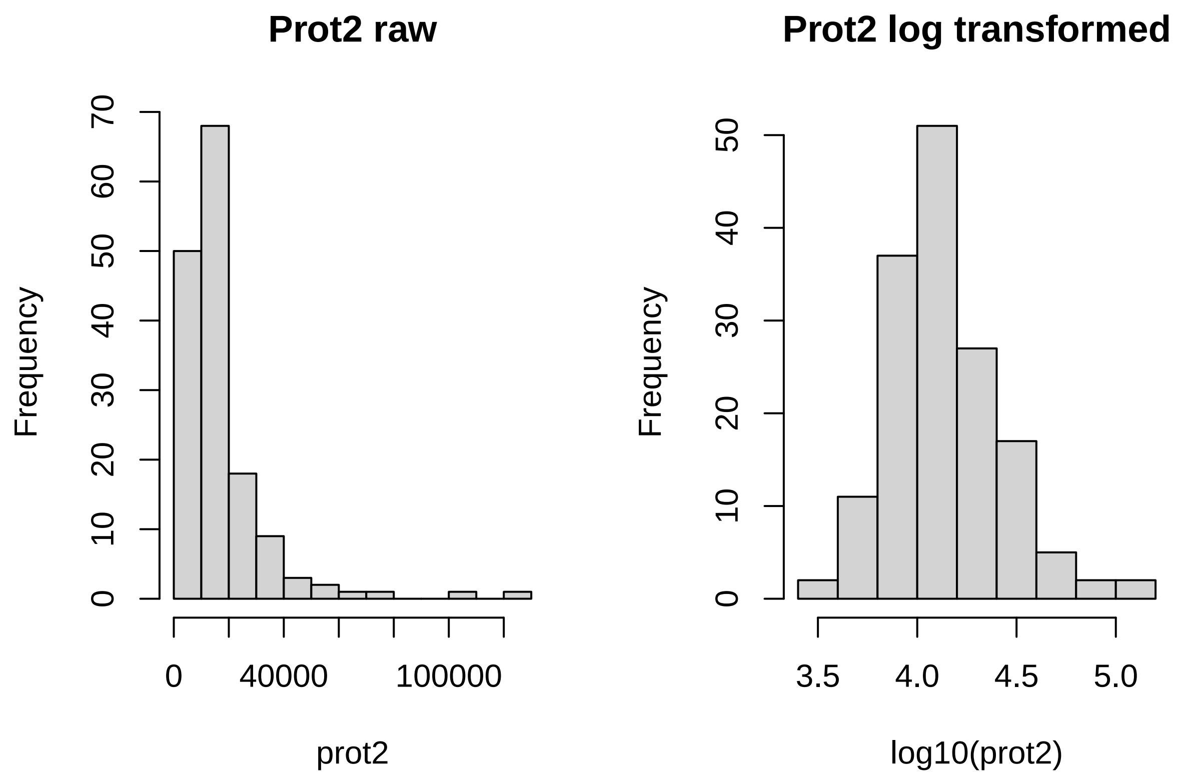
```r
ggplot(as.data.frame(prot4), aes(x = prot4)) +
  geom_histogram(aes(y=after_stat(density)), bins = 30) +
  geom_density(color="red") +
  ggtitle('Protein 4 density')
```



From the density plots, `prot2` looks the most skewed (strongly right-skewed) `prot3` looks slightly right-skewed as well. Both proteins contain large outliers, however the scale of `prot2`'s protein level is much higher, so the same follows for its outliers.

Comparison of `prot2` histogram to its log transformed counterpart:

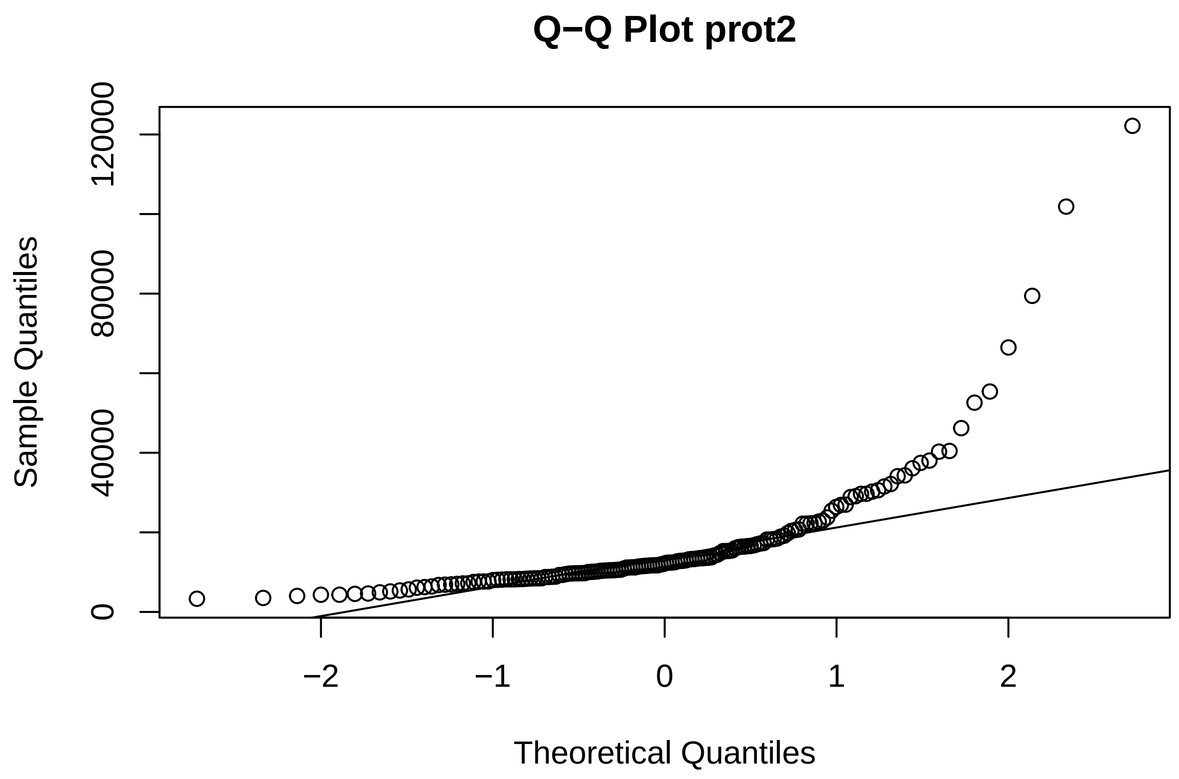
```r
par(mfrow=c(1,2))
hist(prot2, main="Prot2 raw")
hist(log10(prot2), main="Prot2 log transformed")
```



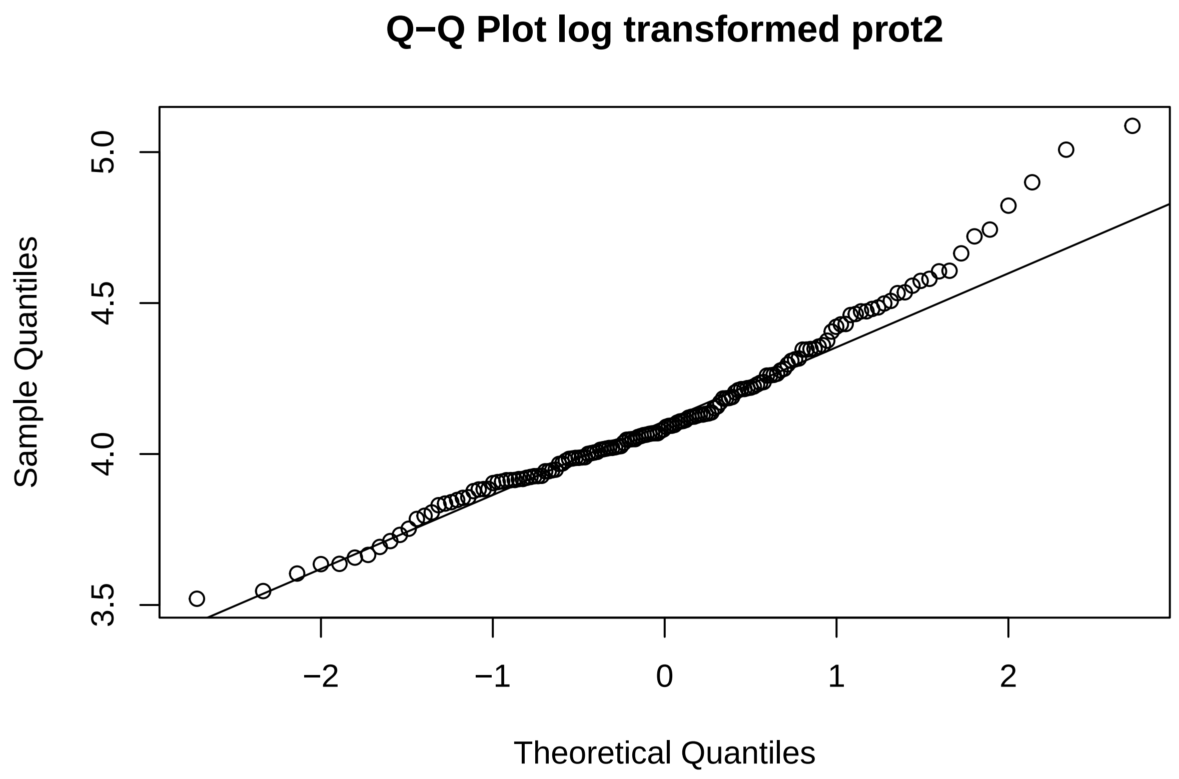
As we can see, after log transforming, the right-skewed `prot2` now appears more symmetric and of a much smaller and more readable scale.

We can check more rigorously to see how normality differs between the raw and log transformed `prot2` with a QQ-Plot.

```r
# normality check for raw data
qqnorm(prot2, main = 'Q-Q Plot prot2')
qqline(prot2)
```



```r
# normality check for log transformed data
qqnorm(log10(prot2), main = 'Q-Q Plot log transformed prot2')
qqline(log10(prot2))
```



After transforming `prot2`, the protein values appear slightly more normal, reducing the influential outliers. Overall, log transforming the protein levels acted as a way to reduce the large scale of the values, and make the data more symmetric.

4. Find a simpler panel that achieves comparable classification strategy using a different method

```r
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-10
```

```r
library(yardstick)
```

```
##
## Attaching package: 'yardstick'

## The following object is masked from 'package:readr':
##
##     spec
```

```
load(here("data", "biomarker-clean.RData"))

X <- biomarker_clean %>%
  select(-c(group, ados)) %>%
  as.matrix()

# turn outcome into a factor
y <- as.factor(ifelse(biomarker_clean$group == "ASD", 1, 0))

set.seed(123)
n <- nrow(X)
train_idx <- sample(1:n, size = floor(0.8 * n))
test_idx <- setdiff(1:n, train_idx)

X_train <- X[train_idx, ]
y_train <- y[train_idx]

X_test <- X[test_idx, ]
y_test <- y[test_idx]

cat("Training set size:", length(train_idx), "\n")
```

```
## Training set size: 123
```

```
cat("Test set size:", length(test_idx), "\n\n")
```

```
## Test set size: 31
```

```
# LASSO on training data

cv_fit <- cv.glmnet(
  x = X_train,
  y = y_train,
  family = "binomial",
  alpha = 1,  # LASSO
  nfolds = 10
)


lasso_coef <- coef(cv_fit, s = "lambda.1se")

selected_proteins <- data.frame(
  protein = rownames(lasso_coef),
  coefficient = as.vector(lasso_coef)
) %>%
```

```r
  filter(protein != "(Intercept)", coefficient != 0) %>%
  arrange(desc(abs(coefficient)))

cat("LASSO selected", nrow(selected_proteins), "proteins\n\n")
```

```
## LASSO selected 20 proteins
```

```r
# Get predicted probabilities
pred_prob <- predict(cv_fit,
                     newx = X_test,
                     s = "lambda.1se",
                     type = "response")[,1]

# Get predicted classes (threshold = 0.5)
pred_class <- as.factor(ifelse(pred_prob > 0.5, 1, 0))

# Performance
# Create data frame for yardstick
results_df <- tibble(
  truth = y_test,
  predicted = pred_class,
  prob_ASD = pred_prob
)

# Calculate metrics
metrics <- metric_set(accuracy, sensitivity, specificity, roc_auc)

performance <- results_df %>%
  metrics(truth = truth,
          estimate = predicted,
          prob_ASD,
          event_level = "second")  # ASD is the "positive" class

cat("Performance Metrics:\n")
```

```
## Performance Metrics:
```

```r
knitr::kable(performance)
```

.metric	.estimator	.estimate
accuracy	binary	0.5806452
sensitivity	binary	0.4444444
specificity	binary	0.7692308
roc_auc	binary	0.6666667

```r
# Extract individual metrics for display
acc <- performance %>% filter(.metric == "accuracy") %>% pull(.estimate)
sens <- performance %>% filter(.metric == "sensitivity") %>% pull(.estimate)
spec <- performance %>% filter(.metric == "specificity") %>% pull(.estimate)
auc <- performance %>% filter(.metric == "roc_auc") %>% pull(.estimate)
```

Evaluation metrics: In Hewitson et al. 2021, a logistic regression model was fit using their final panel of proteins, and they evaluated its performance using AUC, sensitivity, and specificity. To benchmark our method, we'll compare our results to the `inclass-analysis.R` results, where the metrics were

Their results were:

$$\text{roc auc} = 0.883 \text{ ; accuracy} = 0.774 \text{ ; sensitivity} = 0.812 \text{ ; specificity} = 0.733$$

Adding more benchmark results to `bahaar_analysis`:

```r
rmarkdown::render("bahaar_analysis.rmd")
```

```
##   |                                                              |
```

```
## /Applications/RStudio.app/Contents/Resources/app/quarto/bin/tools/aarch64/pandoc +RTS -K512m -RTS bal
```

```r
# Get confusion matrix object
cm <- confusionMatrix(pred_class_lasso, y_test, positive = "ASD")

# Print all metrics
cat("AUC ROC (LASSO classifier) on Test Set: ", auroc_final, "\n")
```

```
## AUC ROC (LASSO classifier) on Test Set:  0.8577778
```

```r
cat("Test Set Accuracy (LASSO classifier): ", cm$overall['Accuracy'], "\n")
```

```
## Test Set Accuracy (LASSO classifier):  0.7666667
```

```r
cat("Sensitivity (LASSO classifier): ", cm$byClass['Sensitivity'], "\n")
```

```
## Sensitivity (LASSO classifier):  0.8
```

```r
cat("Specificity (LASSO classifier): ", cm$byClass['Specificity'], "\n")
```

```
## Specificity (LASSO classifier):  0.7333333
```

Thus, AUC and accuracy are improved in our LASSO regression model (compared to 0.883 and 0.774 respectively), specificity remains the same (0.733), and sensitivity is 0.01 unit lower than that of the `inclass-analysis` (0.812).