SDS 4392 HW 1

Washington University, SP 2025

Aidan Kardan SDS 4392 HW 1

1. [5pts] Consider the dataset `attitude` and use `rating` as the response variable

Load Required Packages

```
library(faraway)
library(datasets)
library(visreg)
library(ggplot2)
library(psych)
```

```
##
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha

## The following object is masked from 'package:faraway':
##
##     logit
```

```
library(broom)
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
```

a) Create an initial data analysis that explores the numerical and graphical characteristics of the data. Be sure to look up the dataset to get a sense of what the variables mean.

```
df <- datasets::attitude
str(df)
```

```
## 'data.frame':    30 obs. of  7 variables:
##  $ rating    : num  43 63 71 61 81 43 58 71 72 67 ...
##  $ complaints: num  51 64 70 63 78 55 67 75 82 61 ...
##  $ privileges: num  30 51 68 45 56 49 42 50 72 45 ...
##  $ learning  : num  39 54 69 47 66 44 56 55 67 47 ...
##  $ raises    : num  61 63 76 54 71 54 66 70 71 62 ...
##  $ critical  : num  92 73 86 84 83 49 68 66 83 80 ...
##  $ advance   : num  45 47 48 35 47 34 35 41 31 41 ...
```

```
summary(df)
```

```
##      rating        complaints     privileges       learning         raises     
##  Min.   :40.00   Min.   :37.0   Min.   :30.00   Min.   :34.00   Min.   :43.00  
##  1st Qu.:58.75   1st Qu.:58.5   1st Qu.:45.00   1st Qu.:47.00   1st Qu.:58.25  
##  Median :65.50   Median :65.0   Median :51.50   Median :56.50   Median :63.50  
##  Mean   :64.63   Mean   :66.6   Mean   :53.13   Mean   :56.37   Mean   :64.63  
##  3rd Qu.:71.75   3rd Qu.:77.0   3rd Qu.:62.50   3rd Qu.:66.75   3rd Qu.:71.00  
##  Max.   :85.00   Max.   :90.0   Max.   :83.00   Max.   :75.00   Max.   :88.00  
##     critical        advance     
##  Min.   :49.00   Min.   :25.00  
##  1st Qu.:69.25   1st Qu.:35.00  
##  Median :77.50   Median :41.00  
##  Mean   :74.77   Mean   :42.93  
##  3rd Qu.:80.00   3rd Qu.:47.75  
##  Max.   :92.00   Max.   :72.00
```

```
colSums(is.na(df))
```

```
##     rating complaints privileges   learning     raises   critical    advance 
##          0          0          0          0          0          0          0
```

```
dim(df)
```

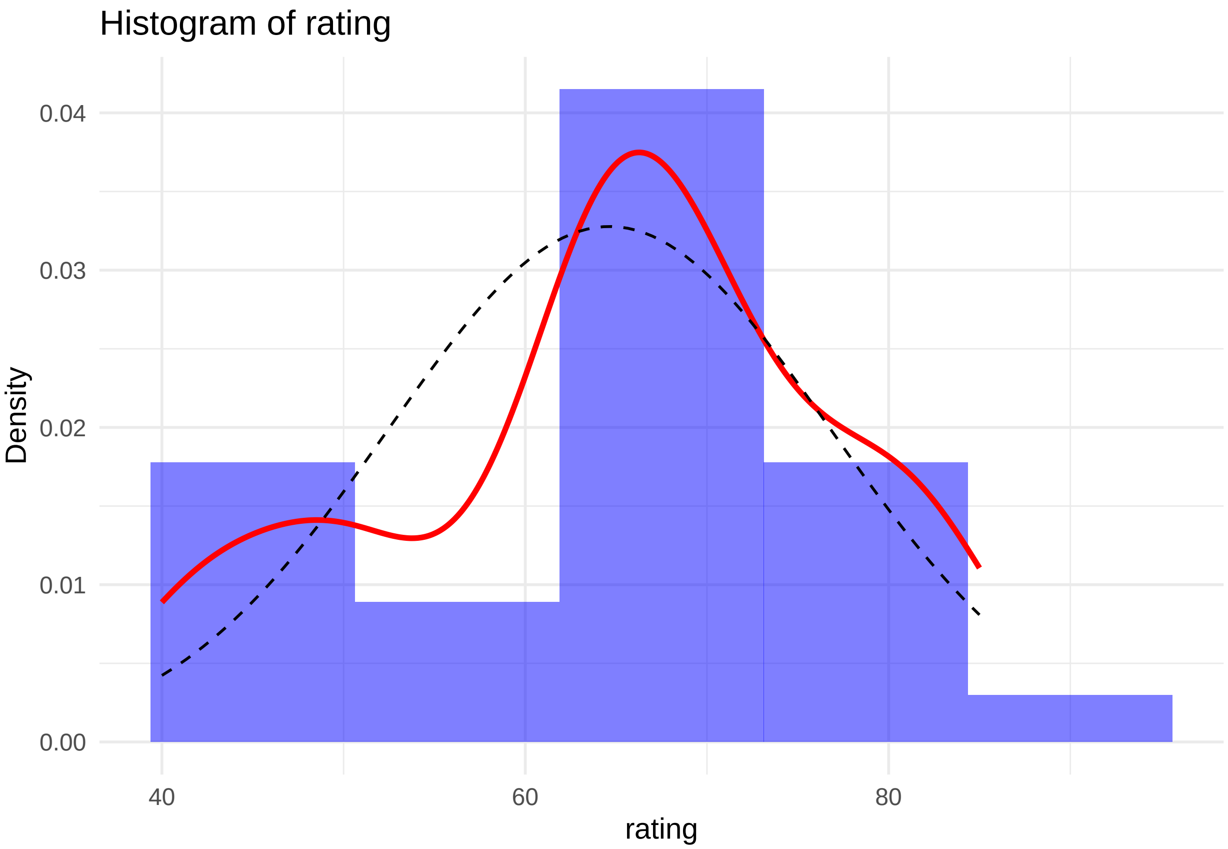
```
## [1] 30  7
```

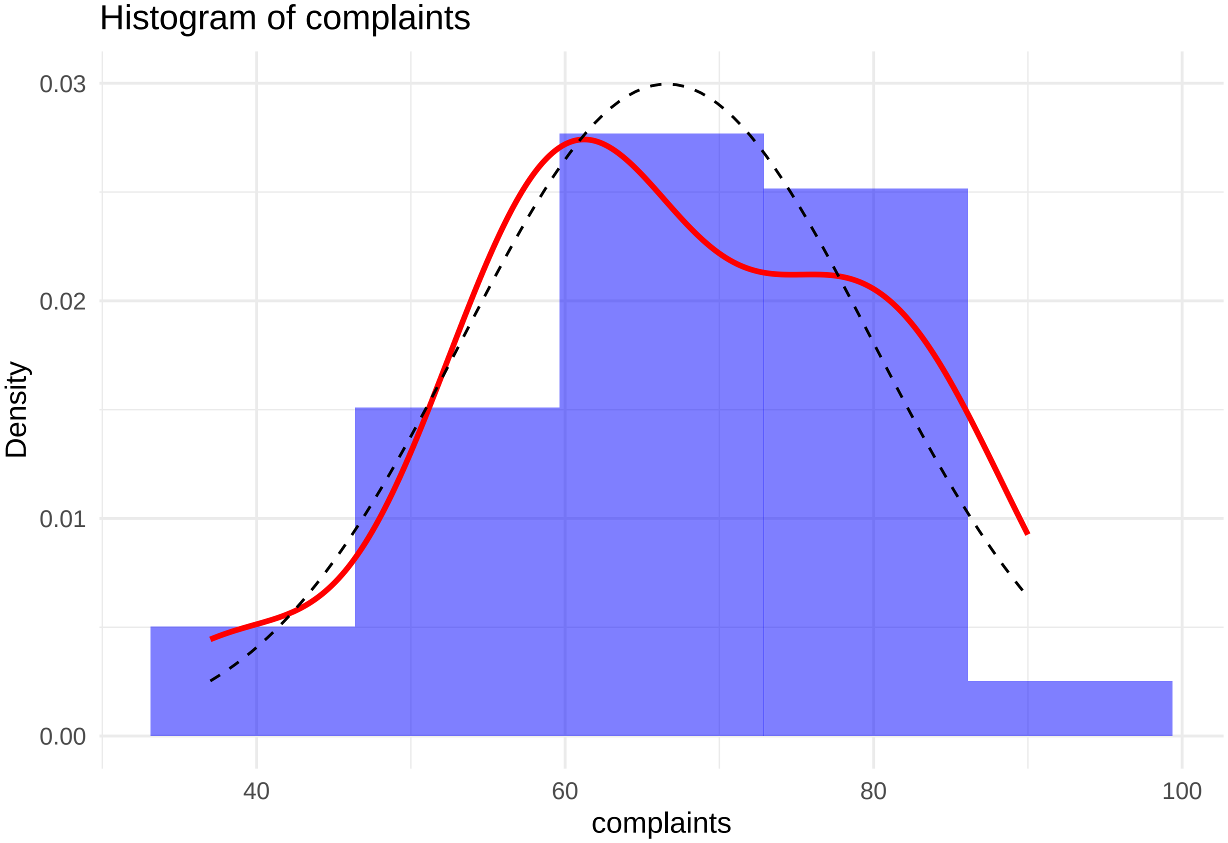
There is no missing data, there are 7 variables on interest, including the response and there are 30 observations.

```
# Function to plot histograms with normal distribution overlay
plot_histogram <- function(data, column) {
  ggplot(data, aes_string(x = column)) +
    geom_histogram(aes(y = after_stat(density)), bins = 5, fill = "blue", alpha = 0.5) +
    geom_density(color = "red", size = 1) +
    stat_function(fun = dnorm, args = list(mean = mean(data[[column]], na.rm = TRUE),
                                           sd = sd(data[[column]], na.rm = TRUE)),
                  color = "black", linetype = "dashed") +
    labs(title = paste("Histogram of", column), x = column, y = "Density") +
    theme_minimal()
}
df_columns <- colnames(df)
for (col in df_columns){
  print(plot_histogram(df, col))
}
```

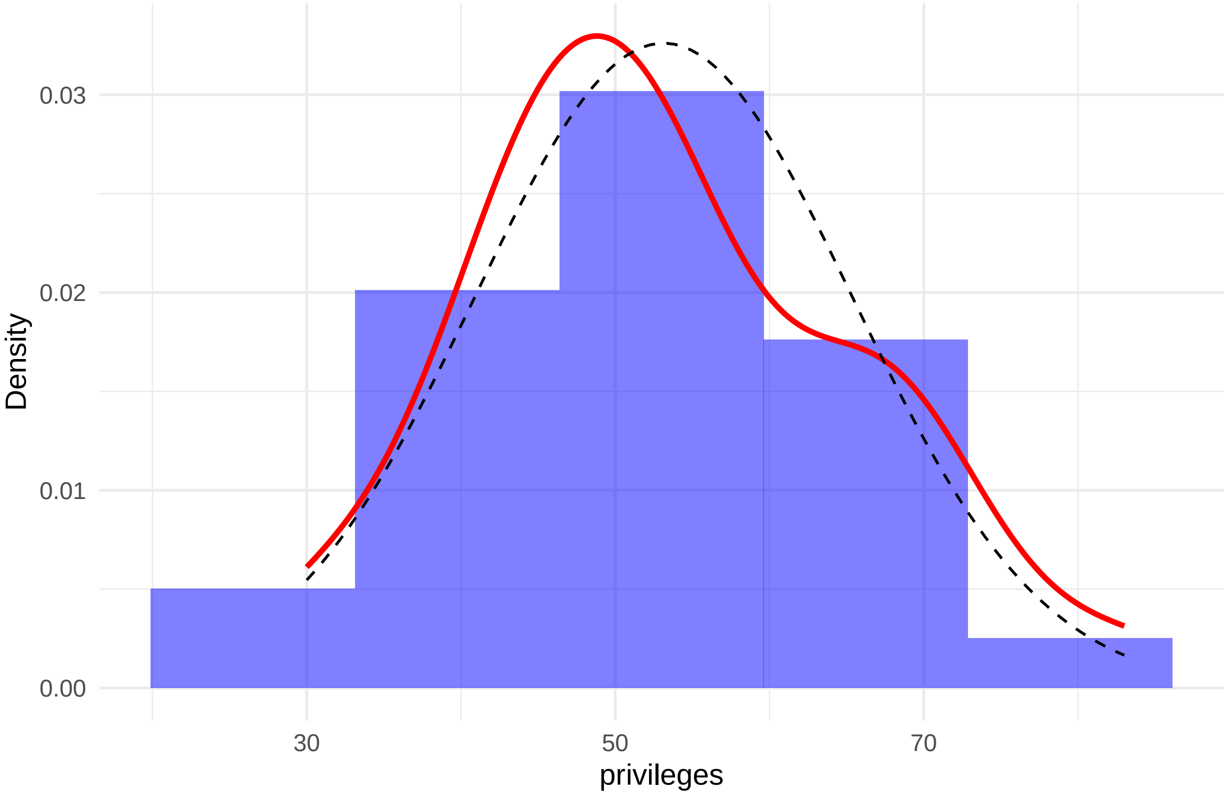
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

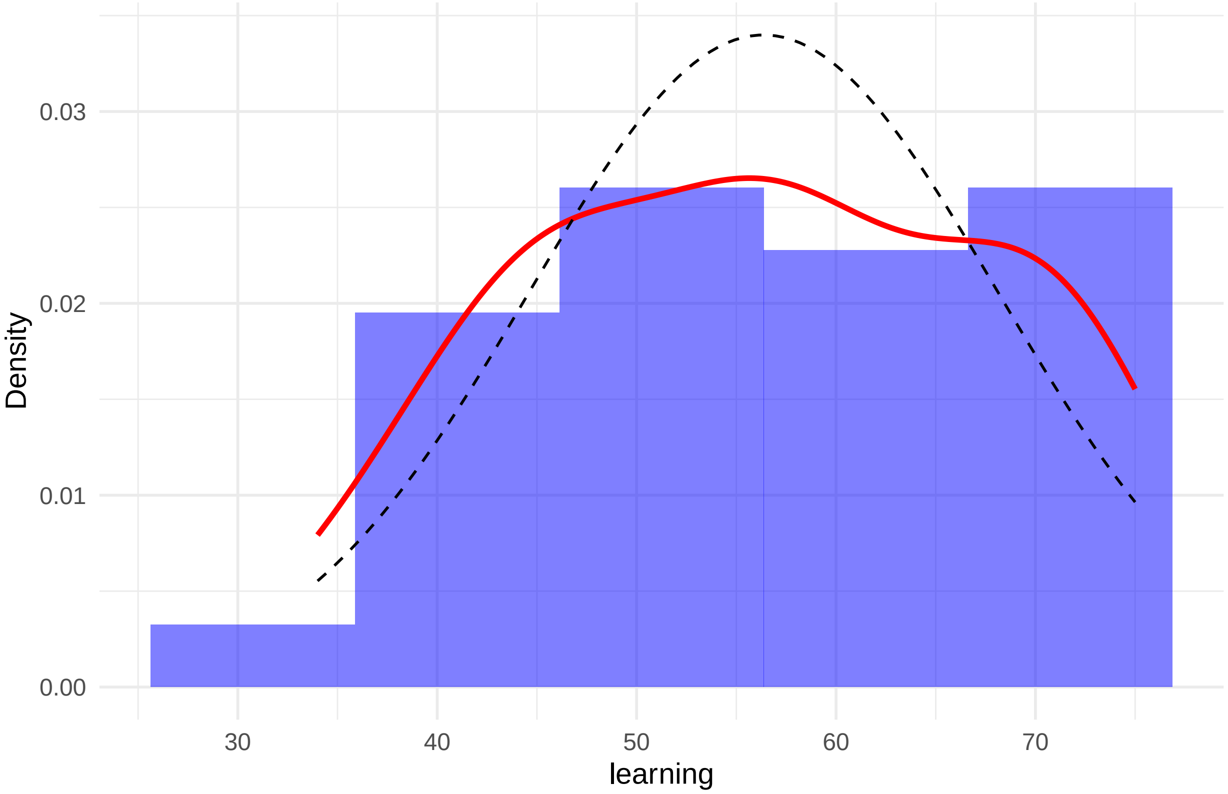

Histogram of rating
Density
0.04
0.03
0.02
0.01
0.00
40
60
80
rating

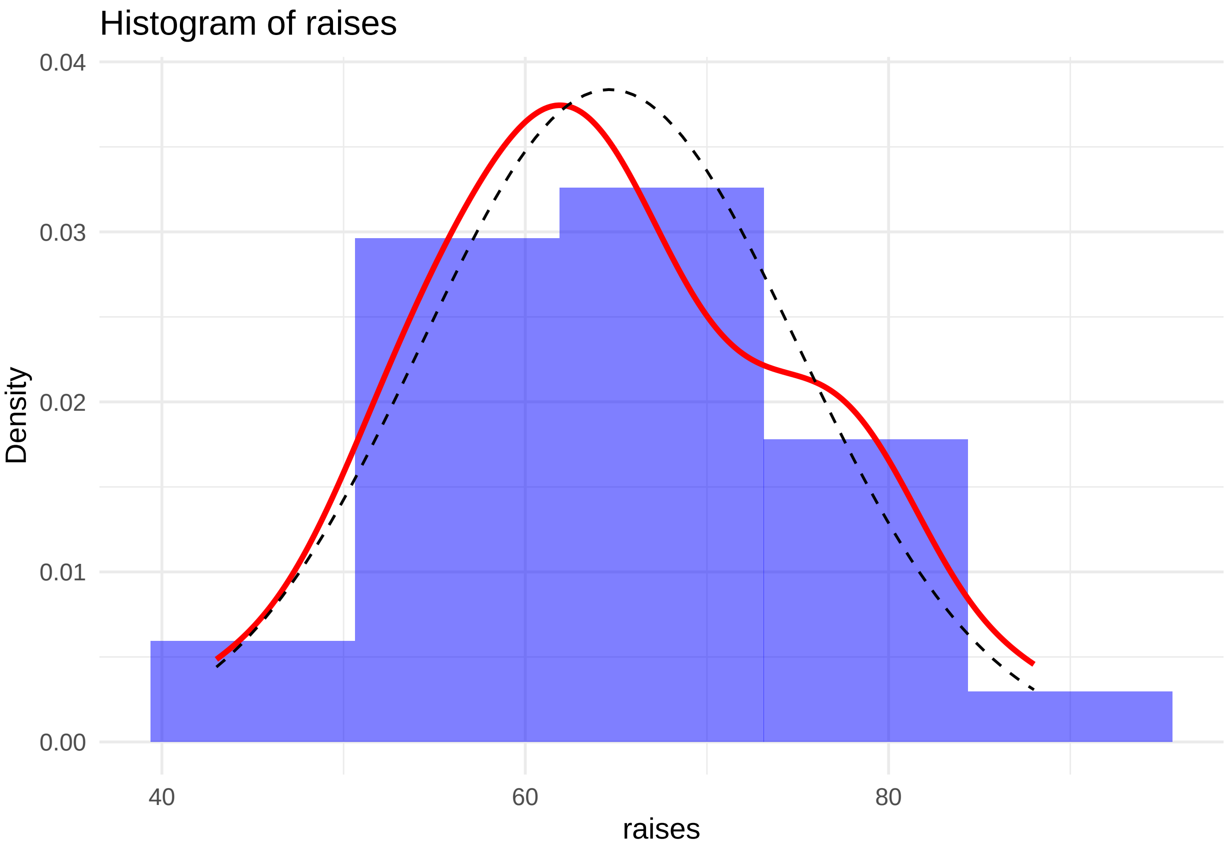

Histogram of complaints
Density
0.03
0.02
0.01
0.00
40
60
80
100
complaints

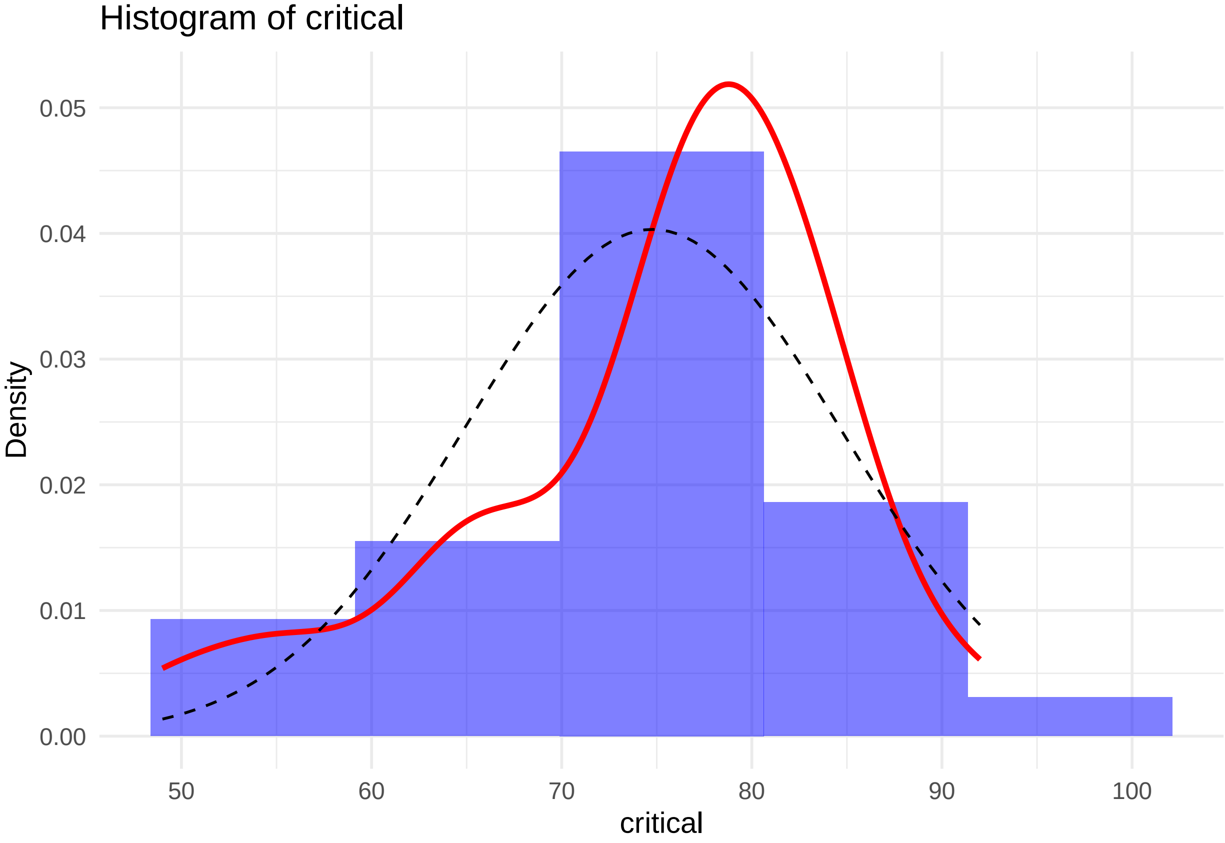
Histogram of privileges

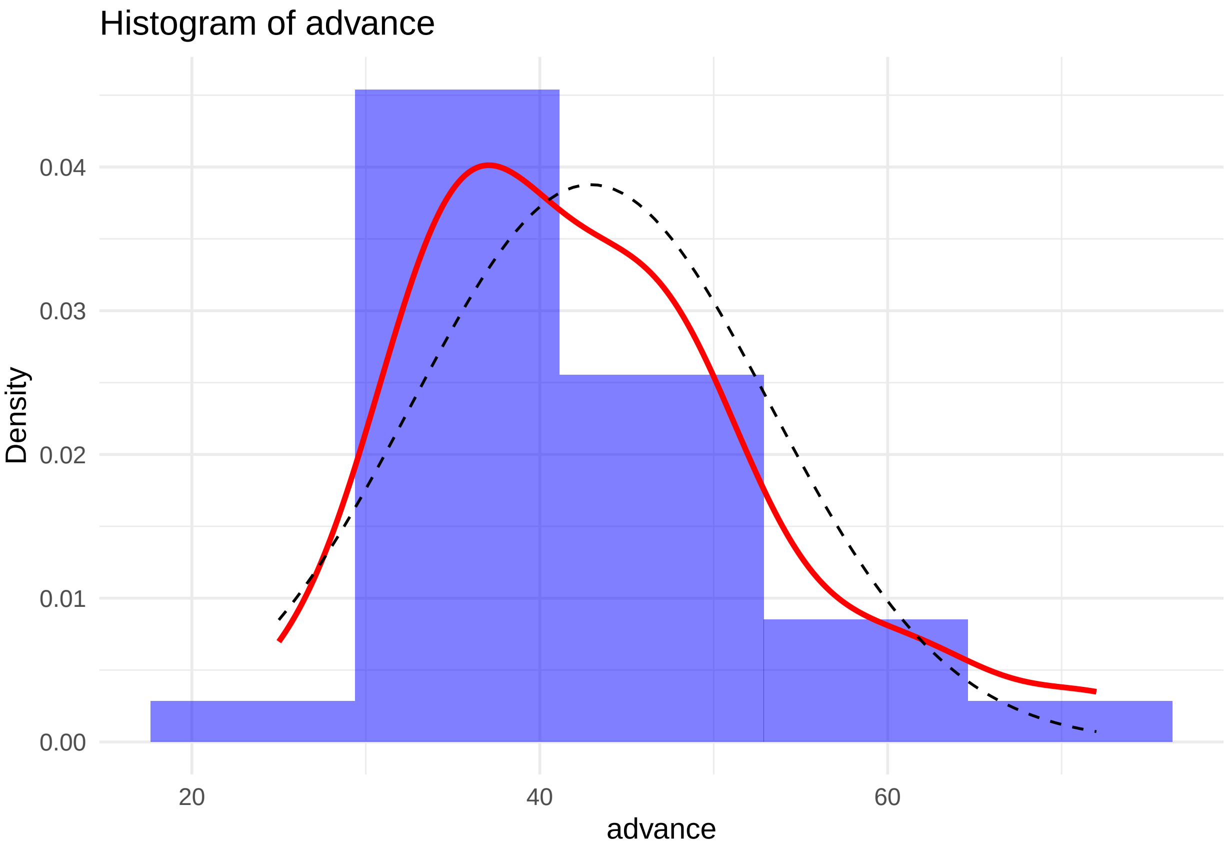


Histogram of learning




Histogram of raises
0.04
0.03
0.02
0.01
0.00
Density
40
60
80
raises


Histogram of critical
0.05
0.04
0.03
0.02
0.01
0.00
Density
50
60
70
80
90
100
critical

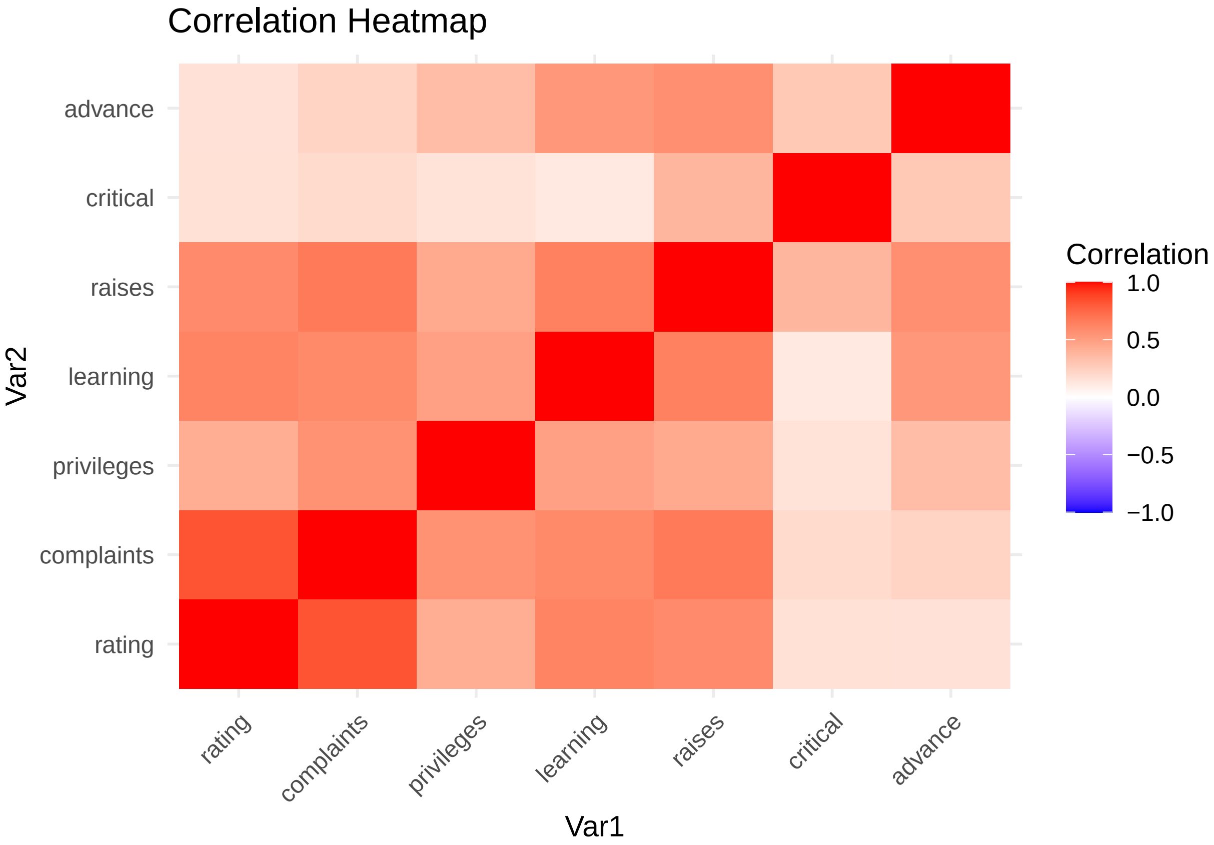


I chose a bin size of 5 for the KDE because there are a total of 30 observations for each variable. The goal of this exploratory analysis is to see how each variable in the data is distributed. In Statistical Analysis, depending on the method chosen, we may need the distribution of the response, or the distribution of the predictors to be normally distributed. This is a good way to not only visually inspect the data but get an understanding of how each variable is distributed. Based on the initial analysis, the response variable may need to be transformed.

```
library(reshape2)

# Compute correlation matrix
corr_matrix <- cor(df, use = "complete.obs")

# Print the correlation matrix
print(corr_matrix)
```

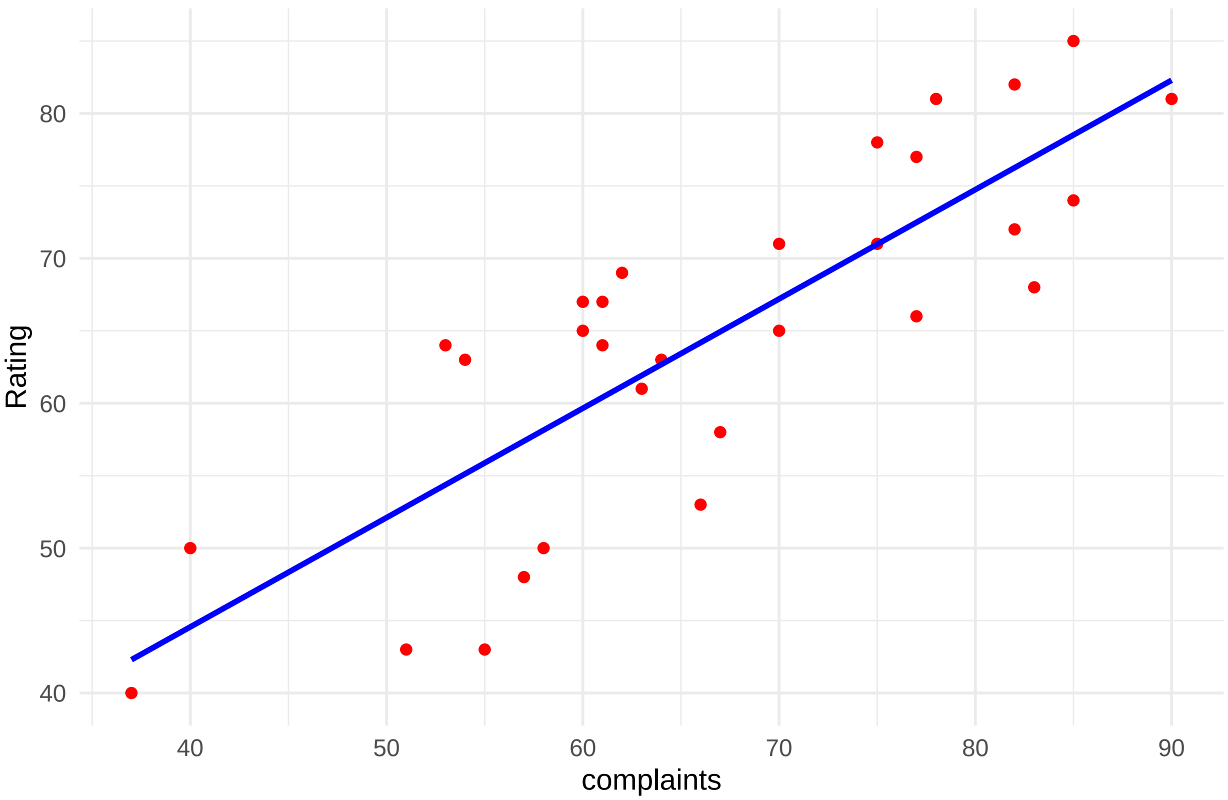
```
##               rating complaints privileges  learning    raises  critical
## rating     1.0000000  0.8254176  0.4261169 0.6236782 0.5901390 0.1564392
## complaints 0.8254176  1.0000000  0.5582882 0.5967358 0.6691975 0.1877143
## privileges 0.4261169  0.5582882  1.0000000 0.4933310 0.4454779 0.1472331
## learning   0.6236782  0.5967358  0.4933310 1.0000000 0.6403144 0.1159652
## raises     0.5901390  0.6691975  0.4454779 0.6403144 1.0000000 0.3768830
## critical   0.1564392  0.1877143  0.1472331 0.1159652 0.3768830 1.0000000
## advance    0.1550863  0.2245796  0.3432934 0.5316198 0.5741862 0.2833432
##              advance
## rating     0.1550863
## complaints 0.2245796
## privileges 0.3432934
## learning   0.5316198
## raises     0.5741862
## critical   0.2833432
## advance    1.0000000
```

```
corr_melted <- melt(corr_matrix)
# Plot heatmap
ggplot(data = corr_melted,
       aes(x = Var1, y = Var2, fill = value)) +   geom_tile() +   scale_fill_gradient2(low = "blue", hig
       limit = c(-1,1), space = "Lab", name="Correlation") +   theme_minimal() +   labs(title = "Correla
```

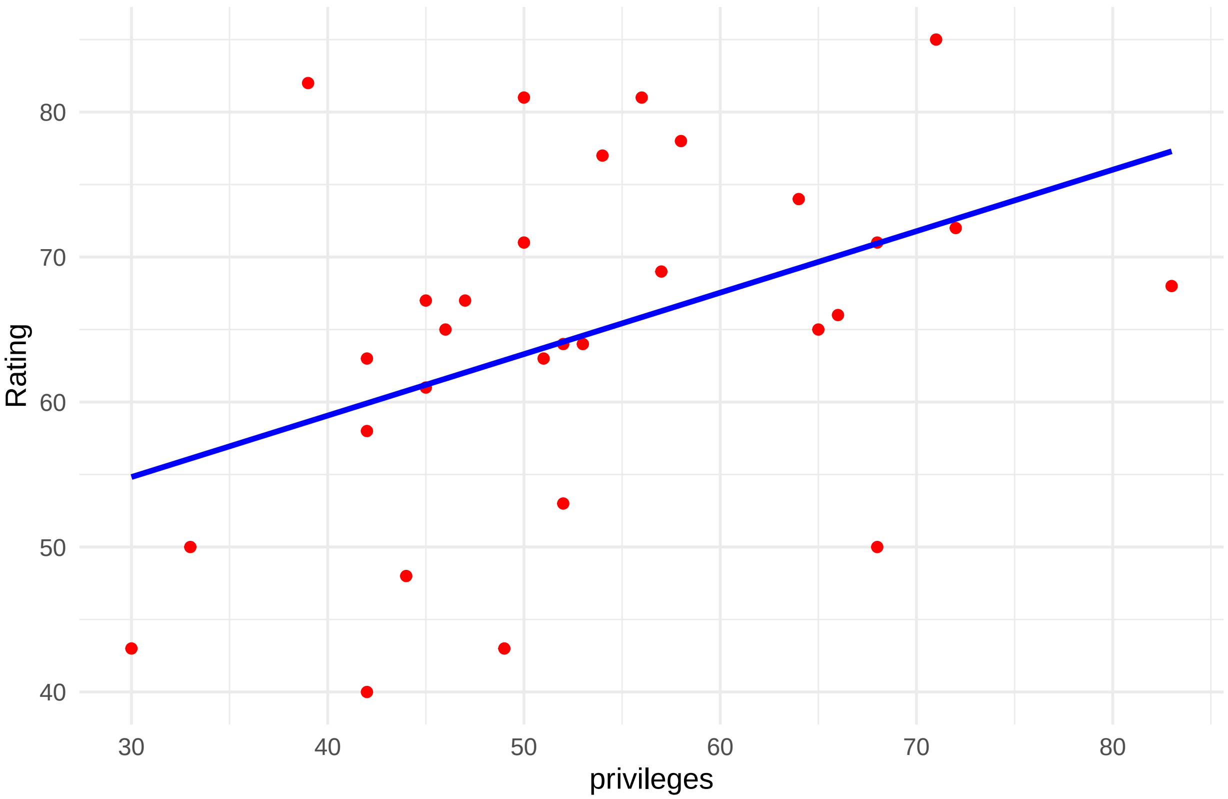


Based on the correlation heap map, complaints has the highest positive correlation to the response, rating. All variables are positively correlated to the response. No predictor variable seems to be highly correlated with one another.

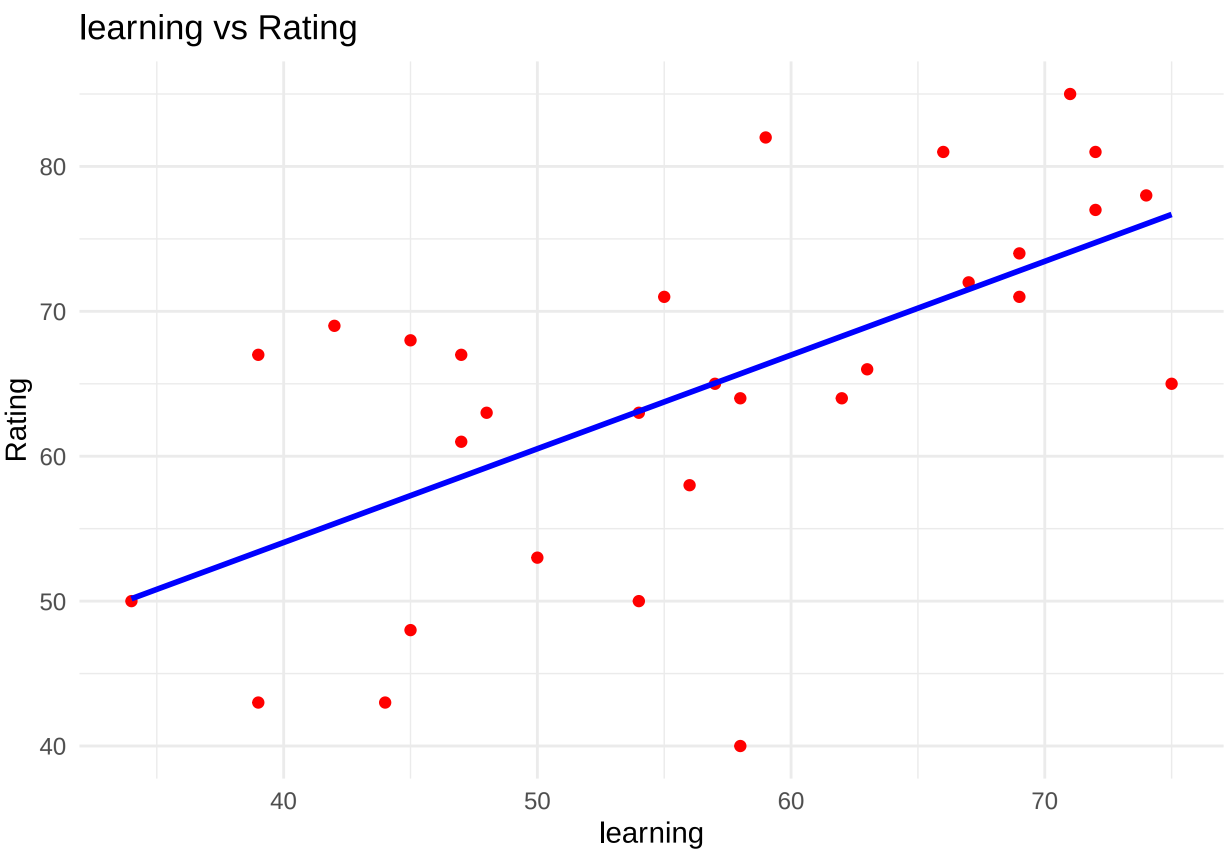
```
library(ggplot2)

predictors <- setdiff(names(df), "rating")

for (var in predictors) {
  print(
    ggplot(df, aes_string(x = var, y = "rating")) +
      geom_point(color = "red") +  # Scatter plot points
      geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "blue") +  # Regression line
      labs(title = paste(var, "vs Rating"), x = var, y = "Rating") +
      theme_minimal()
  )
}
```



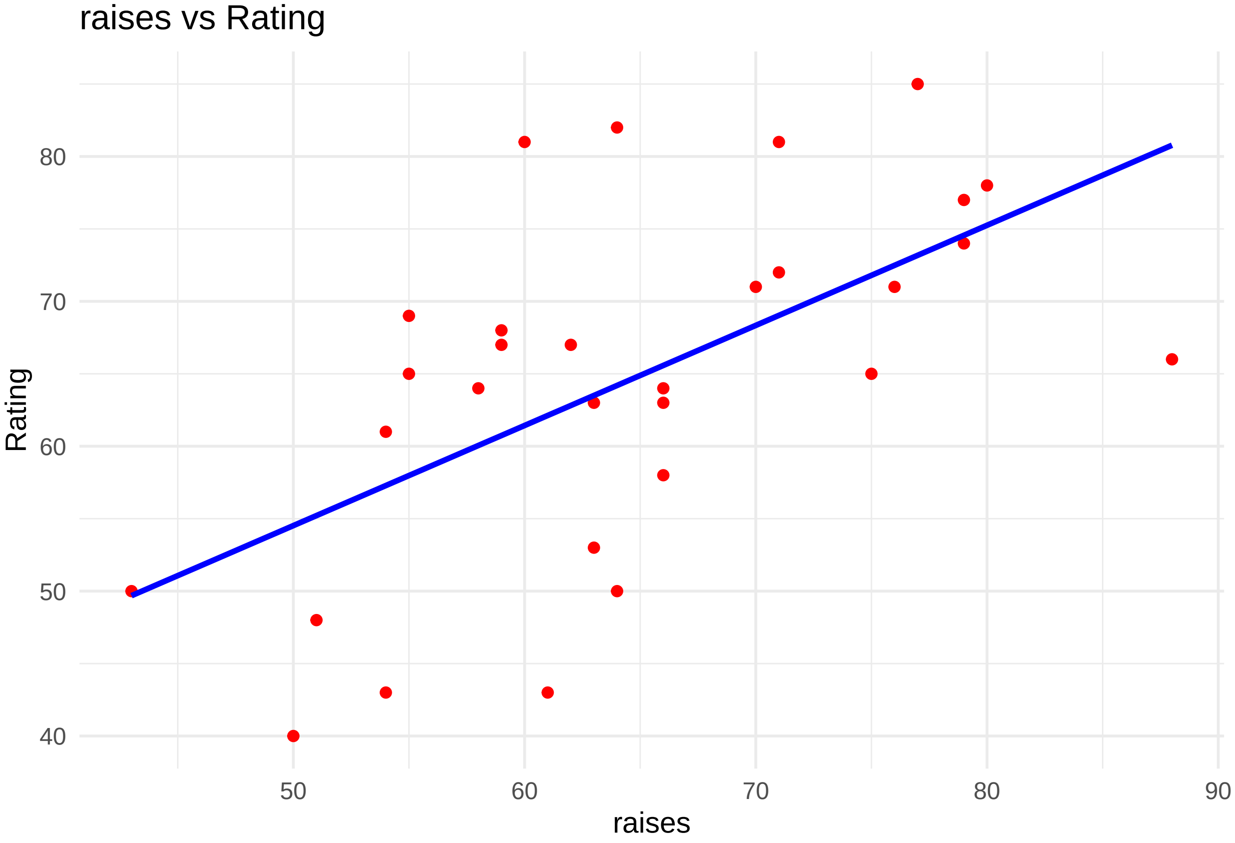
complaints vs Rating



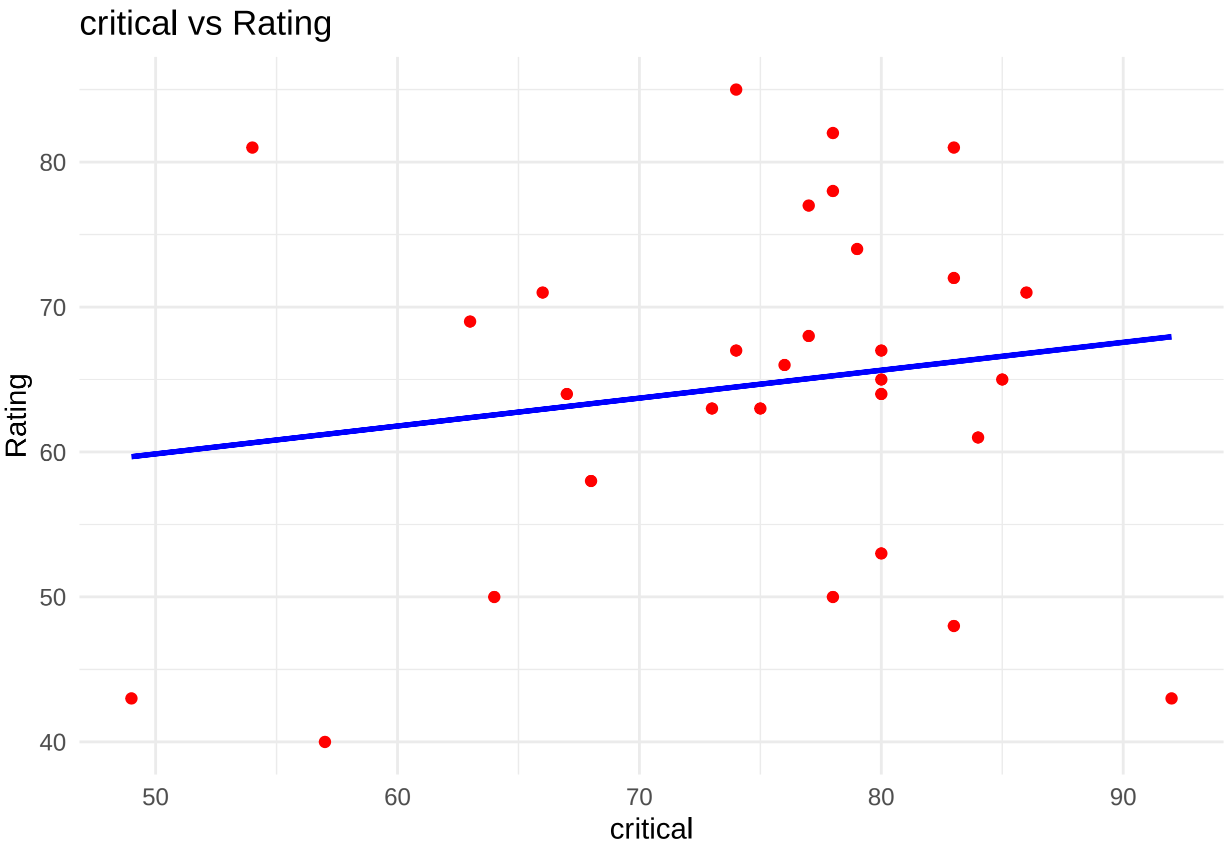
privileges vs Rating

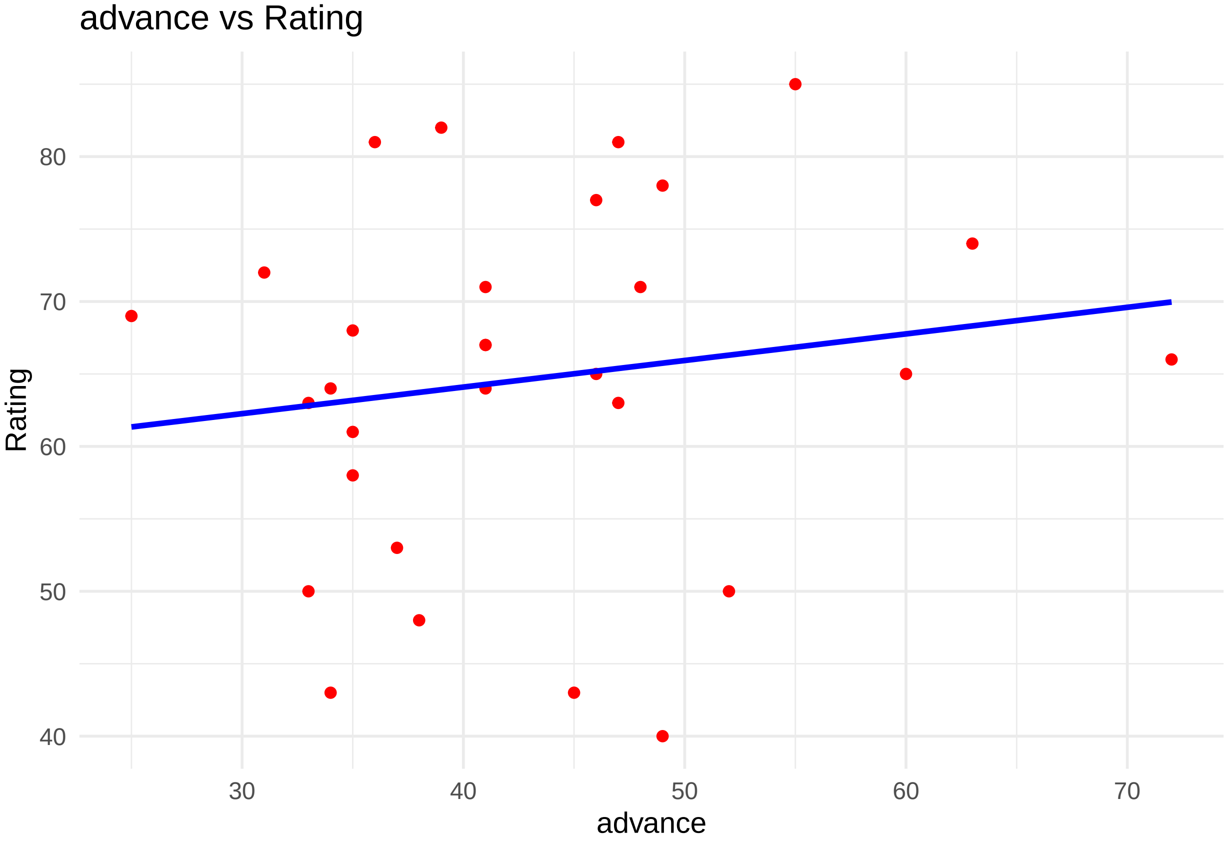


learning vs Rating



raises vs Rating





Based on the scatterplot analysis, we can get a better idea of the relationship between the response and each variable. The response and complaints seem to be quite linearly positively related.

b) Use variable selection to choose the best model (hint: look up the `step()` function)

```
library(MASS)  # Load MASS for stepwise regression
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
# Define full and null models
full_model <- lm(rating ~ ., data = df)  # Model with all predictors
null_model <- lm(rating ~ 1, data = df)  # Model with only intercept

# Perform backward stepwise selection
cat("\nPerforming Backward Stepwise Selection:\n")
```

```
##
## Performing Backward Stepwise Selection:
```

```
backward_model <- step(full_model, direction = "backward", trace = TRUE)
```

```
## Start:  AIC=123.36
## rating ~ complaints + privileges + learning + raises + critical +
##     advance
##
##              Df Sum of Sq    RSS    AIC
## - critical    1      3.41 1152.4 121.45
## - raises      1      6.80 1155.8 121.54
## - privileges  1     14.47 1163.5 121.74
## - advance     1     74.11 1223.1 123.24
## <none>                    1149.0 123.36
## - learning    1    180.50 1329.5 125.74
## - complaints  1    724.80 1873.8 136.04
##
## Step:  AIC=121.45
## rating ~ complaints + privileges + learning + raises + advance
##
##              Df Sum of Sq    RSS    AIC
## - raises      1     10.61 1163.0 119.73
## - privileges  1     14.16 1166.6 119.82
## - advance     1     71.27 1223.7 121.25
## <none>                    1152.4 121.45
## - learning    1    177.74 1330.1 123.75
## - complaints  1    724.70 1877.1 134.09
##
## Step:  AIC=119.73
## rating ~ complaints + privileges + learning + advance
##
##              Df Sum of Sq    RSS    AIC
## - privileges  1     16.10 1179.1 118.14
## - advance     1     61.60 1224.6 119.28
## <none>                    1163.0 119.73
## - learning    1    197.03 1360.0 122.42
## - complaints  1   1165.94 2328.9 138.56
##
## Step:  AIC=118.14
## rating ~ complaints + learning + advance
##
##              Df Sum of Sq    RSS    AIC
## - advance     1     75.54 1254.7 118.00
## <none>                    1179.1 118.14
## - learning    1    186.12 1365.2 120.54
## - complaints  1   1259.91 2439.0 137.94
##
## Step:  AIC=118
## rating ~ complaints + learning
##
##              Df Sum of Sq    RSS    AIC
## <none>                    1254.7 118.00
## - learning    1    114.73 1369.4 118.63
## - complaints  1   1370.91 2625.6 138.16
```

```
# Perform forward stepwise selection
cat("\nPerforming Forward Stepwise Selection:\n")
```

```
##
## Performing Forward Stepwise Selection:
```

```
forward_model <- step(null_model, scope = formula(full_model), direction = "forward", trace = TRUE)
```

```
## Start:  AIC=150.93
## rating ~ 1
##
##              Df Sum of Sq    RSS    AIC
## + complaints  1   2927.58 1369.4 118.63
## + learning    1   1671.41 2625.6 138.16
## + raises      1   1496.48 2800.5 140.09
## + privileges  1    780.22 3516.7 146.92
## <none>                    4297.0 150.93
## + critical    1    105.16 4191.8 152.19
## + advance     1    103.35 4193.6 152.20
##
## Step:  AIC=118.63
## rating ~ complaints
##
##              Df Sum of Sq    RSS    AIC
## + learning    1   114.733 1254.7 118.00
## <none>                    1369.4 118.63
## + raises      1    11.102 1358.3 120.38
## + privileges  1     7.519 1361.9 120.46
## + advance     1     4.151 1365.2 120.54
## + critical    1     0.010 1369.4 120.63
##
## Step:  AIC=118
## rating ~ complaints + learning
##
##              Df Sum of Sq    RSS    AIC
## <none>                    1254.7 118.00
## + advance     1    75.540 1179.1 118.14
## + privileges  1    30.033 1224.6 119.28
## + raises      1     1.188 1253.5 119.97
## + critical    1     0.002 1254.7 120.00
```

```r
# Perform stepwise selection (both directions)
cat("\nPerforming Stepwise Selection (Both Directions):\n")
```

```
##
## Performing Stepwise Selection (Both Directions):
```

```r
stepwise_model <- step(null_model, scope = formula(full_model), direction = "both", trace = TRUE)
```

```
## Start:  AIC=150.93
## rating ~ 1
##
##              Df Sum of Sq    RSS    AIC
## + complaints  1   2927.58 1369.4 118.63
## + learning    1   1671.41 2625.6 138.16
## + raises      1   1496.48 2800.5 140.09
## + privileges  1    780.22 3516.7 146.92
## <none>                    4297.0 150.93
## + critical    1    105.16 4191.8 152.19
## + advance     1    103.35 4193.6 152.20
##
## Step:  AIC=118.63
## rating ~ complaints
##
##              Df Sum of Sq    RSS    AIC
## + learning    1    114.73 1254.6 118.00
## <none>                    1369.4 118.63
## + raises      1     11.10 1358.3 120.38
## + privileges  1      7.52 1361.9 120.46
## + advance     1      4.15 1365.2 120.54
## + critical    1      0.01 1369.4 120.63
## - complaints  1   2927.58 4297.0 150.93
##
## Step:  AIC=118
## rating ~ complaints + learning
##
##              Df Sum of Sq    RSS    AIC
## <none>                    1254.7 118.00
## + advance     1     75.54 1179.1 118.14
## - learning    1    114.73 1369.4 118.63
## + privileges  1     30.03 1224.6 119.28
## + raises      1      1.19 1253.5 119.97
## + critical    1      0.00 1254.7 120.00
## - complaints  1   1370.91 2625.6 138.16
```

```r
# Compare AIC values
aic_values <- data.frame(
  Model = c("Backward", "Forward", "Stepwise"),
  AIC = c(AIC(backward_model), AIC(forward_model), AIC(stepwise_model))
)

# Print AIC comparison
```

```r
print(aic_values)
```

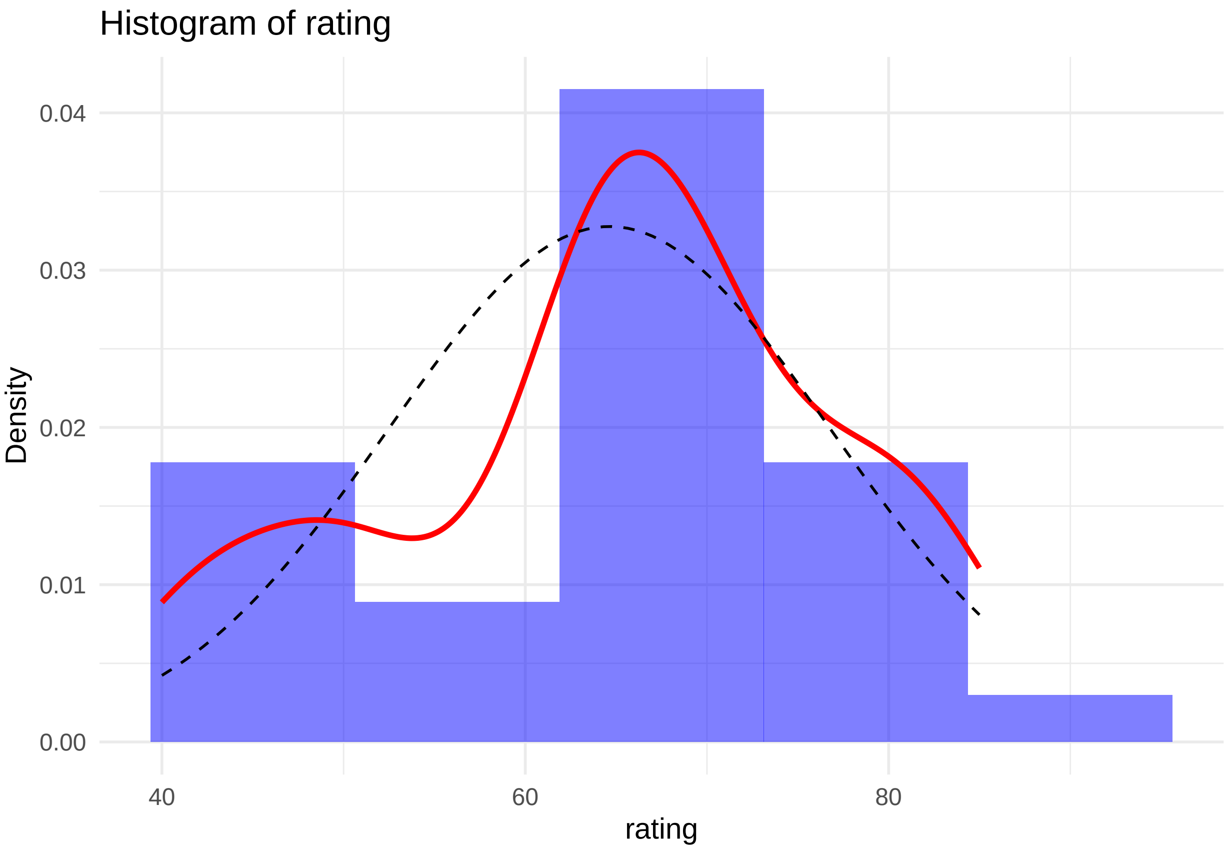
```
##      Model      AIC
## 1 Backward 205.1387
## 2  Forward 205.1387
## 3 Stepwise 205.1387
```

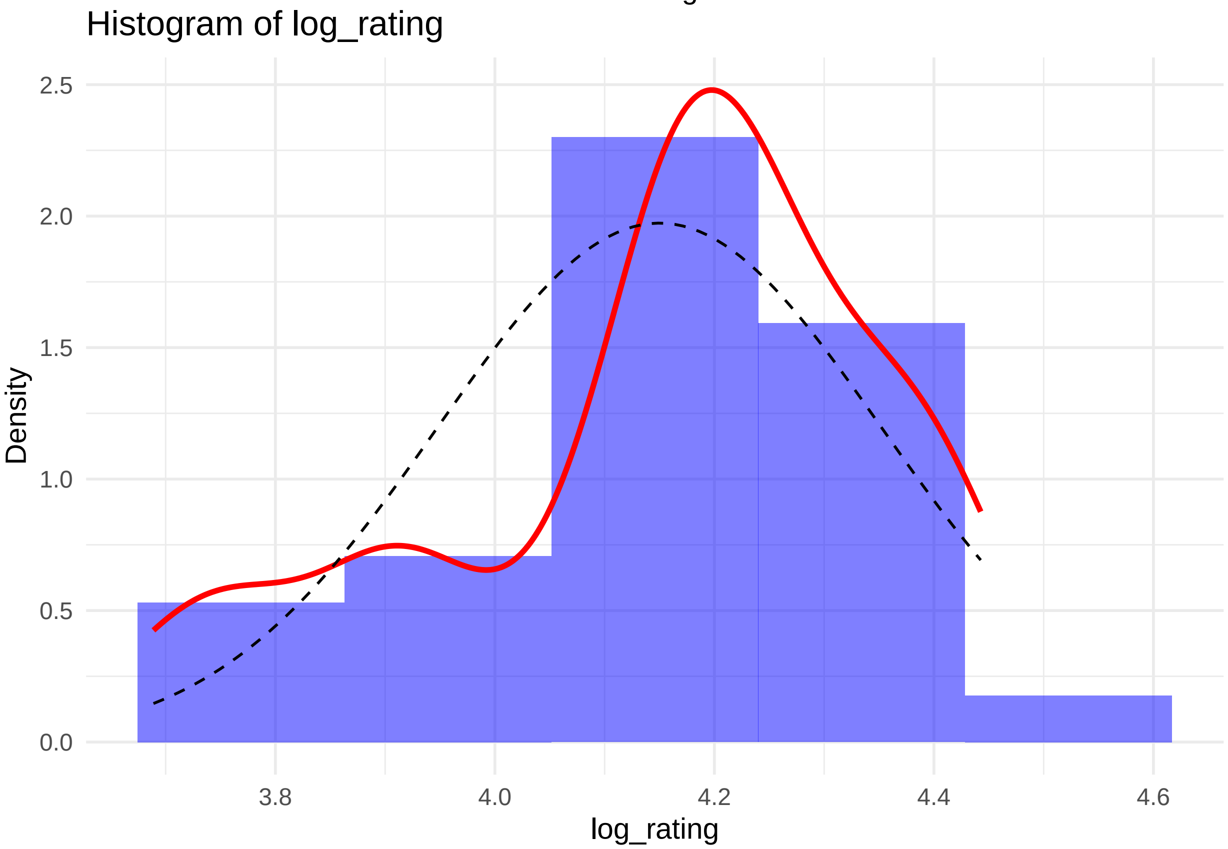
Interestingly, Backward, Forward and Step-wise (Both Directions) all produce the same results for the best model. This is not always the case, and often because of confounding variables, backward selection is not the same as forward selection. Furthermore, the data set is small, 30 observations so it is less likely the methods produce different best models.

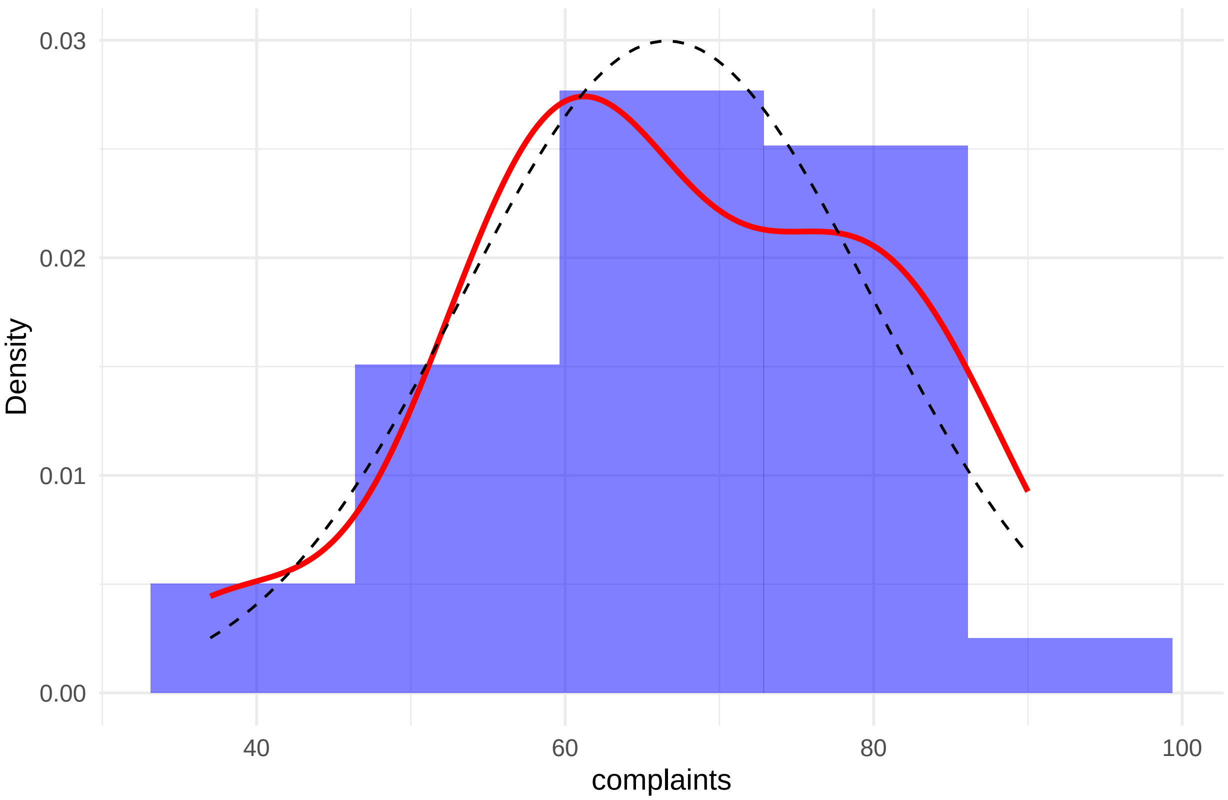
The best model based on AIC criterion and step-wise selection is the model with complaints and learning. This is the best model for backward, forward and step-wise selection.

c) Explore any transformations to improve the fit of the model.

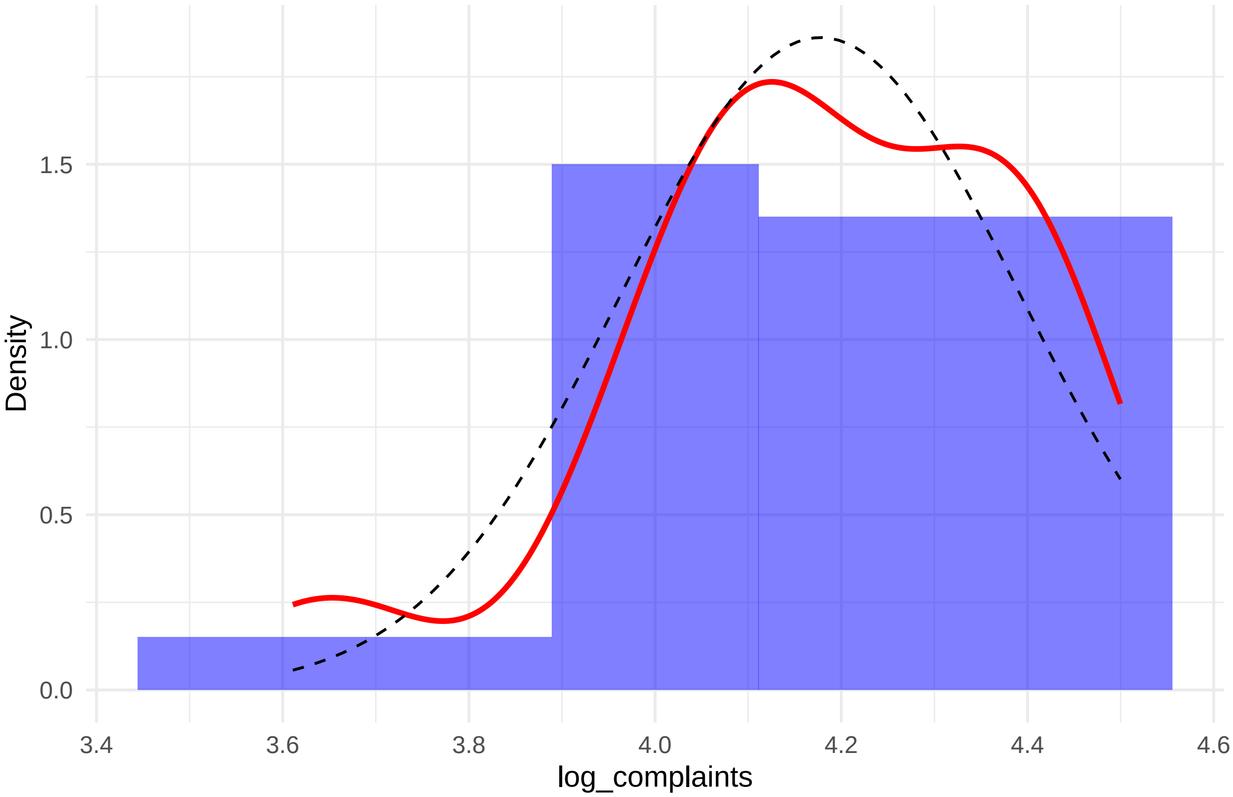
```r
library(ggplot2)

# Manually create log-transformed variables
df$log_rating <- log(df$rating)
df$log_complaints <- log(df$complaints)
df$log_privileges <- log(df$privileges)
df$log_learning <- log(df$learning)
df$log_raises <- log(df$raises)
df$log_advance <- log(df$advance)

# Function to plot histograms
plot_histogram <- function(data, column) {
  ggplot(data, aes_string(x = column)) +
    geom_histogram(aes(y = after_stat(density)), bins = 5, fill = "blue", alpha = 0.5) +
    geom_density(color = "red", size = 1) +
    stat_function(fun = dnorm, args = list(mean = mean(data[[column]], na.rm = TRUE),
                                           sd = sd(data[[column]], na.rm = TRUE)),
                  color = "black", linetype = "dashed") +
    labs(title = paste("Histogram of", column), x = column, y = "Density") +
    theme_minimal()
}

# List of all original and log-transformed variable names
columns_to_plot <- c("rating", "log_rating", "complaints", "log_complaints",
                     "privileges", "log_privileges", "learning", "log_learning",
                     "raises", "log_raises", "advance", "log_advance")

# Loop through all variables and plot
for (col in columns_to_plot) {
  print(plot_histogram(df, col))
}
```


Histogram of rating
Density
0.04
0.03
0.02
0.01
0.00
40
60
80
rating

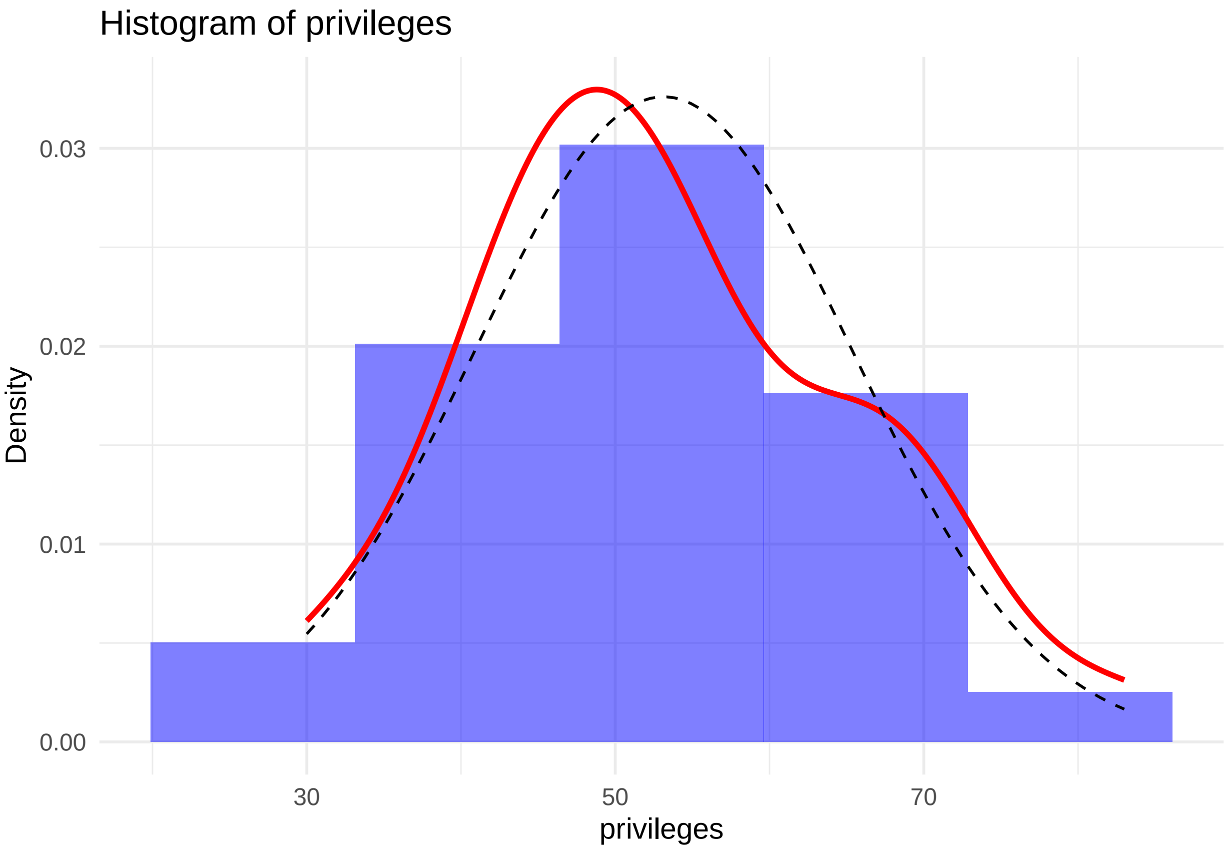

Histogram of log_rating
Density
2.5
2.0
1.5
1.0
0.5
0.0
3.8
4.0
4.2
4.4
4.6
log_rating



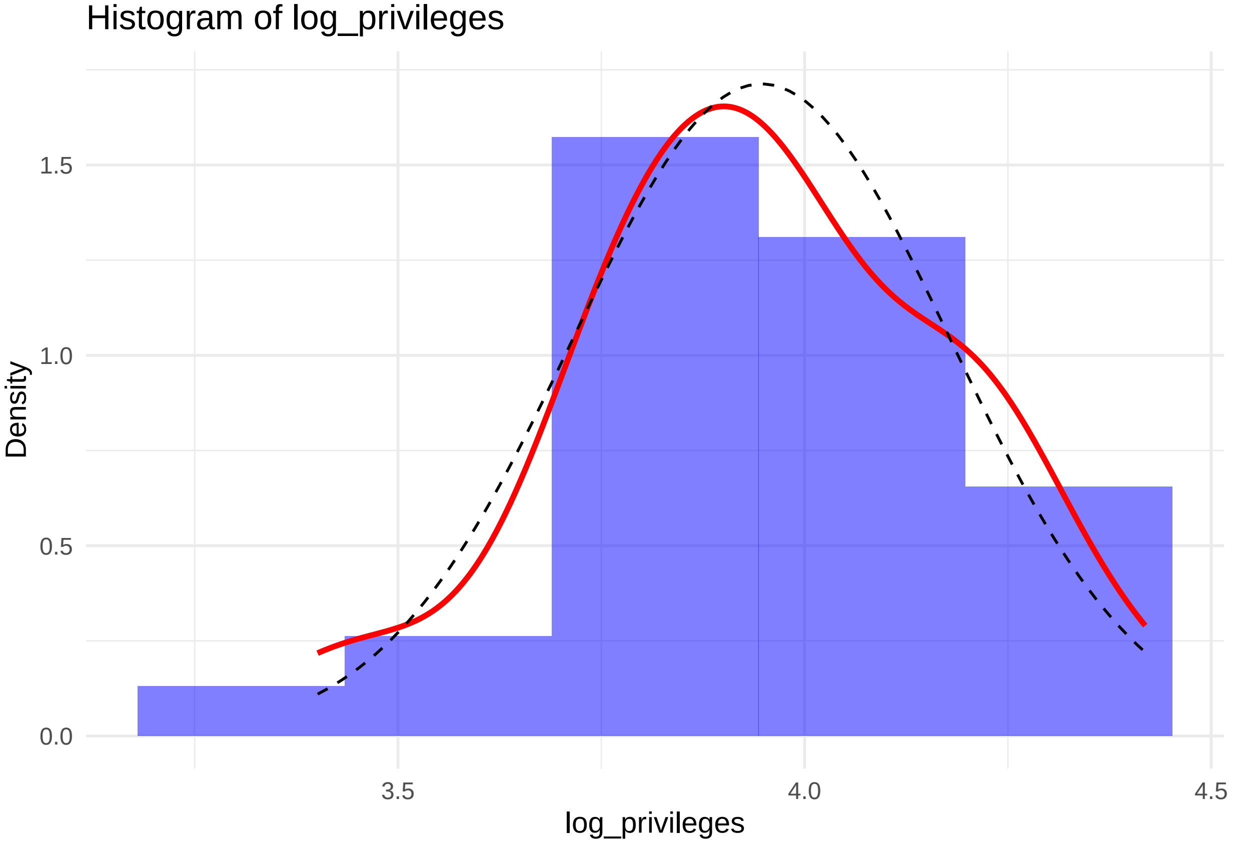
Histogram of complaints



Histogram of log_complaints

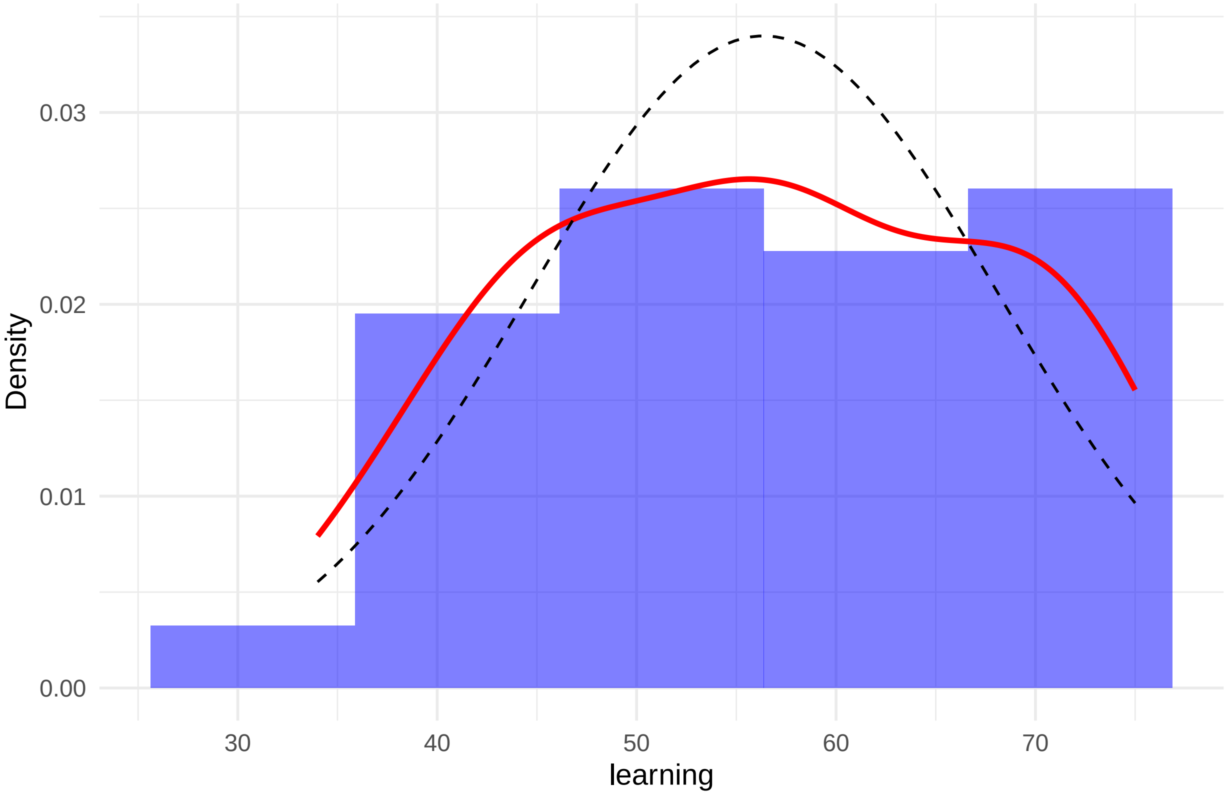


Histogram of privileges

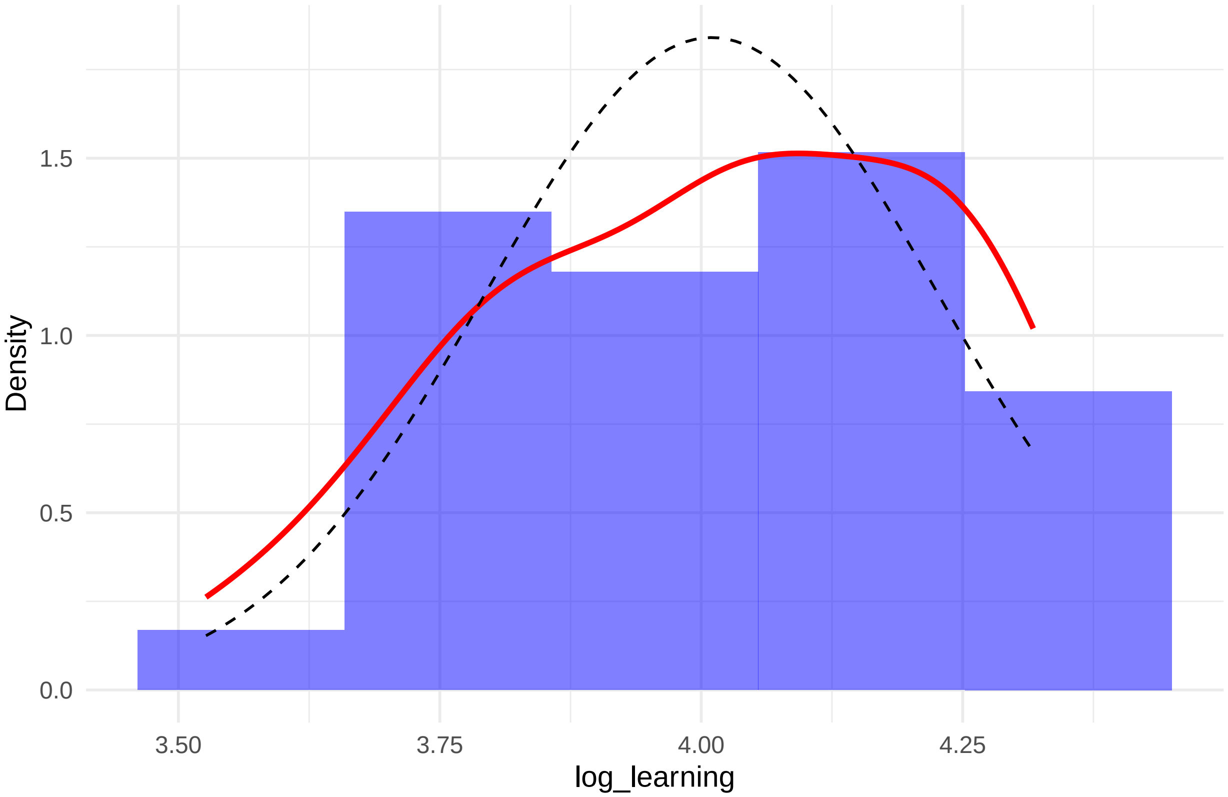


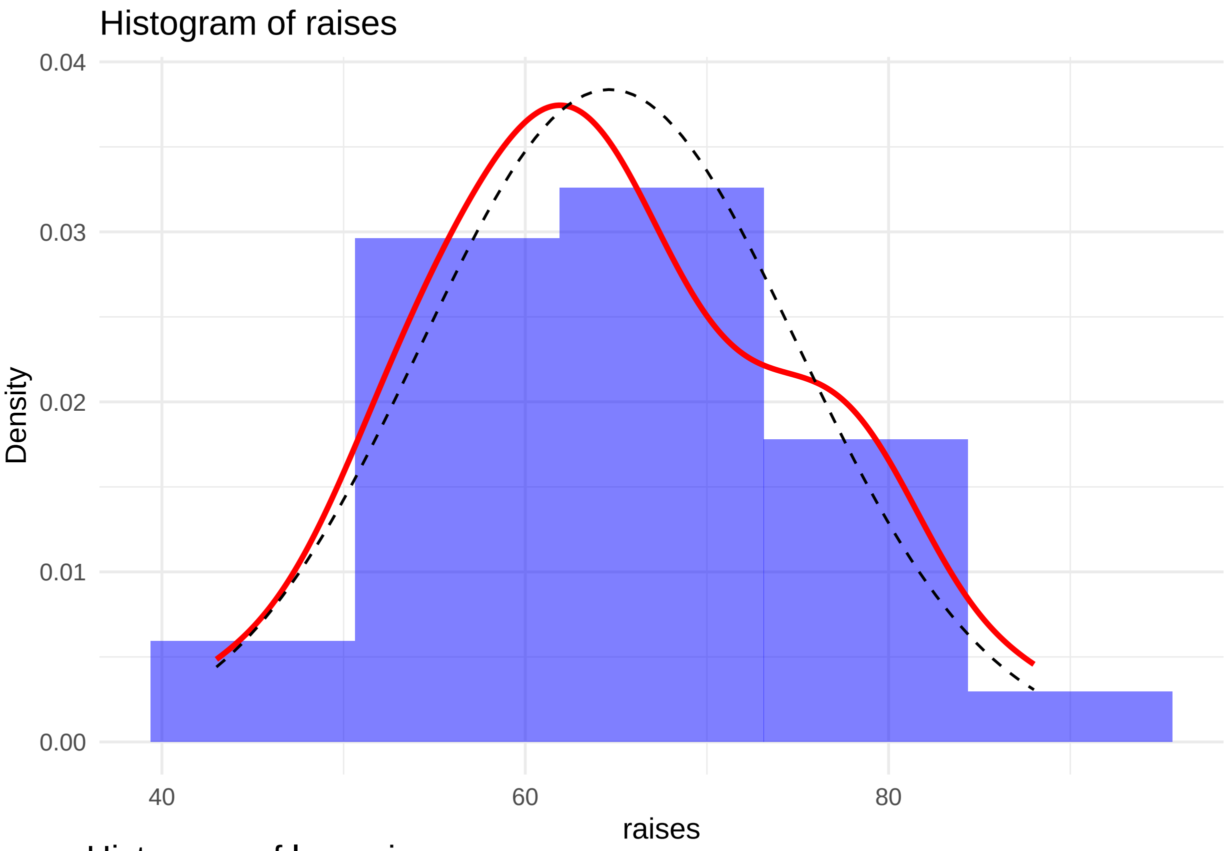
Histogram of log_privileges

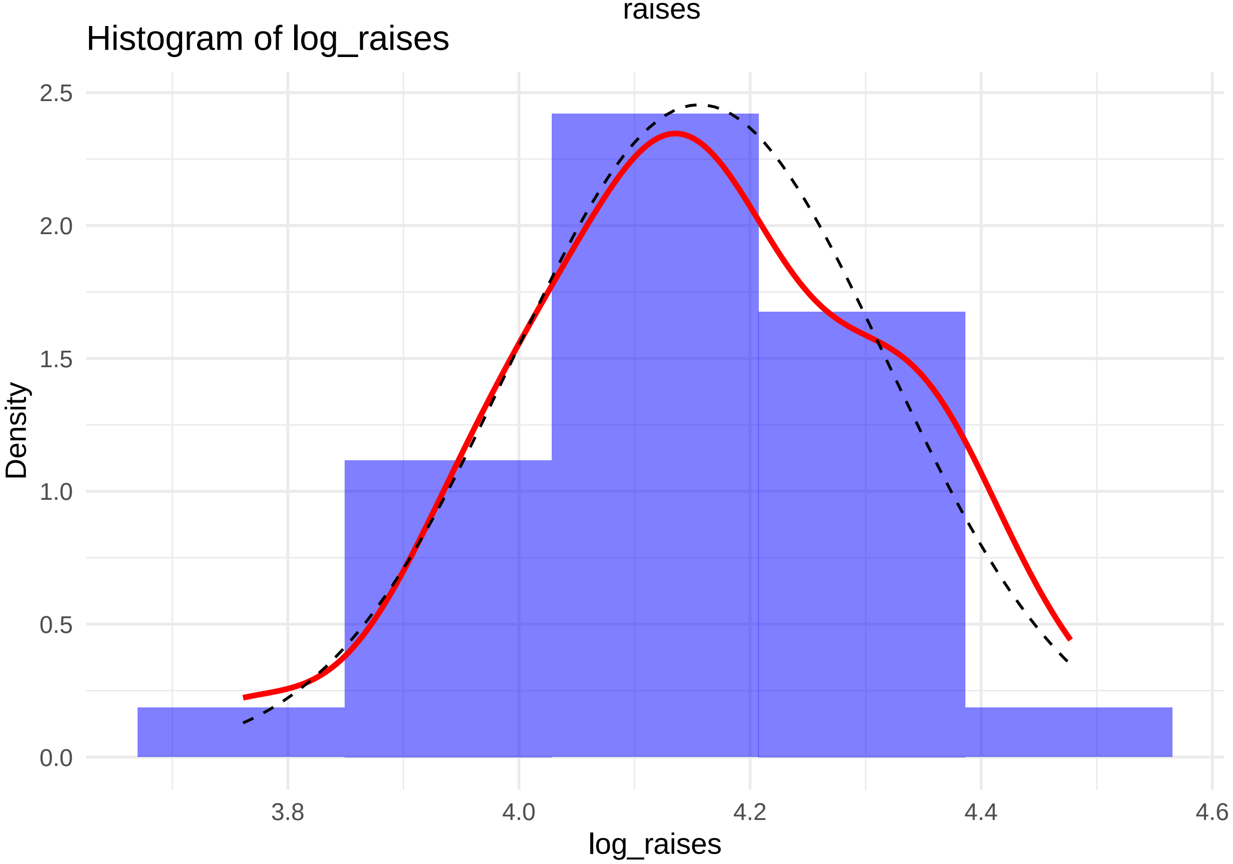
Histogram of learning

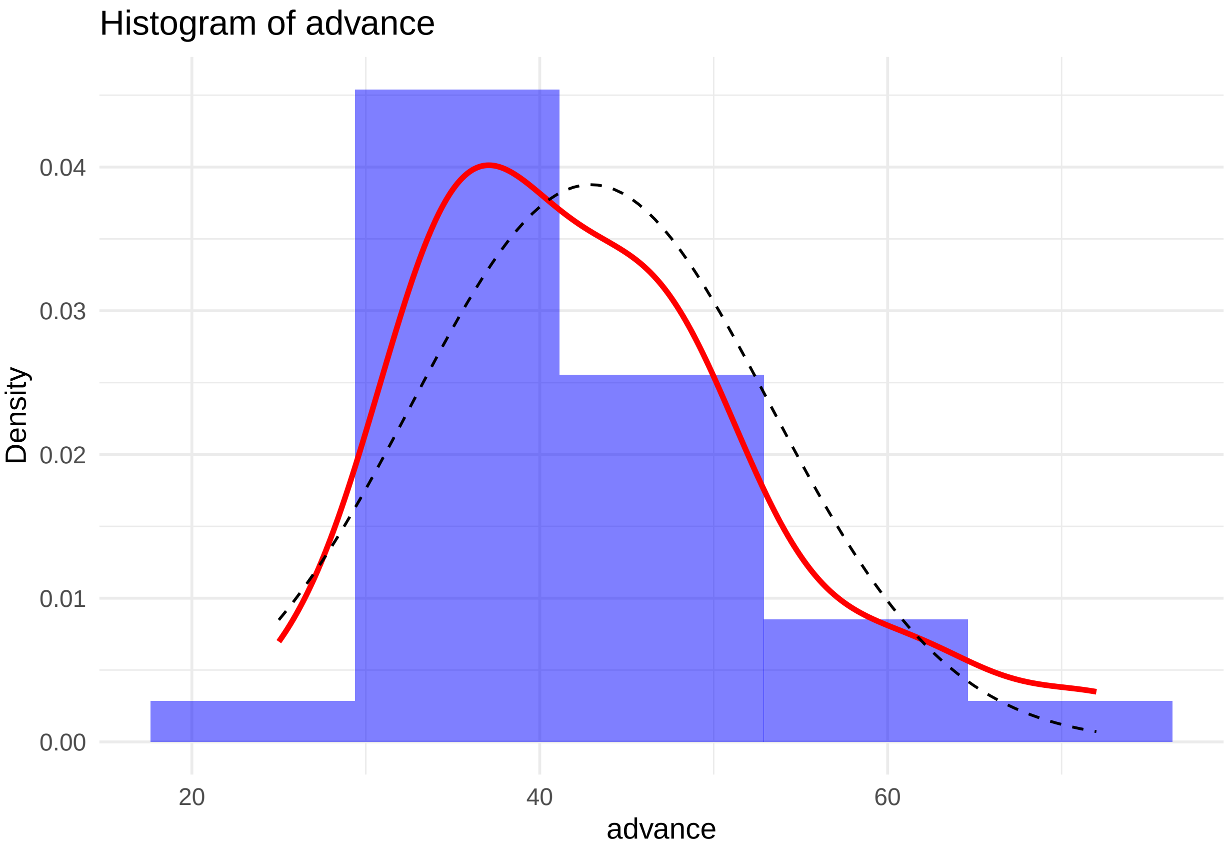


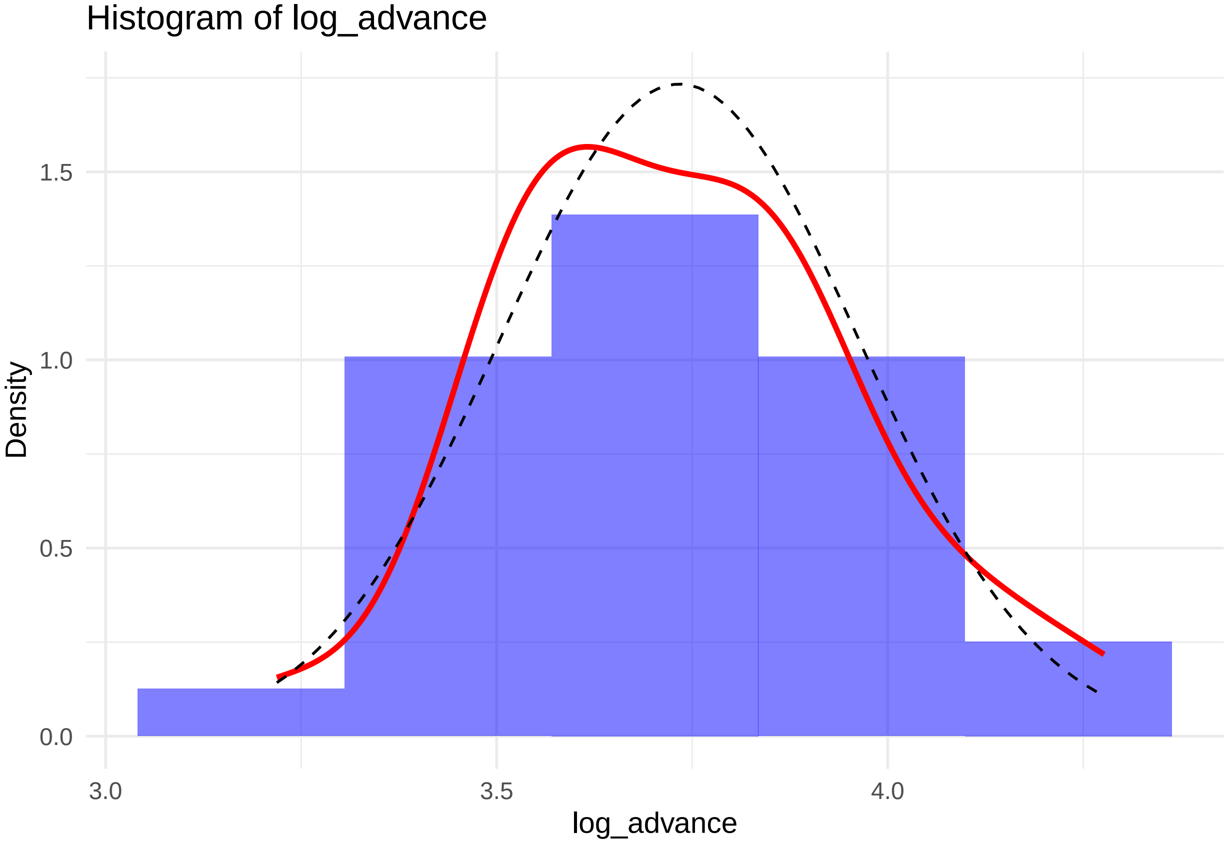
Histogram of log_learning




Histogram of raises
0.04
0.03
0.02
0.01
0.00
Density
40
60
80
raises


Histogram of log_raises
2.5
2.0
1.5
1.0
0.5
0.0
Density
3.8
4.0
4.2
4.4
4.6
log_raises





Based on log transformation analysis, learning, advance and raises appear much more normally distributed.

Furthermore, the response seems to be normally distributed with a log transformation. For the purpose of regression analysis, we want the response to be normally distributed because it ensures the residuals follow a normal distribution in a SLR which is a key assumption.

However, advance and raise are both not predictors in the best model based on AIC and stepwise selection. Therefore, we can see how the model does with a transformed response and transforming learning.

```r
# Fit the log-transformed regression model
log_model <- lm(log_rating ~ complaints + learning, data = df)

# Print model summary
summary(log_model)
```

```
##
## Call:
## lm(formula = log_rating ~ complaints + learning, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.22646 -0.08984  0.02815  0.07793  0.17823
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.262300   0.122263  26.683  < 2e-16 ***
## complaints  0.010776   0.002051   5.253 1.54e-05 ***
## learning    0.003015   0.002327   1.296    0.206
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.118 on 27 degrees of freedom
## Multiple R-squared:  0.6826, Adjusted R-squared:  0.6591
## F-statistic: 29.04 on 2 and 27 DF,  p-value: 1.866e-07
```

```r
# Fit the log-transformed regression model
log_model2 <- lm(log_rating ~ complaints + log_learning, data = df)

# Print model summary
summary(log_model2)
```

```
##
## Call:
## lm(formula = log_rating ~ complaints + log_learning, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.22920 -0.09027  0.02400  0.07684  0.17688
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.843251   0.442285   6.429 6.90e-07 ***
## complaints   0.010955   0.002074   5.283 1.42e-05 ***
## log_learning 0.143926   0.127359   1.130    0.268
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1189 on 27 degrees of freedom
## Multiple R-squared:  0.6781, Adjusted R-squared:  0.6543
## F-statistic: 28.44 on 2 and 27 DF,  p-value: 2.257e-07
```

```
log_model3 <- lm(rating ~ complaints + log_learning, data = df)

summary(log_model3)
```

```
##
## Call:
## lm(formula = rating ~ complaints + log_learning, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.7508  -5.6273   0.6078   6.3720  10.1823
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -19.4238    25.6310  -0.758    0.455
## complaints     0.6562     0.1202   5.461 8.86e-06 ***
## log_learning  10.0641     7.3806   1.364    0.184
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.888 on 27 degrees of freedom
## Multiple R-squared:  0.7018, Adjusted R-squared:  0.6798
## F-statistic: 31.78 on 2 and 27 DF,  p-value: 8.034e-08
```

```
best_stepwise_model <- lm(rating~ complaints + learning, data = df)
summary(best_stepwise_model)
```

```
##
## Call:
## lm(formula = rating ~ complaints + learning, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.5568  -5.7331   0.6701   6.5341  10.3610
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.8709     7.0612   1.398    0.174
## complaints    0.6435     0.1185   5.432 9.57e-06 ***
## learning      0.2112     0.1344   1.571    0.128
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.817 on 27 degrees of freedom
## Multiple R-squared:  0.708,  Adjusted R-squared:  0.6864
## F-statistic: 32.74 on 2 and 27 DF,  p-value: 6.058e-08
```

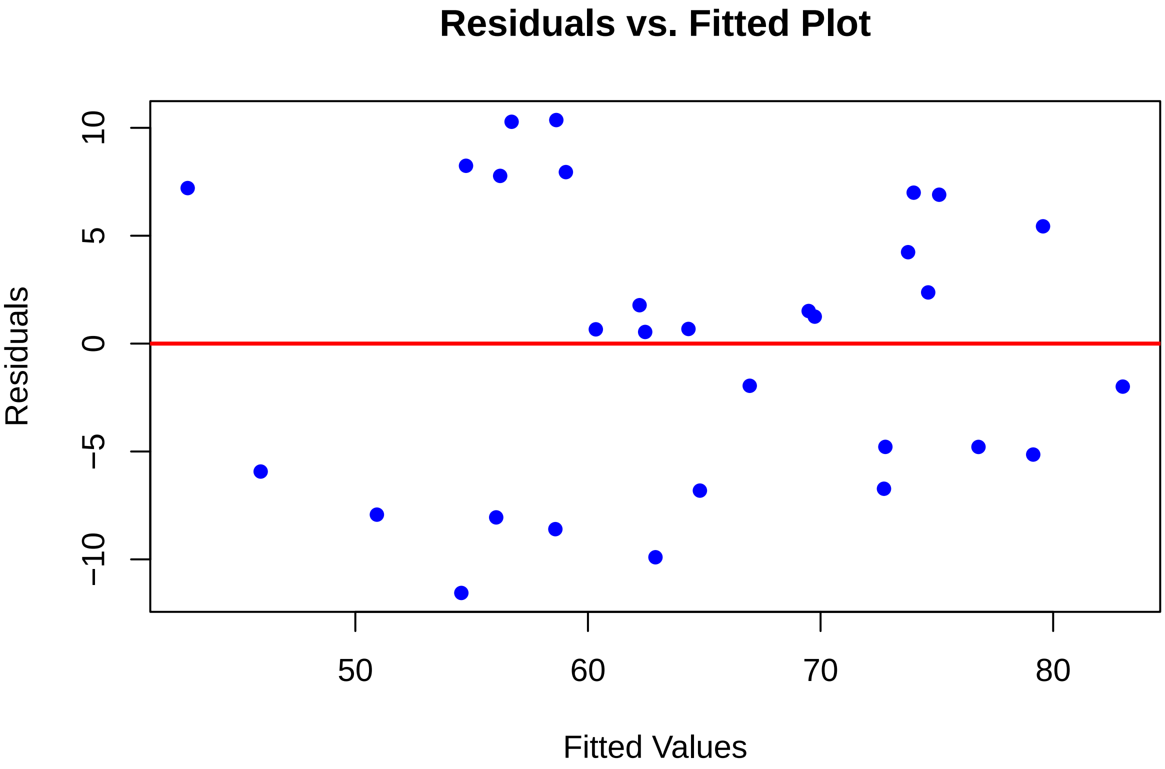
Comparing the original model with a log transformed response, log transformed predictor, and both based on Adjusted R-squared because AIC on different scales is not useful shows us that the original model is better at explaining the variation in the data than any other model.

d) Perform diagnostics to check the assumptions of your model.

Assumptions of a linear model:

Linearity -> Relationship between predictors and response is linear

```
plot(best_stepwise_model$fitted.values, resid(best_stepwise_model),
     main = "Residuals vs. Fitted Plot",
     xlab = "Fitted Values", ylab = "Residuals",
     pch = 16, col = "blue")
abline(h = 0, col = "red", lwd = 2)
```



No clear pattern, so linearity holds.

Independence -> Residuals should be independent (no autocorrelation)

```
library(lmtest)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

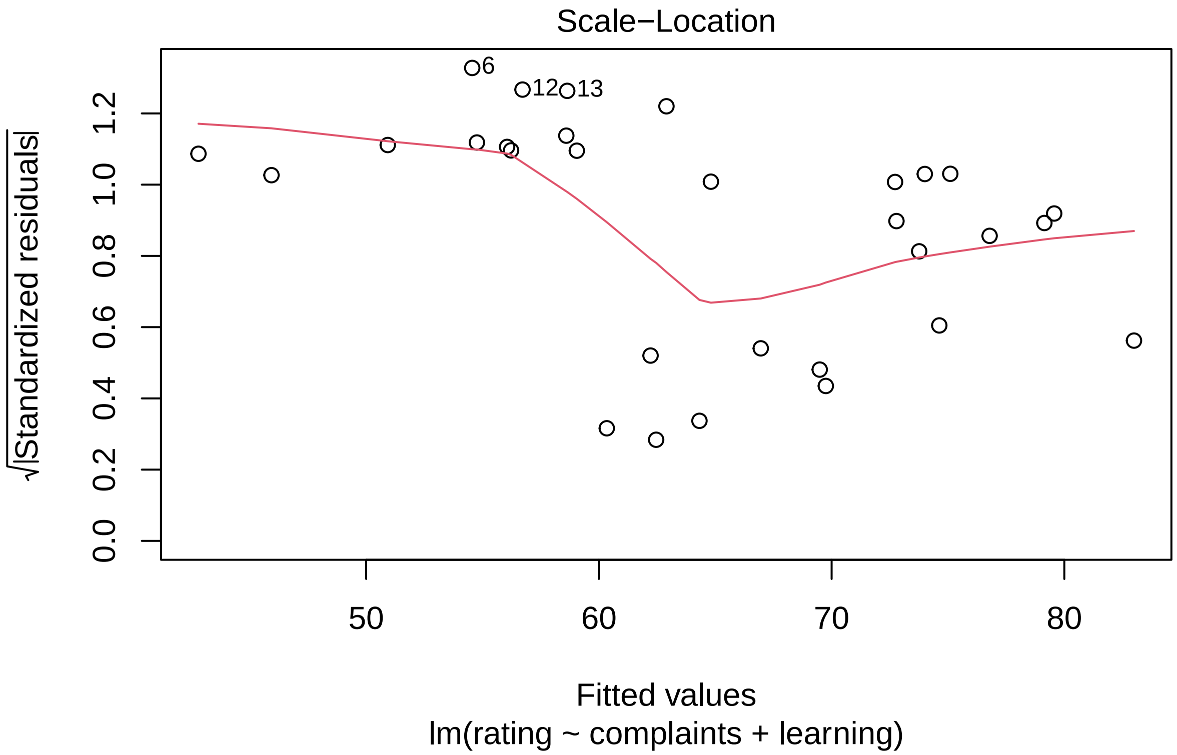
```
dwtest(best_stepwise_model)
```

```
##
##  Durbin-Watson test
##
## data:  best_stepwise_model
## DW = 1.9582, p-value = 0.439
## alternative hypothesis: true autocorrelation is greater than 0
```

P-value > 0.05, no autocorrelation, residuals are independent so independence holds.

Homoscedasticity -> Residuals should have equal variance

```
par(mfrow = c(1, 1))
plot(best_stepwise_model, which = 3)  # Scale-Location Plot
```



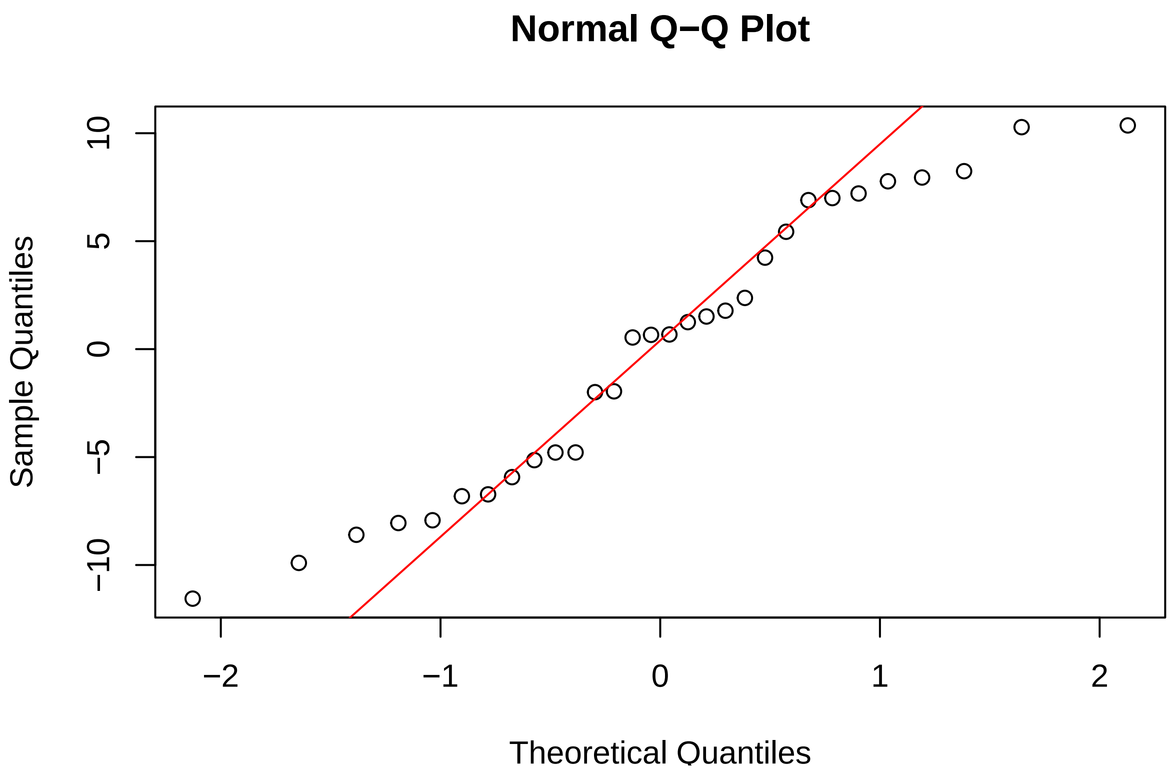
```
bptest(best_stepwise_model)  # Breusch-Pagan Test
```

```
##
##  studentized Breusch-Pagan test
##
## data:  best_stepwise_model
## BP = 11.279, df = 2, p-value = 0.003554
```

The p-value from the BP test < 0.05, which shows that there is non-constant variance. This is confirmed by the Scale-location plot.

Normality -> Residuals should be normally distributed

```
par(mfrow = c(1, 1))
qqnorm(resid(best_stepwise_model))
qqline(resid(best_stepwise_model), col = "red")
```



```
shapiro.test(resid(best_stepwise_model))  # Shapiro-Wilk test
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(best_stepwise_model)
## W = 0.94486, p-value = 0.123
```

P-value > 0.05, residuals are normally distributed, and the points are relatively close to the line.

Multicollinearity -> Predictors should not be highly correlated with each other

For Multicollinearity, we performed correlation analysis and initially observed that none of the predictor variables have an absolute value of correlation higher than 0.7 with another predictor. We will also do VIF analysis.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

```
## The following objects are masked from 'package:faraway':
##
##     logit, vif
```

```r
vif(best_stepwise_model)
```

```
## complaints   learning
##   1.553021   1.553021
```

No Multicollinearity. VIF < 5.

e) Interpret the meaning of the model and comment on potential application

The model almost passes all assumptions in the original form. There is no multicollinearity, linearity, independence and normality seem to hold. However, homoscedasticity does not hold. Therefore, we need to take care of the fact that the variance among residuals is non-constant. We could potentially apply Weighted Least Squares Estimation, WLS, or heteroscedasticity-robust standard errors, HRSE, we could also consider log-transforming the response and evaluating the results again, but the original model seemed to perform better so I decided against that.

Model Interpretation

The best model states that **employee ratings** are best explained by a **linear combination of complaints and learning opportunities**.

Coefficient Interpretation:

- **Intercept (B0 = 9.87)**
  - When complaints = 0 and learning = 0, the predicted rating is **9.87**.

  - This value may not have practical meaning if complaints = 0 is unrealistic.
- **Complaints (B1 = 0.6435, p $<$ 0.001)**
  - **Statistically significant (p $<$ 0.001)**.

  - **For each additional complaint, the employee rating increases by 0.6435, on average**.

  - **This is counter-intuitive**, as we might expect more complaints to lower ratings.

  - This could indicate that employees who voice complaints are **more engaged** or **more proactive**, leading to higher ratings.
- **Learning (B2 = 0.2112, p = 0.128)**
  - **Not statistically significant (p = 0.128)**.
  - This suggests that **learning opportunities may have little or inconsistent impact on ratings**.

  - Since **p $>$ 0.05**, we **cannot confidently conclude** that learning affects rating.

Therefore, considering removing learning is another potential option depending on the objective of the study.

2. [5pts] The National Institute of Diabetes and Digestive and Kidney Diseases conducted a study on 768 adult female Pima Indians living near Phoenix. The purpose of the study was to investigate factors related to diabetes. The data may be found in the the dataset `pima` (available in the Faraway R package).

   a) Create a factor version of the test results and use this to produce an interleaved histogram to show how the distribution of insulin differs between those testing positive and negative. Do you notice anything unbelievable about the plot?

```r
# Load the dataset
library(faraway)
pima <- faraway::pima
# Convert test result (diabetes diagnosis) to a factor
```

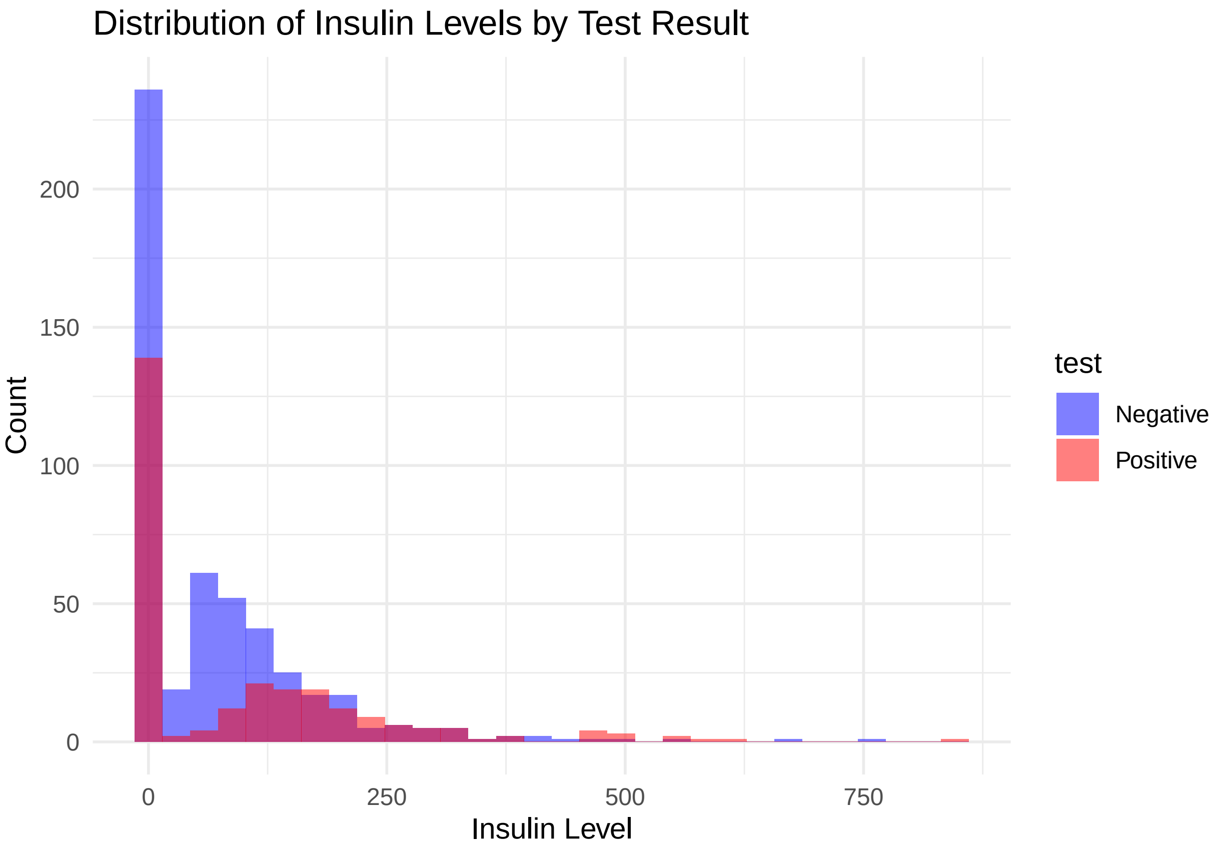
```r
pima$test <- factor(pima$test, labels = c("Negative", "Positive"))

# Check data structure
str(pima)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ pregnant : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ glucose  : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ diastolic: int  72 66 64 66 40 74 50 0 70 96 ...
##  $ triceps  : int  35 29 0 23 35 0 32 0 45 0 ...
##  $ insulin  : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ bmi      : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ diabetes : num  0.627 0.351 0.672 0.167 2.288 ...
##  $ age      : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ test     : Factor w/ 2 levels "Negative","Positive": 2 1 2 1 2 1 2 1 2 2 ...
```

```r
summary(pima)
```

```
##     pregnant         glucose        diastolic         triceps
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     insulin           bmi           diabetes           age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780   Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437   1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725   Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719   Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262   3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200   Max.   :81.00
##        test
##  Negative:500
##  Positive:268
##
##
##
##
```

```r
library(ggplot2)

ggplot(pima, aes(x = insulin, fill = test)) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 30) +
  scale_fill_manual(values = c("blue", "red")) +
  labs(title = "Distribution of Insulin Levels by Test Result",
       x = "Insulin Level",
       y = "Count") +
  theme_minimal()
```



Yes, there is a large number of people who are testing positive for diabetes with insulin levels of 0. This is quite unbelievable.

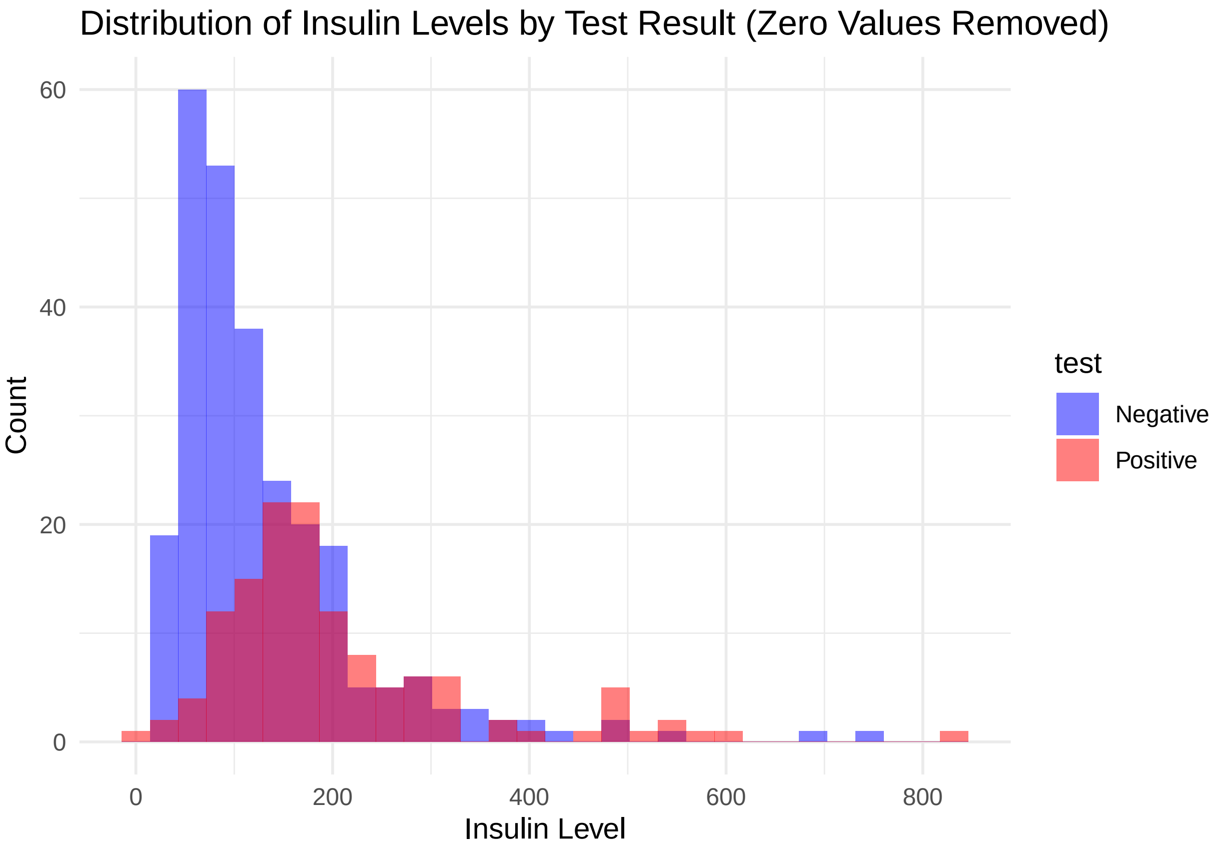
b) Replace the zero values of insulin with the missing value code NA. Recreate the interleaved histogram plot and comment on the distribution.

```
# Convert insulin = 0 to NA (indicating missing data)
pima$insulin[pima$insulin == 0] <- NA

# Check how many values are now NA
sum(is.na(pima$insulin))  # Count 0 insulin values
```

```
## [1] 374
```

```
library(ggplot2)

ggplot(pima, aes(x = insulin, fill = test)) +
  geom_histogram(position = "identity", alpha = 0.5, bins = 30) +
  scale_fill_manual(values = c("blue", "red")) +
  labs(title = "Distribution of Insulin Levels by Test Result (Zero Values Removed)",
       x = "Insulin Level",
       y = "Count") +
  theme_minimal()
```

```
## Warning: Removed 374 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



This plot makes much more sense, we see that as insulin level increases, the conviction of tests being positive increases and the count increases, also the count of testing negative significantly decreases over time, which is expected.

c) Replace the incredible zeroes in other variables with the missing value code. Fit a model with the result of the diabetes test as the response and all the other variables as predictors. How many observations were used in the model fitting? Why is this less than the number of observations in the data frame?

```
# Convert zero values to NA
pima$glucose[pima$glucose == 0] <- NA
pima$diastolic[pima$diastolic == 0] <- NA
pima$triceps[pima$triceps == 0] <- NA
pima$insulin[pima$insulin == 0] <- NA
pima$bmi[pima$bmi == 0] <- NA
pima$pregnant[pima$pregnant == 0] <- NA
```

We will fit a logistic regression model because we transformed the response into a factor.

```
# Fit logistic regression model with all variables as predictors
full_model <- glm(test ~ ., data = pima, family = binomial)

# View model summary
summary(full_model)
```

```
##
## Call:
## glm(formula = test ~ ., family = binomial, data = pima)
##
## Coefficients:
```

```
##               Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -1.083e+01  1.423e+00  -7.610 2.73e-14 ***
## pregnant     7.364e-02  5.973e-02   1.233   0.2176    
## glucose      3.616e-02  6.249e-03   5.785 7.23e-09 ***
## diastolic    5.993e-03  1.320e-02   0.454   0.6497    
## triceps      1.110e-02  1.869e-02   0.594   0.5527    
## insulin      3.231e-05  1.445e-03   0.022   0.9822    
## bmi          7.615e-02  3.174e-02   2.399   0.0164 *  
## diabetes     1.097e+00  4.777e-01   2.297   0.0216 *  
## age          4.075e-02  1.919e-02   2.123   0.0337 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 426.34  on 335  degrees of freedom
## Residual deviance: 288.92  on 327  degrees of freedom
##   (432 observations deleted due to missingness)
## AIC: 306.92
## 
## Number of Fisher Scoring iterations: 5
```

```r
# Check how many observations were used in the model
nobs(full_model)
```

```
## [1] 336
```

336 observations were used, linear regression cannot have empty or NA values for any specific row or column to be statistically sound, therefore, removing all 0's caused the number of observations to go from 768 to 336.

d) Refit the model but now without the insulin and triceps predictors. How many observations were used in fitting this model? Devise a test to compare this model with that in the previous question.

```r
# Refit the model without insulin and triceps
reduced_model <- glm(test ~ pregnant + glucose + diastolic + bmi + diabetes + age,
                     data = pima, family = binomial)

# View model summary
summary(reduced_model)
```

```
## 
## Call:
## glm(formula = test ~ pregnant + glucose + diastolic + bmi + diabetes + 
##     age, family = binomial, data = pima)
## 
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -9.354750   0.915697 -10.216  < 2e-16 ***
## pregnant     0.130695   0.037880   3.450  0.00056 ***
## glucose      0.035337   0.003900   9.061  < 2e-16 ***
## diastolic   -0.008673   0.009422  -0.920  0.35734    
## bmi          0.098547   0.017768   5.546 2.92e-08 ***
## diabetes     1.020669   0.336136   3.036  0.00239 ** 
## age          0.016642   0.010553   1.577  0.11478    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
```

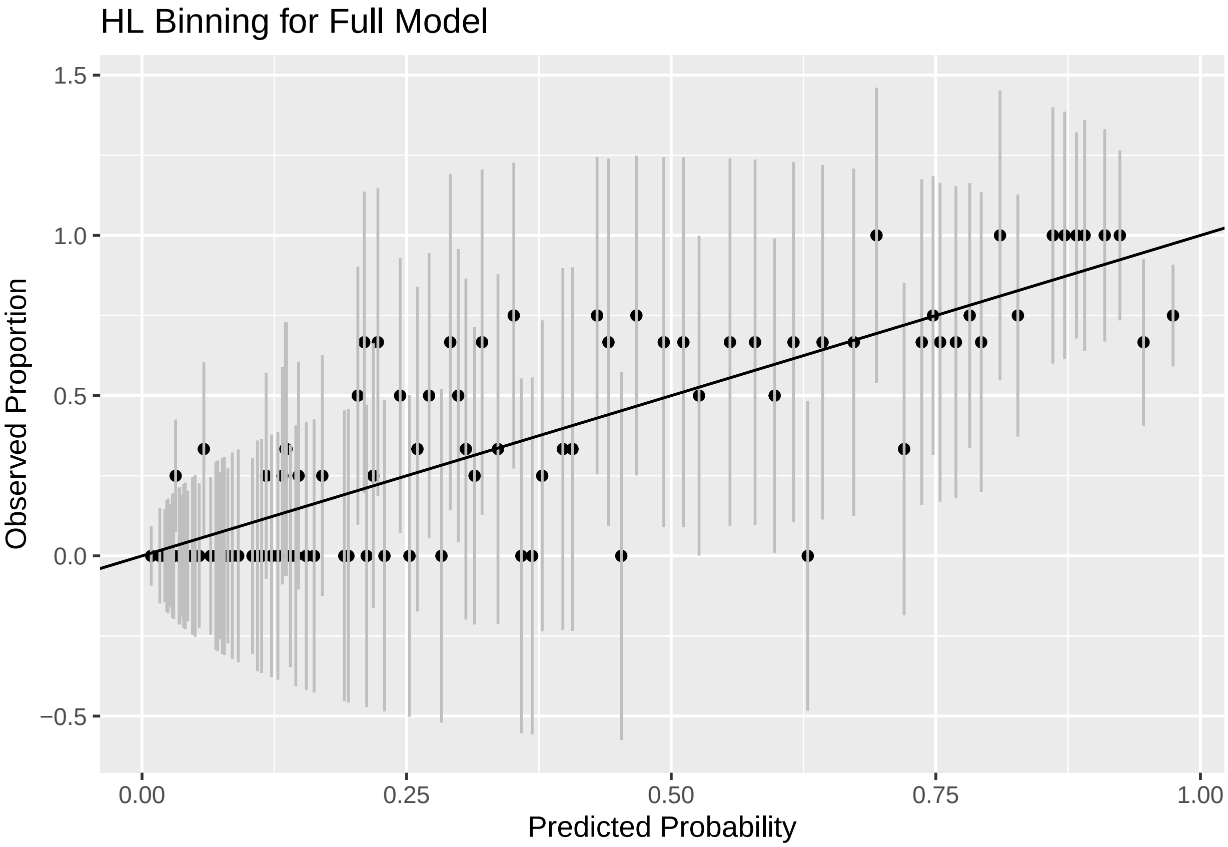
```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 807.12  on 624  degrees of freedom
## Residual deviance: 577.80  on 618  degrees of freedom
##   (143 observations deleted due to missingness)
## AIC: 591.8
##
## Number of Fisher Scoring iterations: 5
```

```r
nobs(reduced_model)
```

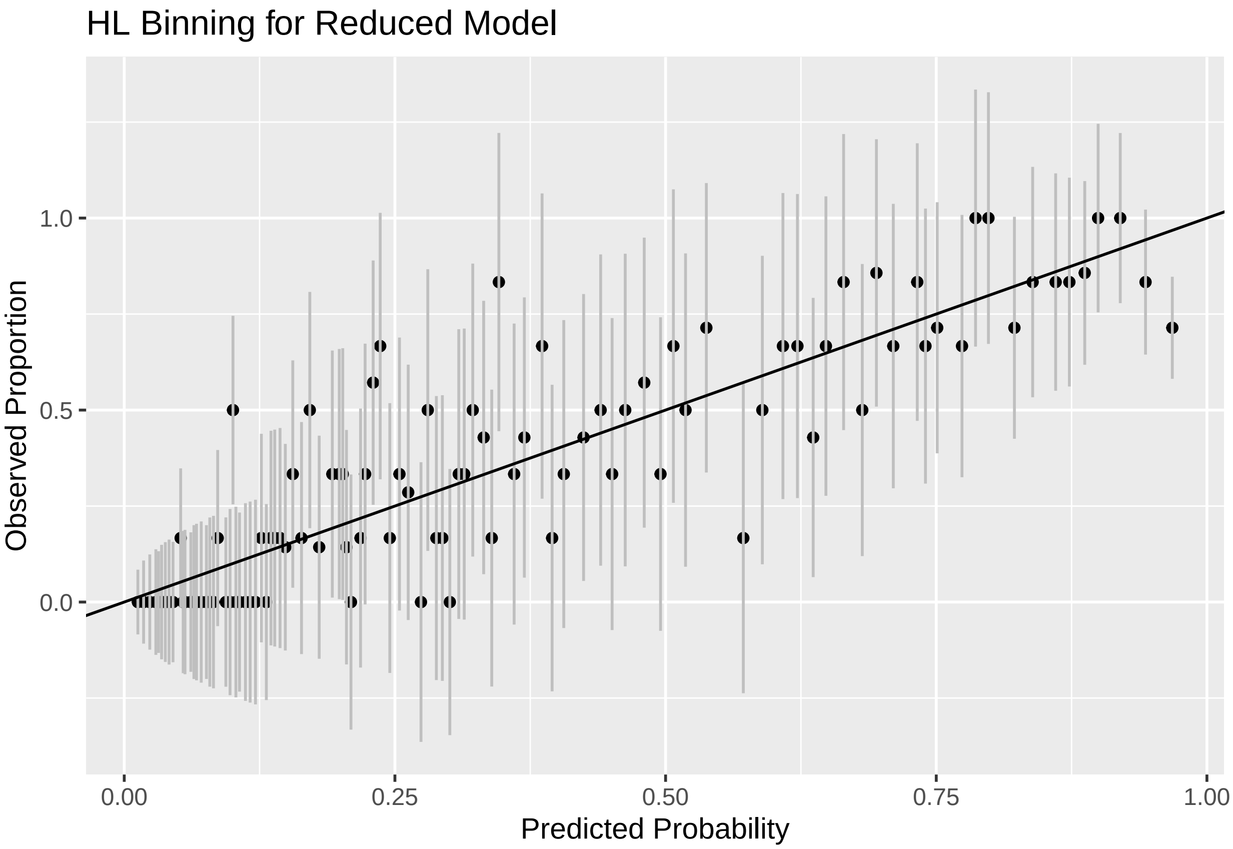
```
## [1] 625
```

There are 625 observations used in fitting this model. There are more observations in this model than the previous model therefore we cannot do a simple ANOVA test for comparison because one model is not nested within the other. We will do HL binning, Specificity and Sensitity Analysis, and ROC curves to conduct model comparison.

```r
library(dplyr)

# Store rows used in each model separately
pima_full <- pima[!is.na(pima$pregnant) & !is.na(pima$glucose) &
                   !is.na(pima$diastolic) & !is.na(pima$triceps) &
                   !is.na(pima$insulin) & !is.na(pima$bmi) &
                   !is.na(pima$diabetes) & !is.na(pima$age), ]

pima_reduced <- pima[!is.na(pima$pregnant) & !is.na(pima$glucose) &
                      !is.na(pima$diastolic) & !is.na(pima$bmi) &
                      !is.na(pima$diabetes) & !is.na(pima$age), ]

# Generate predictions for both models
pima_full <- mutate(pima_full,
                    linpred_full = fitted(full_model),
                    predprob_full = predict(full_model, type="response"))

pima_reduced <- mutate(pima_reduced,
                       linpred_reduced = fitted(reduced_model),
                       predprob_reduced = predict(reduced_model, type="response"))

library(ggplot2)

# Function to perform HL binning for a given model
perform_hl_binning <- function(data, linpred_col, predprob_col, model_name) {

  # Ensure unique bin breaks
  bin_breaks <- unique(quantile(data[[linpred_col]], probs = seq(0, 1, length.out = 101), na.rm = TRUE)

  # Create bins
  gdf <- group_by(data, cut(!!sym(linpred_col), breaks=bin_breaks, include.lowest = TRUE))

  # Compute observed vs predicted proportions
  hldf <- summarise(gdf,
                    y = sum(as.numeric(test) - 1, na.rm = TRUE),
                    ppred = mean(!!sym(predprob_col), na.rm = TRUE),
                    count = n()) %>%
    mutate(se.fit = sqrt(ppred * (1 - ppred) / count))
  # Generate HL Binning Plot
  ggplot(hldf, aes(x=ppred, y=y/count, ymin=y/count - 2 * se.fit, ymax=y/count + 2 * se.fit)) +
    geom_point() +
    geom_linerange(color=grey(0.75)) +
    geom_abline(intercept=0, slope=1) +
    xlab("Predicted Probability") +
    ylab("Observed Proportion") +
    ggtitle(paste("HL Binning for", model_name))
}

# Perform HL binning for Full Model
perform_hl_binning(pima_full, "linpred_full", "predprob_full", "Full Model")
```

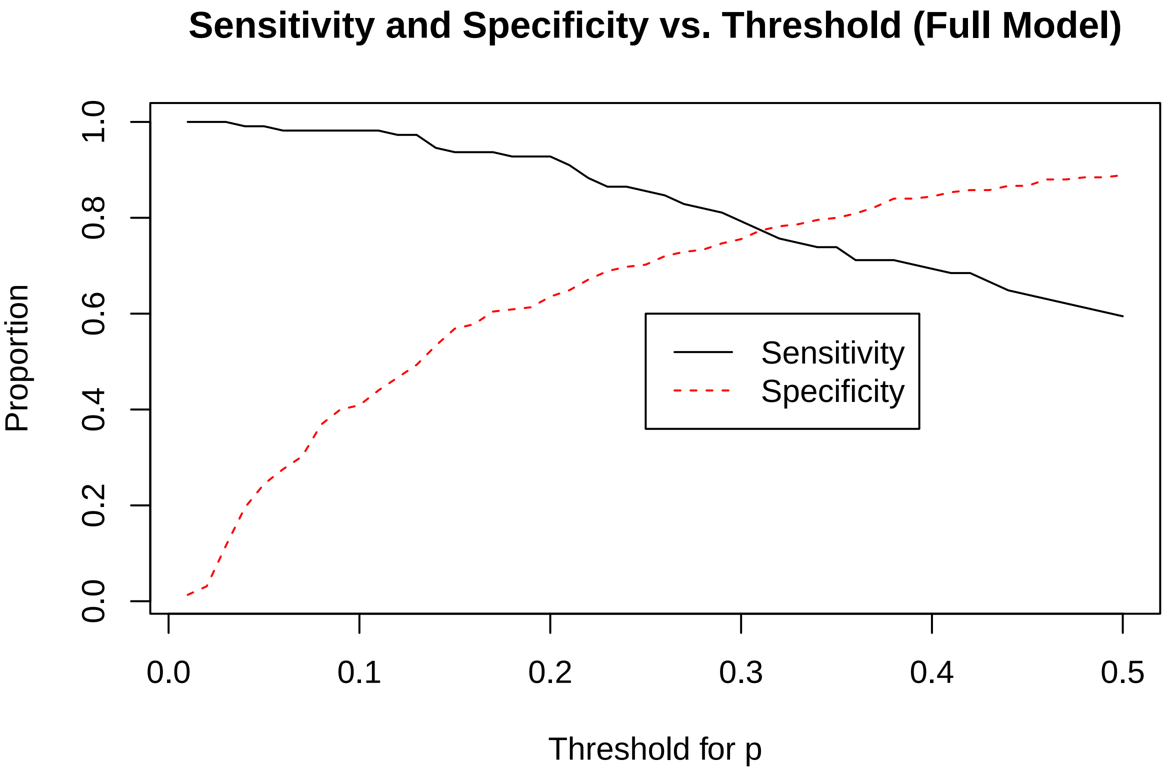


```
# Perform HL binning for Reduced Model
perform_hl_binning(pima_reduced, "linpred_reduced", "predprob_reduced", "Reduced Model")
```

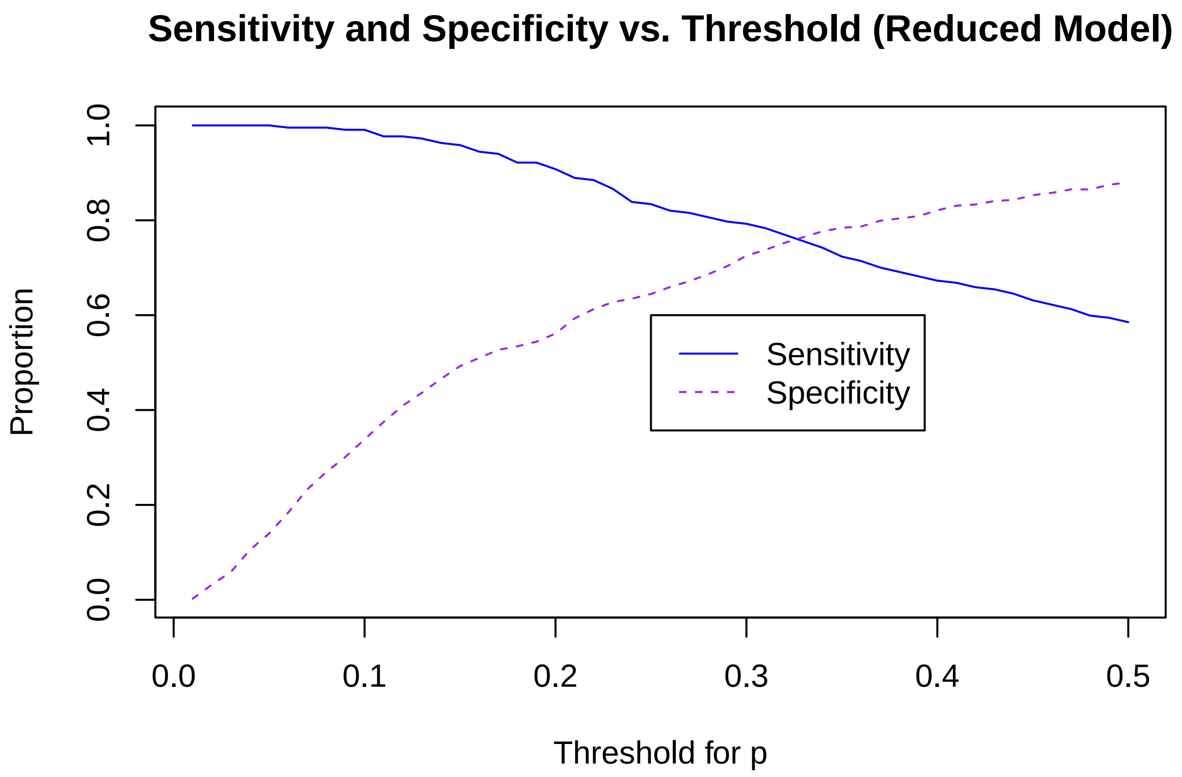


```
thresh <- seq(0.01, 0.5, 0.01)

# Initialize vectors
Sensitivity_full <- numeric(length(thresh))
Specificity_full <- numeric(length(thresh))

Sensitivity_reduced <- numeric(length(thresh))
Specificity_reduced <- numeric(length(thresh))

# Compute ROC values for Full Model
for (j in seq(along=thresh)) {
  pp_full <- ifelse(pima_full$predprob_full < thresh[j], "Negative", "Positive")

  # Ensure factor levels match
  pima_full$test_factor <- factor(pima_full$test, levels = c("Negative", "Positive"))
  pp_full <- factor(pp_full, levels = c("Negative", "Positive"))

  xx_full <- xtabs(~ test_factor + pp_full, pima_full)

  Specificity_full[j] <- xx_full[1,1] / (xx_full[1,1] + xx_full[1,2])
  Sensitivity_full[j] <- xx_full[2,2] / (xx_full[2,1] + xx_full[2,2])
}

# Compute ROC values for Reduced Model
for (j in seq(along=thresh)) {
  pp_reduced <- ifelse(pima_reduced$predprob_reduced < thresh[j], "Negative", "Positive")
  # Ensure factor levels match
  pima_reduced$test_factor <- factor(pima_reduced$test, levels = c("Negative", "Positive"))
  pp_reduced <- factor(pp_reduced, levels = c("Negative", "Positive"))

  xx_reduced <- xtabs(~ test_factor + pp_reduced, pima_reduced)

  Specificity_reduced[j] <- xx_reduced[1,1] / (xx_reduced[1,1] + xx_reduced[1,2])
  Sensitivity_reduced[j] <- xx_reduced[2,2] / (xx_reduced[2,1] + xx_reduced[2,2])
}
```

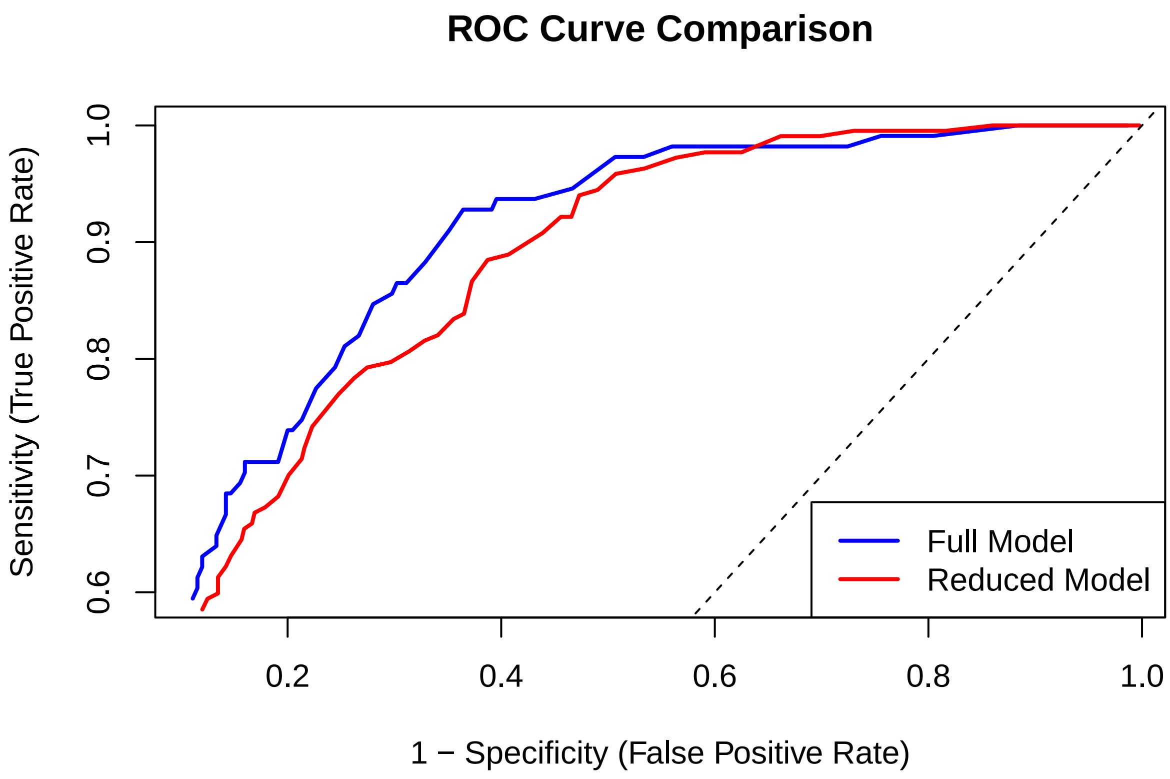
```
matplot(thresh, cbind(Sensitivity_full, Specificity_full), type="l",
        xlab="Threshold for p", ylab="Proportion", lty=1:2, col=c("black", "red"),
        main="Sensitivity and Specificity vs. Threshold (Full Model)")
legend(0.25, 0.6, lty=1:2, col=c("black", "red"),
       legend=c("Sensitivity", "Specificity"))
```



```
matplot(thresh, cbind(Sensitivity_reduced, Specificity_reduced), type="l",
        xlab="Threshold for p", ylab="Proportion", lty=1:2, col=c("blue", "purple"),
        main="Sensitivity and Specificity vs. Threshold (Reduced Model)")
legend(0.25, 0.6, lty=1:2, col=c("blue", "purple"),
       legend=c("Sensitivity", "Specificity"))
```



```
# ROC Curve Plot
plot(1 - Specificity_full, Sensitivity_full, type="l", col='blue', lwd=2,
     main="ROC Curve Comparison", xlab="1 - Specificity (False Positive Rate)",
     ylab="Sensitivity (True Positive Rate)")
lines(1 - Specificity_reduced, Sensitivity_reduced, col='red', lwd=2)

# Add reference line
abline(0,1, lty=2, col="black")

# Add legend
legend("bottomright", legend=c("Full Model", "Reduced Model"), col=c("blue", "red"), lwd=2)
```



Conclusion:

After comparing the Full Model (using all predictors) and the Reduced Model (excluding `insulin` and `triceps`), we find that:

- The **Full Model performed slightly better** in ROC analysis but had **fewer observations (336 vs. 625)**.
- **Both models were well-calibrated** based on HL Binning.
- **Sensitivity and Specificity were similar**, meaning classification ability did not drop significantly in the Reduced Model.
- **While the Full Model explained more variance**, the difference in fit was small.
- **The Reduced Model used almost twice as many observations**, making it more statistically robust.

Final Decision:

Since the **loss in model fit is minimal**, but the **Reduced Model retains significantly more observations**, I **prefer the Reduced Model**. This model provides a good balance between **predictive accuracy and data availability**, making it the **best choice for practical application**.

e) Use AIC to select a model. You will need to take account of the missing values. Which predictors are selected? How many cases are used in your selected model?

In order for Step-wise Selection with AIC to occur, we need to take a subset of the data in which there are no NA values, so we will take a look at a full model, but considering only non-null observations.

```
# Remove all rows with missing values in relevant predictors
pima_clean <- na.omit(pima)  # Drops any row with NA

# Refit the Full Model on the cleaned dataset
full_model_clean <- glm(test ~ ., data = pima_clean, family = binomial)
```

```r
# Check the number of remaining observations
cat("Number of observations after removing NAs:", nrow(pima_clean), "\n")
```

```
## Number of observations after removing NAs: 336
```

There are 336 observations in our model selection criterion based on stepwise selection and AIC.

```r
library(MASS)

# Perform stepwise AIC selection
best_aic_model <- step(full_model_clean, direction="both", trace=TRUE)
```

```
## Start:  AIC=306.92
## test ~ pregnant + glucose + diastolic + triceps + insulin + bmi +
##     diabetes + age
##
##             Df Deviance    AIC
## - insulin    1   288.92 304.92
## - diastolic  1   289.13 305.13
## - triceps    1   289.27 305.27
## - pregnant   1   290.45 306.45
## <none>           288.92 306.92
## - age        1   293.62 309.62
## - diabetes   1   294.35 310.35
## - bmi        1   294.81 310.81
## - glucose    1   327.93 343.93
##
## Step:  AIC=304.92
## test ~ pregnant + glucose + diastolic + triceps + bmi + diabetes +
##     age
##
##             Df Deviance    AIC
## - diastolic  1   289.13 303.13
## - triceps    1   289.27 303.27
## - pregnant   1   290.45 304.45
## <none>           288.92 304.92
## + insulin    1   288.92 306.92
## - age        1   293.68 307.68
## - diabetes   1   294.38 308.38
## - bmi        1   295.05 309.05
## - glucose    1   342.36 356.36
##
## Step:  AIC=303.13
## test ~ pregnant + glucose + triceps + bmi + diabetes + age
##
##             Df Deviance    AIC
## - triceps    1   289.49 301.49
## - pregnant   1   290.75 302.75
## <none>           289.13 303.13
## + diastolic  1   288.92 304.92
## + insulin    1   289.13 305.13
## - age        1   294.37 306.37
## - diabetes   1   294.59 306.59
## - bmi        1   296.04 308.04
## - glucose    1   343.60 355.60
##
## Step:  AIC=301.49
## test ~ pregnant + glucose + bmi + diabetes + age
##
##             Df Deviance    AIC
## - pregnant   1   291.12 301.12
## <none>           289.49 301.49
## + triceps    1   289.13 303.13
## + diastolic  1   289.27 303.27
## + insulin    1   289.49 303.49
## - age        1   295.20 305.20
## - diabetes   1   295.29 305.29
## - bmi        1   303.66 313.66
## - glucose    1   344.70 354.70
##
## Step:  AIC=301.12
## test ~ glucose + bmi + diabetes + age
##
##             Df Deviance    AIC
## <none>           291.12 301.12
## + pregnant   1   289.49 301.49
## + triceps    1   290.75 302.75
## + diastolic  1   290.82 302.82
## + insulin    1   291.11 303.11
## - diabetes   1   296.43 304.43
## - bmi        1   305.38 313.38
## - age        1   309.18 317.18
## - glucose    1   345.86 353.86
```

```
# Summary of the best model
summary(best_aic_model)
```

```
##
## Call:
## glm(formula = test ~ glucose + bmi + diabetes + age, family = binomial,
##     data = pima_clean)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.810466   1.253806  -8.622  < 2e-16 ***
## glucose       0.036394   0.005495   6.624 3.51e-11 ***
## bmi           0.089165   0.024301   3.669 0.000243 ***
## diabetes      1.055880   0.465979   2.266 0.023455 *
## age           0.059405   0.014515   4.093 4.26e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 426.34  on 335  degrees of freedom
## Residual deviance: 291.12  on 331  degrees of freedom
## AIC: 301.12
##
## Number of Fisher Scoring iterations: 5
```

```r
# Print AIC of the final model
cat("\nAIC for Selected Model:", AIC(best_aic_model), "\n")
```

```
##
## AIC for Selected Model: 301.1248
```

```r
# Print number of observations used
cat("\nNumber of observations used:", nobs(best_aic_model), "\n")
```

```
##
## Number of observations used: 336
```

There were 336 cases used in the selected AIC model. The best AIC model includes **glucose, bmi, diabetes and age** as the selected predictors.

3. Recall

$$\hat{y} = X\hat{\beta} = X(X'X)^{-1}X'y = Hy,$$

where $y = (y_1, \dots, y_n)$ and $X = [1|x_1|x_2|...|x_p], x_i = (x_{i1}, ..x_{in})$.

Let $h_1, \dots, h_n$ be the diagonal values of H, then $e_{(i)} = \frac{e_i}{1-h_i}$ is the prediction error that would be obtained if the i^{th} observation were omitted.

[2pts] (a) Create a synthetic data set using R with vectors x_1, x_2, x_3, y, each vector consists of $n = 30$ observations and y is the response variable.

```r
set.seed(123)  # Ensure reproducibility

n <- 30  # Number of observations

# Generate random predictors
x1 <- rnorm(n, mean = 5, sd = 2)
x2 <- rnorm(n, mean = 10, sd = 3)
x3 <- rnorm(n, mean = 15, sd = 4)

# Generate response variable y based on a linear relationship
beta_0 <- 3
beta_1 <- 2
beta_2 <- -1
beta_3 <- 0.5

y <- beta_0 + beta_1*x1 + beta_2*x2 + beta_3*x3 + rnorm(n, mean = 0, sd = 1)  # Add noise

# Create a data frame
df <- data.frame(y, x1, x2, x3)

# Check the structure of the dataset
str(df)
```

```
## 'data.frame':    30 obs. of  4 variables:
##  $ y : num  8.73 10.01 13.62 5.48 7.77 ...
##  $ x1: num  3.88 4.54 8.12 5.14 5.26 ...
##  $ x2: num  11.28 9.11 12.69 12.63 12.46 ...
##  $ x3: num  16.5 13 13.7 10.9 10.7 ...
```

[1pt] (b) Fit two linear models to the following : one to the full dataset you created in (a), and another one to the dataset with a row (of your choice) removed, let's call this row i. Show the coefficients for both models.

```r
# Fit full linear model
full_model <- lm(y ~ x1 + x2 + x3, data = df)

# Remove one observation (let's remove row `i = 5`)
df_removed <- df[-5, ]

# Fit the model without row 5
removed_model <- lm(y ~ x1 + x2 + x3, data = df_removed)

# Print coefficients for comparison
cat("Coefficients for Full Model:\n")
```

```
## Coefficients for Full Model:
```

```r
print(coef(full_model))
```

```
## (Intercept)          x1          x2          x3
##   2.4075941   2.0249121  -0.9688249   0.5031709
```

```r
cat("\nCoefficients for Model with Row 5 Removed:\n")
```

```
##
## Coefficients for Model with Row 5 Removed:
```

```r
print(coef(removed_model))
```

```
## (Intercept)          x1          x2          x3
##   2.2586344   2.0122719  -0.9802793   0.5216947
```

As we can see the coefficients from the two models changes slightly, signifying the influence of the removal of that specific data point from the data set (leverage).

[2pts] (c) Computationally verify that $\hat{\beta}_{(i)} = \hat{\beta} - \frac{e_i}{1-h_i}(X'X)^{-1}x_i$, where e_i is the model error for the i^{th} observation from the original model.

```r
# Extract X matrix correctly (including intercept)
X <- model.matrix(full_model)

# Compute (X'X)^(-1)
XtX_inv <- solve(t(X) %*% X)

# Compute Hat Matrix H
H <- X %*% XtX_inv %*% t(X)

# Extract leverage value for row 5
h_i <- H[5,5]

# Compute residual e_i for row 5
e_i <- resid(full_model)[5]

# Extract the predictor values for the removed observation (row 5), including intercept
x_i <- X[5, , drop=FALSE]  # Ensures it's a row vector

# Compute beta adjustment term (ensuring dimensions match)
adjustment <- (e_i / (1 - h_i)) * XtX_inv %*% t(x_i)

# Compute leave-one-out beta estimate
```

```r
beta_LOO <- coef(full_model) - adjustment

# Print results
cat("\nBeta from Model with Row 5 Removed:\n")
```

```
##
## Beta from Model with Row 5 Removed:
```

```r
print(coef(removed_model))
```

```
## (Intercept)          x1          x2          x3
##   2.2586344   2.0122719  -0.9802793   0.5216947
```

```r
cat("\nBeta from Leave-One-Out Computation:\n")
```

```
##
## Beta from Leave-One-Out Computation:
```

```r
print(beta_LOO)
```

```
##                      5
## (Intercept)  2.2586344
## x1           2.0122719
## x2          -0.9802793
## x3           0.5216947
```

As we can see, the values from the computations match, therefore we have computationally verified that $\hat{\beta}_{(i)} = \hat{\beta} - \frac{e_i}{1-h_i}(X'X)^{-1}x_i$, where e_i is the model error for the i^{th} observation from the original model.