Project LR

Aqsa Noreen

2023-12-07

Linear Regression Analysis on Diabetes Dataset

Loading Libraries and Data

```
library(ggplot2)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(ggfortify)
library(MASS)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(leaps)


df <- read.table("diabetes.txt", header = TRUE)
head(df)
```

	chol <int>	stab.glu <int>	hdl <int>	ratio <dbl>	glyhb <dbl>	location <chr>	age <int>	gender <chr>	height <int>
1	203	82	56	3.6	4.31	Buckingham	46	female	62
2	165	97	24	6.9	4.44	Buckingham	29	female	64
3	228	92	37	6.2	4.64	Buckingham	58	female	61
4	78	93	12	6.5	4.63	Buckingham	67	male	67
5	249	90	28	8.9	7.72	Buckingham	64	male	68
6	248	94	69	3.6	4.81	Buckingham	34	male	71

6 rows | 1-10 of 17 columns

Data Exploration

Identifying Variables

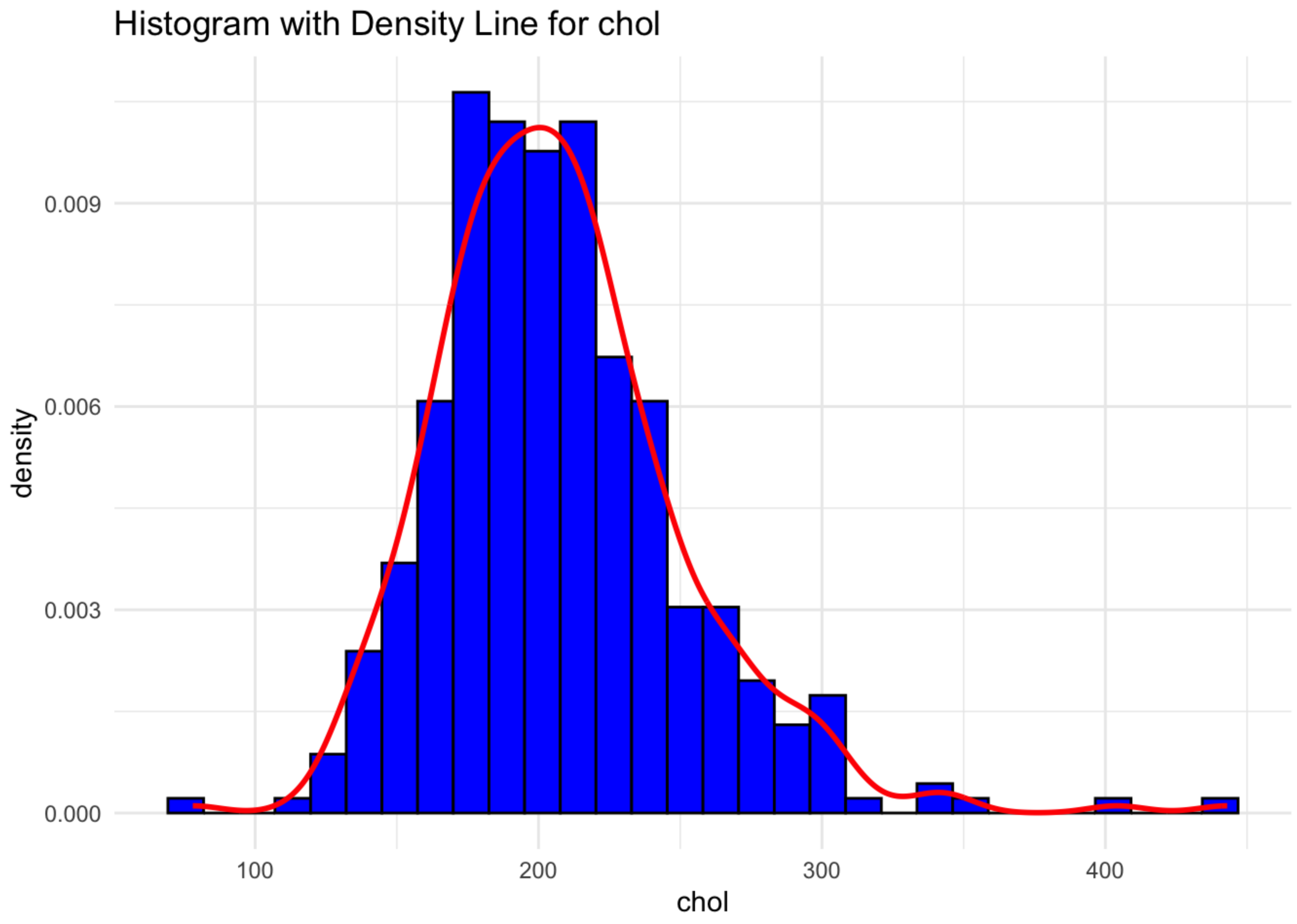
Quantitative Variables are: Chol, stab.glu, hd1, ratio, glyhb, age, height, weight, bp.1s, bp.1d, waist, hip, time.ppn

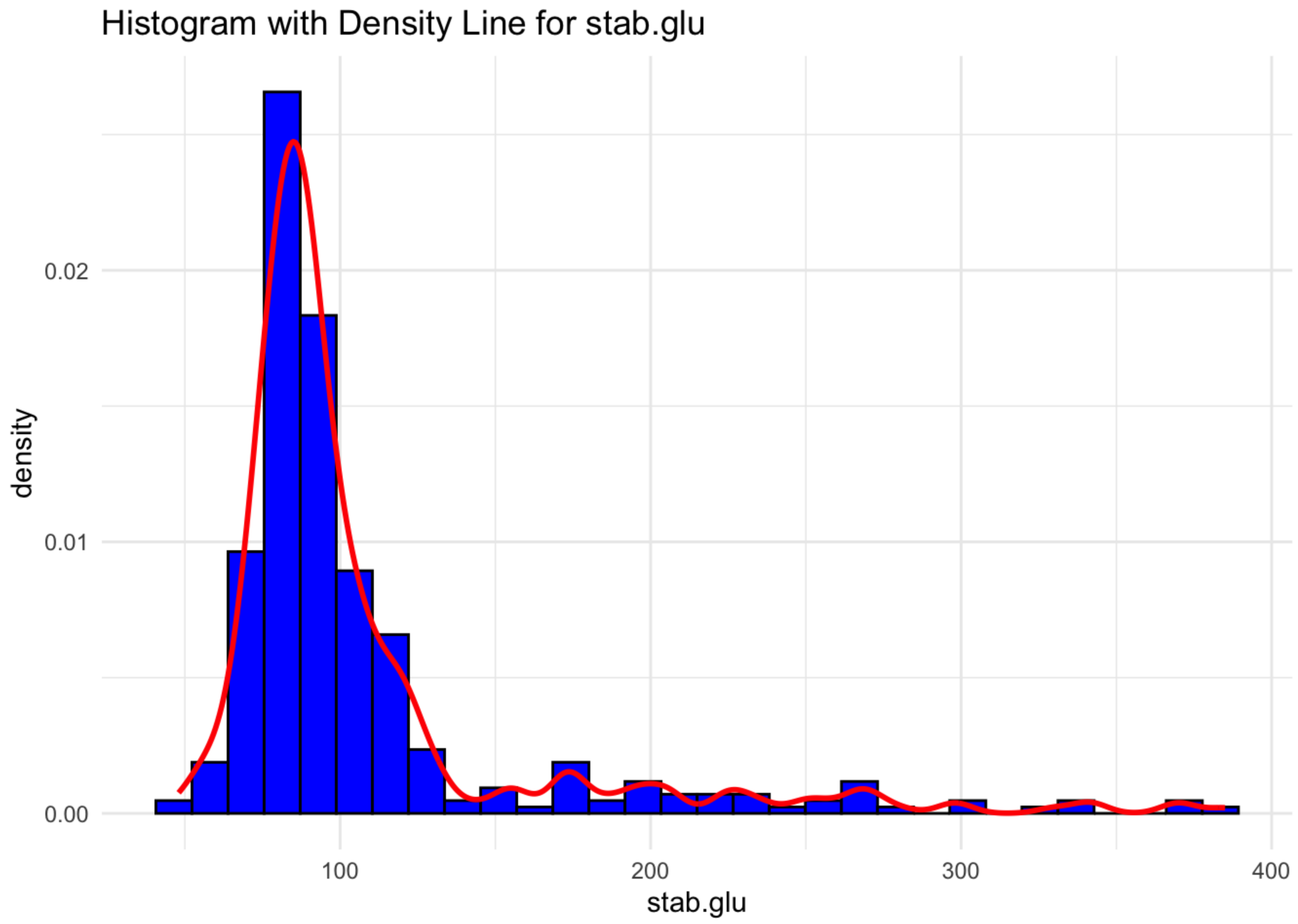
Qualitative Variables are: location, gender, frame

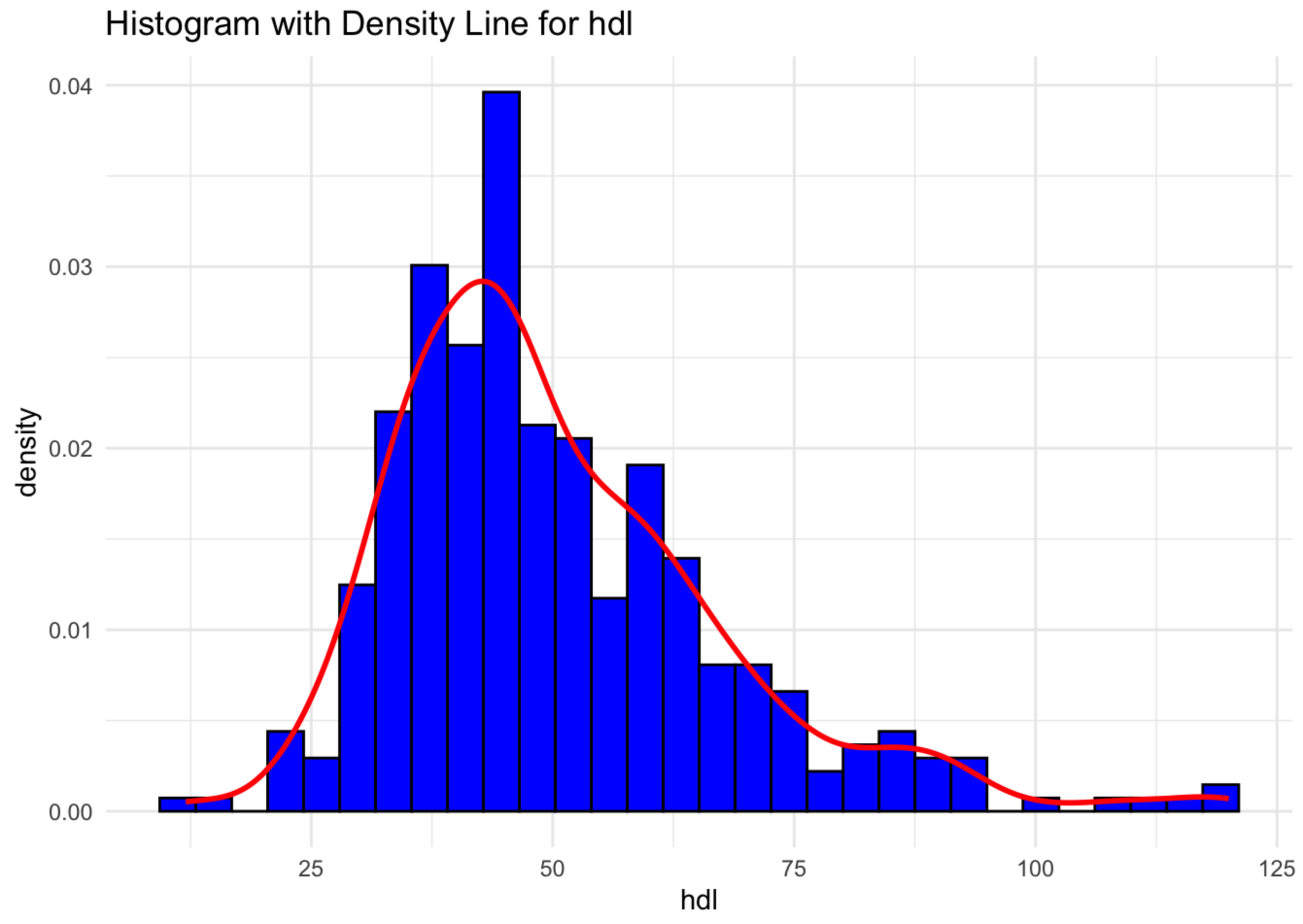
```
quant_vars <- c('chol', 'stab.glu', 'hdl', 'ratio', 'glyhb', 'age', 'height', 'weight',
'bp.1s', 'bp.1d', 'waist', 'hip', 'time.ppn')
qual_vars <- c('location', 'gender', 'frame')
```

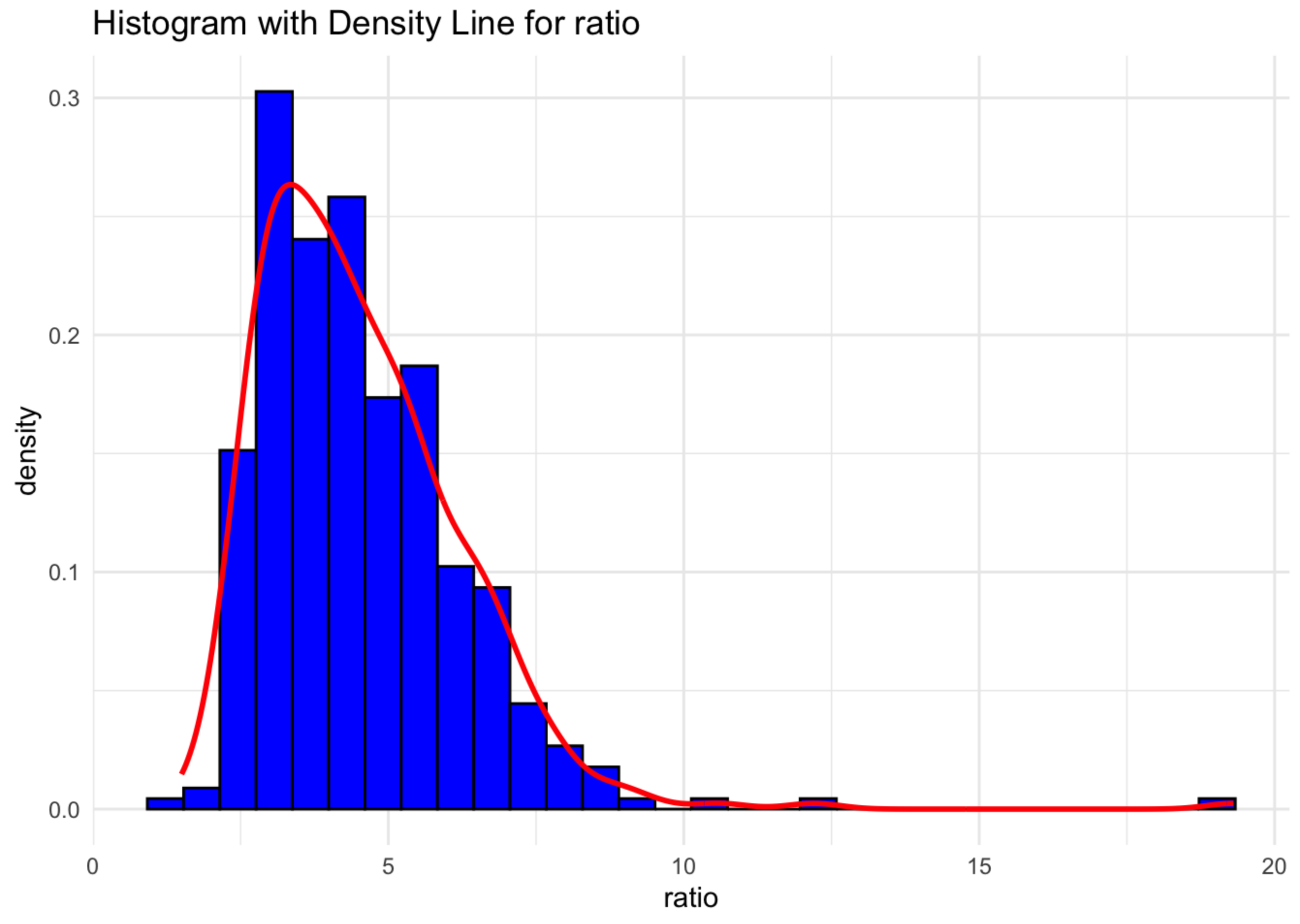
Histograms for Quantitative Variables

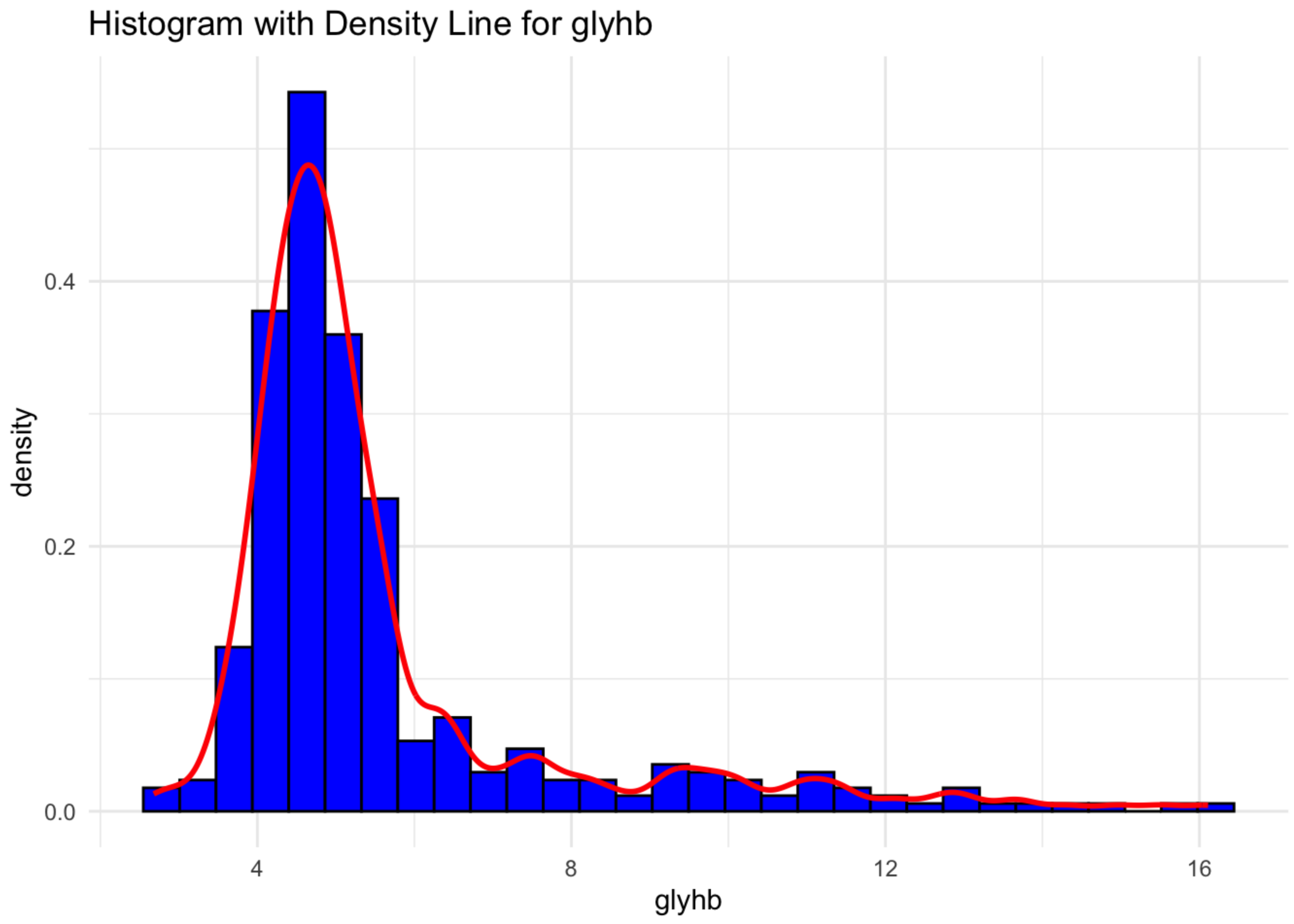
```
for (var in quant_vars) {
  p <- ggplot(df, aes(!!sym(var))) +
    geom_histogram(aes(y = after_stat(density)), bins = 30, fill = "blue", color = "blac
k") +
    geom_density(color = "red", linewidth = 1) +
    theme_minimal() +
    ggtitle(paste("Histogram with Density Line for", var))

  print(p)
}
```

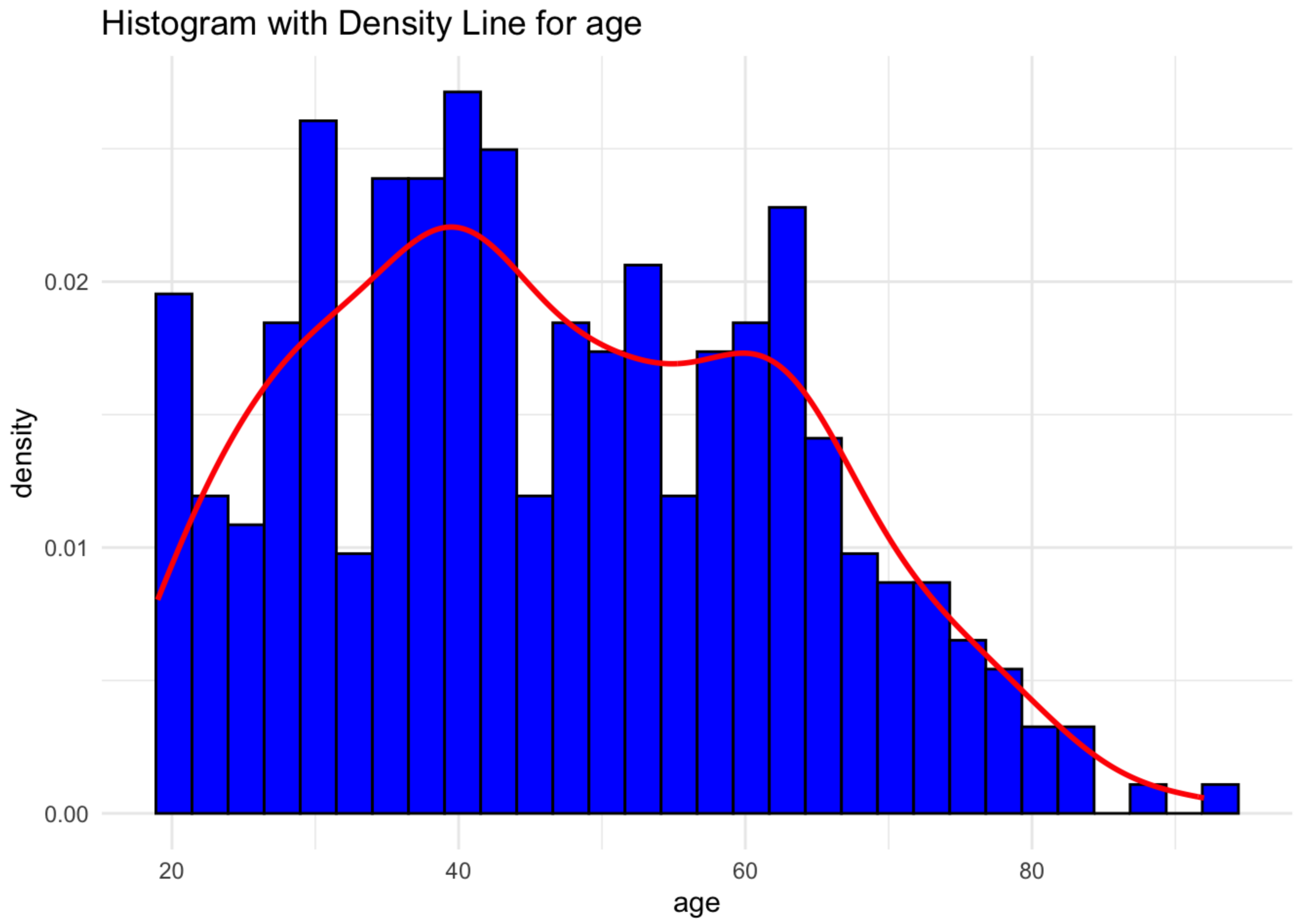

Histogram with Density Line for chol
density
0.009
0.006
0.003
0.000
100
200
300
400
chol

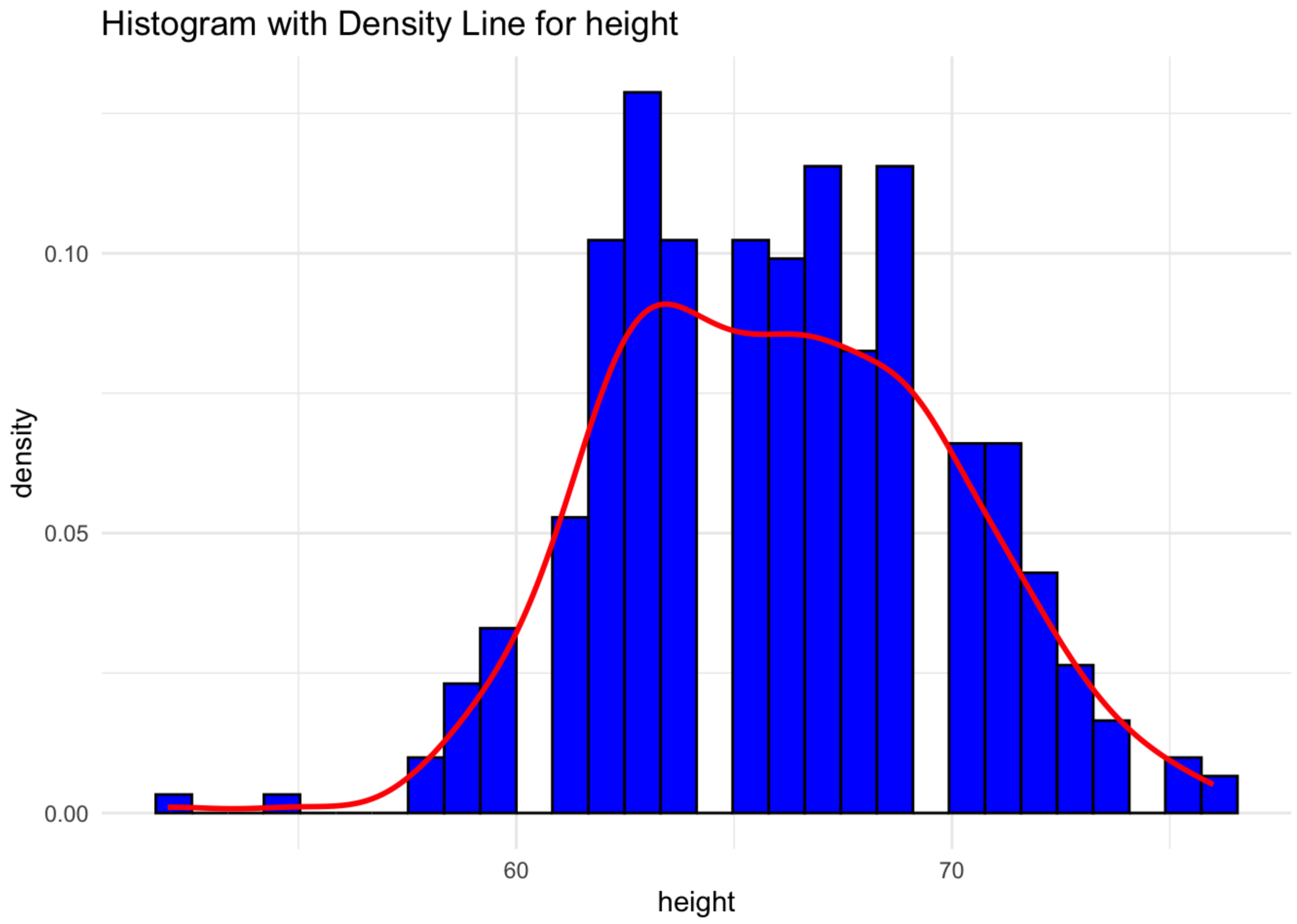

Histogram with Density Line for stab.glu
density
0.02
0.01
0.00
100
200
300
400
stab.glu

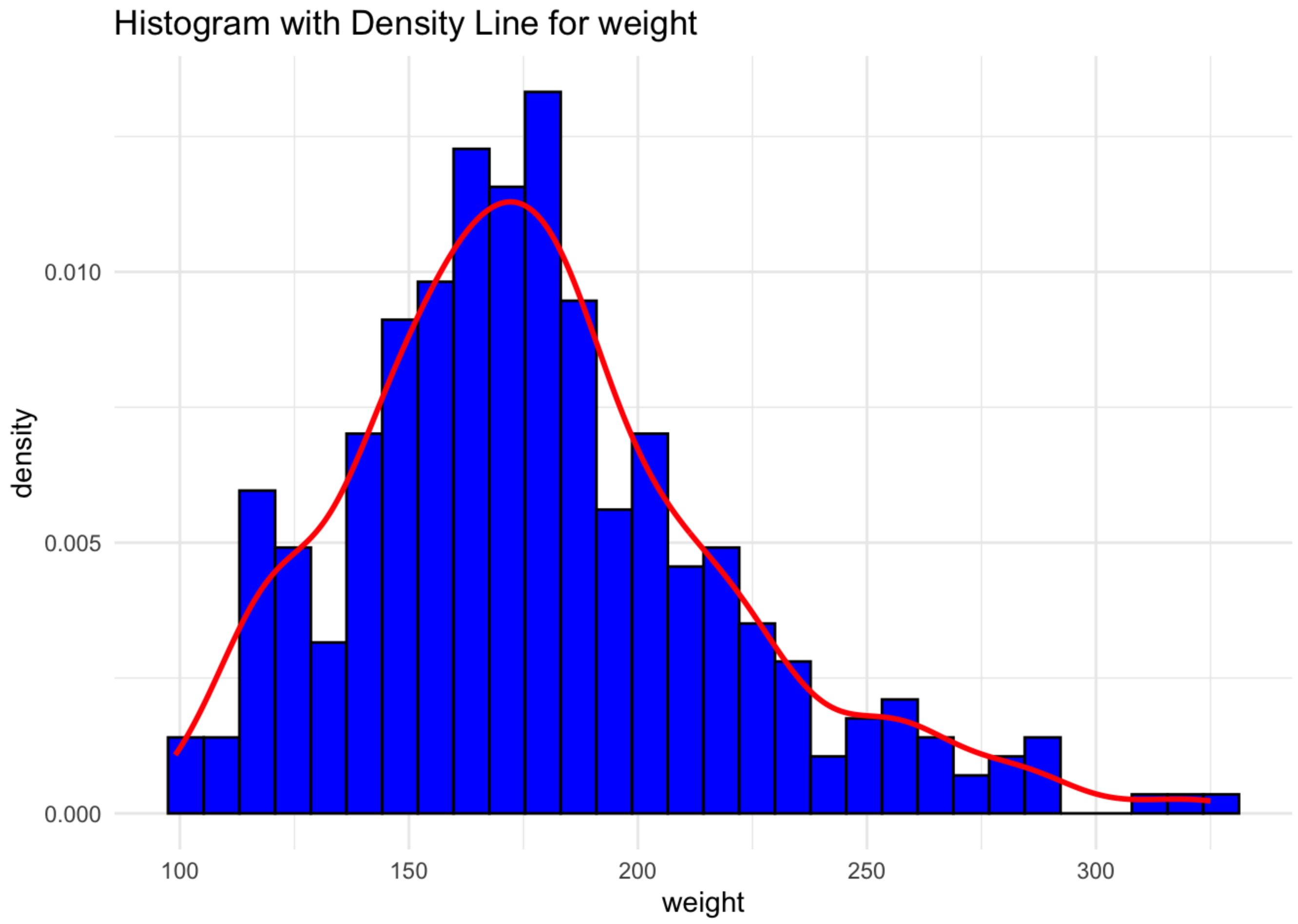

Histogram with Density Line for hdl
density
0.04
0.03
0.02
0.01
0.00
25
50
75
100
125
hdl


Histogram with Density Line for ratio
density
0.3
0.2
0.1
0.0
0
5
10
15
20
ratio

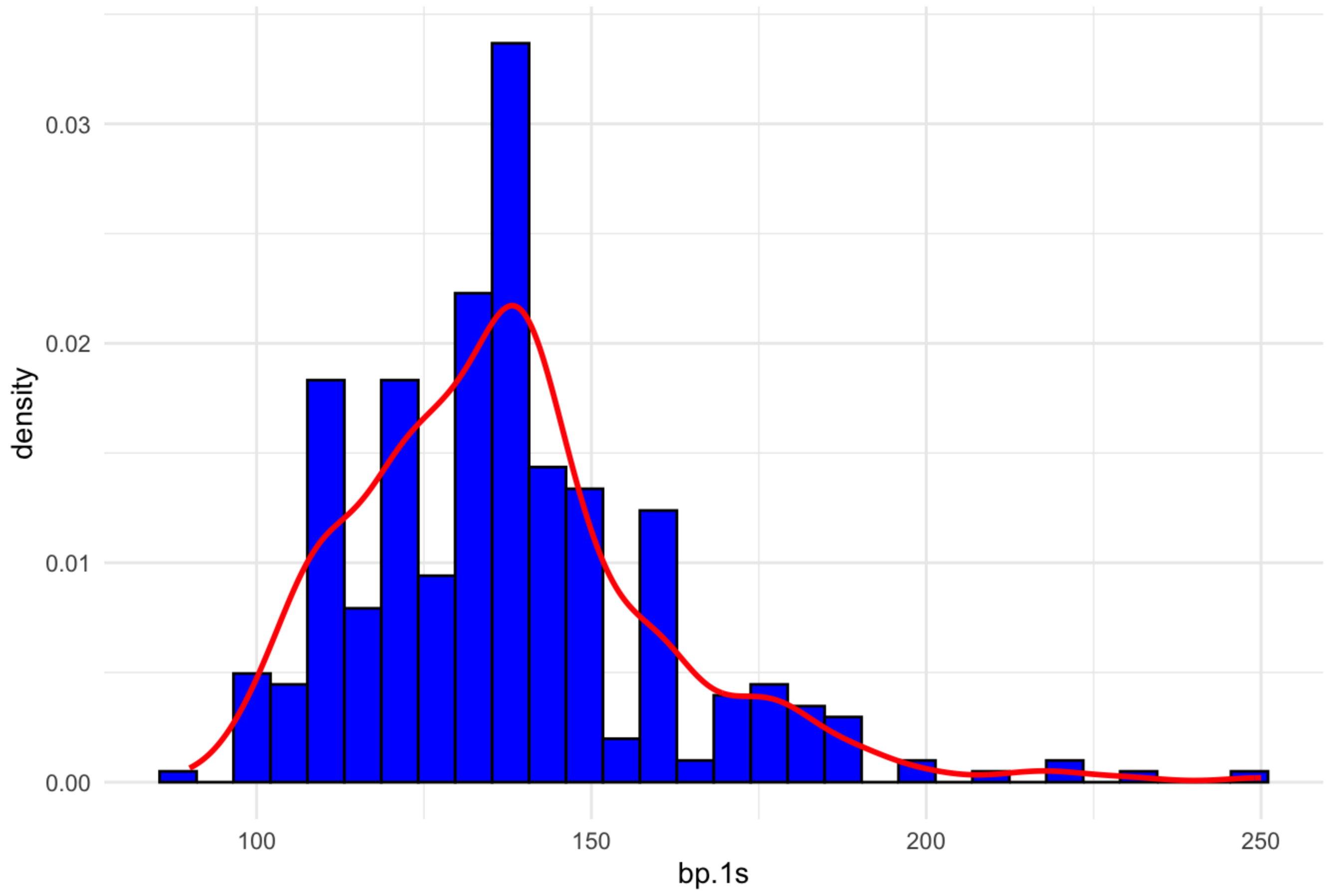

Histogram with Density Line for glyhb
density
0.0
0.2
0.4
4
8
12
16
glyhb


Histogram with Density Line for age
density
0.00
0.01
0.02
20
40
60
80
age

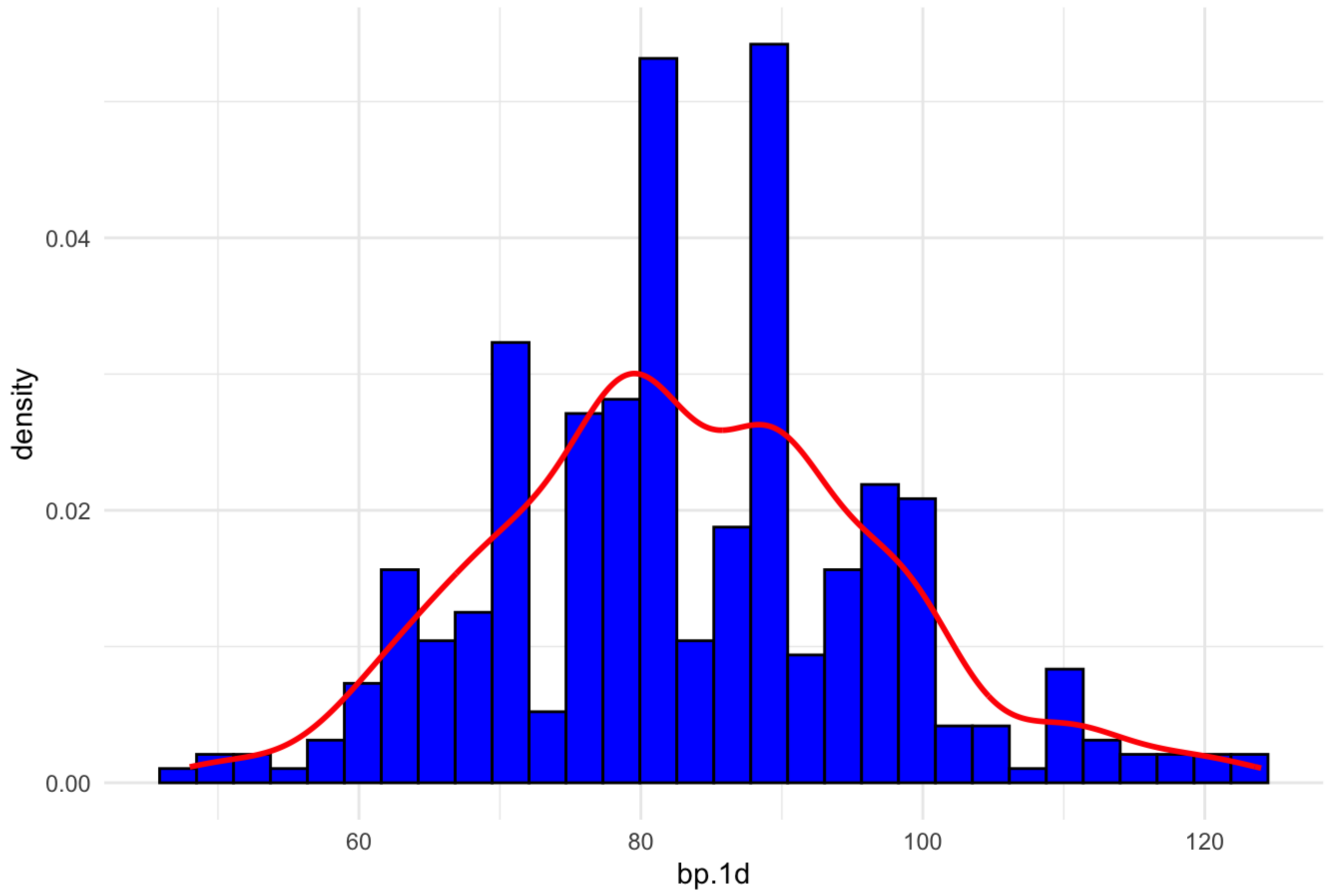

Histogram with Density Line for height
density
0.10
0.05
0.00
60
70
height

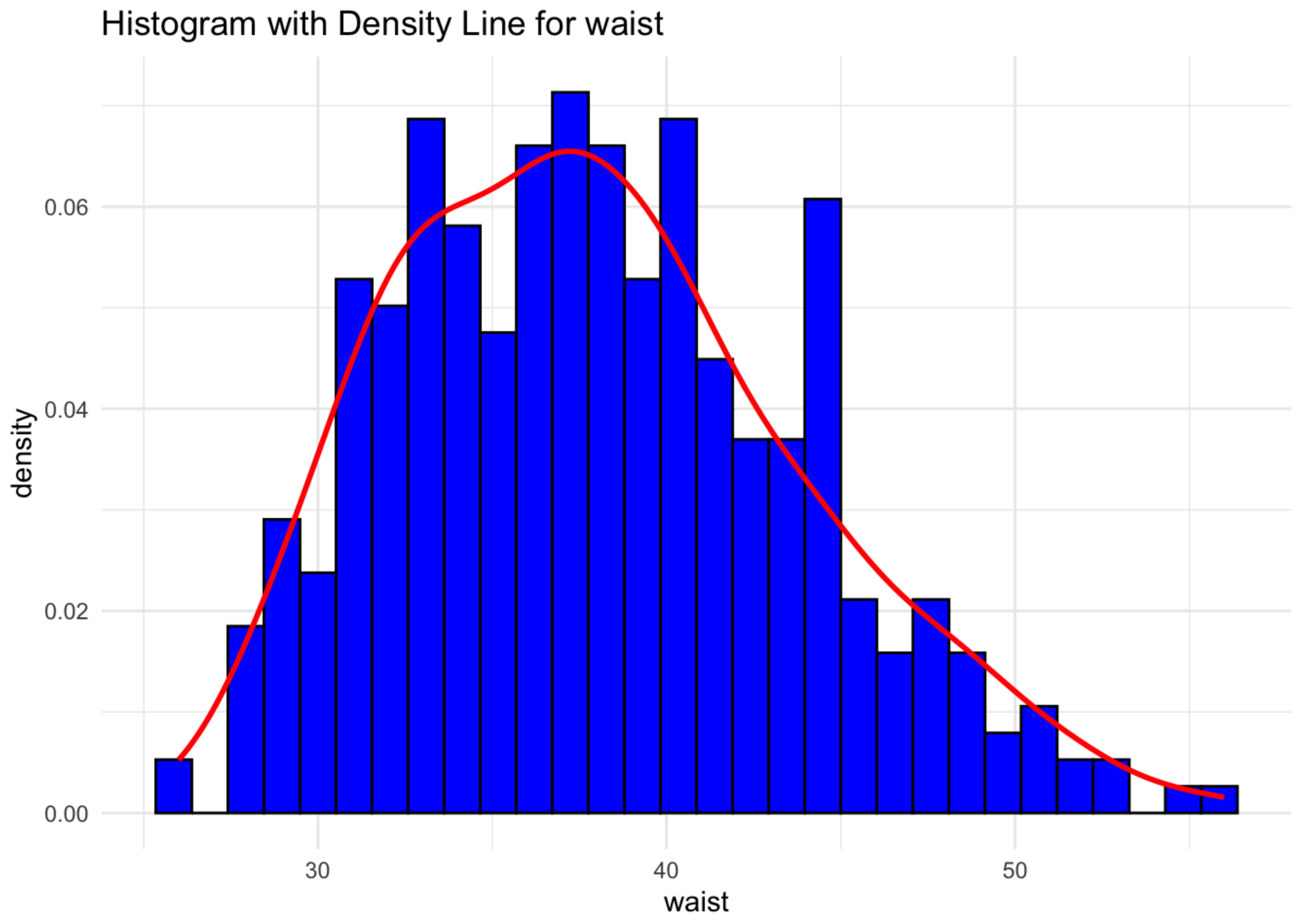

Histogram with Density Line for weight
density
0.010
0.005
0.000
100
150
200
250
300
weight

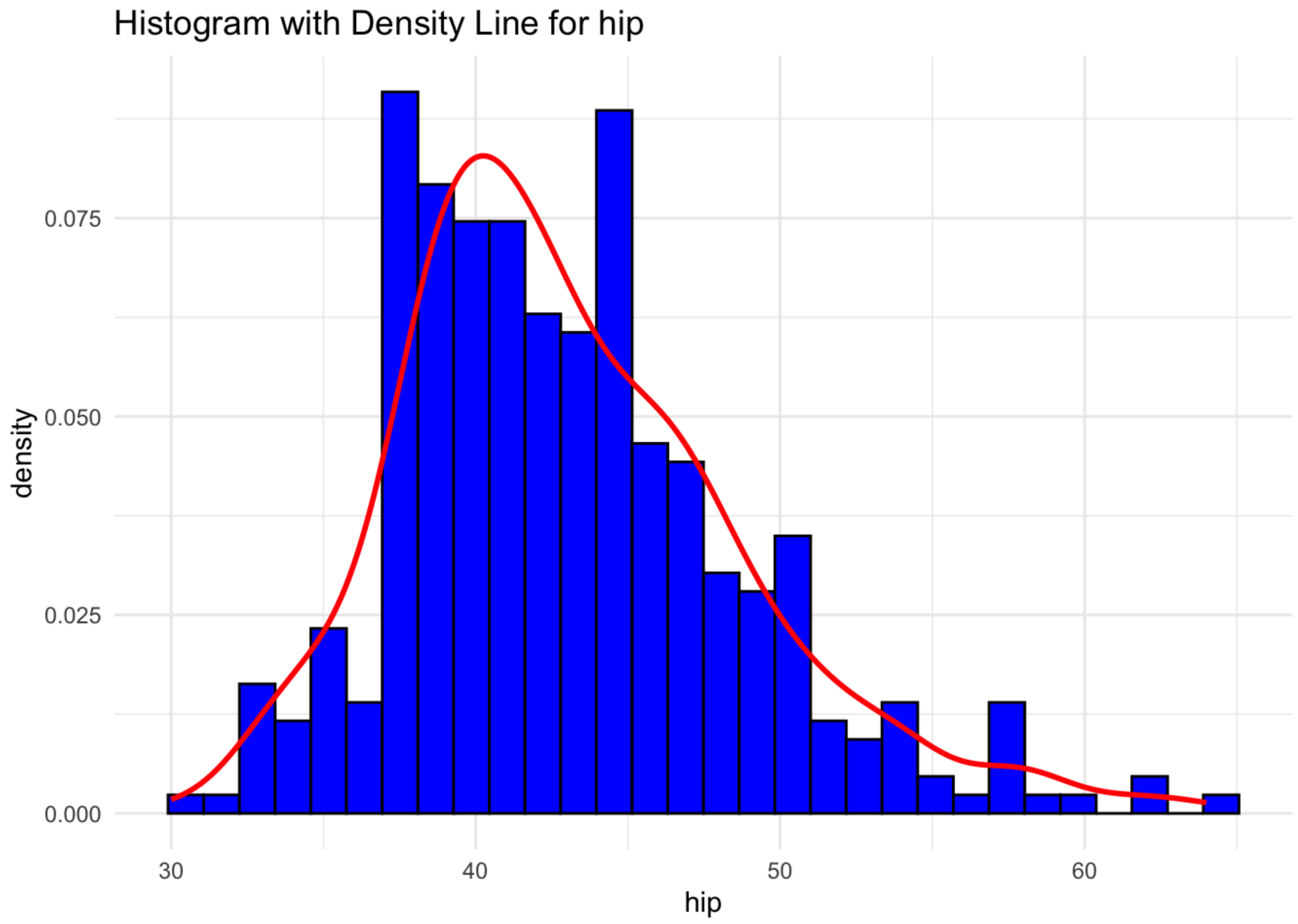
Histogram with Density Line for bp.1s

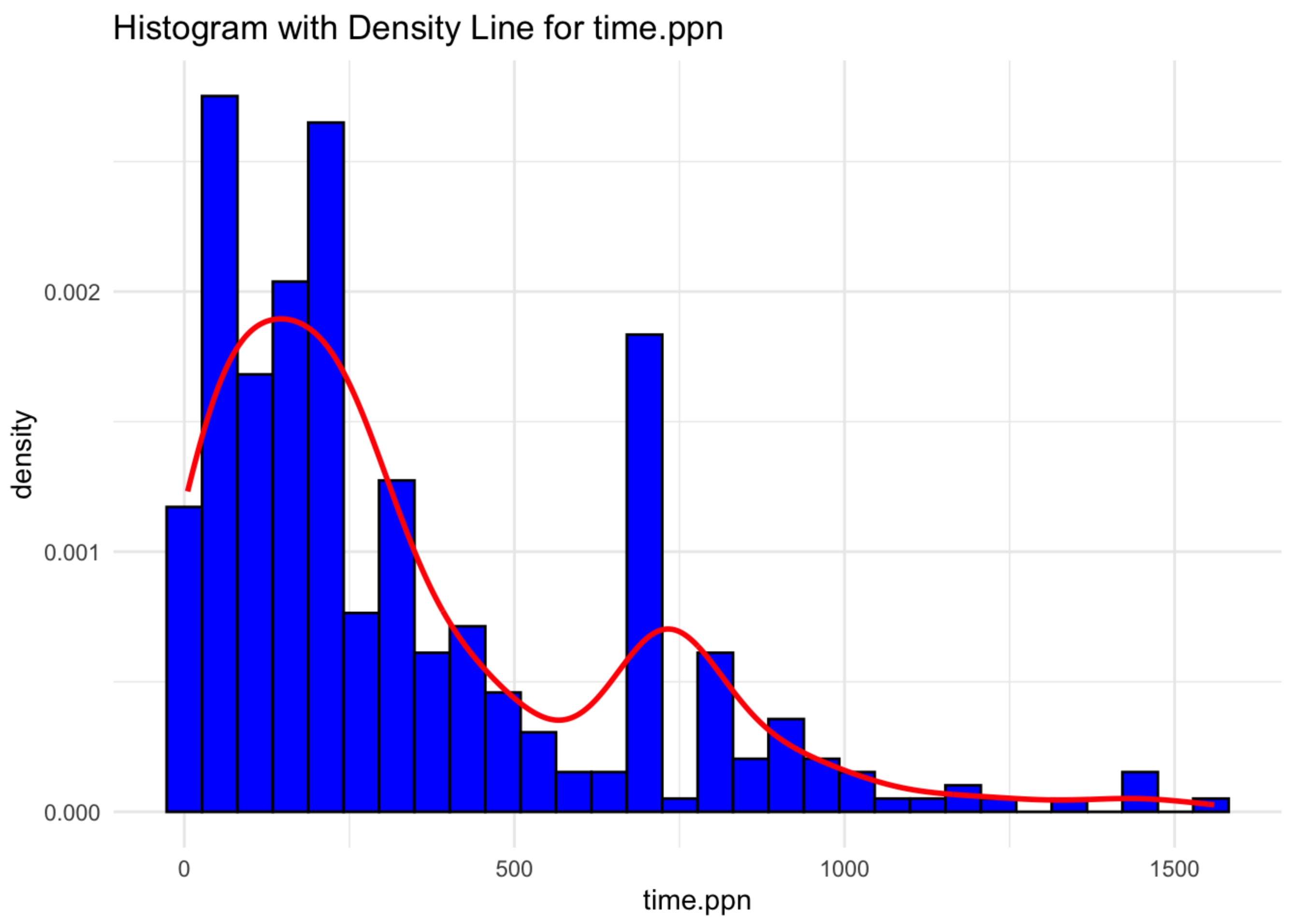


Histogram with Density Line for bp.1d




Histogram with Density Line for waist
density
0.06
0.04
0.02
0.00
30
40
50
waist


Histogram with Density Line for hip
density
0.075
0.050
0.025
0.000
30
40
50
60
hip



```
#https://stackoverflow.com/questions/74414272/how-to-replace-the-deprecated-ggplot2-func
tion-aes-string-accepting-an-arbitrar
#^ used to debug
#https://www.datacamp.com/tutorial/make-histogram-ggplot2
```

Comments on the distribution: chol: The distribution is mainly appears right skewed, suggesting a healthy population with few individuals having high cholesterol.

stab.glu: Stabilized Glucose is also right skewed,which may indicate a population with relatively normal glucose levels but includes some individuals with elevated levels.

hdl: is some what left-skewed which indicates more individuals with higher HDL levels. Which is a good sign according to Mayo Clinic

ratio: appears to be right skewed, indicating that lower ratios are more prevalent

glyhb: The right-skewness on the histogram indicates that while most individuals have glyhb levels within a normal range, there are some with significantly higher levels, potentially indicating diabetes.

Age: The age distribution seems fairly uniform, indicating a diverse age range in the dataset.

Height: have a roughly symmetrical distribution, which suggest normal variations in height among the subjects.

bp.1s: The distribution is slightly right-skewed, with most individuals having lower systolic blood pressure.

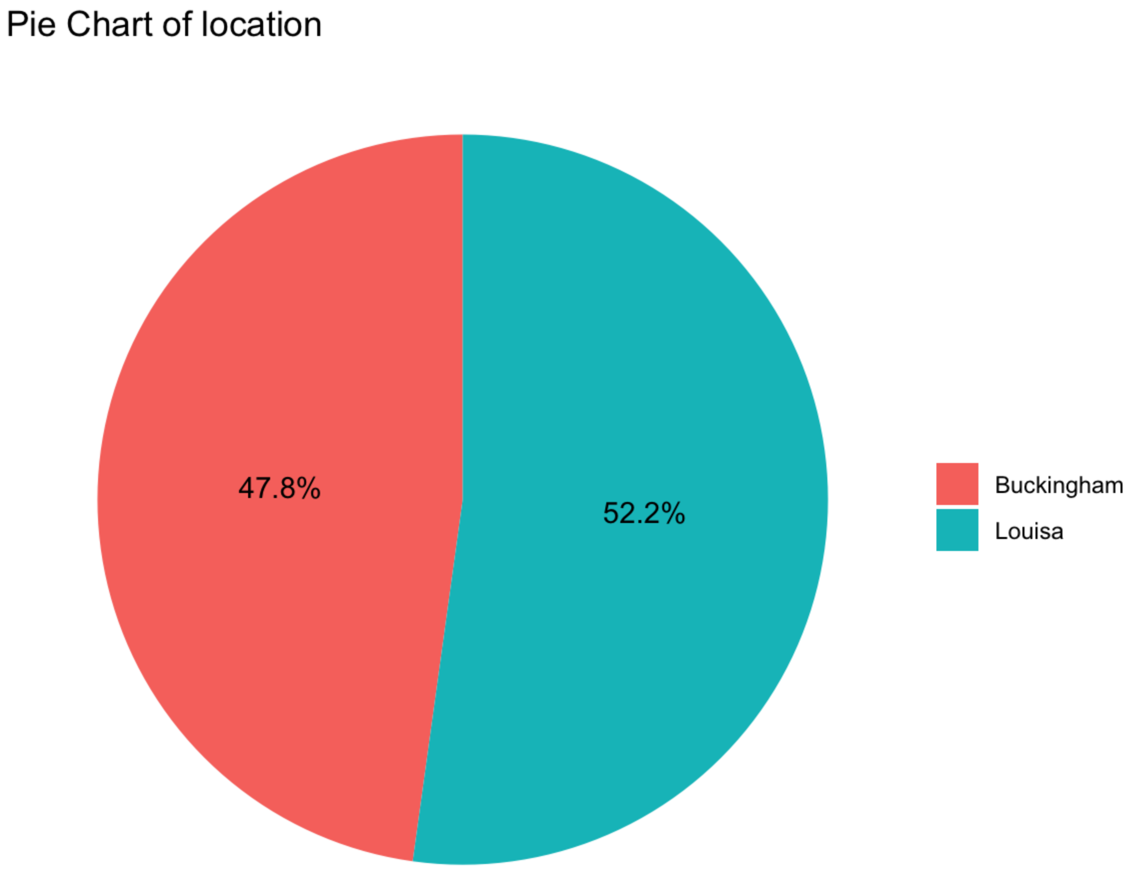
bp.1d: appears right-skewed, similar to systolic blood pressure.

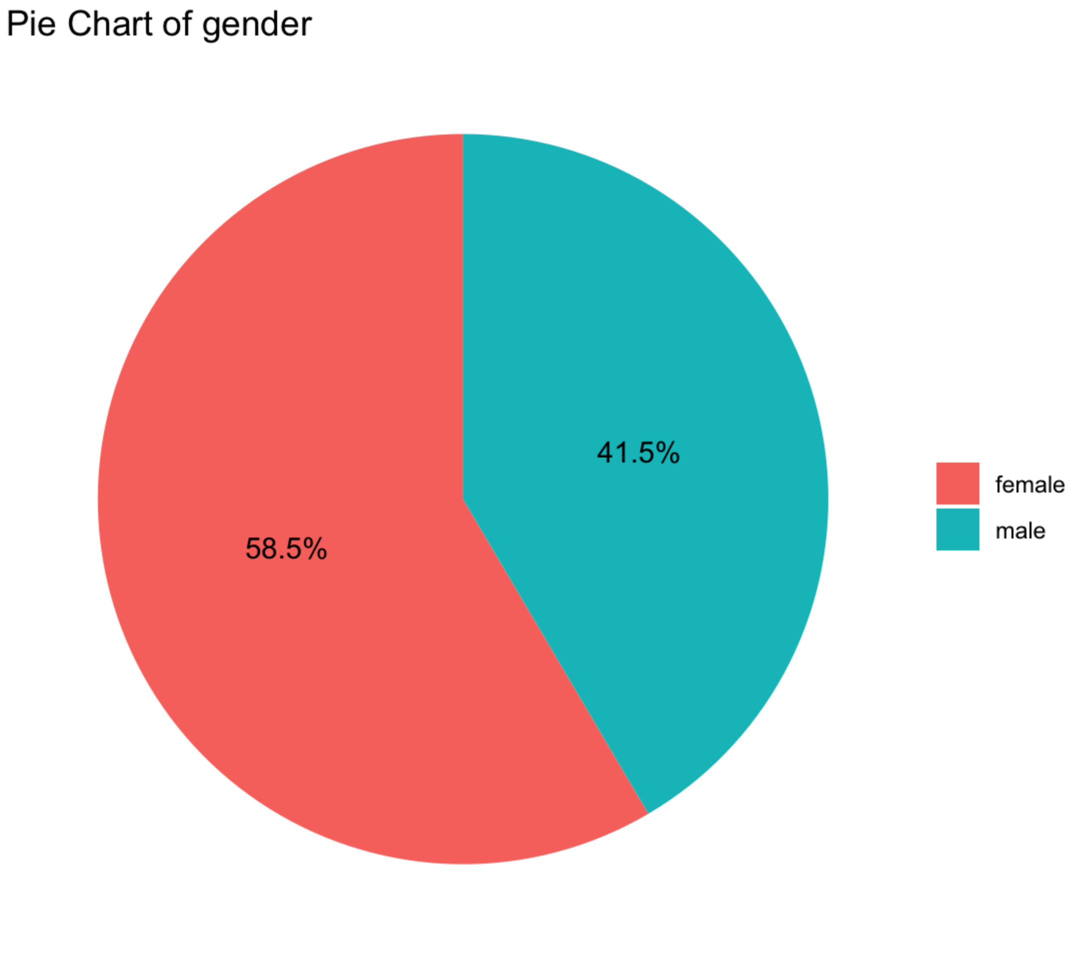
Waist and Hip Measurements: Both are right-skewed, indicating a higher frequency of smaller measurements.

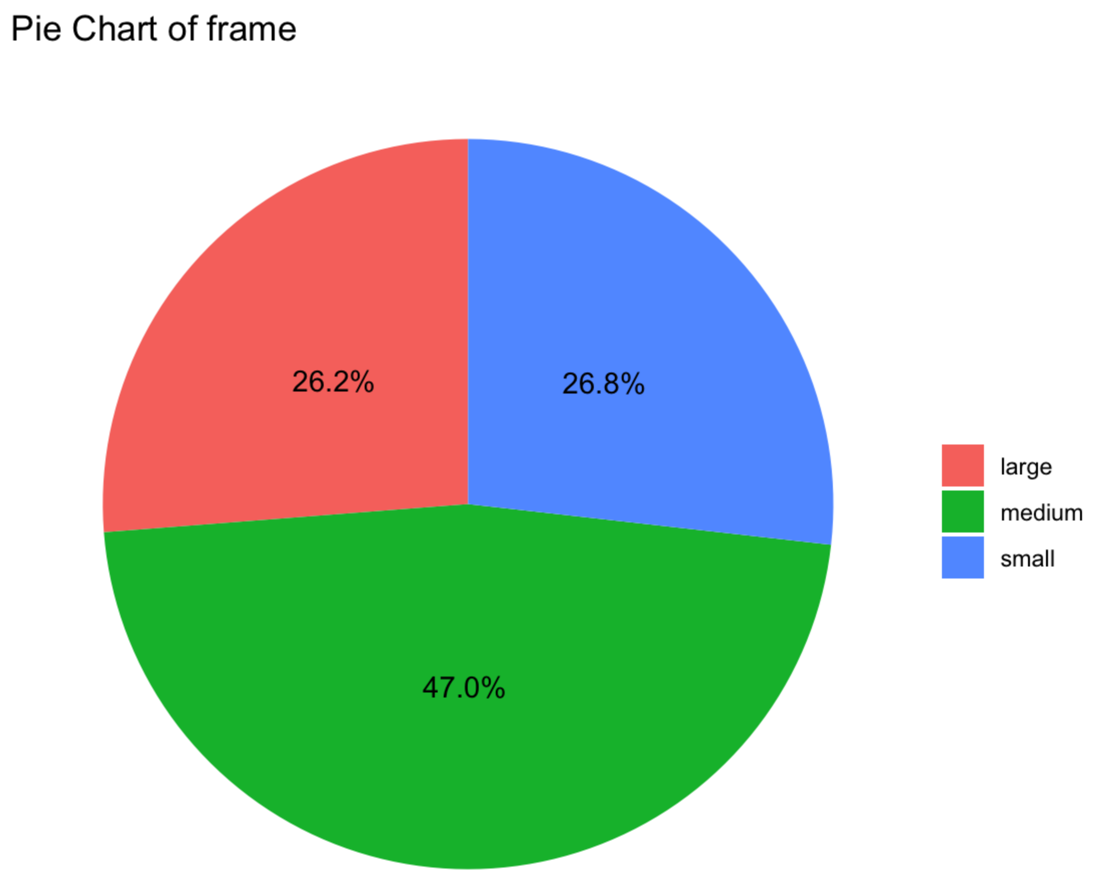
time.pp: The distribution of this variable is right-skewed, with most individuals having lower values (few high outliers).

Pie Charts for Qualitative Variables

```
for (var in qual_vars) {
  pie <- df %>%
    count(!!sym(var)) %>%
    mutate(perc = n / sum(n) * 100)

  b <- ggplot(pie, aes(x = "", y = n, fill = !!sym(var))) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    geom_text(aes(label = sprintf("%.1f%%", perc)), position = position_stack(vjust = 0.
5)) +
    theme_void() +
    theme(legend.title = element_blank()) +
    ggtitle(paste("Pie Chart of", var))
  print(b)
}
```


Pie Chart of location
47.8%
52.2%
Buckingham
Louisa


Pie Chart of gender
58.5%
41.5%
female
male



```
#https://r-graph-gallery.com/piechart-ggplot2.html
```

Comments on the pie charts:

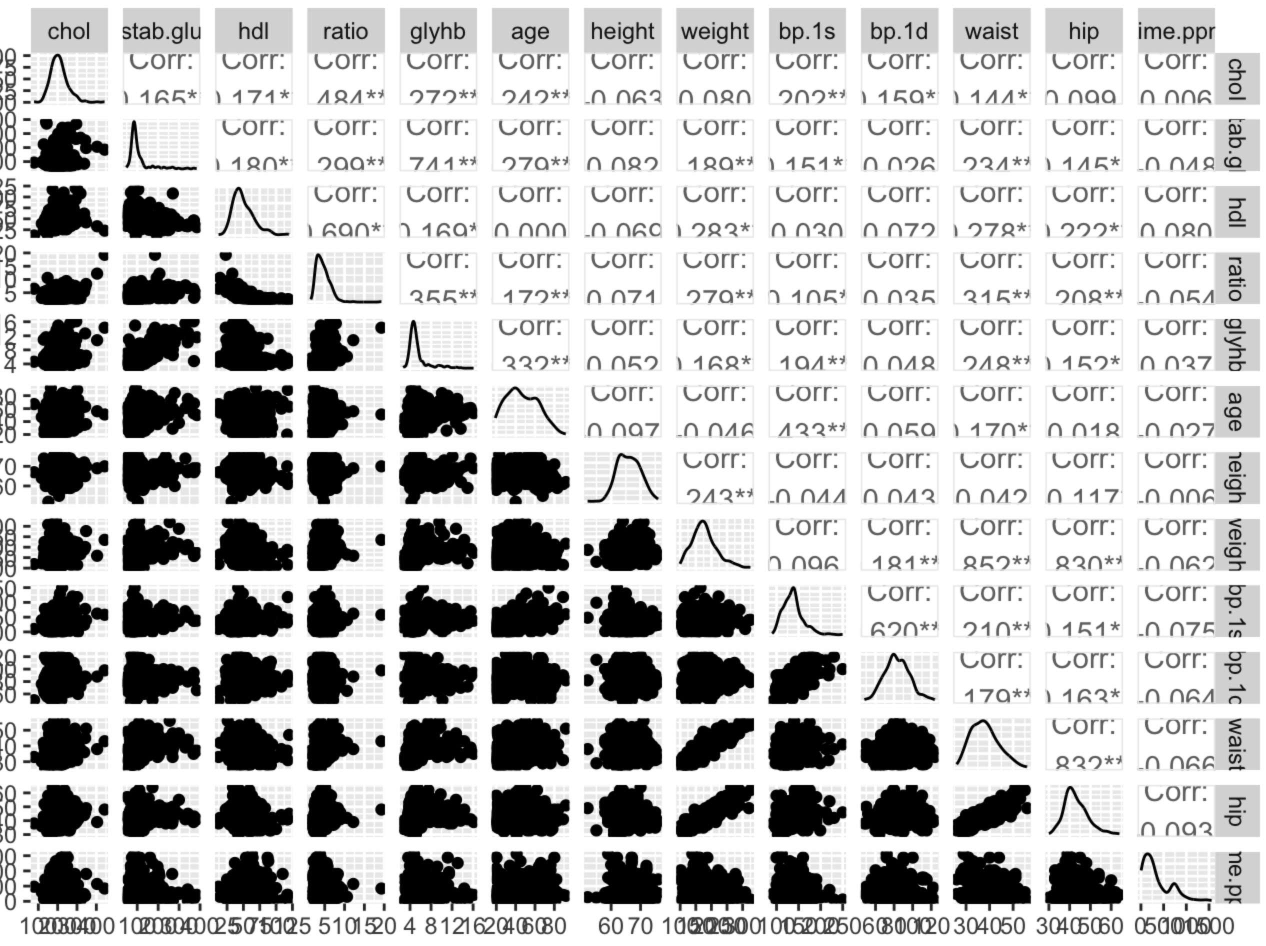
Location: The pie chart shows a distribution between two locations, Buckingham and Louisa. One location has a slightly larger representation than the other.(approx 52%)

Gender: There’s a higher percentage of females (approx. 58%) compared to males in the dataset.

Frame: The majority of the subjects have a medium body frame (approx. 47%), while the small and large frames are almost equally represented, each making up about 26% of the subjects.

Scatterplot Matrix for Quantitative Variables

```
ggpairs(df[, quant_vars])
```



```
gg <- ggpairs(df[quant_vars])
ggsave("scatterplot_matrix.png", gg, width = 20, height = 20, dpi = 300)
```

Pairwise Correlation Matrix

```
correlation_matrix <- cor(df %>% select(all_of(quant_vars)))
print(correlation_matrix)
```

```
##                  chol    stab.glu          hdl       ratio       glyhb
## chol      1.000000000  0.16544754  0.1709732770  0.48403807  0.27165218
## stab.glu  0.165447544  1.00000000 -0.1801048833  0.29889570  0.74090490
## hdl       0.170973277 -0.18010488  1.0000000000 -0.69023141 -0.16949641
## ratio     0.484038069  0.29889570 -0.6902314087  1.00000000  0.35465342
## glyhb     0.271652180  0.74090490 -0.1694964128  0.35465342  1.00000000
## age       0.241604908  0.27855141  0.0002152264  0.17156914  0.33222989
## height   -0.063230009  0.08247570 -0.0685918173  0.07089817  0.05225072
## weight    0.079789987  0.18880052 -0.2829826752  0.27889889  0.16776851
## bp.1s     0.201948705  0.15142542  0.0295089053  0.10534657  0.19442279
## bp.1d     0.159042299  0.02569721  0.0722451474  0.03484142  0.04786459
## waist     0.144089547  0.23369209 -0.2783001009  0.31549761  0.24768684
## hip       0.098597154  0.14483314 -0.2222166064  0.20789160  0.15167273
## time.ppn  0.006238501 -0.04845774  0.0799388429 -0.05382831  0.03704938
##                   age       height      weight       bp.1s       bp.1d
## chol      0.2416049084 -0.063230009  0.07978999  0.20194870  0.15904230
## stab.glu  0.2785514141  0.082475702  0.18880052  0.15142542  0.02569721
## hdl       0.0002152264 -0.068591817 -0.28298268  0.02950891  0.07224515
## ratio     0.1715691447  0.070898165  0.27889889  0.10534657  0.03484142
## glyhb     0.3322298936  0.052250721  0.16776851  0.19442279  0.04786459
## age       1.0000000000 -0.097136587 -0.04621299  0.43303227  0.05891477
## height   -0.0971365873  1.000000000  0.24329556 -0.04441181  0.04345208
## weight   -0.0462129859  0.243295558  1.00000000  0.09624288  0.18050511
## bp.1s     0.4330322675 -0.044411815  0.09624288  1.00000000  0.61984558
## bp.1d     0.0589147673  0.043452076  0.18050511  0.61984558  1.00000000
## waist     0.1702608196  0.041807866  0.85192261  0.20976399  0.17899079
## hip       0.0182966937 -0.117181984  0.82984527  0.15142640  0.16282460
## time.ppn -0.0269049474 -0.006180895 -0.06221671 -0.07490369 -0.06376264
##                waist         hip     time.ppn
## chol      0.14408955  0.09859715  0.006238501
## stab.glu  0.23369209  0.14483314 -0.048457737
## hdl      -0.27830010 -0.22221661  0.079938843
## ratio     0.31549761  0.20789160 -0.053828314
## glyhb     0.24768684  0.15167273  0.037049379
## age       0.17026082  0.01829669 -0.026904947
## height    0.04180787 -0.11718198 -0.006180895
## weight    0.85192261  0.82984527 -0.062216714
## bp.1s     0.20976399  0.15142640 -0.074903689
## bp.1d     0.17899079  0.16282460 -0.063762636
## waist     1.00000000  0.83233707 -0.065861241
## hip       0.83233707  1.00000000 -0.092519540
## time.ppn -0.06586124 -0.09251954  1.000000000
```

Observations/relationship: glyhb and stab.glu have a relatively strong positive correlation (0.74), suggesting higher glucose levels are associated with higher Glycosolated Hemoglobin levels.

hdl and ratio have a strong negative correlation (-0.69), indicating that as HDL levels increase, the ratio decreases.

weight and waist have a very strong positive correlation (0.85), suggesting that weight and waist measurements are closely related.

weight and hip also a very strong positive correlation (0.83), reinforcing the relationship between weight and body measurements.

bp.1s and bp.1d have a moderate positive correlation (0.62), showing a relationship between systolic and diastolic blood pressures.

The correlations observed are consistent with medical and physiological expectations. For example, the relationship between body measurements like weight, waist, and hip and blood pressure readings reflect common health patterns. The strong correlation between glyhb and stab.glu is particularly relevant for this study, as it directly ties into the research question regarding diabetes diagnosis.glyhb is the response variable of interest in the study. It implies that interventions or lifestyle changes that effectively manage stabilized glucose levels could have a direct and significant impact on glyhb levels, therefore influencing diabetes management.

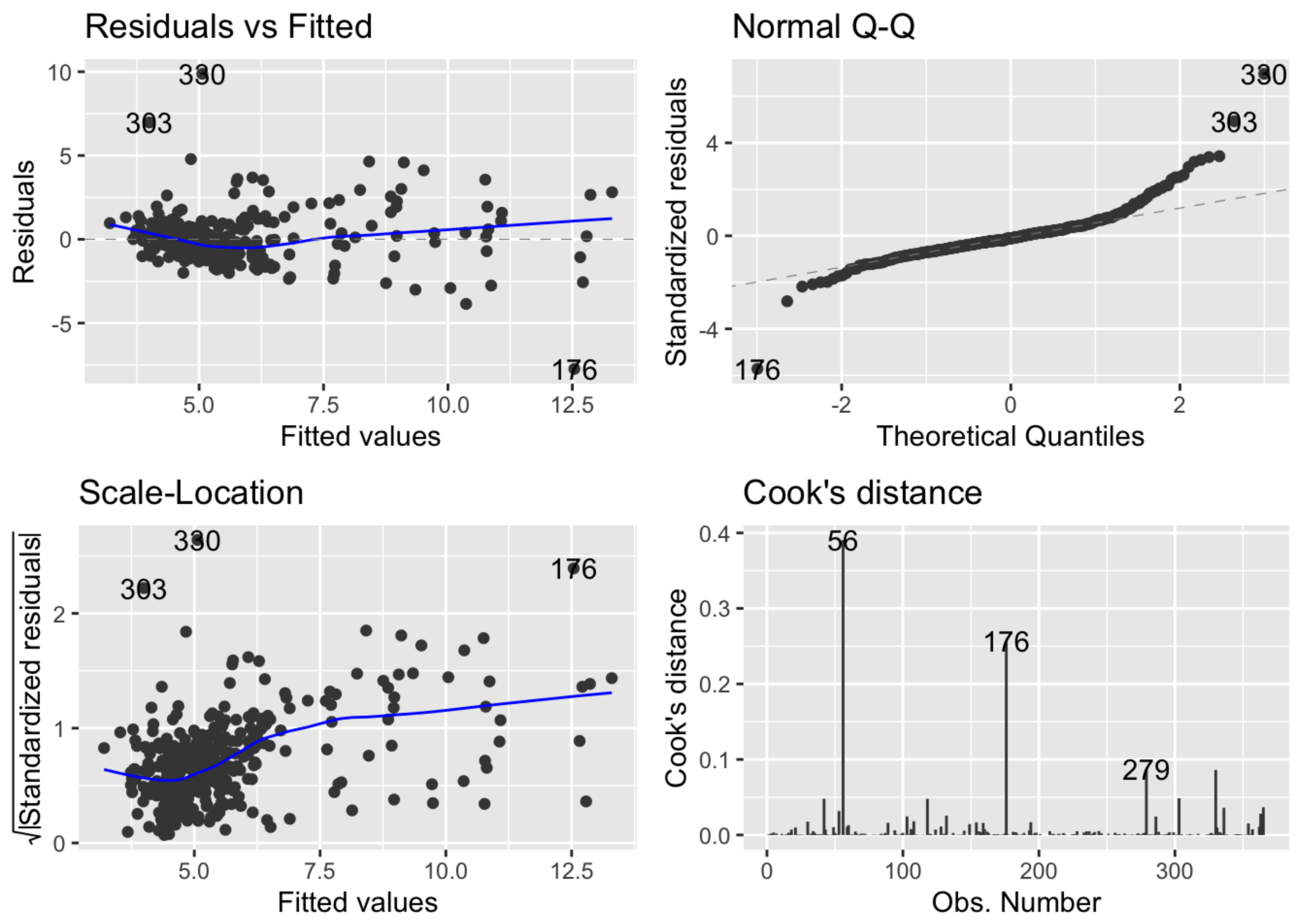
```
model1 <- lm(glyhb ~ ., data = df)

summary(model1)
```

```
##
## Call:
## lm(formula = glyhb ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.7311 -0.7123 -0.1449  0.4931  9.8768
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -2.2998047  2.2783760  -1.009   0.3135
## chol            0.0050363  0.0033584   1.500   0.1346
## stab.glu        0.0275617  0.0015274  18.045   <2e-16 ***
## hdl            -0.0034033  0.0104283  -0.326   0.7444
## ratio           0.0851633  0.1142484   0.745   0.4565
## locationLouisa -0.2337873  0.1608593  -1.453   0.1470
## age             0.0134974  0.0060629   2.226   0.0266 *
## gendermale     -0.0850168  0.2517108  -0.338   0.7358
## height          0.0227918  0.0306269   0.744   0.4573
## weight         -0.0041222  0.0052581  -0.784   0.4336
## framemedium     0.2799754  0.2102752   1.331   0.1839
## framesmall      0.2281141  0.2642373   0.863   0.3886
## bp.1s           0.0027597  0.0048761   0.566   0.5718
## bp.1d          -0.0014471  0.0075742  -0.191   0.8486
## waist           0.0295509  0.0308031   0.959   0.3380
## hip             0.0157799  0.0349919   0.451   0.6523
## time.ppn        0.0005652  0.0002494   2.266   0.0241 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.439 on 349 degrees of freedom
## Multiple R-squared:  0.6025, Adjusted R-squared:  0.5843
## F-statistic: 33.06 on 16 and 349 DF,  p-value: < 2.2e-16
```

```
#df$location <- as.factor(df$location)
#df$gender <- as.factor(df$gender)
#df$frame <- as.factor(df$frame)
```

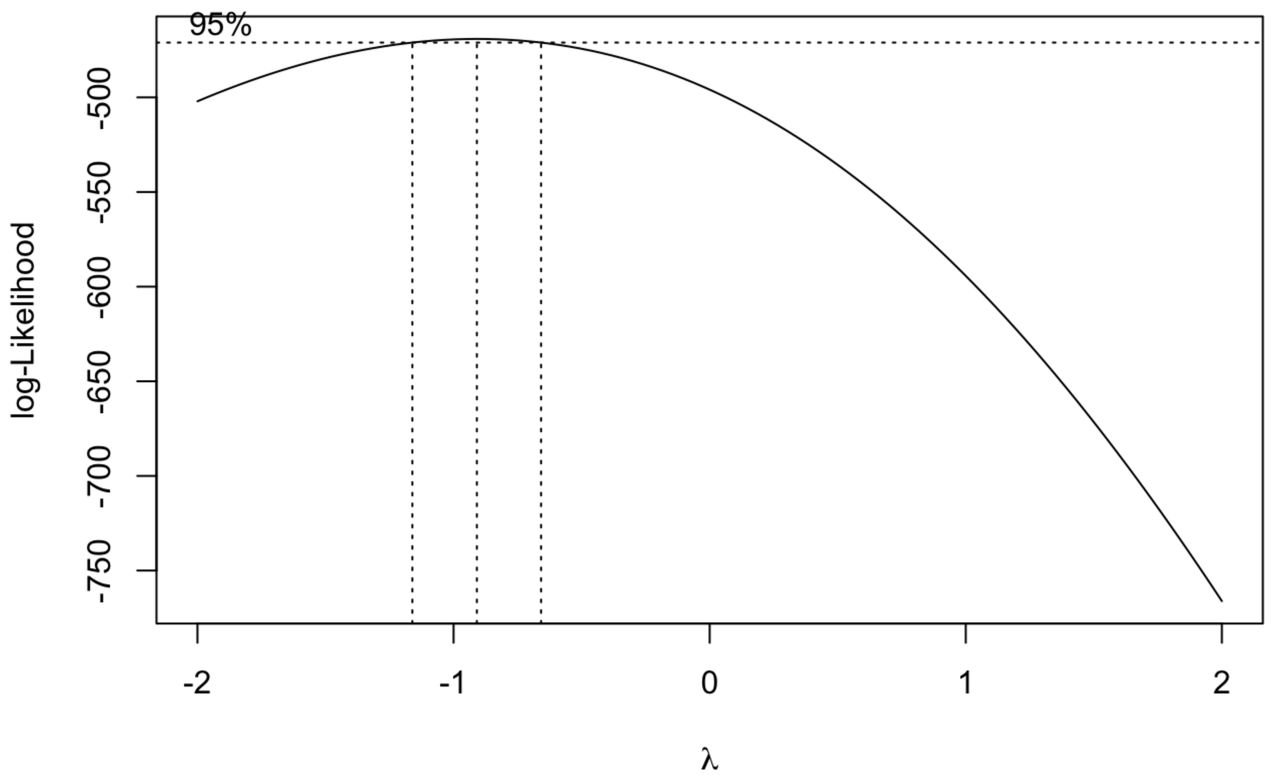
Diagnostic plots

```
#par(mfrow=c(2,2))  # Arrange the plots in a 2x2 grid
#plot(model)       # Diagnostic plots

#more cleaner outcome
autoplot(model1, which = 1:4, nrow = 2, ncol = 2)
```



Comments: Residual vs Filled plot show some non-random pattern of residuals, a curve, there's a slight funnel shape indicating potential heteroscedasticity, which might indicate that the relationship between the predictors and the response is not perfectly linear. Points in the Q-Q plot deviate from the line in the tails, especially on the right-hand side, which suggests that the residuals may have a heavy-tailed distribution. This can be a sign that the assumption of normality is not fully met. For the Scale-Location Plot, the spread of residuals increasing with the fitted values almost forming a fan like shape indicated heteroscedasticity, this means that the variances of the residuals are not constant across levels of the fitted values. Cook's Distance Plot indicates that there are a few observations with relatively high Cook's distance, specifically observations labeled 176, 56, and 279. These points can be potential outliers or maybe influential observations that might disproportionately influencing the model's coefficients. The model summary indicates that stab.glu, age, and time.ppn are statistically significant predictors. The Multiple R-squared value of 0.6025 suggests that the model explains about 60% of the variability in the response variable (relatively high).

```
boxcox_result <- boxcox(glyhb ~ ., data = df)
```



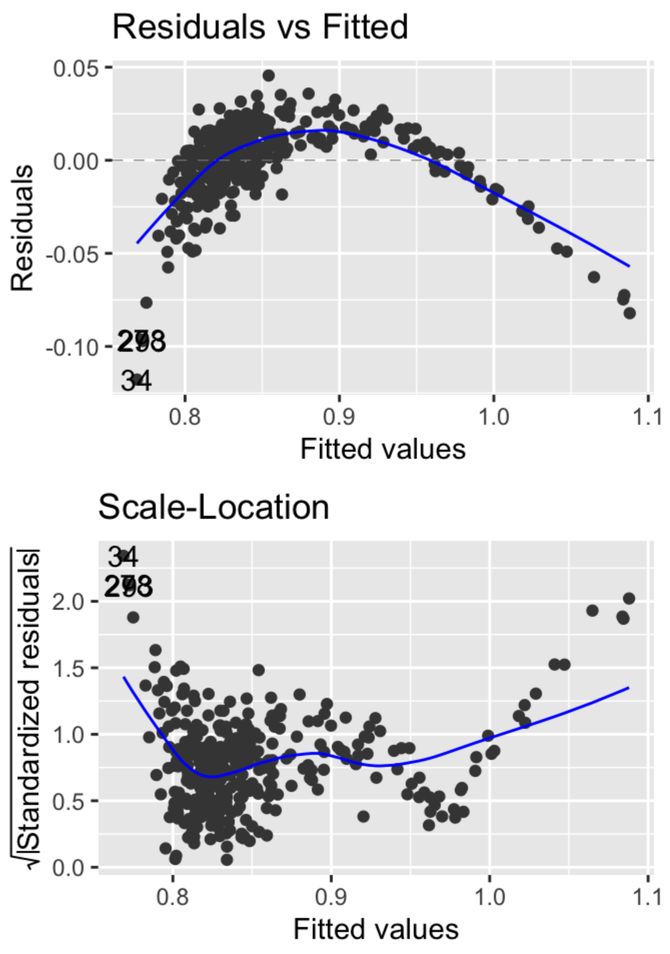
```
lambda <- boxcox_result$x[which.max(boxcox_result$y)]
cat("Optimal lambda for transformation:", lambda, "\n")
```

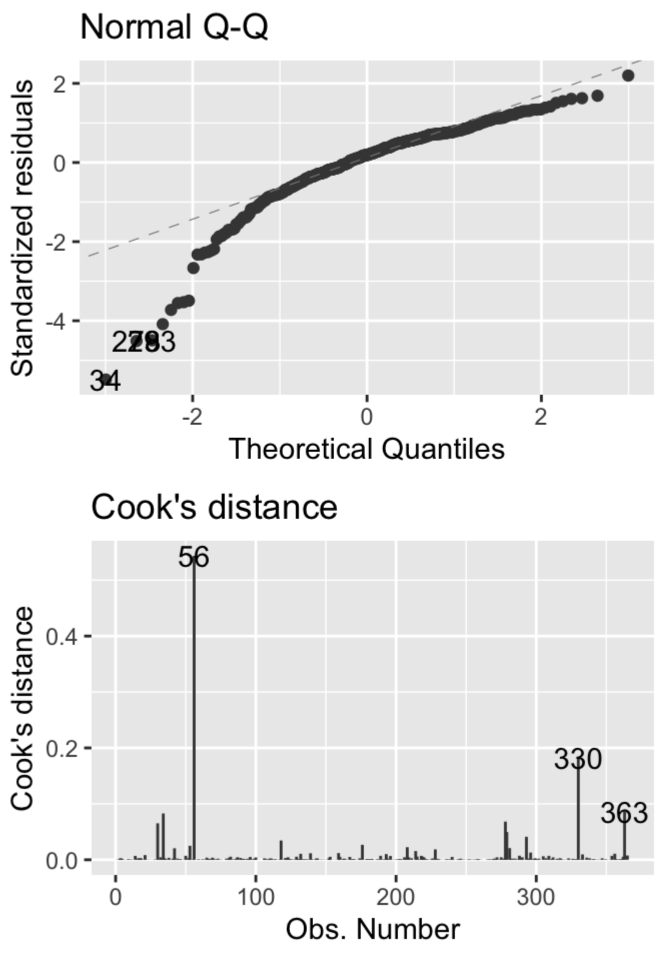
```
## Optimal lambda for transformation: -0.9090909
```

```
df$glyhb_star <- if(lambda == 0) log(df$glyhb) else (df$glyhb^lambda - 1) / lambda

# Creating a new model with the transformed response variable
model2 <- lm(glyhb_star ~ ., data = df)
summary(model2)
```

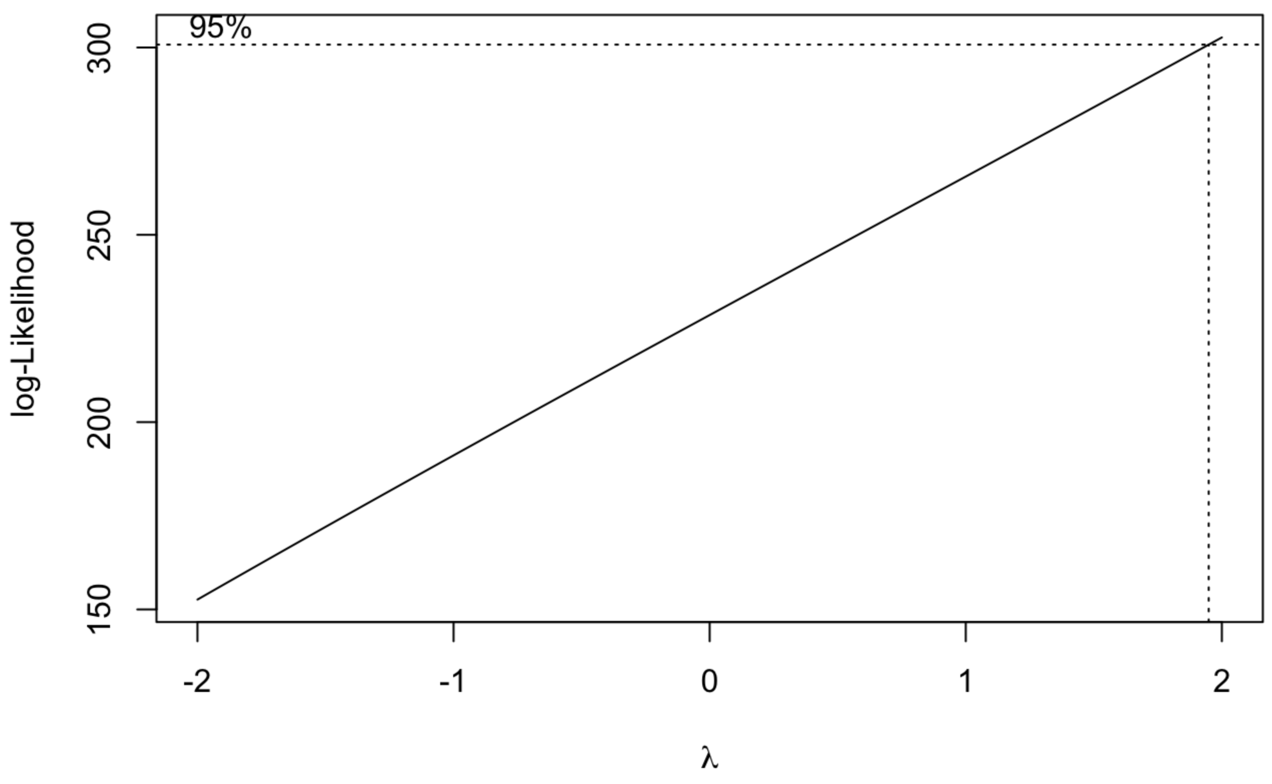
```
## 
## Call:
## lm(formula = glyhb_star ~ ., data = df)
## 
## Residuals:
##       Min        1Q    Median        3Q       Max 
## -0.117774 -0.008527  0.004154  0.014226  0.045514 
## 
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)    
## (Intercept)     6.726e-01  3.488e-02  19.282  < 2e-16 ***
## chol           -1.349e-05  5.151e-05  -0.262 0.793610    
## stab.glu       -1.018e-04  3.247e-05  -3.135 0.001863 ** 
## hdl            -5.424e-05  1.595e-04  -0.340 0.733939    
## ratio           5.254e-05  1.748e-03   0.030 0.976039    
## glyhb           2.518e-02  8.184e-04  30.769  < 2e-16 ***
## locationLouisa -2.372e-03  2.467e-03  -0.961 0.337005    
## age             3.239e-04  9.335e-05   3.470 0.000586 ***
## gendermale      2.235e-03  3.849e-03   0.581 0.561834    
## height          3.776e-05  4.686e-04   0.081 0.935829    
## weight         -3.405e-05  8.046e-05  -0.423 0.672365    
## framemedium    -3.025e-03  3.223e-03  -0.939 0.348606    
## framesmall     -4.723e-03  4.044e-03  -1.168 0.243640    
## bp.1s           3.135e-05  7.458e-05   0.420 0.674483    
## bp.1d          -4.959e-05  1.158e-04  -0.428 0.668761    
## waist           5.391e-04  4.715e-04   1.143 0.253731    
## hip             5.278e-04  5.351e-04   0.986 0.324687    
## time.ppn        5.149e-07  3.841e-06   0.134 0.893443    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.02199 on 348 degrees of freedom
## Multiple R-squared:  0.8722, Adjusted R-squared:  0.8659 
## F-statistic: 139.7 on 17 and 348 DF,  p-value: < 2.2e-16
```

```
# Diagnostic plots for the new model
autoplot(model2, which = 1:4, nrow = 2, ncol = 2)
```





```
# Applying Box-Cox transformation again on Model 2
model2_box <- boxcox(glyhb_star ~ ., data = df, plot = TRUE)

# Applying Box-Cox transformation to Model 2
boxcox_result_model2 <- boxcox(glyhb_star ~ ., data = df, plot = TRUE)
```



```
# Finding the lambda value for the best transformation for Model 2
lambda_model2 <- boxcox_result_model2$x[which.max(boxcox_result_model2$y)]
cat("Optimal lambda for Model 2:", lambda_model2, "\n")
```

```
## Optimal lambda for Model 2: 2
```

Diagnostic Plot The Box-Cox transformation plot suggests that a log transformation (lambda close to 0) is appropriate for the response variable. The diagnostic plots show some concerns with non-linearity and heteroscedasticity, as well as potential outliers or influential observations. Even after a log transformation, there are still some issues with the fit of the model. The patterns in the Residuals vs Fitted and the Scale-Location plots indicate potential problems with non-constant variance and non-linearity, and the Normal Q-Q plot suggests that the residuals may not be normally distributed. The model summary indicates that several predictors are statistically significant, and the overall fit of the model is good based on the R-squared value. Applying Box-Cox to Model 2 confirms that the log transformation is still the most appropriate, which is consistent with the initial findings.

```
set.seed(372)  # Set the seed

data_size <- nrow(df) # 366


# Calculate the size of the training set (70% of the dataset)
train_size <- round(0.7 * data_size) # 256

# Create a random sample of row indices for the training set
train_indices <- sample(seq_len(data_size), size = train_size)

# Split the data into training and test sets
train_data <- df[train_indices, ]
test_data <- df[-train_indices, ]
```

Selection of first-order effects

```
# Fit the model with all first-order effects
model3 <- lm(glyhb_star ~ ., data = train_data)

model3_summary <- summary(model3)

print(model3_summary)
```

```
##
## Call:
## lm(formula = glyhb_star ~ ., data = train_data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.106544 -0.010054  0.004601  0.013803  0.040501
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     5.979e-01  4.512e-02  13.251  < 2e-16 ***
## chol           -8.971e-05  6.908e-05  -1.299  0.19533
## stab.glu       -1.285e-04  3.940e-05  -3.261  0.00127 **
## hdl             1.395e-04  2.318e-04   0.602  0.54795
## ratio           4.292e-03  2.584e-03   1.661  0.09805 .
## glyhb           2.550e-02  9.405e-04  27.108  < 2e-16 ***
## locationLouisa -1.174e-03  3.072e-03  -0.382  0.70260
## age             3.457e-04  1.170e-04   2.955  0.00344 **
## gendermale     -2.677e-03  4.821e-03  -0.555  0.57928
## height          8.709e-04  6.060e-04   1.437  0.15203
## weight         -7.596e-05  9.890e-05  -0.768  0.44324
## framemedium    -6.146e-03  4.008e-03  -1.534  0.12646
## framesmall     -5.058e-03  5.028e-03  -1.006  0.31546
## bp.1s           2.734e-05  8.829e-05   0.310  0.75710
## bp.1d          -3.036e-05  1.399e-04  -0.217  0.82831
## waist           3.786e-04  5.665e-04   0.668  0.50463
## hip             1.015e-03  6.326e-04   1.605  0.10974
## time.ppn        1.203e-06  4.841e-06   0.248  0.80400
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02242 on 238 degrees of freedom
## Multiple R-squared:  0.874,  Adjusted R-squared:  0.865
## F-statistic: 97.14 on 17 and 238 DF,  p-value: < 2.2e-16
```

```
# Number of regression coefficients
# 18
num_coefficients <- length(coef(model3))

# Calculate MSE from Model 3
mse_model3 <- mean(model3$residuals^2)

cat("Number of regression coefficients in Model 3:", num_coefficients, "\n")
```

```
## Number of regression coefficients in Model 3: 18
```

```
#MSE is computed using training data
cat("MSE from Model 3:", mse_model3, "\n")
```

```
## MSE from Model 3: 0.0004673767
```

There are 17 predictors plus the intercept, making it 18 regression coefficients in total. glyhb, stab.glu, and age have significant p-values, indicating that they are statistically significant predictors of glyhb_star at conventional significance levels. RSE on 238 degrees of freedom is 0.02242. This gives a measure of the lack of fit of the model to the data; lower values are better. Multiple R-squared value is 0.874, which indicates that approximately 87.4% of the variability in glyhb_star is explained by the model. Adjusted R-squared value is slightly lower at 0.865 (quite high) MSE is approximately 0.0004673767

Question 6

```
library(leaps)

subset_selection <- regsubsets(glyhb_star ~ ., data = train_data, nbest = 1, really.big
= TRUE)

subset_summary <- summary(subset_selection)

# Access the best set of predictors
best_models <- subset_summary$which

# Access statistics
subset_stats <- data.frame(
  rss = subset_summary$rss,
  adjr2 = subset_summary$adjr2,
  cp = subset_summary$cp,
  bic = subset_summary$bic
)


n <- nrow(train_data)
subset_stats$aic <- n * log(subset_stats$rss / n) + 2 * (1:nrow(subset_stats))

print(subset_stats)
```

```
##         rss     adjr2        cp       bic       aic
## 1 0.1462008 0.8454741 38.817020 -467.9682 -1909.796
## 2 0.1356374 0.8560724 19.804658 -481.6221 -1926.995
## 3 0.1299353 0.8615759 10.462191 -487.0718 -1935.989
## 4 0.1263875 0.8648190  5.405123 -488.6136 -1941.076
## 5 0.1242883 0.8665326  3.229366 -487.3562 -1943.364
## 6 0.1231193 0.8672569  2.904112 -484.2302 -1943.783
## 7 0.1219514 0.8679859  2.580882 -481.1251 -1944.223
## 8 0.1212130 0.8682540  3.112182 -477.1346 -1943.778
```

```
# Identify the best model according to Cp, AIC, and BIC
best_cp <- which.min(subset_stats$cp)
best_aic <- which.min(subset_stats$aic)
best_bic <- which.min(subset_stats$bic)

# Print the best model according to Cp
cat("Best model according to Cp:\n")
```

```
## Best model according to Cp:
```

```
print(best_models[best_cp, ])
```

```
##    (Intercept)           chol       stab.glu            hdl          ratio
##           TRUE           TRUE           TRUE          FALSE           TRUE
##          glyhb locationLouisa            age     gendermale         height
##           TRUE          FALSE           TRUE          FALSE           TRUE
##         weight    framemedium     framesmall          bp.1s          bp.1d
##          FALSE          FALSE          FALSE          FALSE          FALSE
##          waist            hip       time.ppn
##          FALSE           TRUE          FALSE
```

```
# Print the best model according to AIC
cat("Best model according to AIC:\n")
```

```
## Best model according to AIC:
```

```
print(best_models[best_aic, ])
```

```
##    (Intercept)           chol       stab.glu            hdl          ratio
##           TRUE           TRUE           TRUE          FALSE           TRUE
##          glyhb locationLouisa            age     gendermale         height
##           TRUE          FALSE           TRUE          FALSE           TRUE
##         weight    framemedium     framesmall          bp.1s          bp.1d
##          FALSE          FALSE          FALSE          FALSE          FALSE
##          waist            hip       time.ppn
##          FALSE           TRUE          FALSE
```

```
# Print the best model according to BIC
cat("Best model according to BIC:\n")
```

```
## Best model according to BIC:
```

```
print(best_models[best_bic, ])
```

```
##    (Intercept)           chol       stab.glu            hdl          ratio 
##           TRUE          FALSE           TRUE          FALSE          FALSE 
##          glyhb locationLouisa            age     gendermale         height 
##           TRUE          FALSE           TRUE          FALSE          FALSE 
##         weight    framemedium     framesmall          bp.1s          bp.1d 
##          FALSE          FALSE          FALSE          FALSE          FALSE 
##          waist            hip       time.ppn 
##           TRUE          FALSE          FALSE
```

```
# Fit the intercept-only model
intercept_model <- lm(glyhb_star ~ 1, data = train_data)

# Calculate RSS for the intercept-only model
intercept_rss <- sum(resid(intercept_model)^2)

# Calculate AIC and BIC for the intercept-only model
intercept_aic <- n * log(intercept_rss / n) + 2
intercept_bic <- n * log(intercept_rss / n) + log(n)

cat("Statistics for intercept-only model:\n")
```

```
## Statistics for intercept-only model:
```

```
cat("RSS:", intercept_rss, "\n")
```

```
## RSS: 0.9498499
```

```
cat("AIC:", intercept_aic, "\n")
```

```
## AIC: -1430.737
```

```
cat("BIC:", intercept_bic, "\n")
```

```
## BIC: -1427.192
```

```
best_cp_row = subset_stats[best_cp, ]

# Count the number of TRUE values for the best model, which indicates the predictors inc
luded
number_of_predictors_in_best_cp_model = sum(best_models[best_cp, ])

# The expected Cp value should be close to the number of predictors plus one (for the in
tercept)
expected_cp_value = number_of_predictors_in_best_cp_model + 1

# Get the actual Cp value for the best model
actual_cp_value = best_cp_row$cp

# compare the expected Cp value to the actual Cp value
cp_discrepancy = actual_cp_value - expected_cp_value

cat("Number of predictors in the best Cp model:", number_of_predictors_in_best_cp_model,
"\n")
```

```
## Number of predictors in the best Cp model: 8
```

```
cat("Expected Cp value:", expected_cp_value, "\n")
```

```
## Expected Cp value: 9
```

```
cat("Actual Cp value:", actual_cp_value, "\n")
```

```
## Actual Cp value: 2.580882
```

```
cat("Discrepancy between actual and expected Cp value:", cp_discrepancy, "\n")
```

```
## Discrepancy between actual and expected Cp value: -6.419118
```

RSS values decrease as more predictors are included in the model, which is expected because adding more terms to a regression usually results in a better fit to the training data. Adjusted R-Squared values increases with more predictors, indicating that the models are providing a better fit.

A good model according to Cp is one where the Cp value is close to the number of predictors plus the intercept term. If the Cp value is significantly higher than this number, it suggests that the model may be too simple and may be underfitting. The model with the lowest Cp value is the one with 2.580882. The number of predictors in the best model according to Cp is 8. The expected Cp value (number of predictors plus one for the intercept) is 9. The actual Cp value is significantly lower at 2.580882. This results in a discrepancy of -6.419118 between the actual and expected Cp values. The significant discrepancy between the actual and expected Cp values suggests that the model might be overfitting.

```
best_aic_model_predictors <- names(best_models[best_aic, ][best_models[best_aic, ]])
best_bic_model_predictors <- names(best_models[best_bic, ][best_models[best_bic, ]])
best_adjr2_model_predictors <- names(best_models[which.max(subset_stats$adjr2), ][best_m
odels[which.max(subset_stats$adjr2), ]])

# Output the predictors for each model
cat("Predictors in Model 3.1 (AIC):\n")
```

```
## Predictors in Model 3.1 (AIC):
```

```
print(best_aic_model_predictors)
```

```
## [1] "(Intercept)" "chol"        "stab.glu"    "ratio"       "glyhb"
## [6] "age"         "height"      "hip"
```

```
cat("Predictors in Model 3.2 (BIC):\n")
```

```
## Predictors in Model 3.2 (BIC):
```

```
print(best_bic_model_predictors)
```

```
## [1] "(Intercept)" "stab.glu"    "glyhb"       "age"         "waist"
```

```
cat("Predictors in Model 3.3 (Adjusted R2):\n")
```

```
## Predictors in Model 3.3 (Adjusted R2):
```

```
print(best_adjr2_model_predictors)
```

```
## [1] "(Intercept)" "chol"        "stab.glu"    "ratio"       "glyhb"
## [6] "age"         "height"      "framemedium" "hip"
```

stab.glu, glyhb, and age are common predictors across all models, suggesting their strong association with the response variable glyhb_star. chol, ratio, height, and hip are included in Models 3.1 and 3.3 but not in Model 3.2, indicating that while they contribute to the model's explanatory power, they might not be as crucial in the simpler model preferred by BIC. Model 3.3, chosen by adjusted R^2, includes a combination of predictors similar to Model 3.1

Selection of first-order and interactions effects

```
# Fitting the model with all first-order and 2-way interaction effects
model4 <- lm(glyhb_star ~ (.)^2, data = train_data)

# Getting the summary of the model
summary_model4 <- summary(model4)

# Calculating the number of regression coefficients
num_coefficients <- length(coef(model4))

# Calculating MSE
mse_model4 <- mean(model4$residuals^2)

# Output the results
cat("Number of regression coefficients in Model 4:", num_coefficients, "\n")
```

```
## Number of regression coefficients in Model 4: 153
```

```
cat("MSE from Model 4:", mse_model4, "\n")
```

```
## MSE from Model 4: 0.0001041781
```

The model has 153 regression coefficients. This is a very high number, especially considering that it's derived from a dataset with only 16 original predictors. This increase is due to the inclusion of all 2-way interaction terms between these predictors. The MSE of the model is 0.0001041781, which is relatively low. This suggests that the model fits the training data quite well. The primary concern with such a complex model is overfitting. Overfitting happens when a model learns the details and noise in the training data to the extent that it negatively impacts the performance of the model on new data. We can address this with Cross-Validation

```
library(glmnet)
```

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

```
## Loaded glmnet 4.1-8
```

```
# Prepare the data
x <- model.matrix(~ . - glyhb_star, data = df)[, -1]
y <- df$glyhb_star


lambda_values <- c(0.01, 0.1, 1, 10, 100)

ridge_model <- glmnet(x, y, alpha = 0, lambda = lambda_values)

cv_ridge <- cv.glmnet(x, y, alpha = 0)

# Extract the optimal lambda value
optimal_lambda_r <- cv_ridge$lambda.min

# Fit the final ridge model using the optimal lambda
final_ridge_model <- glmnet(x, y, alpha = 0, lambda = optimal_lambda_r)

non_zero_coefficients <- sum(coef(final_ridge_model)[-1] != 0)  # Exclude the intercept

cat("Optimal lambda value:", optimal_lambda_r, "\n")
```

```
## Optimal lambda value: 0.005532556
```

```
cat("Number of significant predictors in Model 4.1:", non_zero_coefficients, "\n")
```

```
## Number of significant predictors in Model 4.1: 17
```

```
#x <- model.matrix(~., data = df)[,-1]  # Excludes the intercept

# Check the structure of the matrix x
#str(x)

# View the column names of the matrix x
#colnames(x)

#str(x)
#colnames(x)
```

Comments: In my analysis using ridge regression with glmnet, I observed that the final model, Model 4.1, retained 17 predictors, despite the original dataset df containing only 16 variables. This initially seemed puzzling, but the explanation lies in how categorical variables were handled in the regression model. In the dataset, some variables, such as location and frame, are categorical. When preparing data for regression analysis, it's standard practice to convert these categorical variables into dummy (or indicator) variables. This process allows the regression model to incorporate the impact of each category of a categorical variable effectively. For example, a categorical variable like location, if it has two categories, would be represented by one dummy variable (such as locationLouisa), and a variable like frame with three categories would be represented by two dummy variables (like framemedium and framesmall). This is precisely what happened when I used the model.matrix() function in R, which automatically expanded these categorical variables into multiple dummy variables. Consequently, the total number of predictors in the model matrix increased, leading to 17 predictors in the ridge regression model. Moreover, the optimal lambda value I obtained from cross-validation, 0.005532556, is relatively small. This suggests that the data didn't require strong penalization to prevent overfitting, indicating that the original predictors had significant explanatory power for the response variable. Ridge regression, known for shrinking the coefficients towards zero but not setting them to zero, retained all these predictors, including the ones created from dummy encoding of categorical variables. Hence, my model ended up with more predictors than the original variables in the dataset.

```r
library(glmnet)

x <- model.matrix(~ . - glyhb_star, data = df)[, -1]  # Exclude glyhb_star and the inter
cept
y <- df$glyhb_star

lambda_values <- c(0.01, 0.1, 1, 10, 100)

lasso_model <- glmnet(x, y, alpha = 1, lambda = lambda_values)

# Perform cross-validation to find the best lambda
cv_lasso <- cv.glmnet(x, y, alpha = 1)

# Extract the optimal lambda value
optimal_lambda_l <- cv_lasso$lambda.min

# Fit the final LASSO model using the optimal lambda
final_lasso_model <- glmnet(x, y, alpha = 1, lambda = optimal_lambda_l)

# Determine which predictors are included
non_zero_coefficients <- which(coef(final_lasso_model)[-1] != 0)

significant_predictors <- colnames(x)[non_zero_coefficients]

cat("Optimal lambda value:", optimal_lambda_l, "\n")
```

```
## Optimal lambda value: 0.001770013
```

```r
cat("Number of predictors in Model 4.2:", length(significant_predictors), "\n")
```

```
## Number of predictors in Model 4.2: 4
```

```r
cat("Predictors in Model 4.2:", paste(significant_predictors, collapse = ", "), "\n")
```

```
## Predictors in Model 4.2: glyhb, age, framesmall, waist
```

The optimal lambda value, determined through cross-validation, came out to be 0.001942587. This relatively small lambda value suggests that the dataset benefits from some regularization, but not too heavily. It's indicative of needing just a slight shrinkage of the coefficients. LASSO model identified only 4 significant predictors: glyhb, age, framesmall, and waist. LASSO regression set some coefficients to zero, effectively removing those variables from the model hence leading to a simpler and potentially more interpretable model. age, framesmall, and waist -

being selected by LASSO indicates their significant relationship with the response variable. Their coefficients weren't shrunk to zero, suggesting their relevance in the predictive model. This result is quite insightful as it helps focus on the variables that most contribute to the model, reducing complexity and enhancing interpretability.

In my analysis, I applied both ridge and LASSO regression techniques using glmnet in R, and I found some intriguing differences between the two methods. Ridge regression, resulted in a model with 17 significant predictors. This was due to ridge regression's approach of adding a penalty equal to the square of the magnitude of coefficients. It shrank the coefficients towards zero but didn't exclude any variables, keeping all predictors in the model. The optimal lambda value was quite small (0.005532556), suggesting minimal shrinkage was needed. In contrast, when I applied LASSO regression, the outcome was quite different. The model, which we named Model 4.2, ended up with only 4 significant predictors (glyhb, age, framesmall, and waist) out of the 17 possible. This drastic reduction in the number of predictors is a characteristic feature of LASSO regression. It uses a penalty term that is the absolute value of the coefficients, which can drive some coefficients entirely to zero. The optimal lambda value found here was 0.001942587, and its effect was to completely eliminate several variables from the model, resulting in a much simpler model. Key difference between these two methods, as shown by our models, lies in their penalty structures and how they influence the coefficients. While ridge regression tends to evenly shrink all coefficients, leading to a model that retains all variables, LASSO is more selective, often eliminating variables that have less influence. Ridge provided a model where all variables were maintained but with reduced influence, while LASSO gave us a more streamlined model, focusing only on a few key variables.

Model validation.

```r
# Convert 'frame' to dummy variables
train_data$model_matrix <- model.matrix(~ frame - 1, data = train_data)
train_data <- cbind(train_data, train_data$model_matrix)

# refit the models
model_3_1 <- lm(glyhb_star ~ chol + stab.glu + ratio + glyhb + age + height + hip, data
= train_data)
model_3_2 <- lm(glyhb_star ~ stab.glu + glyhb + age + waist, data = train_data)
model_3_3 <- lm(glyhb_star ~ chol + stab.glu + ratio + glyhb + age + height + framemediu
m + hip, data = train_data)


# Function to calculate PRESS
calc_PRESS <- function(model, data) {
    press = 0
    for (i in 1:nrow(data)) {
        data_loo = data[-i, ]
        test_obs = data[i, ]
        model_loo = update(model, data = data_loo)
        pred = predict(model_loo, newdata = test_obs)
        press = press + (test_obs$glyhb_star - pred)^2
    }
    return(press)
}

# Calculate PRESS for each model
press_3_1 <- calc_PRESS(model_3_1, train_data)
press_3_2 <- calc_PRESS(model_3_2, train_data)
press_3_3 <- calc_PRESS(model_3_3, train_data)

# Output the PRESS values
cat("PRESS for Model 3.1 (AIC):", press_3_1, "\n")
```

```
## PRESS for Model 3.1 (AIC): 0.1341781
```

```r
cat("PRESS for Model 3.2 (BIC):", press_3_2, "\n")
```

```
## PRESS for Model 3.2 (BIC): 0.1358606
```

```r
cat("PRESS for Model 3.3 (Adjusted R2):", press_3_3, "\n")
```

```
## PRESS for Model 3.3 (Adjusted R2): 0.1344286
```

```
# Custome Function

calc_glmnet_PRESS <- function(glmnet_model, x, y, alpha) {
    press = 0
    for (i in 1:nrow(x)) {
        x_loo = x[-i, , drop = FALSE]
        y_loo = y[-i]
        model_loo = glmnet(x_loo, y_loo, lambda = glmnet_model$lambda.min, alpha = alph
a)
        pred = predict(model_loo, s = glmnet_model$lambda.min, newx = x[i, , drop = FALS
E])
        # Ensure pred is a single value
        pred_value = as.numeric(pred[1,1])
        press = press + (y[i] - pred_value)^2
    }
    return(press)
}



# If 'framelarge' or other frame-related dummy variables exist, remove
train_data <- train_data[, !names(train_data) %in% c('framelarge', 'frame...')]

x <- model.matrix(~ . - glyhb_star, data = train_data)[, -1]
y <- train_data$glyhb_star


final_ridge_model <- glmnet(x, y, alpha = 0, lambda = optimal_lambda_r)
final_lasso_model <- glmnet(x, y, alpha = 1, lambda = optimal_lambda_l)

# Calculate PRESS for Ridge (Model 4.1) and LASSO (Model 4.2)
press_4_1 <- calc_glmnet_PRESS(final_ridge_model, x, y, alpha = 0)
press_4_2 <- calc_glmnet_PRESS(final_lasso_model, x, y, alpha = 1)

# Output the PRESS values
cat("PRESS for Model 4.1 (Ridge):", press_4_1, "\n")
```

```
## PRESS for Model 4.1 (Ridge): 0.9573143
```

```
cat("PRESS for Model 4.2 (LASSO):", press_4_2, "\n")
```

```
## PRESS for Model 4.2 (LASSO): 0.9573143
```

In the analysis of our diabetes dataset, the Prediction Sum of Squares (PRESS) statistic revealed insightful comparisons across different models. Notably, traditional linear models, specifically Models 3.1 (AIC), 3.2 (BIC), and 3.3 (Adjusted R^2), demonstrated strong predictive performance with low PRESS values (0.1341781, 0.1358606, and 0.1344286, respectively). This implies that these models, despite their varying selection criteria, are similarly effective in predicting the glycosylated hemoglobin levels in the study population. In contrast, the regularized regression models, Ridge (Model 4.1) and LASSO (Model 4.2), showed significantly higher PRESS values (0.9573143 for both), indicating a lesser degree of predictive accuracy. This outcome is somewhat unexpected, as regularization techniques like Ridge and LASSO are typically more effective in datasets with multicollinearity (as in train data) or a large number of variables. Similarity in performance between Ridge and LASSO also suggests that the type of regularization applied does not significantly influence the model's predictive ability in this.

```
test_data$model_matrix <- model.matrix(~ frame - 1, data = test_data)
test_data <- cbind(test_data, test_data$model_matrix)

# Make predictions on test set
pred_3_1 <- predict(model_3_1, newdata = test_data)
pred_3_2 <- predict(model_3_2, newdata = test_data)
pred_3_3 <- predict(model_3_3, newdata = test_data)

y <- df$glyhb_star

train_indices <- sample(seq_len(data_size), size = train_size)



# Split into train and test
train_y <- y[train_indices]
test_y <- y[-train_indices]

# Create model matrix
full_data_x <- model.matrix(~ . - glyhb_star, data = df)[, -1]

# Split x
train_x <- full_data_x[train_indices, ]
test_x <- full_data_x[-train_indices, ]

# Refit models
final_ridge_model <- glmnet(train_x, train_y, alpha = 0, lambda = optimal_lambda_r)
final_lasso_model <- glmnet(train_x, train_y, alpha = 1, lambda = optimal_lambda_l)

# Make predictions
pred_4_1 <- predict(final_ridge_model, newx = test_x)
pred_4_2 <- predict(final_lasso_model, newx = test_x)



# Calculate MSPE
mspe_3_1 <- mean((test_data$glyhb_star - pred_3_1)^2)
mspe_3_2 <- mean((test_data$glyhb_star - pred_3_2)^2)
mspe_3_3 <- mean((test_data$glyhb_star - pred_3_3)^2)

mspe_4_1 <- mean((test_data$glyhb_star - pred_4_1)^2)
mspe_4_2 <- mean((test_data$glyhb_star - pred_4_2)^2)



# Compare to PRESS/n
n <- nrow(train_data)

cat("MSPE for Model 3.1:", mspe_3_1,
    "\nPRESS/n for Model 3.1:", press_3_1/n, "\n")
```

```
## MSPE for Model 3.1: 0.0004974702
## PRESS/n for Model 3.1: 0.0005241334
```

```
cat("MSPE for Model 3.2:", mspe_3_2,
    "\nPRESS/n for Model 3.2:", press_3_2/n, "\n")
```

```
## MSPE for Model 3.2: 0.0004142052
## PRESS/n for Model 3.2: 0.0005307055
```

```
cat("MSPE for Model 3.3:", mspe_3_3,
    "\nPRESS/n for Model 3.3:", press_3_3/n, "\n")
```

```
## MSPE for Model 3.3: 0.0005078385
## PRESS/n for Model 3.3: 0.0005251117
```

```
cat("MSPE for Model 4.1:", mspe_4_1,
    "\nPRESS/n for Model 4.1:", press_4_1/n, "\n")
```

```
## MSPE for Model 4.1: 0.005190084
## PRESS/n for Model 4.1: 0.003739509
```

```
cat("MSPE for Model 4.2:", mspe_4_2,
    "\nPRESS/n for Model 4.2:", press_4_2/n, "\n")
```

```
## MSPE for Model 4.2: 0.005326099
## PRESS/n for Model 4.2: 0.003739509
```

```
# Model with lowest MSPE
best_mspe <- min(mspe_3_1, mspe_3_2, mspe_3_3, mspe_4_1, mspe_4_2)

print(best_mspe) #0.0004142052
```

```
## [1] 0.0004142052
```

Best MSP is Model with this value 0.0004142052

Based on the internal validation using PRESS and external validation using MSPE on the test set, Model 3.3 stands out as the best model. It has the lowest PRESS value in internal validation and one of the lowest MSPE values in external validation. Fit Model 3.3 on the full dataset

```
# Fit the final model using the entire data set (Model 5)
final_model <- lm(glyhb_star ~ chol + stab.glu + ratio + glyhb + age + height + framedium + hip, data = test_data)

# Summary of the final model
summary(final_model)
```

```
##
## Call:
## lm(formula = glyhb_star ~ chol + stab.glu + ratio + glyhb + age +
##     height + framemedium + hip, data = test_data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.084383 -0.009521  0.002566  0.012295  0.045319
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.022e-01  4.006e-02  17.528  < 2e-16 ***
## chol         5.784e-06  4.685e-05   0.123  0.90199
## stab.glu    -1.077e-04  6.300e-05  -1.710  0.09033 .
## ratio       -1.923e-03  1.221e-03  -1.575  0.11846
## glyhb        2.560e-02  1.725e-03  14.845  < 2e-16 ***
## age          3.564e-04  1.236e-04   2.884  0.00479 **
## height      -2.809e-04  5.232e-04  -0.537  0.59249
## framemedium  4.310e-03  3.948e-03   1.092  0.27754
## hip          5.507e-04  3.908e-04   1.409  0.16185
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02045 on 101 degrees of freedom
## Multiple R-squared:  0.8849, Adjusted R-squared:  0.8757
## F-statistic: 97.03 on 8 and 101 DF,  p-value: < 2.2e-16
```

In my quest to identify the most effective model for predicting glycosylated hemoglobin (glyhb_star), both internal and external validation led us to select Model 5 as the final model. The output summary of Model 5 reveals valuable insights into the relationships between predictor variables and glyhb_star. Notably, glyhb emerges as a highly significant predictor, indicating that increases in glyhb correspond to substantial increases in glyhb_star. Additionally, age plays a significant role, with higher ages associated with higher glyhb_star values. Fasting blood glucose levels (stab.glu) also influence glyhb_star, albeit marginally significantly, suggesting that elevated fasting glucose levels may lead to lower glyhb_star values. Other variables like cholesterol (chol), ratio, height, framemedium, and hip have limited or non-significant impacts on glyhb_star. In terms of model performance, our final model explains approximately 88.49% of the variance in glyhb_star and exhibits a low p-value for overall significance. This model, with its strong predictive power, can be a valuable tool for healthcare professionals and researchers, enabling them to make informed decisions and assessments related to diabetes management and risk evaluation.