Stat131A_FinalProject

Introduction

We load the data set into R, load all the required libraries, and take a quick glance at the values in the data set.

```
cholangitis <- read.csv("/Users/setushah/OneDrive/Documents/Classes/Fall 2021/Stat C131A/Final Project/
suppressMessages(library(dplyr))
suppressMessages(library(ggplot2))
suppressMessages(library(tidyverse))
suppressMessages(library(pheatmap))
suppressMessages(library(vioplot))
suppressMessages(library(leaps))
glimpse(cholangitis)
```

```
## Rows: 418
## Columns: 20
## $ id            <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1~
## $ n_days        <int> 400, 4500, 1012, 1925, 1504, 2503, 1832, 2466, 2400, 51,~
## $ status        <chr> "D", "C", "D", "D", "CL", "D", "C", "D", "D", "D", "D", ~
## $ drug          <chr> "D-penicillamine", "D-penicillamine", "D-penicillamine",~
## $ age           <int> 21464, 20617, 25594, 19994, 13918, 24201, 20284, 19379, ~
## $ sex           <chr> "F", "F", "M", "F", "F", "F", "F", "F", "F", "F", "F", "~
## $ ascites       <chr> "Y", "N", "N", "N", "N", "N", "N", "N", "N", "Y", "N", "~
## $ hepatomegaly  <chr> "Y", "Y", "N", "Y", "Y", "Y", "Y", "N", "N", "N", "Y", "~
## $ spiders       <chr> "Y", "Y", "N", "Y", "Y", "N", "N", "N", "Y", "Y", "Y", "~
## $ edema         <chr> "Y", "N", "S", "S", "N", "N", "N", "N", "N", "Y", "N", "~
## $ bilirubin     <dbl> 14.5, 1.1, 1.4, 1.8, 3.4, 0.8, 1.0, 0.3, 3.2, 12.6, 1.4,~
## $ cholesterol   <int> 261, 302, 176, 244, 279, 248, 322, 280, 562, 200, 259, 2~
## $ albumin       <dbl> 2.60, 4.14, 3.48, 2.54, 3.53, 3.98, 4.09, 4.00, 3.08, 2.~
## $ copper        <int> 156, 54, 210, 64, 143, 50, 52, 52, 79, 140, 46, 94, 40, ~
## $ alk_phos      <dbl> 1718.0, 7394.8, 516.0, 6121.8, 671.0, 944.0, 824.0, 4651~
## $ sgot          <dbl> 137.95, 113.52, 96.10, 60.63, 113.15, 93.00, 60.45, 28.3~
## $ tryglicerides <int> 172, 88, 55, 92, 72, 63, 213, 189, 88, 143, 79, 95, 130,~
## $ platelets     <int> 190, 221, 151, 183, 136, 361, 204, 373, 251, 302, 258, 7~
## $ prothrombin   <dbl> 12.2, 10.6, 12.0, 10.3, 10.9, 11.0, 9.7, 11.0, 11.0, 11.~
## $ stage         <int> 4, 3, 4, 4, 3, 3, 3, 3, 2, 4, 4, 4, 3, 4, 3, 3, 4, 4, 3,~
```

Cleaning Data

Now we will change character variables to factor variables to help make the data points make logical sense. We change age to number of years (instead of number of days).

```r
cholangitis <- mutate_if(cholangitis, is.character, as.factor)
cholangitis$stage <- factor(cholangitis$stage, levels = c(1,2,3,4))
cholangitis$age <- round(cholangitis$age / 365)

#removing the patients who got a liver transplant
cholangitis <- subset(cholangitis, status != "CL")
glimpse(cholangitis)
```

```
## Rows: 393
## Columns: 20
## $ id            <int> 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ~
## $ n_days        <int> 400, 4500, 1012, 1925, 2503, 1832, 2466, 2400, 51, 3762,~
## $ status        <fct> D, C, D, D, D, C, D, D, D, D, D, C, D, D, C, D, D, C, D,~
## $ drug          <fct> D-penicillamine, D-penicillamine, D-penicillamine, D-pen~
## $ age           <dbl> 59, 56, 70, 55, 66, 56, 53, 43, 71, 54, 59, 46, 56, 65, ~
## $ sex           <fct> F, F, M, F, F, F, F, F, F, F, F, F, M, F, F, F, F, F, F,~
## $ ascites       <fct> Y, N, N, N, N, N, N, N, Y, N, N, N, Y, N, N, N, N, N, N,~
## $ hepatomegaly  <fct> Y, Y, N, Y, Y, Y, N, N, N, Y, N, N, Y, N, N, Y, Y, Y, Y,~
## $ spiders       <fct> Y, Y, N, Y, N, N, N, Y, Y, Y, Y, N, N, N, N, N, Y, N, N,~
## $ edema         <fct> Y, N, S, S, N, N, N, N, Y, N, N, N, Y, N, N, N, Y, S, N,~
## $ bilirubin     <dbl> 14.5, 1.1, 1.4, 1.8, 0.8, 1.0, 0.3, 3.2, 12.6, 1.4, 3.6,~
## $ cholesterol   <int> 261, 302, 176, 244, 248, 322, 280, 562, 200, 259, 236, 2~
## $ albumin       <dbl> 2.60, 4.14, 3.48, 2.54, 3.98, 4.09, 4.00, 3.08, 2.74, 4.~
## $ copper        <int> 156, 54, 210, 64, 50, 52, 52, 79, 140, 46, 94, 40, 43, 1~
## $ alk_phos      <dbl> 1718.0, 7394.8, 516.0, 6121.8, 944.0, 824.0, 4651.2, 227~
## $ sgot          <dbl> 137.95, 113.52, 96.10, 60.63, 93.00, 60.45, 28.38, 144.1~
## $ tryglicerides <int> 172, 88, 55, 92, 63, 213, 189, 88, 143, 79, 95, 130, 151~
## $ platelets     <int> 190, 221, 151, 183, 361, 204, 373, 251, 302, 258, 71, 24~
## $ prothrombin   <dbl> 12.2, 10.6, 12.0, 10.3, 11.0, 9.7, 11.0, 11.0, 11.5, 12.~
## $ stage         <fct> 4, 3, 4, 4, 3, 3, 3, 2, 4, 4, 4, 3, 4, 3, 3, 4, 4, 3, 4,~
```

Exploratory Data Analysis

After dropping the patients who got a liver transplant, we take a look at the missing (i.e. NA) values in the data set.

```r
summary(is.na(cholangitis))
```

```
##      id            n_days          status           drug
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:393       FALSE:393       FALSE:393       FALSE:293
##                                                  TRUE :100
##     age             sex           ascites         hepatomegaly
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:393       FALSE:393       FALSE:393       FALSE:393
##
##   spiders          edema         bilirubin       cholesterol
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:393       FALSE:393       FALSE:393       FALSE:388
##                                                  TRUE :5
##   albumin          copper         alk_phos          sgot
```

```
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:393       FALSE:393       FALSE:393       FALSE:393
##
##  tryglicerides   platelets       prothrombin       stage
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:388       FALSE:393       FALSE:393       FALSE:393
##  TRUE :5
```

```
cholangitis[which(is.na(cholangitis$cholesterol)), ]
```

```
##      id n_days status            drug age sex ascites hepatomegaly spiders
## 41   41   1350      D D-penicillamine  34   F       N            Y       N
## 106 106   3222      D D-penicillamine  69   F       Y            Y       N
## 146 146   2615      C         Placebo  34   F       N            N       N
## 178 178   2580      C D-penicillamine  70   F       N            N       N
## 190 190   2504      C D-penicillamine  55   F       N            N       Y
##     edema bilirubin cholesterol albumin copper alk_phos   sgot tryglicerides
## 41      N       6.8          NA    3.26     96     1215 151.90            NA
## 106     N       2.1          NA    3.90     50     1087 103.85            NA
## 146     S       1.2          NA    3.89     58     1284 173.60            NA
## 178     N       0.6          NA    4.08     51      665  74.40            NA
## 190     N       2.3          NA    3.93     24     1828 133.30            NA
##     platelets prothrombin stage
## 41        226        11.7     4
## 106       137        10.6     2
## 146       239         9.4     3
## 178       325        10.2     4
## 190       327        10.2     2
```

We see that `tryglicerides` and `cholesterol` have 5 missing (NA) values each. It appears that this is a data collection error because the missing values refer to data for 5 female patients of varying ages. We will drop the rows from our data for easier computation and since there are so few of them. The 100 missing values for the `drug` variable refer to the 100 out of the 106 patients that did not consent to randomization so they received neither the drug nor the placebo. Since they makes up a fair chunk of our data set, we will keep them in the data set for now.

Next we, check the summary statistics after cleaning the data.

```
summary(cholangitis)
```

```
##        id           n_days     status                drug          age
##  Min.   :  1   Min.   :  41   C :232   D-penicillamine:148   Min.   :26.00
##  1st Qu.:100   1st Qu.:1119   CL:  0   Placebo        :145   1st Qu.:44.00
##  Median :204   Median :1769   D :161   NA's           :100   Median :52.00
##  Mean   :207   Mean   :1941                                  Mean   :51.35
##  3rd Qu.:314   3rd Qu.:2657                                  3rd Qu.:59.00
##  Max.   :418   Max.   :4795                                  Max.   :78.00
##
##  sex     ascites hepatomegaly spiders edema     bilirubin       cholesterol
##  F:352   N:365   N:193        N:279   N:332   Min.   : 0.300   Min.   : 120.0
##  M: 41   Y: 28   Y:200        Y:114   S: 41   1st Qu.: 0.800   1st Qu.: 247.8
##                                       Y: 20   Median : 1.300   Median : 308.5
##                                               Mean   : 3.199   Mean   : 363.6
```

```
##                                                3rd Qu.: 3.300   3rd Qu.: 401.0
##                                                Max.   :28.000   Max.   :1775.0
##                                                                 NA's   :5
##     albumin          copper          alk_phos          sgot
##  Min.   :1.960   Min.   :  4.00   Min.   :  289   Min.   : 26.35
##  1st Qu.:3.260   1st Qu.: 41.00   1st Qu.:  857   1st Qu.: 80.60
##  Median :3.530   Median : 70.00   Median : 1243   Median :113.15
##  Mean   :3.498   Mean   : 94.44   Mean   : 1961   Mean   :121.44
##  3rd Qu.:3.770   3rd Qu.:123.00   3rd Qu.: 2045   3rd Qu.:151.90
##  Max.   :4.640   Max.   :588.00   Max.   :13862   Max.   :457.25
##
##  tryglicerides     platelets      prothrombin    stage
##  Min.   : 33.0   Min.   : 62.0   Min.   : 9.00   1: 21
##  1st Qu.: 84.0   1st Qu.:188.0   1st Qu.:10.00   2: 89
##  Median :110.5   Median :246.0   Median :10.60   3:146
##  Mean   :122.8   Mean   :254.1   Mean   :10.75   4:137
##  3rd Qu.:151.0   3rd Qu.:316.0   3rd Qu.:11.10
##  Max.   :598.0   Max.   :721.0   Max.   :18.00
##  NA's   :5
```

Here, I'm creating subsets of the original dataset that I will use throughout my analysis. I have presented them at the top for ease of use.

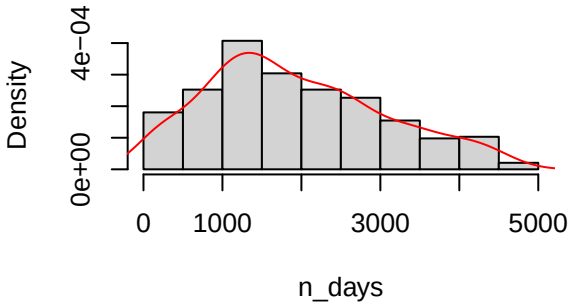
```
#keeping only numeric variables and dropping id column as it provides no new information
chol_sub <- cholangitis[, names(which(sapply(cholangitis, is.numeric)))]
chol_sub <- subset(chol_sub, select = -id)

#creating another data frame for running regression. We remove the id variable here as well
chol_updated <- na.omit(cholangitis)
chol_updated <- subset(chol_updated, select = -id)
invisible(droplevels(chol_updated$status))
chol_updated$status <- as.numeric(chol_updated$status == "C")
chol_updated$status <- as.factor(chol_updated$status)
```

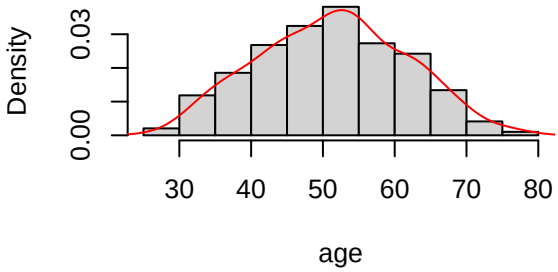
```
#dropping rows with NA values for cholesterol and tryglicerides
chol_sub <- na.omit(chol_sub)
```

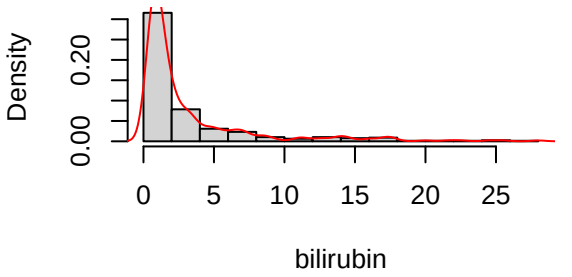
Visualization

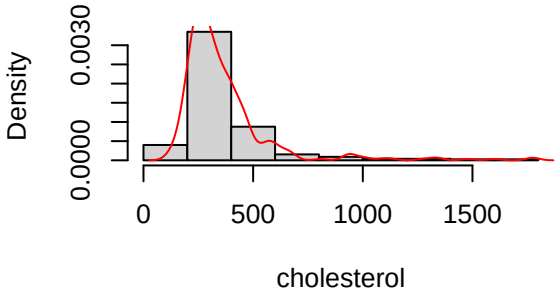
Now, we do some visualizations for exploratory data analysis. We will plot histograms with density curves overlaid to see the distribution of the numeric variables.

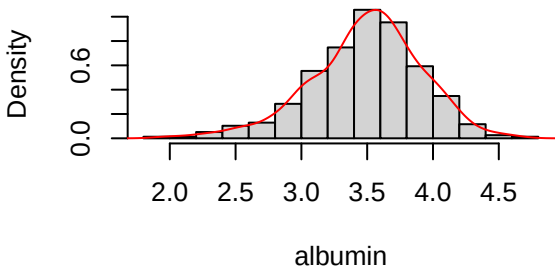
```
#EDA - density histograms, we omit NA values to prevent errors
par(mfrow = c(2,2))
for (i in names(chol_sub)) { hist(chol_sub[, i],  freq=F, xlab = i, main = "")
  lines(density(chol_sub[, i]), col = "red")}
```

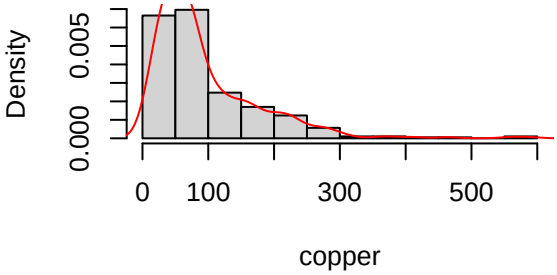

Density
0e+00
4e−04
0
1000
3000
5000
n_days

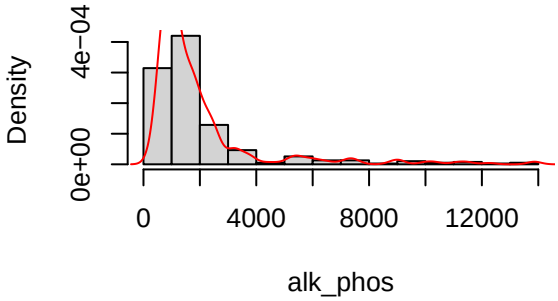

Density
0.00
0.03
30
40
50
60
70
80
age

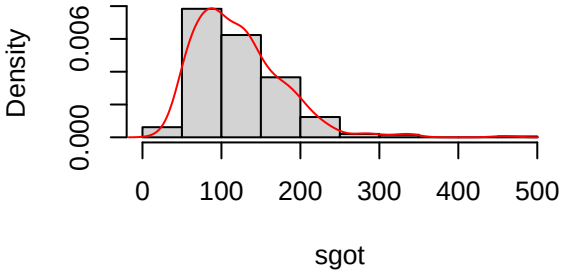

Density
0.00
0.20
0
5
10
15
20
25
bilirubin

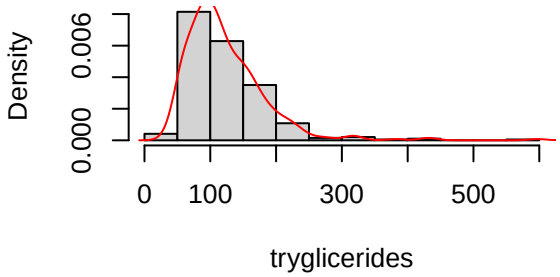

Density
0.0000
0.0030
0
500
1000
1500
cholesterol

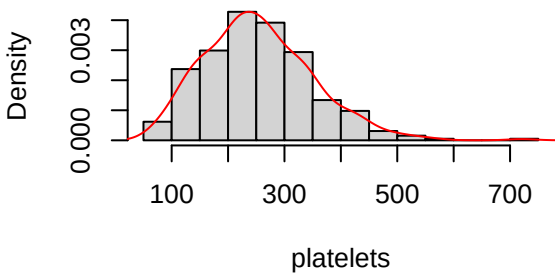

Density
0.0
0.6
2.0
2.5
3.0
3.5
4.0
4.5
albumin

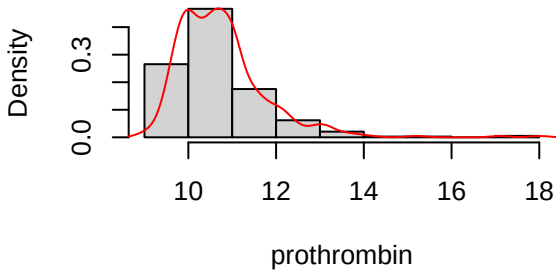

Density
0.000
0.005
0
100
300
500
copper


Density
0e+00
4e−04
0
4000
8000
12000
alk_phos


Density
0.000
0.006
0
100
200
300
400
500
sgot

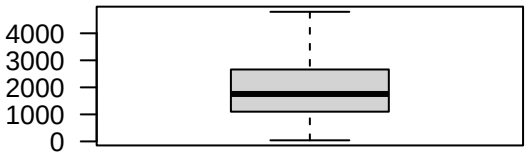




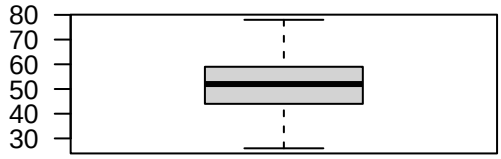


Next, we look at boxplots. They are useful as they show us the whiskers (1.5 x IQR) of the data, and can help us identify outliers in the data.

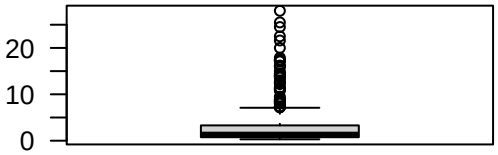
```
par(mfrow = c(2,2))
for (i in names(chol_sub)) { boxplot(chol_sub[, i], xlab = i, las = 2) }
```



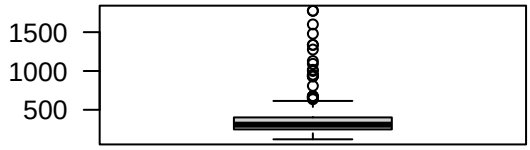
n_days



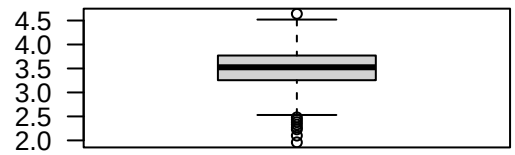
age



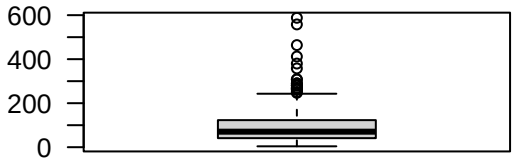
bilirubin



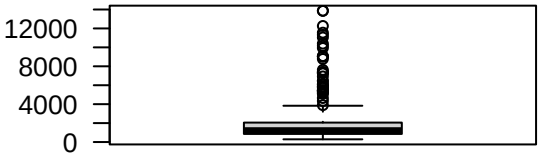
cholesterol



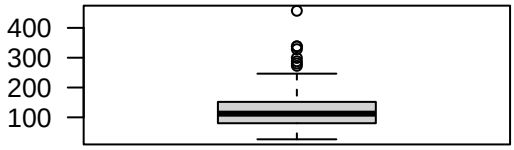
albumin



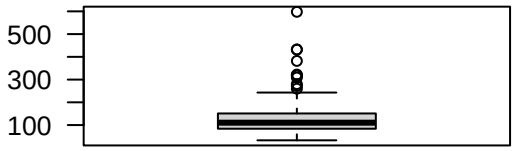
copper



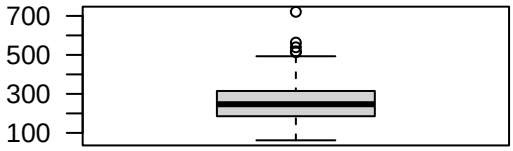
alk_phos



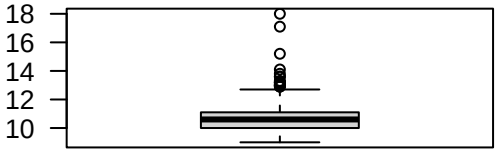
sgot



trygliceride



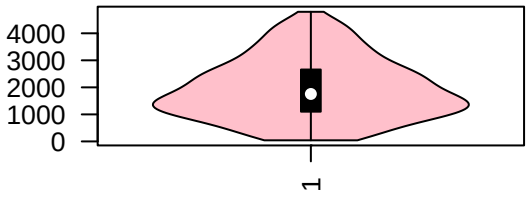
platelets



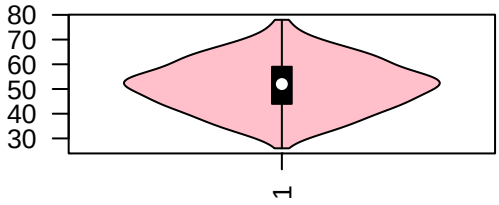
prothrombin

We also look at violin plots of the numeric variables to check for variability in these explanatory variables. Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator.

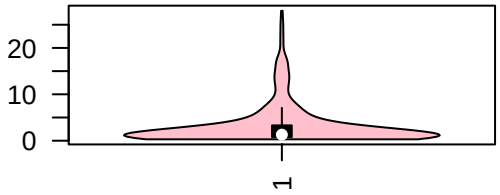
```
par(mfrow=c(2,2))
for (i in names(chol_sub)) { vioplot(chol_sub[, i], xlab = i, col = "pink", las = 2) }
```



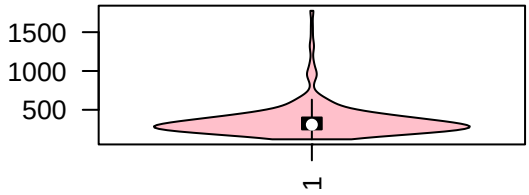
n_days



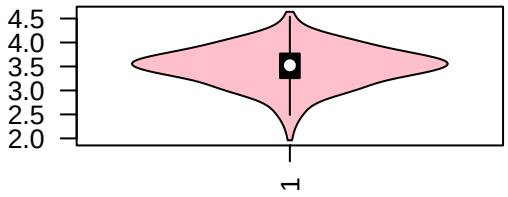
age



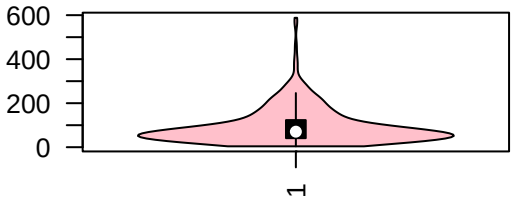
bilirubin



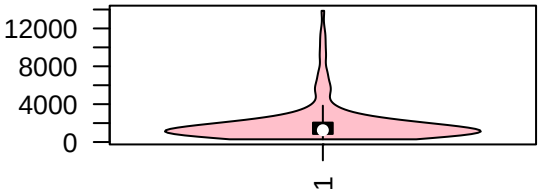
cholesterol



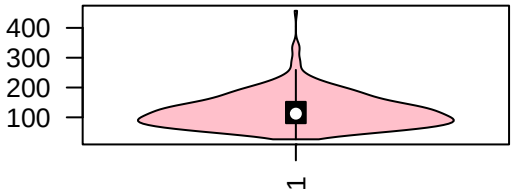
albumin



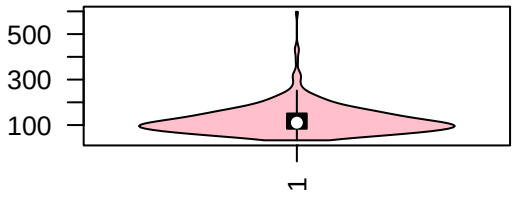
copper



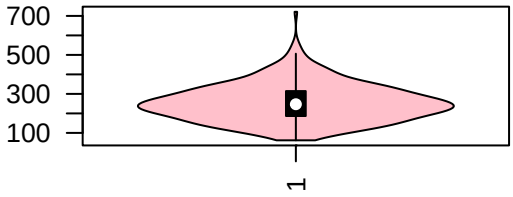
alk_phos



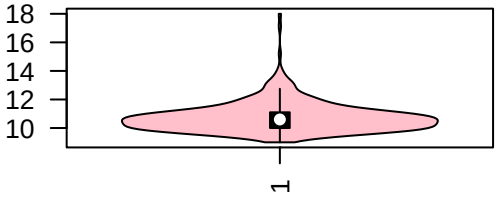
sgot



trygliceride



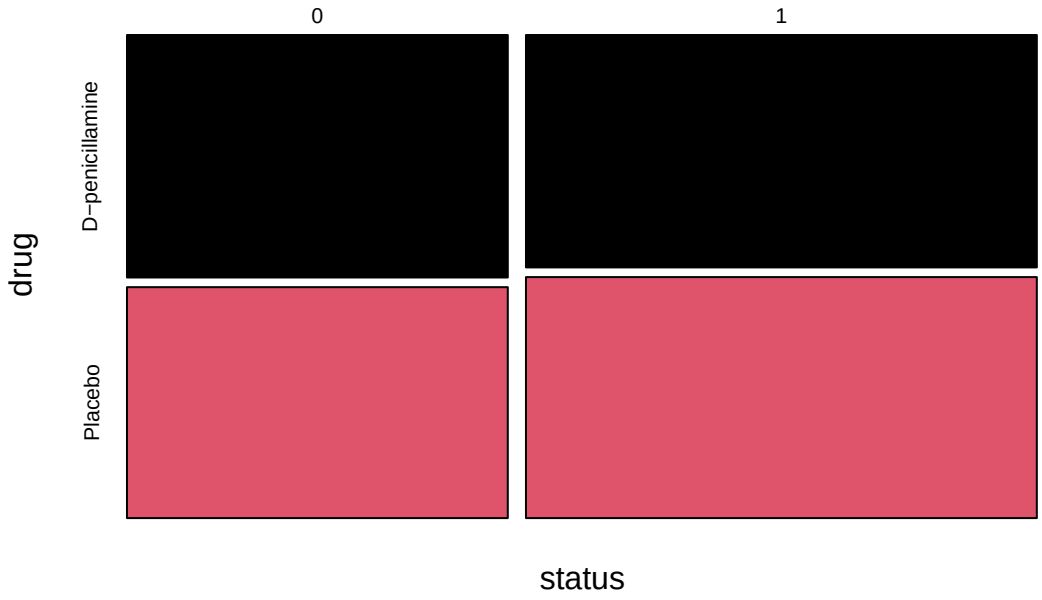
platelets

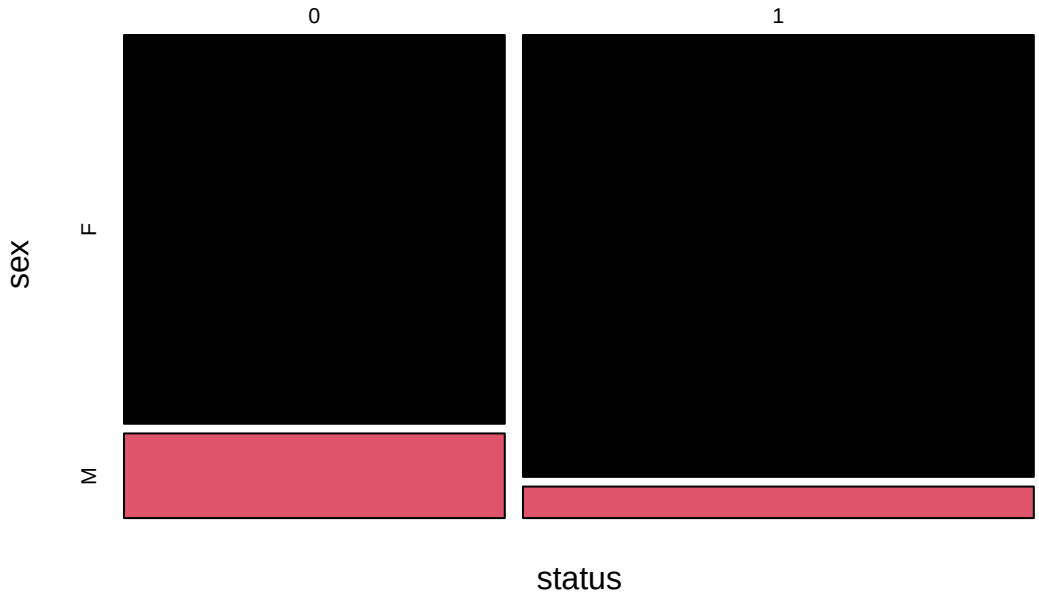


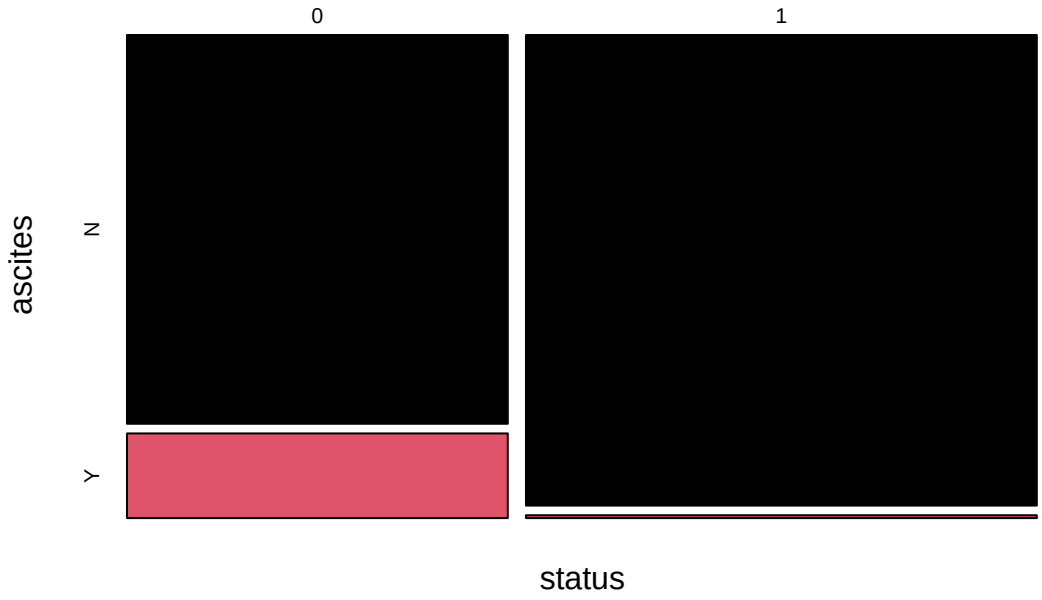
prothrombin

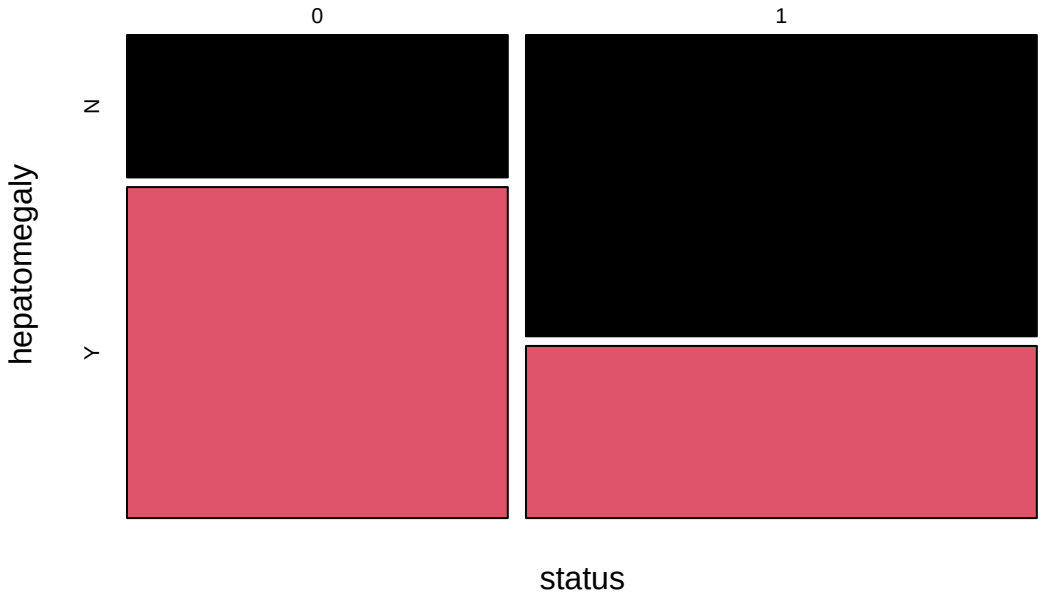
From the previous plots we see that many of the variables are right skewed, i.e. they have some large values that might be outliers.We can try log transforming the variables as it will reduce some of the skewness and might improve our analysis.

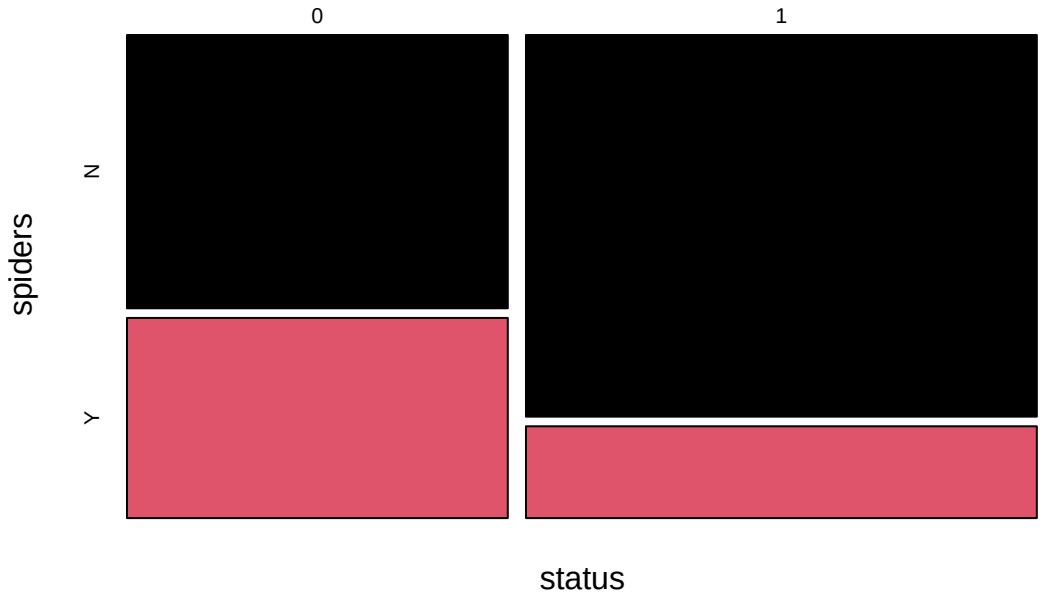
```
for (i in names(chol_updated[,c(3, 5, 6:9, 19)])) {
  mosaicplot(status ~ chol_updated[, i], data=na.omit(chol_updated),col = palette(),
             ylab = i, main = str_c("Mosaic Plot of status (0=dead, 1=alive) and ", i))}
```

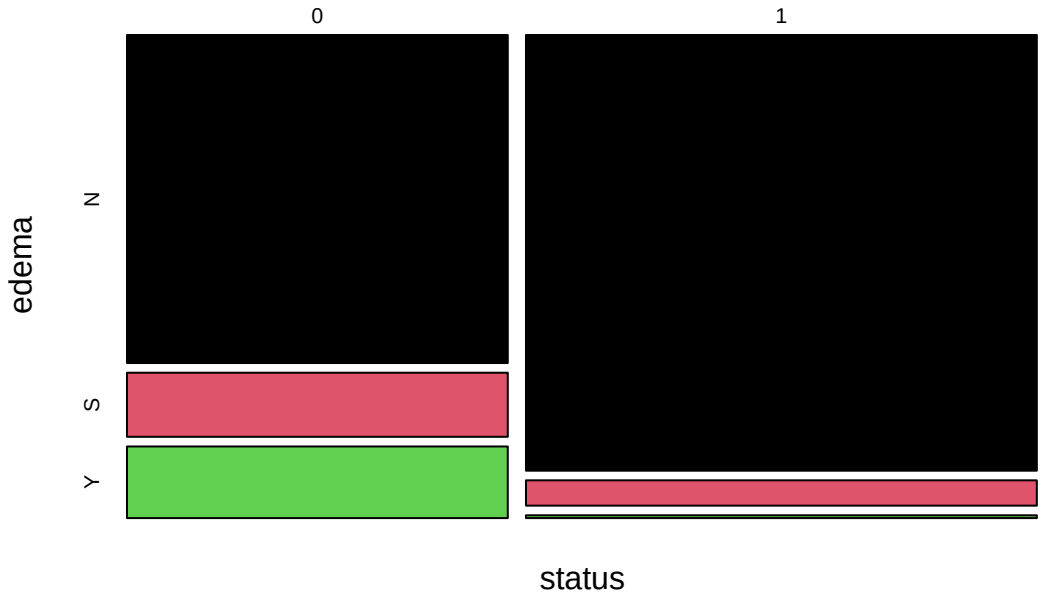

Mosaic Plot of status (0=dead, 1=alive) and drug
0
1
D−penicillamine
Placebo
drug
status

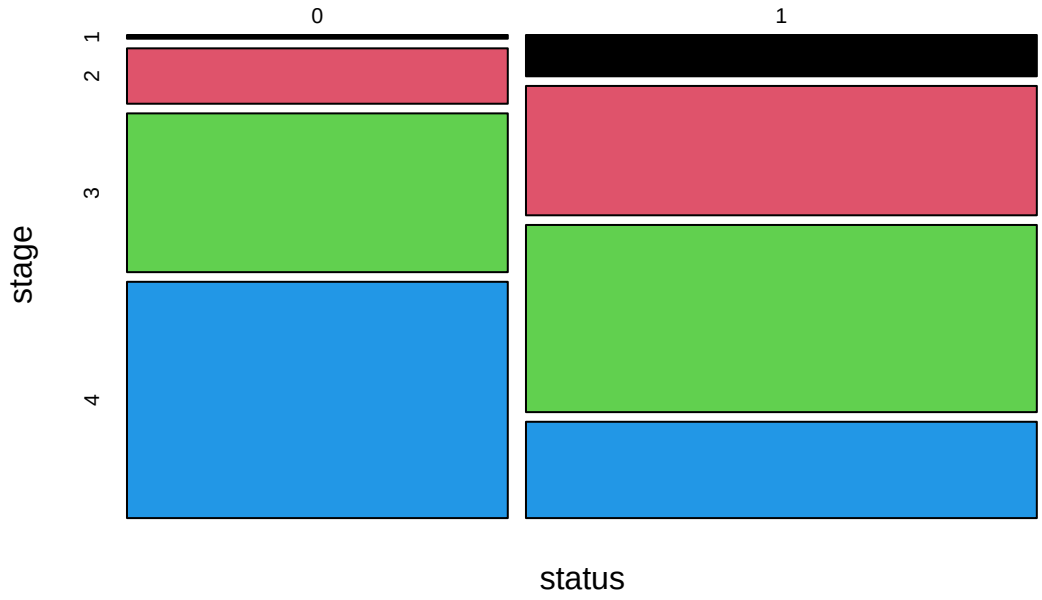

Mosaic Plot of status (0=dead, 1=alive) and sex
0
1
sex
F
M
status


Mosaic Plot of status (0=dead, 1=alive) and ascites
0
1
ascites
N
Y
status


Mosaic Plot of status (0=dead, 1=alive) and hepatomegaly
0
1
N
Y
hepatomegaly
status


Mosaic Plot of status (0=dead, 1=alive) and spiders
0
1
N
Y
spiders
status

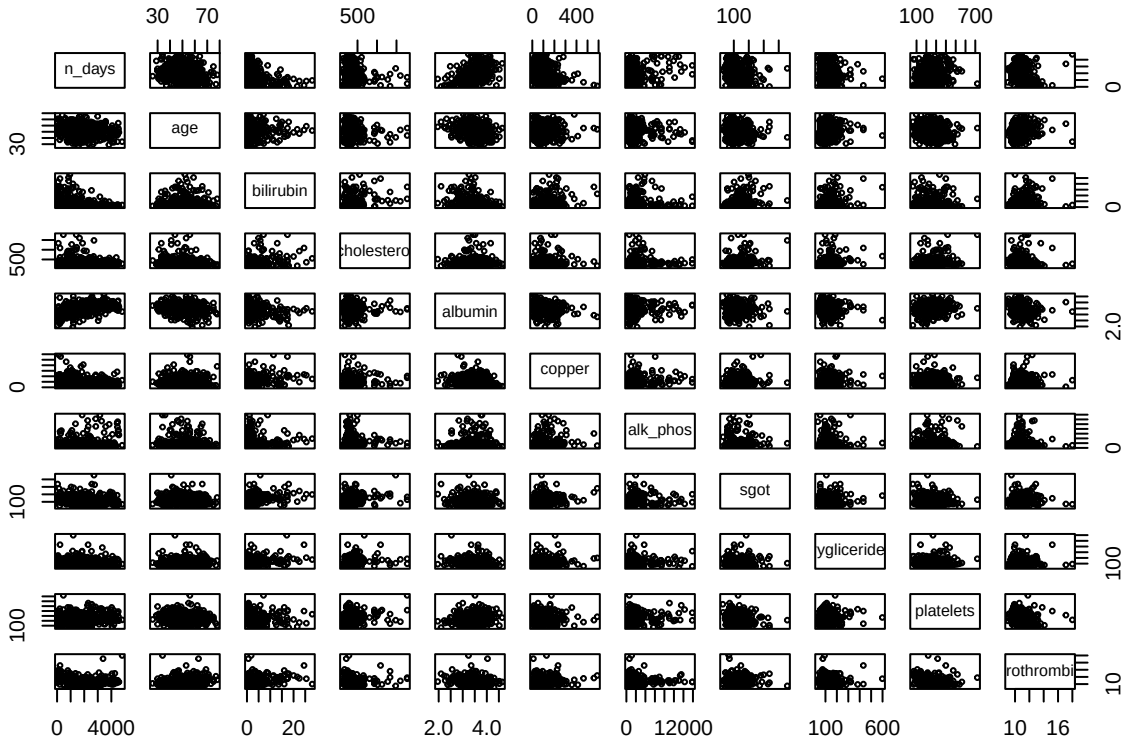

Mosaic Plot of status (0=dead, 1=alive) and edema
0
1
edema
N
S
Y
status



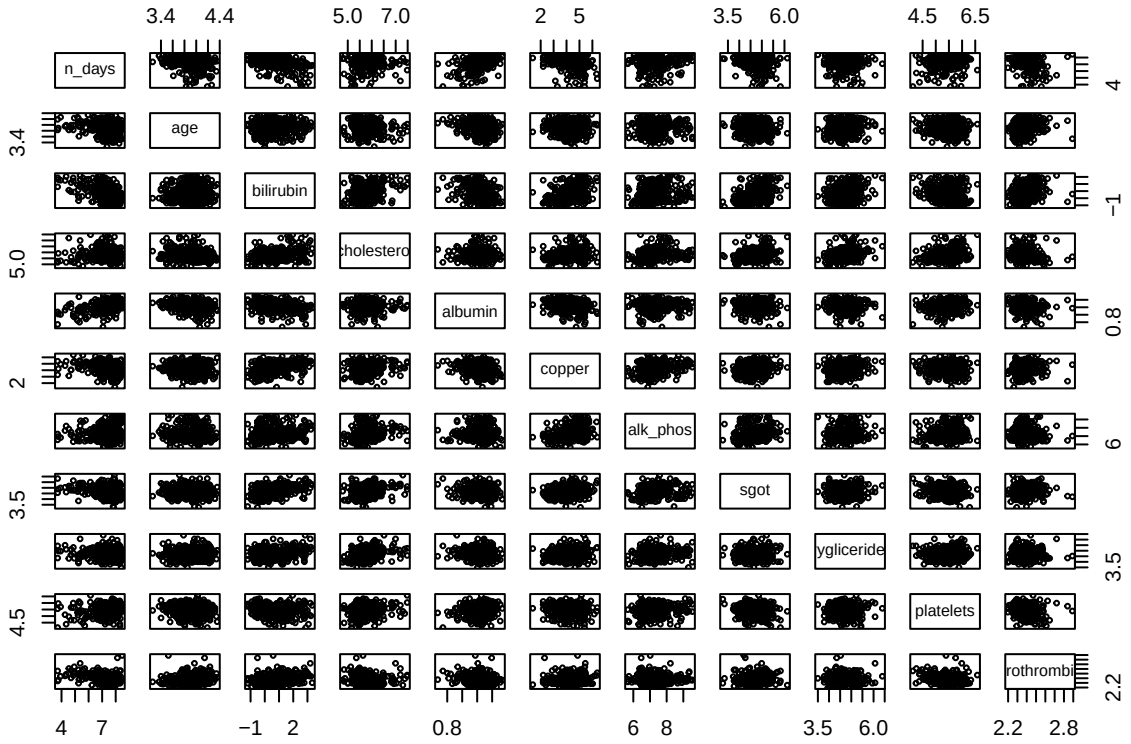
From the mosaic plots, we observe that the majority of the patients in the study were female (255), yet the proportion of alive patients was greater for males. We also note that almost none of the alive patients had `ascites` present. Additionally, almost none of the alive patients had `edema` as well. From the mosaic plot of `drug`, it seems that the administration of the drug did not have an effect on the outcome of the study.

We look at pairs plots to observe any patterns in the data points or covariance between explanatory variables. The second plot shows the log transformed values.

```
#pairs(chol_sub, cex=0.5, col = chol_updated$status)
pairs(chol_sub, cex = 0.5)
```




```
pairs(log(chol_sub), cex = 0.5)
```



From the above pairs plots, none of the variables seem to be highly correlated or have any obvious linear relationships. We observe that log transformation of the values does improve heteroscedasticity, pulls in the large values, and spreads out values close to zero. It seems like a useful tool for our data set.

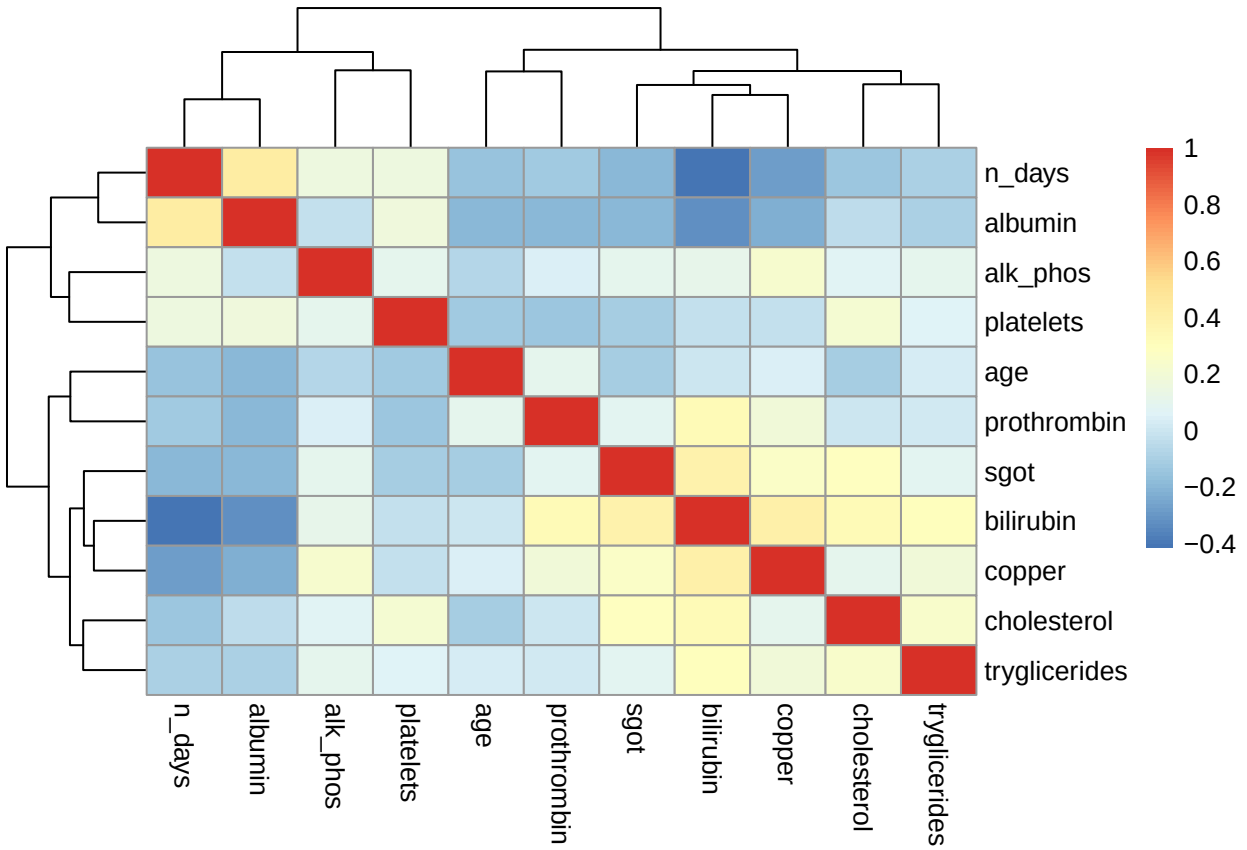
We also look at the correlation matrix to check the numerical values of covariance. We use heat maps to visualize the correlation matrix (i.e. linear relationships among the explanatory variables).

```
cor((chol_sub))
```

```
##                    n_days          age    bilirubin  cholesterol     albumin
## n_days         1.00000000 -0.157104423 -0.415294596 -0.140355522  0.42878749
## age           -0.15710442  1.000000000  0.004278639 -0.114019424 -0.19669603
## bilirubin     -0.41529460  0.004278639  1.000000000  0.326789201 -0.31667663
## cholesterol   -0.14035552 -0.114019424  0.326789201  1.000000000 -0.04040746
## albumin        0.42878749 -0.196696035 -0.316676634 -0.040407457  1.00000000
## copper        -0.28353170  0.040030966  0.401446408  0.102523386 -0.21777968
## alk_phos       0.16100394 -0.062796410  0.122140239  0.078094608 -0.03288182
## sgot          -0.19974621 -0.107487272  0.383117839  0.279632506 -0.19513341
## tryglicerides -0.09137315  0.033087950  0.304268395  0.243914178 -0.09859746
## platelets      0.15181116 -0.125884463 -0.030700976  0.215039918  0.16639650
## prothrombin   -0.11903369  0.106448761  0.330834496  0.006585721 -0.20021796
##                    copper    alk_phos        sgot tryglicerides   platelets
## n_days        -0.28353170  0.16100394 -0.19974621   -0.09137315  0.15181116
## age            0.04003097 -0.06279641 -0.10748727    0.03308795 -0.12588446
## bilirubin      0.40144641  0.12214024  0.38311784    0.30426839 -0.03070098
## cholesterol    0.10252339  0.07809461  0.27963251    0.24391418  0.21503992
```

```
## albumin       -0.21777968 -0.03288182 -0.19513341   -0.09859746  0.16639650
## copper         1.00000000  0.22827051  0.25551862    0.18153460 -0.02873867
## alk_phos       0.22827051  1.00000000  0.10144480    0.09868158  0.10293471
## sgot           0.25551862  0.10144480  1.00000000    0.08099354 -0.11346838
## tryglicerides  0.18153460  0.09868158  0.08099354    1.00000000  0.06272690
## platelets     -0.02873867  0.10293471 -0.11346838    0.06272690  1.00000000
## prothrombin    0.19035345  0.04906747  0.08866652    0.01542020 -0.14597076
##                prothrombin
## n_days        -0.119033689
## age            0.106448761
## bilirubin      0.330834496
## cholesterol    0.006585721
## albumin       -0.200217959
## copper         0.190353447
## alk_phos       0.049067475
## sgot           0.088666515
## tryglicerides  0.015420203
## platelets     -0.145970763
## prothrombin    1.000000000
```

```
pheatmap(cor(chol_sub))
```



`Bilirubin` is the only variable that seems to have some degree of correlation with some of the other explanatory variables. However, since this correlation is not very high, we will not drop this variable for now.

Now that we finished with our initial exploratory data analysis and visualization, we move onto regression analysis.

Multivariate Regression

We run a multivariate regression model with `n_days` (number of days between registration and the earlier of: death, transplantation, or end of study) as our response variable. Then, we look at summary statistics of the linear model and plot the regression diagnostics. We also do variable selection to choose the optimal model.

```
#chol_updated is cholangitis with id variable dropped
summary(lm(n_days~., data = chol_updated))
```

```
##
## Call:
## lm(formula = n_days ~ ., data = chol_updated)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2322.31  -622.92    -1.42   565.50  2270.68
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.524e+03  1.016e+03  -1.501 0.134665
## status1        6.011e+02  1.369e+02   4.390 1.63e-05 ***
## drugPlacebo   -2.293e+01  1.085e+02  -0.211 0.832792
## age           -2.574e+00  5.757e+00  -0.447 0.655188
## sexM           1.503e+02  1.826e+02   0.823 0.411127
## ascitesY      -6.060e+01  2.701e+02  -0.224 0.822684
## hepatomegalyY -3.164e+01  1.267e+02  -0.250 0.803024
## spidersY      -8.161e+01  1.325e+02  -0.616 0.538586
## edemaS        -1.241e+02  1.978e+02  -0.627 0.531038
## edemaY        -3.467e+02  2.900e+02  -1.196 0.232923
## bilirubin     -5.132e+01  1.737e+01  -2.955 0.003408 **
## cholesterol   -1.388e-01  2.905e-01  -0.478 0.633270
## albumin        5.473e+02  1.525e+02   3.589 0.000395 ***
## copper        -1.914e+00  7.712e-01  -2.482 0.013679 *
## alk_phos       1.375e-01  2.523e-02   5.450 1.15e-07 ***
## sgot           6.313e-01  1.090e+00   0.579 0.562971
## tryglicerides  7.036e-01  9.521e-01   0.739 0.460565
## platelets      3.900e-01  6.078e-01   0.642 0.521637
## prothrombin    1.581e+02  6.332e+01   2.497 0.013144 *
## stage2        -2.199e+02  2.577e+02  -0.853 0.394233
## stage3        -3.302e+02  2.534e+02  -1.303 0.193600
## stage4        -5.013e+02  2.737e+02  -1.832 0.068095 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 877.1 on 266 degrees of freedom
## Multiple R-squared:  0.4543, Adjusted R-squared:  0.4112
## F-statistic: 10.55 on 21 and 266 DF,  p-value: < 2.2e-16
```

We observe that none of the categorical variables except `status` are good predictors (i.e. they are not statistically significant). So, we can fit another model by dropping all categorical variables except `status` and compare the results.

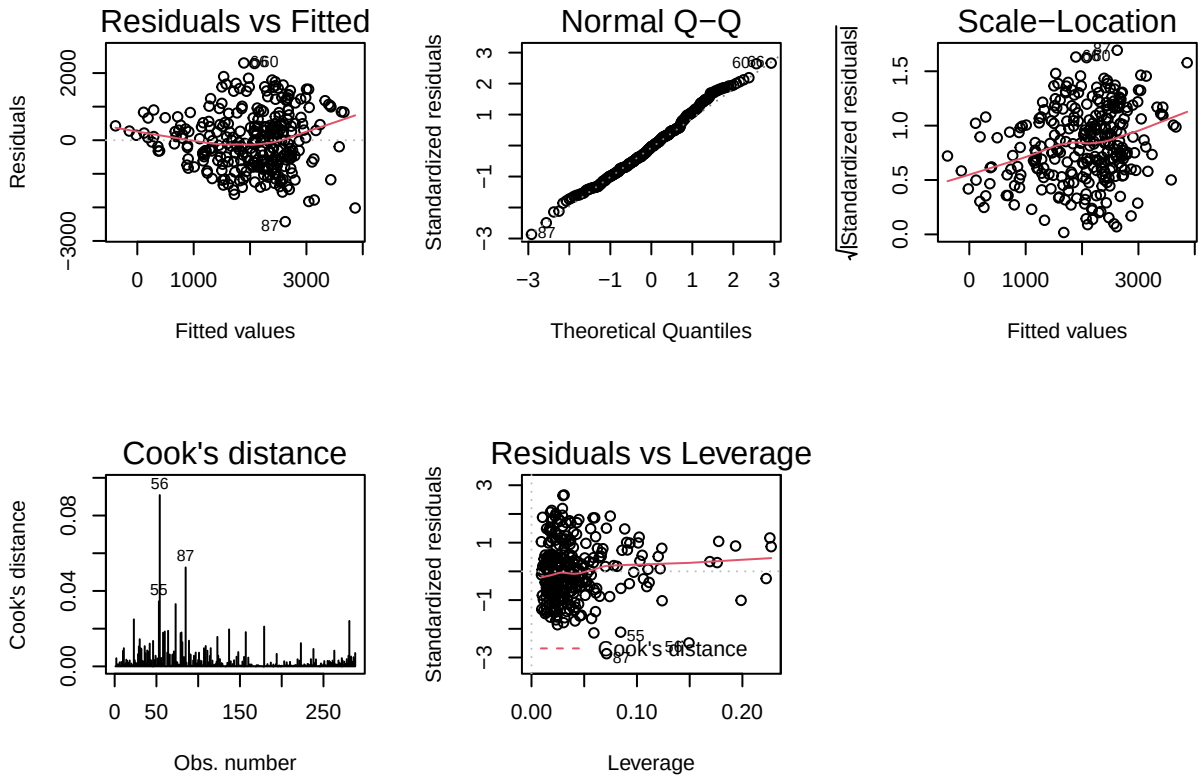
```
#create new subset by dropping categorical variables and omitting NA values
chol_alt <- subset(na.omit(chol_updated), select = -c(3,5,6,7,8,9,19))
chol_fit <- lm(n_days~., data = chol_alt)
summary(chol_fit)
```

```
##
## Call:
## lm(formula = n_days ~ ., data = chol_alt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2426.43  -611.69   -31.09   497.57  2300.82
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.307e+03  9.541e+02  -2.418 0.016251 *
## status1        6.669e+02  1.329e+02   5.020 9.28e-07 ***
## age           -2.433e+00  5.422e+00  -0.449 0.654033
## bilirubin     -6.225e+01  1.618e+01  -3.848 0.000148 ***
## cholesterol   -6.562e-03  2.803e-01  -0.023 0.981338
## albumin        7.142e+02  1.381e+02   5.173 4.45e-07 ***
## copper        -2.027e+00  7.329e-01  -2.766 0.006065 **
## alk_phos       1.420e-01  2.508e-02   5.661 3.77e-08 ***
## sgot           7.071e-01  1.073e+00   0.659 0.510422
## tryglicerides  5.552e-01  9.280e-01   0.598 0.550115
## platelets      6.393e-01  5.875e-01   1.088 0.277481
## prothrombin    1.268e+02  5.982e+01   2.120 0.034876 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 878.3 on 276 degrees of freedom
## Multiple R-squared:  0.4322, Adjusted R-squared:  0.4096
## F-statistic:  19.1 on 11 and 276 DF,  p-value: < 2.2e-16
```

The significance of `bilirubin` and `copper` increased by one star each as we dropped the categorical variables. It appears that our model has improved and we seem to be on the right track. Although dropping the categorical columns may come across as cherry-picking of the data (or "p-hacking"), I ran multiple models and applied multiple comparison correction (i.e. Bonferroni correction) and only after that did I decide to drop said variables.

Now we plot the regression diagnostics to visualize our fitted model.

```
par(mfrow=c(2,3))
plot(chol_fit, which = 1:5)
```



There are 3 observations with large residuals and large Cook's distances (55,56,87). From the Q-Q plot, we observe that our data largely conforms to the normality condition. However, we notice some heteroscedasticity in the first plot, i.e. large values have higher variance (and mean of larger residuals is increasing).

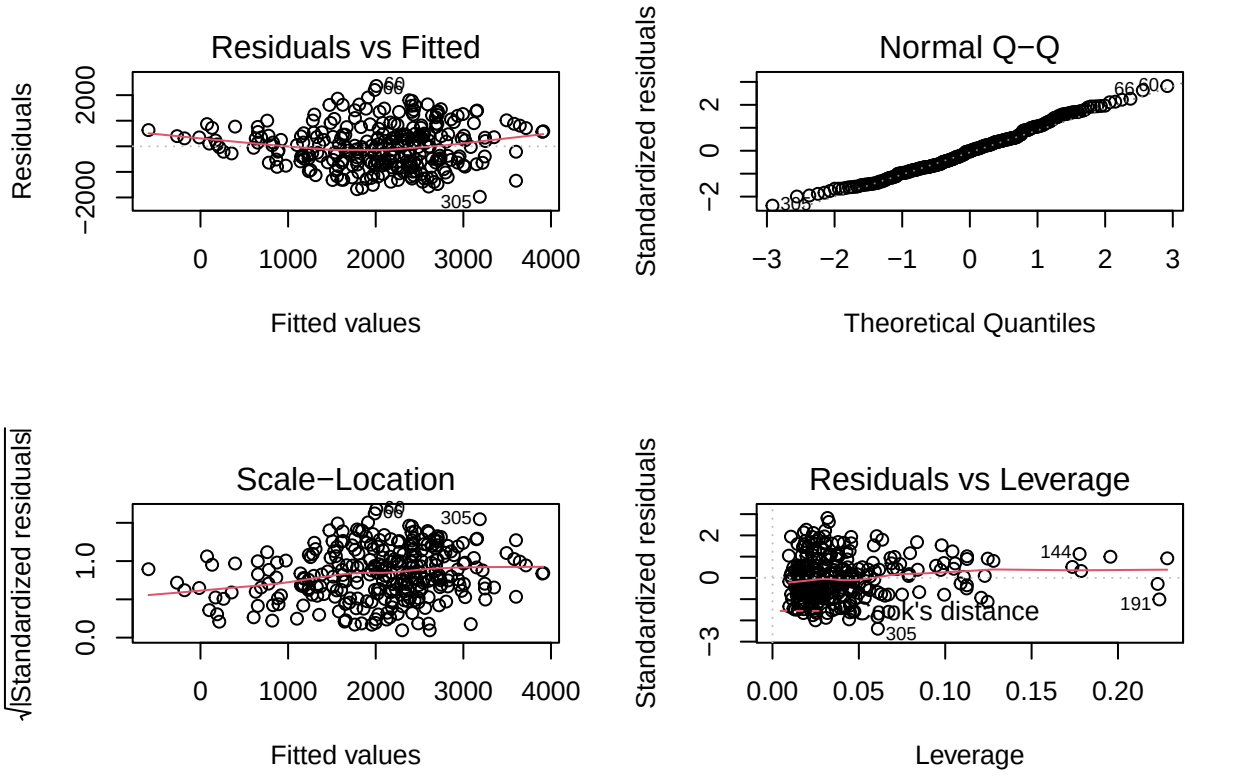
To deal with the violation of the homoscedasticity regression assumption, we can transform the data (log or square root) and check if that improves our model. But first, we will remove the outliers we identified and recompute our model and compare the results.

```
#removing outliers and plotting regression diagnostics again
outliers <- which(cooks.distance(chol_fit) > 0.03)
chol_alt <- chol_alt[-outliers,]
chol_alt_fit <- lm(n_days~., data = chol_alt)
summary(chol_alt_fit)
```

```
##
## Call:
## lm(formula = n_days ~ ., data = chol_alt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1971.78  -608.31   -26.88   478.59  2358.21
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -2.682e+03  9.281e+02  -2.890  0.00416 **
## status1        5.855e+02  1.300e+02   4.505 9.86e-06 ***
```

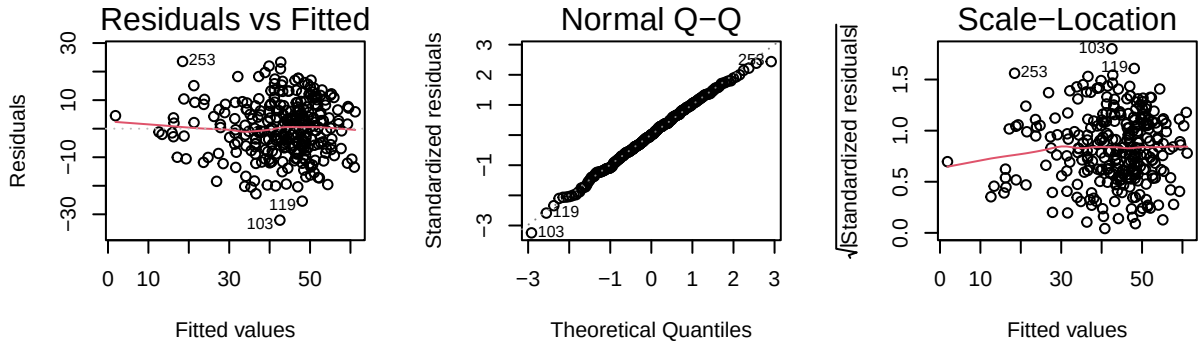
```
## age           -2.831e+00  5.327e+00  -0.532  0.59545
## bilirubin     -6.822e+01  1.574e+01  -4.333 2.07e-05 ***
## cholesterol   -3.872e-02  2.724e-01  -0.142  0.88708
## albumin        8.264e+02  1.367e+02   6.045 4.91e-09 ***
## copper        -2.176e+00  7.111e-01  -3.060  0.00244 **
## alk_phos       1.748e-01  2.619e-02   6.675 1.39e-10 ***
## sgot           9.156e-01  1.050e+00   0.872  0.38378
## tryglicerides  5.990e-01  1.020e+00   0.587  0.55755
## platelets      7.233e-01  5.762e-01   1.255  0.21043
## prothrombin    1.265e+02  5.797e+01   2.183  0.02992 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 850.2 on 272 degrees of freedom
## Multiple R-squared:  0.4691, Adjusted R-squared:  0.4476
## F-statistic: 21.85 on 11 and 272 DF,  p-value: < 2.2e-16
```

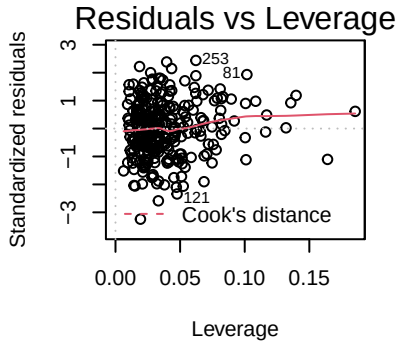
```
par(mfrow=c(2,2))
plot(chol_alt_fit)
```



Removing the outliers seems to have improved our model as our R^2 and Adjusted R^2 increased. Also, we now observe that`status` has become significant for both factor levels. `status1` refers to the patient being alive and is highly significant with a p-value close to zero. The intercept term is `status` = 0, which refers to the patient being dead and it is significant with a p-value slightly above 0.1. Hence, `status` is the only categorical variable important to our model.

```
par(mfrow=c(2,3))
temp <- lm(n_days~., data = sqrt(subset(chol_alt, select = -status)))
plot(temp)
```



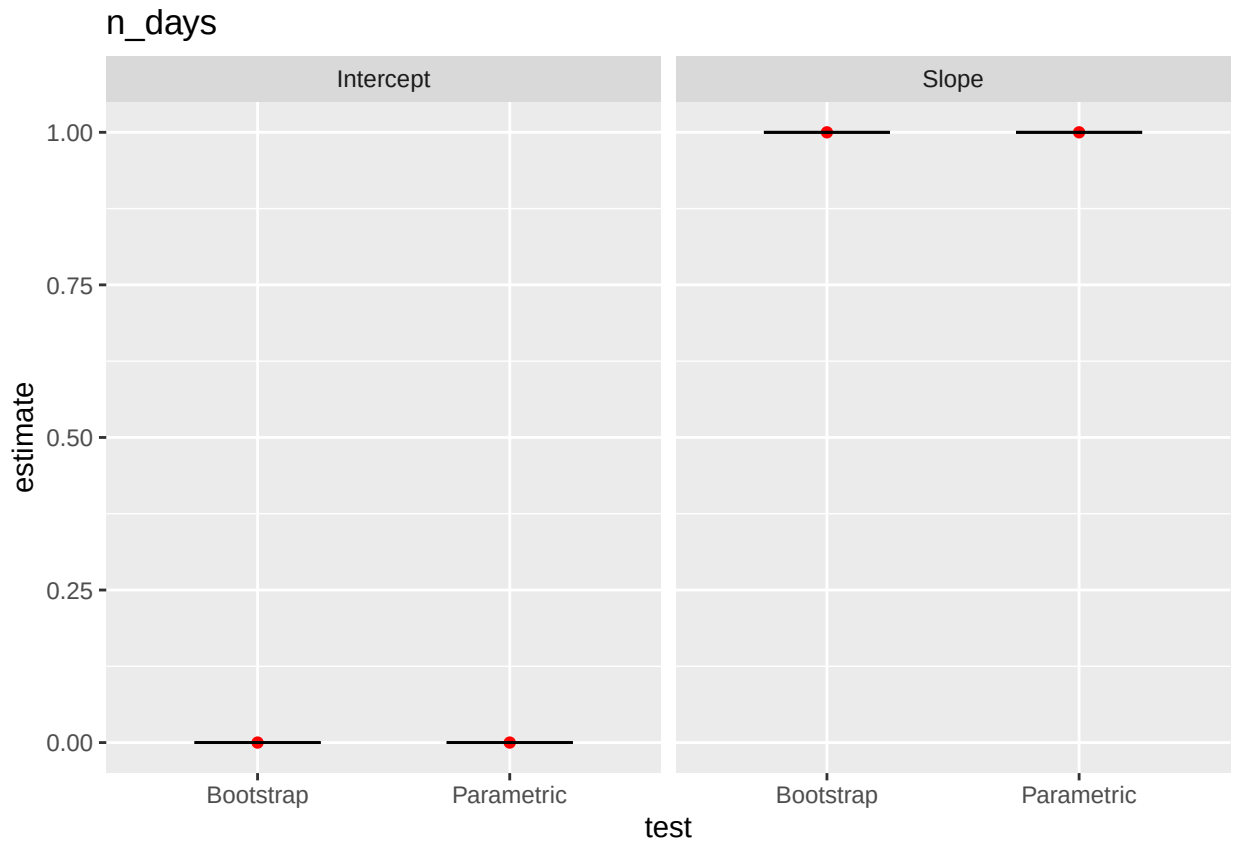


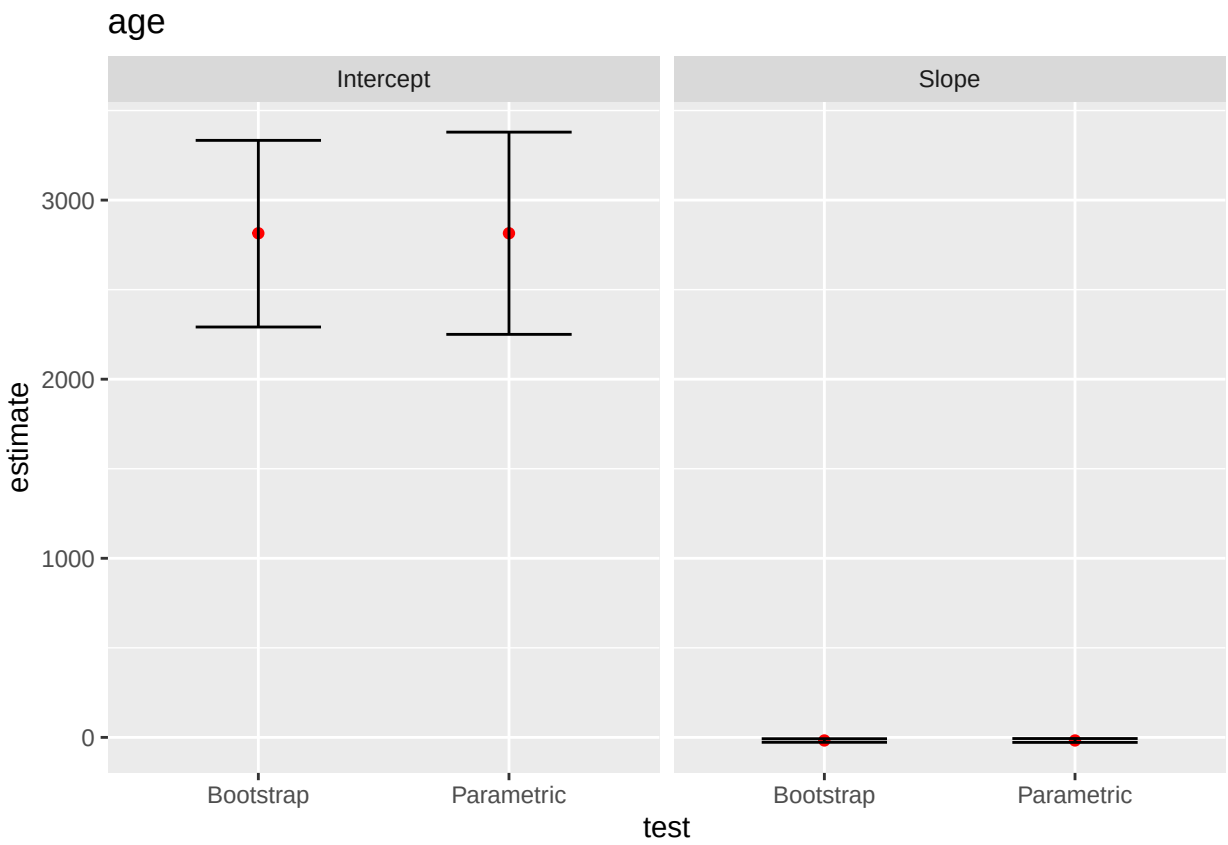
Next we construct Bootstrap and Parametric Models for Global Fit (numerical variables only). We do this to check the robustness of our analysis. We compare the two confidence intervals and plot them to observe the differences visually.

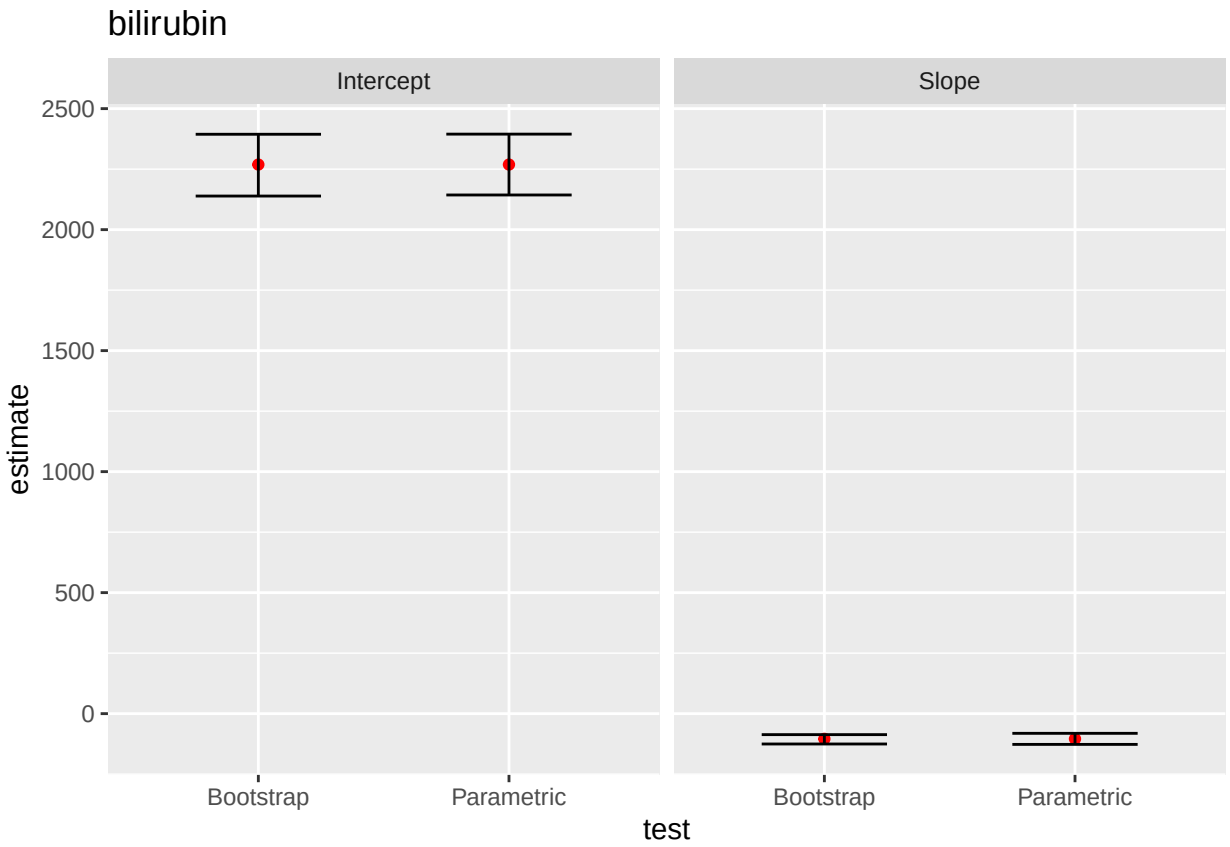
```
bootstrapLM <- function(y,x, repetitions, df= chol_sub, confidence.level= 0.95){
  stat.obs <- coef(lm(y~x))
  bootFun<-function(){
      sampled <- sample(1:length(y), size=length(y),replace = TRUE)
      coef(lm(y[sampled]~x[sampled]))
  }
  stat.boot<-replicate(repetitions,bootFun())
  nm <-deparse(substitute(x))

  row.names(stat.boot)[2]<-nm
  level<-1-confidence.level
  confidence.interval <- apply(stat.boot,1,quantile,probs=c(level/2,1-level/2))
  out<-cbind("lower"=confidence.interval[1,],
             "estimate"=stat.obs,"upper"=confidence.interval[2,])
  return(list(confidence.interval = out))
}

for (i in names(chol_sub)) {
```

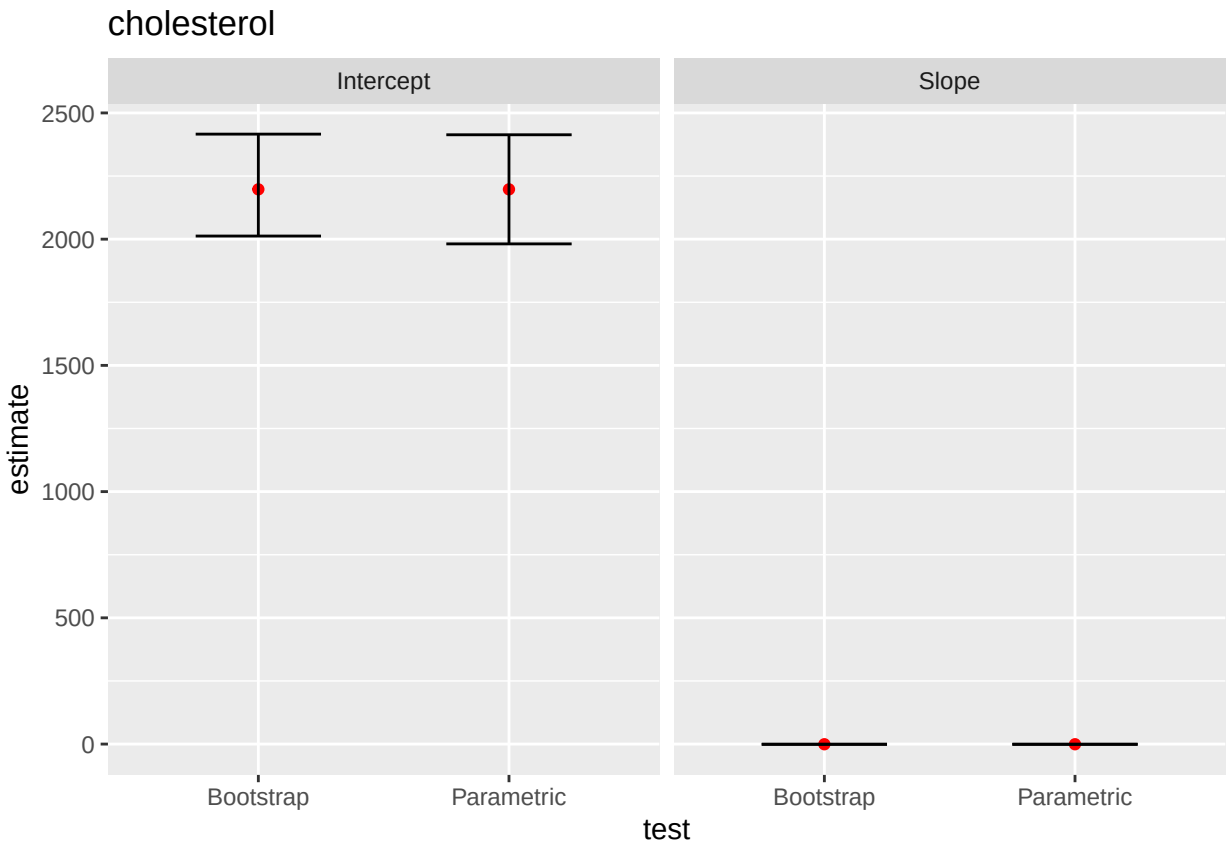
```
boot_LM <- bootstrapLM(chol_sub$n_days, chol_sub[, i], 1000)
boot_CI <- boot_LM$confidence.interval
BS_fit <- lm(n_days~chol_sub[,i], data = chol_sub)
parametric_CI <- confint(BS_fit)

boot_test_CI = rbind(c(lower= parametric_CI[1,1], estimate = unname(coef(BS_fit)[1]),
                     upper = parametric_CI[1,2]), boot_CI[1,])

perm_test_CI = rbind(c(lower = parametric_CI[2,1], estimate = unname(coef(BS_fit)[2]),
                       upper = parametric_CI[2,2]), boot_CI[2,])

df_plot = data.frame(rbind(boot_test_CI, perm_test_CI))
df_plot$test = c("Parametric", "Bootstrap", "Parametric", "Bootstrap")
df_plot$type = c("Intercept", "Intercept", "Slope", "Slope")
print(ggplot(df_plot, aes(test, estimate)) + geom_point(aes(y= estimate), color = "red") +
        geom_errorbar(aes(x= test, ymin = lower, ymax = upper), color = "black", width = 0.5) + facet_g
}
```

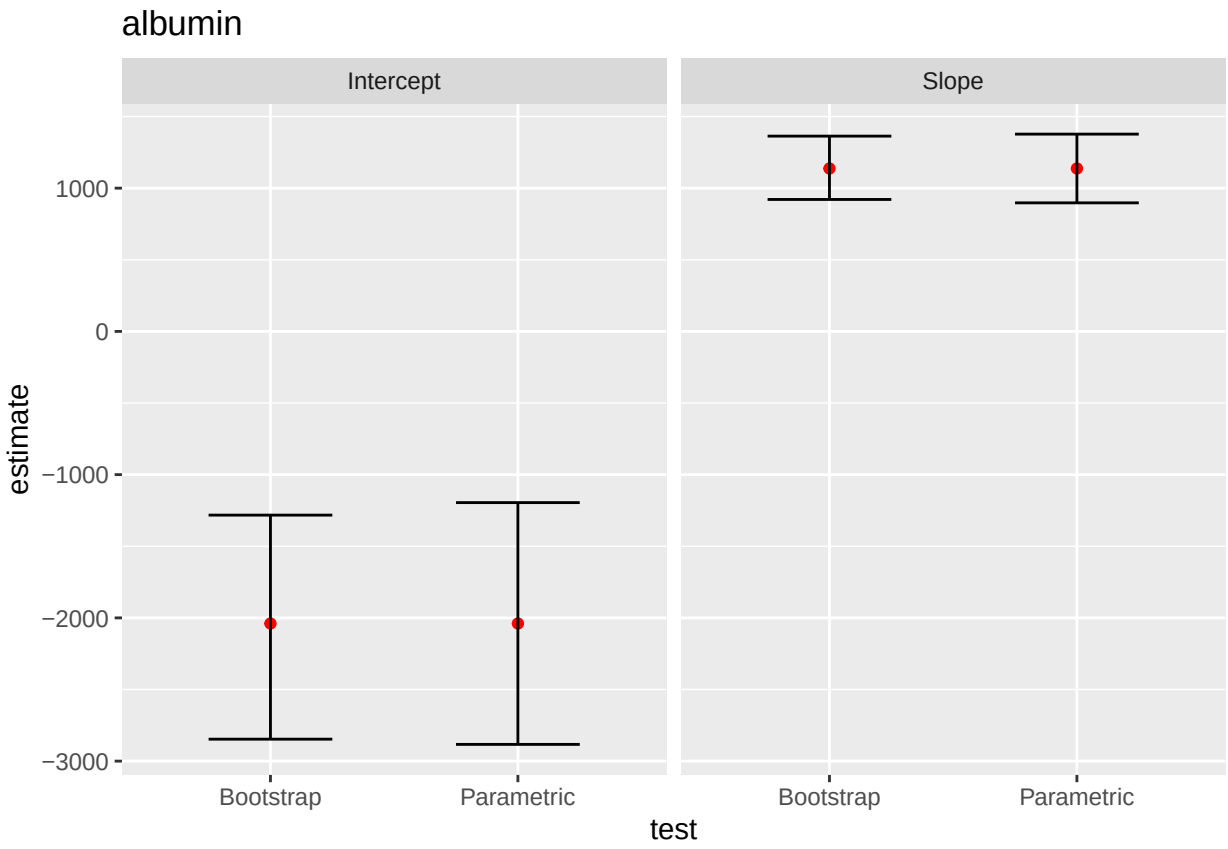
```
## Warning in summary.lm(object, ...): essentially perfect fit: summary may be
## unreliable
```

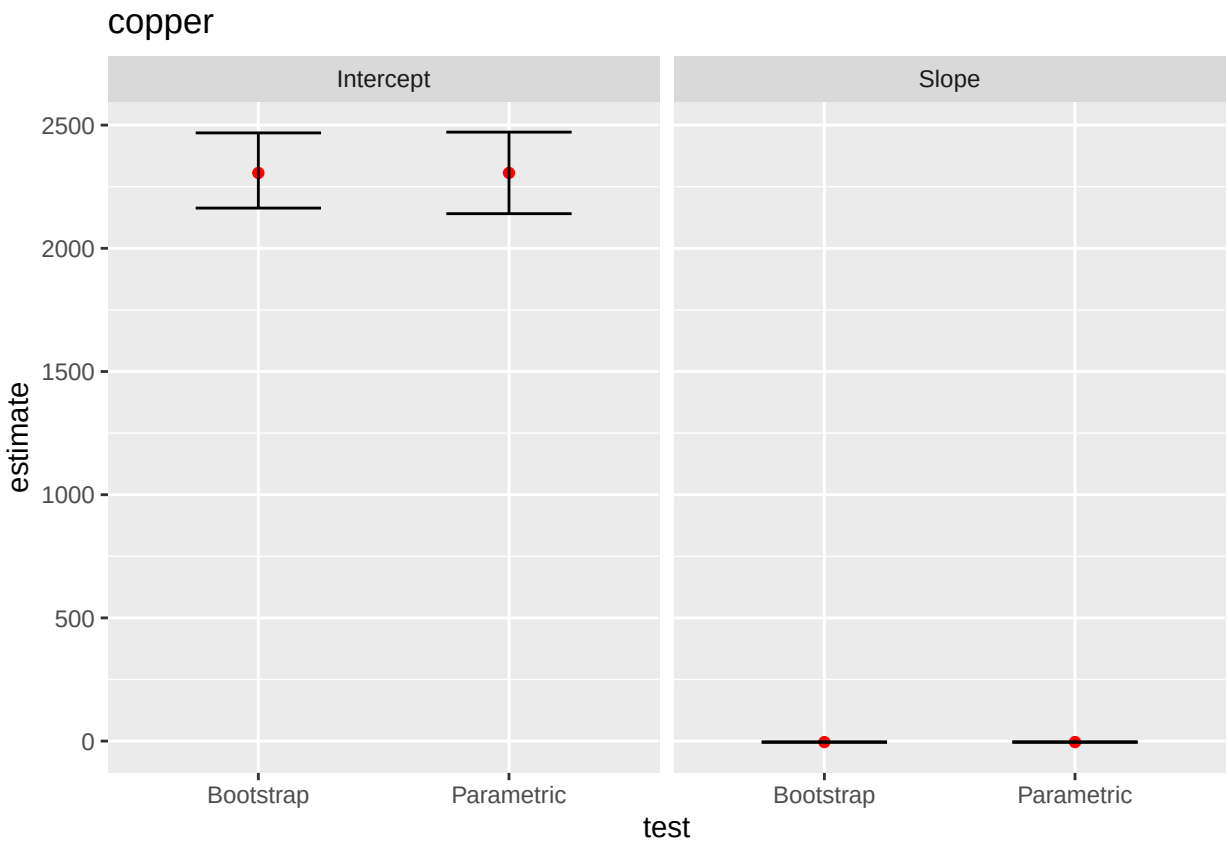


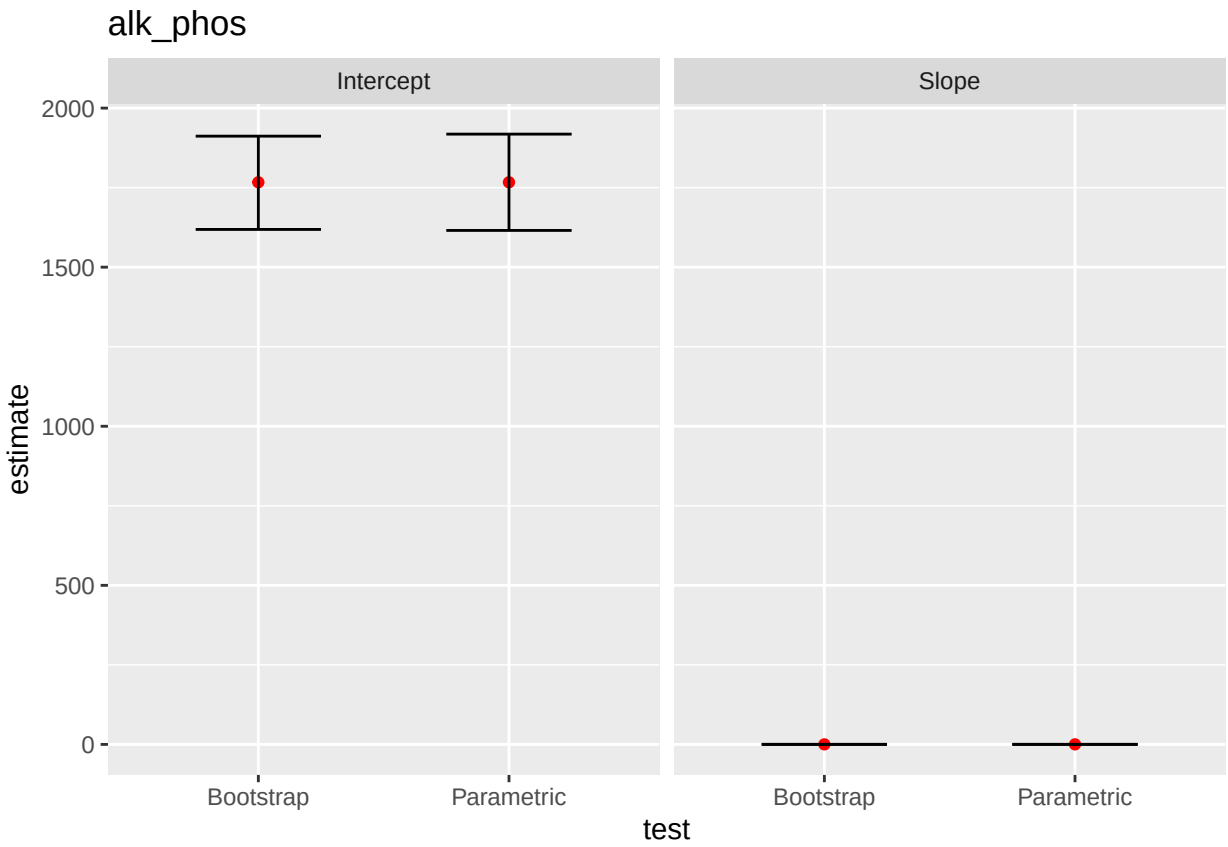

age
Intercept
Slope
3000
2000
1000
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test

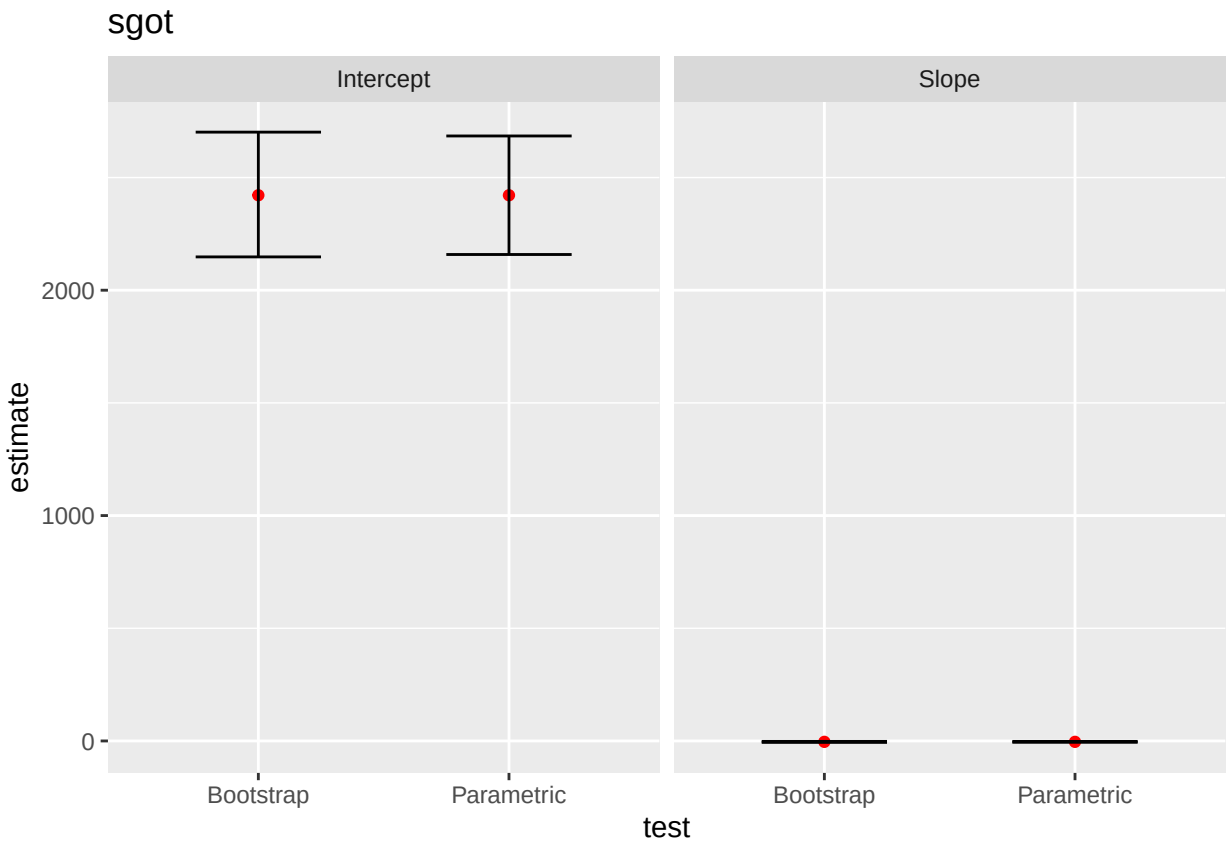

bilirubin
Intercept
Slope
2500
2000
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test

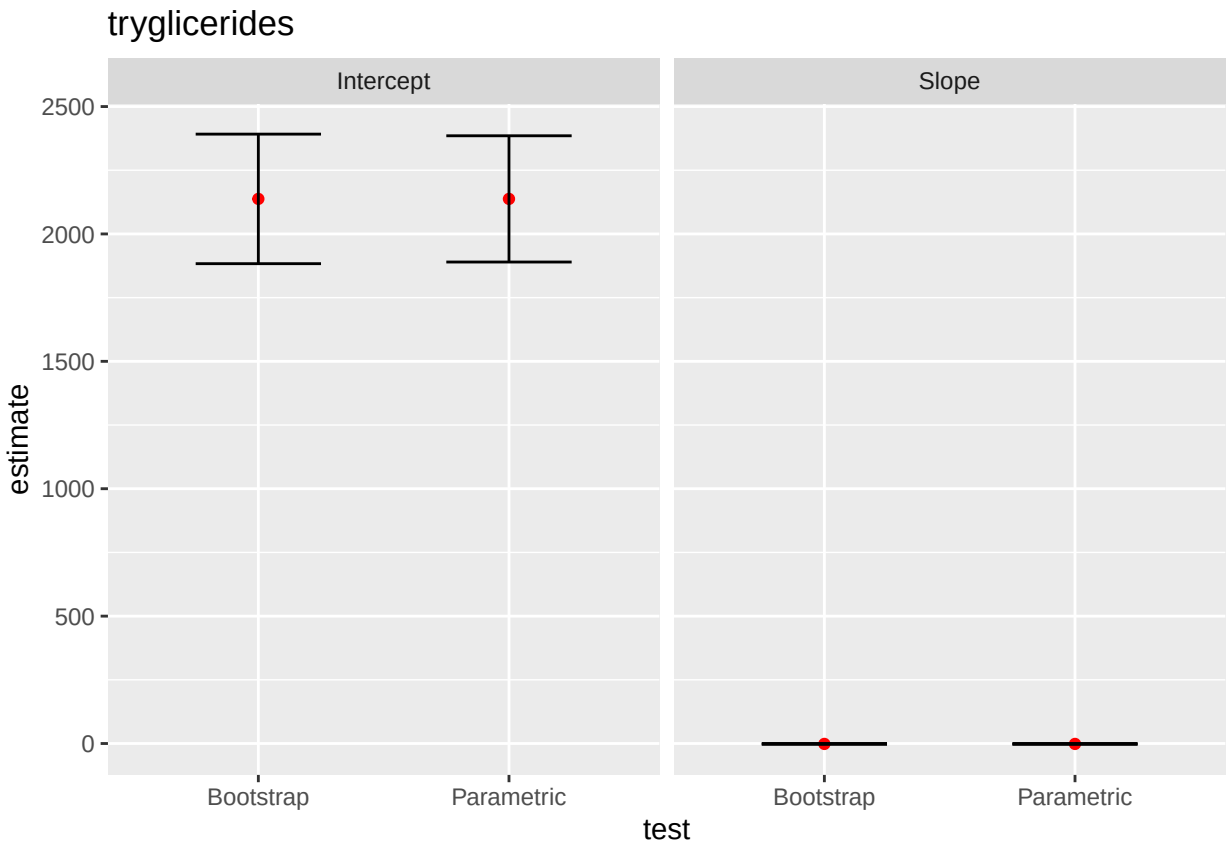

cholesterol
Intercept
Slope
2500
2000
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test

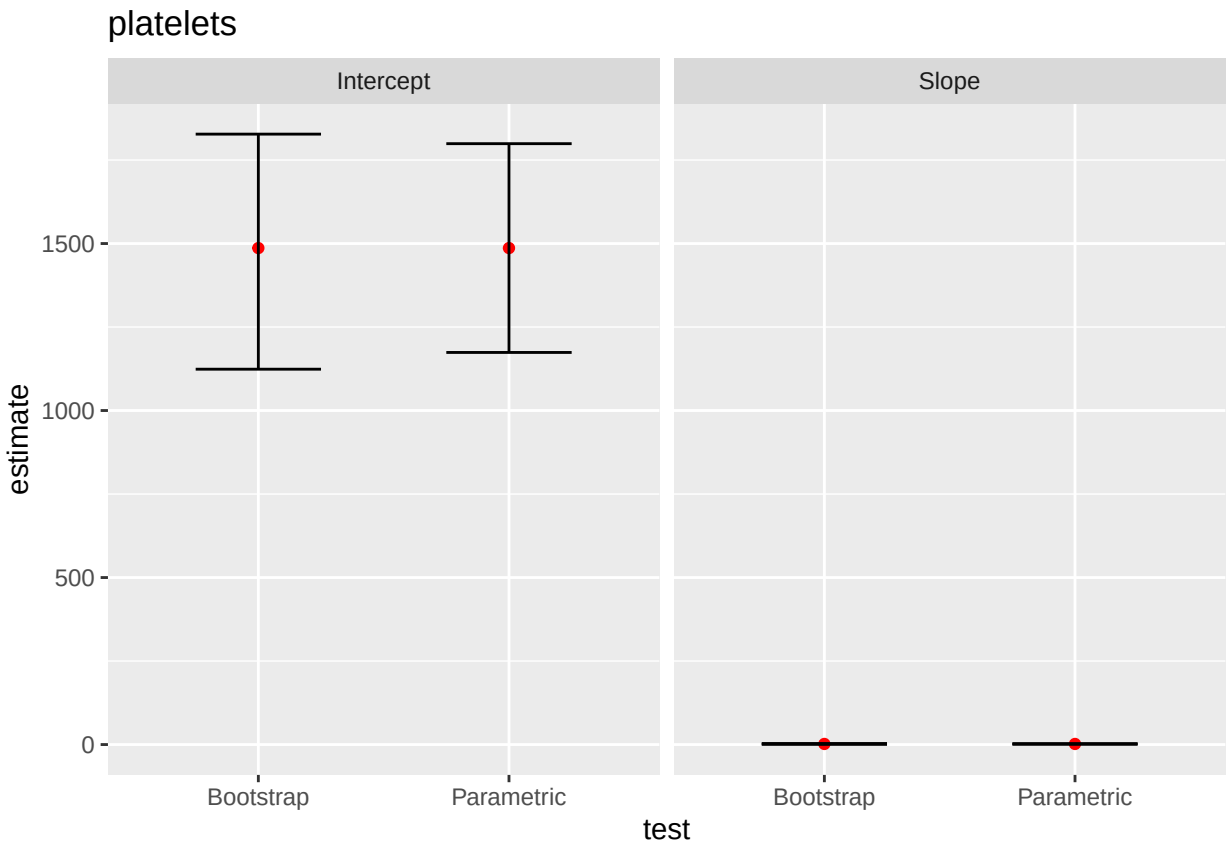

albumin
Intercept
Slope
1000
0
−1000
−2000
−3000
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test

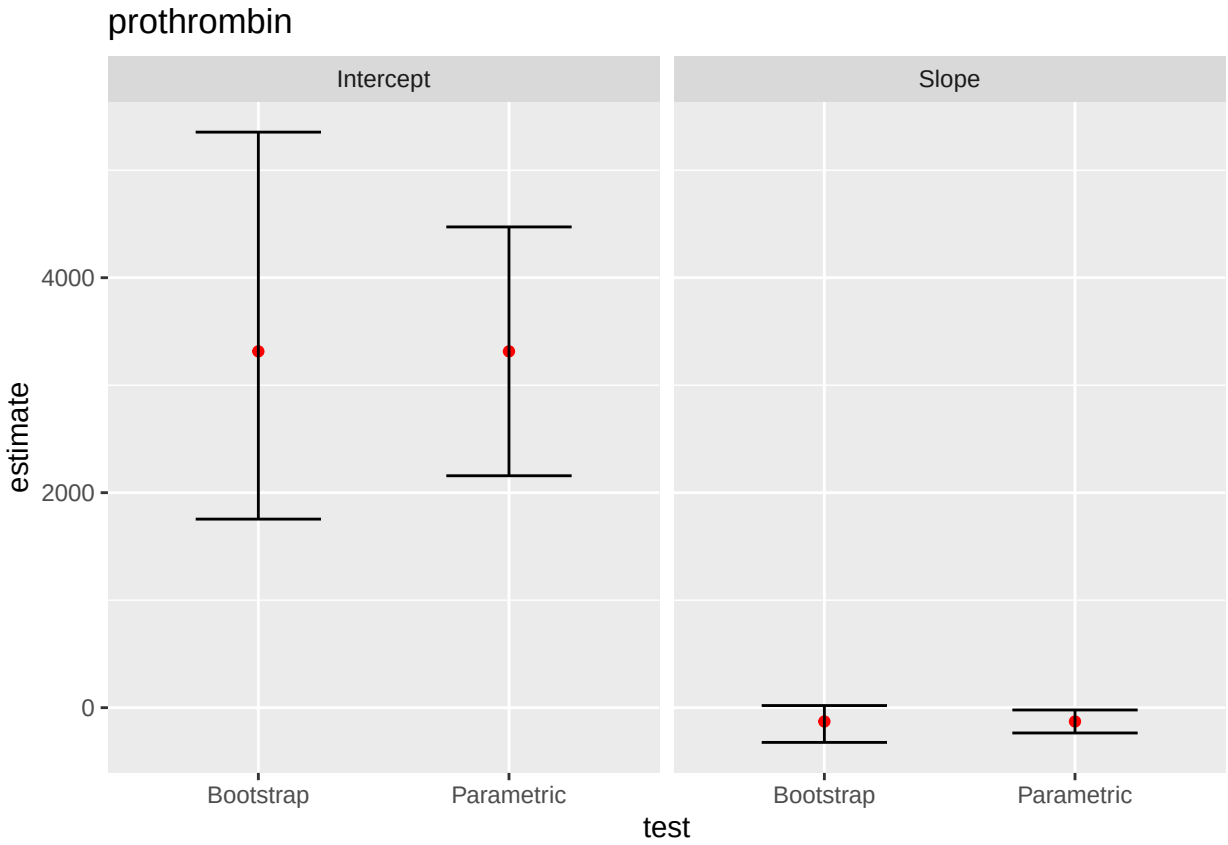

copper
Intercept
Slope
2500
2000
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test


alk_phos
Intercept
Slope
2000
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test


sgot
Intercept
Slope
2000
1000
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test


tryglicerides
Intercept
Slope
2500
2000
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test


platelets
Intercept
Slope
1500
1000
500
0
estimate
Bootstrap
Parametric
Bootstrap
Parametric
test



From the graphs, we notice that the parametric confidence intervals are slightly wider than the bootstrap ones. This is expected because of the difference in assumptions between the two models and so the parametric model has a little more variability. Even so, we can conclude that our model meets robustness criteria.

Next, we do variable selection using the `step` function to find the optimal submodel.

```
step(chol_alt_fit, direction = "backward", trace = F)
```

```
##
## Call:
## lm(formula = n_days ~ status + bilirubin + albumin + copper +
##     alk_phos + prothrombin, data = chol_alt)
##
## Coefficients:
## (Intercept)      status1    bilirubin      albumin       copper     alk_phos
##  -2389.4038     595.6851     -60.3022     860.5169      -2.1170       0.1846
## prothrombin
##    102.7772
```

Our optimal model from using `step` to predict `n_days` is `Status` + `Bilirubin` + `Albumin` + `Copper` + `Alk_Phos` + `Prothrombin`.

Next, we use `regsubsets` to do variable selection again to check if the results match our results from using the `step` function.

```r
chol_mod_selection <- subset(chol_alt, select = -status)
reg.sub.out <- regsubsets(n_days~., data = chol_mod_selection)
summary(reg.sub.out)
```

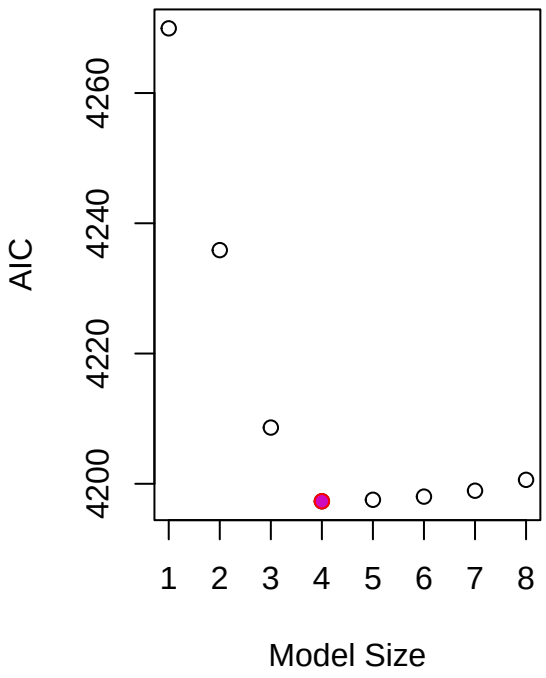
```
## Subset selection object
## Call: regsubsets.formula(n_days ~ ., data = chol_mod_selection)
## 10 Variables  (and intercept)
##               Forced in Forced out
## age               FALSE      FALSE
## bilirubin         FALSE      FALSE
## cholesterol       FALSE      FALSE
## albumin           FALSE      FALSE
## copper            FALSE      FALSE
## alk_phos          FALSE      FALSE
## sgot              FALSE      FALSE
## tryglicerides     FALSE      FALSE
## platelets         FALSE      FALSE
## prothrombin       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          age bilirubin cholesterol albumin copper alk_phos sgot tryglicerides
## 1  ( 1 ) " " " "       " "         "*"     " "    " "      " "  " "
## 2  ( 1 ) " " "*"       " "         "*"     " "    " "      " "  " "
## 3  ( 1 ) " " "*"       " "         "*"     " "    "*"      " "  " "
## 4  ( 1 ) " " "*"       " "         "*"     "*"    "*"      " "  " "
## 5  ( 1 ) " " "*"       " "         "*"     "*"    "*"      " "  " "
## 6  ( 1 ) "*" "*"       " "         "*"     "*"    "*"      " "  " "
## 7  ( 1 ) "*" "*"       " "         "*"     "*"    "*"      " "  " "
## 8  ( 1 ) "*" "*"       "*"         "*"     "*"    "*"      " "  " "
##          platelets prothrombin
## 1  ( 1 ) " "       " "
## 2  ( 1 ) " "       " "
## 3  ( 1 ) " "       " "
## 4  ( 1 ) " "       " "
## 5  ( 1 ) "*"       " "
## 6  ( 1 ) "*"       " "
## 7  ( 1 ) "*"       "*"
## 8  ( 1 ) "*"       "*"
```

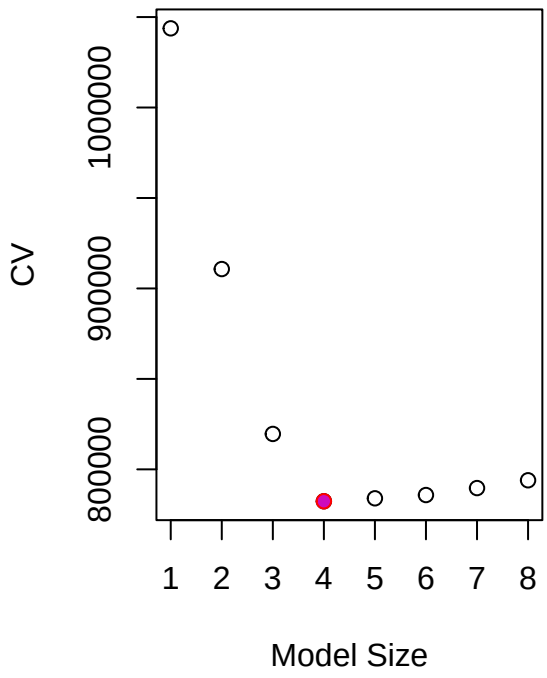
So, our optimal model from `regsubsets` is `Albumin` + `Bilirubin` + `Alk_Phos` + `Copper` + `Platelets` + `Age` + `Prothrombin` + `Cholesterol`. The two models gives us different optimal models, which was expected as `regsubsets` does not work well with categorical variables and it is possible to miss the optimal model with `step`.

Next, we do model selection by comparing results from AIC and LOOCV (Leave One Out Cross Validation). We plot the results from these methods to check if they give us the same answer.

```r
permut <- sample(1:nrow(chol_mod_selection))
folds <- cut(1:nrow(chol_mod_selection), breaks = 10, labels = F)
PredErrorMat <- matrix(nrow = 10, ncol = nrow(summary(reg.sub.out)$which))
for (i in 1:10) {
  test_indices <- which(folds == i, arr.ind = T)
  test_data <- chol_mod_selection[permut, ][test_indices, ]
```

```
  train_data <- chol_mod_selection[permut, ][-test_indices, ]
  pred_error <- apply(summary(reg.sub.out)$which[, -1], 1,
                      function(x) {
                        lm_obj <- lm(train_data$n_days ~.,
                                     data = train_data[, c("n_days", names(x)[x]), drop = FALSE])
                        AIC(lm_obj)
                      })
  PredErrorMat[i, ] <- pred_error
}

chol_AIC <- colMeans(PredErrorMat)

LOOCV <- function(lm) {
  vals <- residuals(lm)/(1 - lm.influence(lm)$hat)
  sum(vals^2)/length(vals) }

permutation <- sample(1:nrow(chol_mod_selection))
folds_2 <- cut(1:nrow(chol_mod_selection), breaks = 10, labels = F)
PredErrorMatrix <- matrix(nrow = 10, ncol = nrow(summary(reg.sub.out)$which))
for (i in 1:10) {
  test_indices <- which(folds_2 == i, arr.ind = T)
  test_data <- chol_mod_selection[permutation, ][test_indices, ]
  train_data <- chol_mod_selection[permutation, ][-test_indices, ]
  pred_error <- apply(summary(reg.sub.out)$which[, -1], 1,
                      function(x) {
                        lm_obj <- lm(train_data$n_days ~.,
                                     data = train_data[, c("n_days", names(x)[x]), drop = FALSE])
                        LOOCV(lm_obj)
                      })
  PredErrorMatrix[i, ] <- pred_error
}

chol_cv <- colMeans(PredErrorMatrix)

par(mfrow=c(1,2))
plot(1:8, chol_AIC, xlab = "Model Size", ylab = "AIC")
points(which.min(chol_AIC), min(chol_AIC), col = "red", bg = 22, pch = 21)
plot(1:8, chol_cv, xlab = "Model Size", ylab = "CV")
points(which.min(chol_cv), min(chol_cv), col = "red", bg = 22, pch = 21)
```






```
coef(reg.sub.out, which.min(chol_AIC))
```

```
##   (Intercept)     bilirubin       albumin        copper      alk_phos
## -1138.5753968   -71.0903879   952.5119459    -2.7405560     0.1704562
```

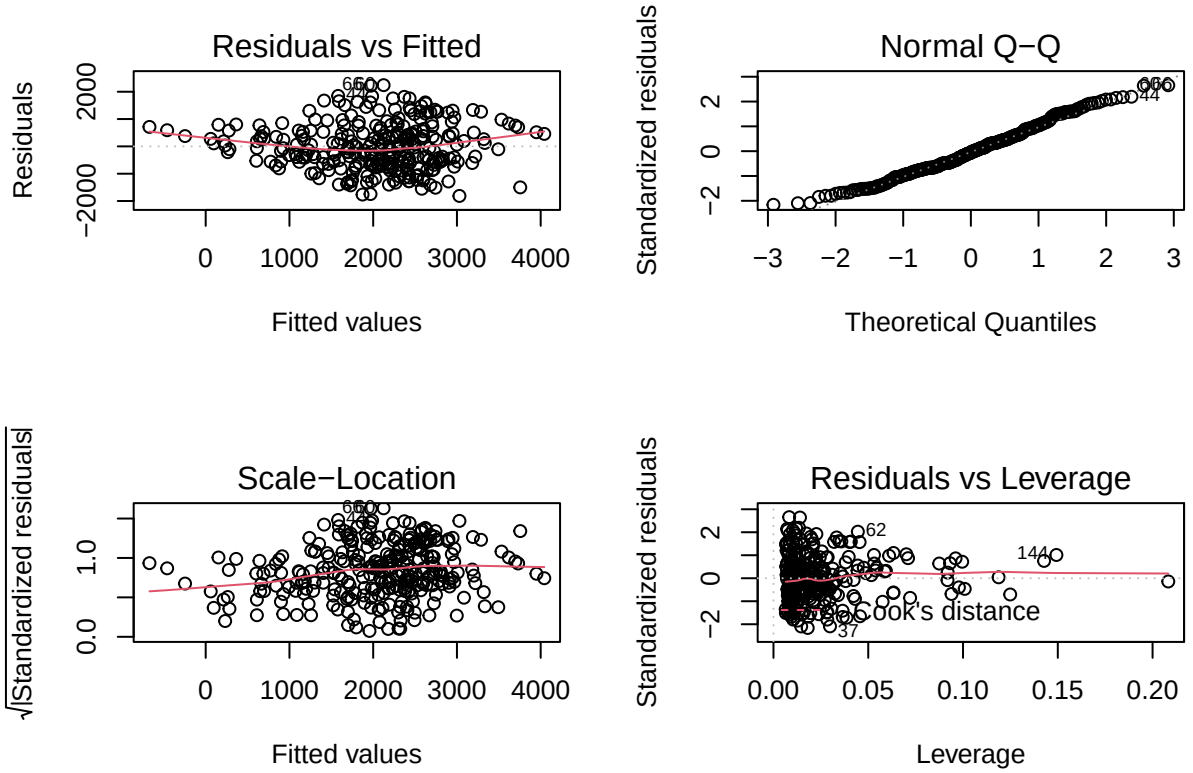
Thus, the results of cross validation (LOOCV) and AIC gives us the same optimal model size (=4). Our optimal model is `Bilirubin` + `Albumin` + `Copper` + `Alk_Phos`. However, it is different than the optimal mode size (=6) that we got from using the `step` function as we removed the `status` categorical variable for running `regsubsets`. We can compare our two nested models by using the `anova` function that uses the F-test.

```
anova(lm(n_days~., data = chol_mod_selection), chol_alt_fit)
```

```
## Analysis of Variance Table
##
## Model 1: n_days ~ age + bilirubin + cholesterol + albumin + copper + alk_phos +
##     sgot + tryglicerides + platelets + prothrombin
## Model 2: n_days ~ status + age + bilirubin + cholesterol + albumin + copper +
##     alk_phos + sgot + tryglicerides + platelets + prothrombin
##   Res.Df       RSS Df Sum of Sq      F    Pr(>F)
## 1    273 211305101
## 2    272 196633196  1  14671904 20.295 9.861e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Thus, we conclude that adding the `status` variable is highly significant as our p-value from running `anova` is close to zero. So, our overall optimal model (size = 6) is the one given by using the `steps` function.

```
par(mfrow = c(2,2))
plot(lm(formula = n_days ~ status + bilirubin + albumin + copper +
    alk_phos + prothrombin, data = chol_alt))
```



Running regression diagnostics on our final model shows us that there is still some heteroscedasticity in our data and we still have outliers that affect our plots. However, we cannot remove these outliers without sufficient domain knowledge. For the same reason, we did not stick with transformations of the data also. Our assumption of normality is not violated, so that is one positive.

Logistic Regression

The next step in our analysis is logistic regression. We fit a generalized linear model by using the `glm` function and input "binomial" in the `family` argument to specify that we want to fit a logistic regression model.

We create an intercept only model that we use to check if our model with all the explanatory variables is better. We will use both models to perform variable selection as before.

We plot the regression diagnostics and look at the summary statistics of our bigger model (`chol.log.fit`).

```
chol.int.fit <- glm(status ~ 1, data = chol_updated, family = "binomial" )
chol.log.fit <- glm(status~., data = chol_updated, family = "binomial")
summary(chol.int.fit)
```

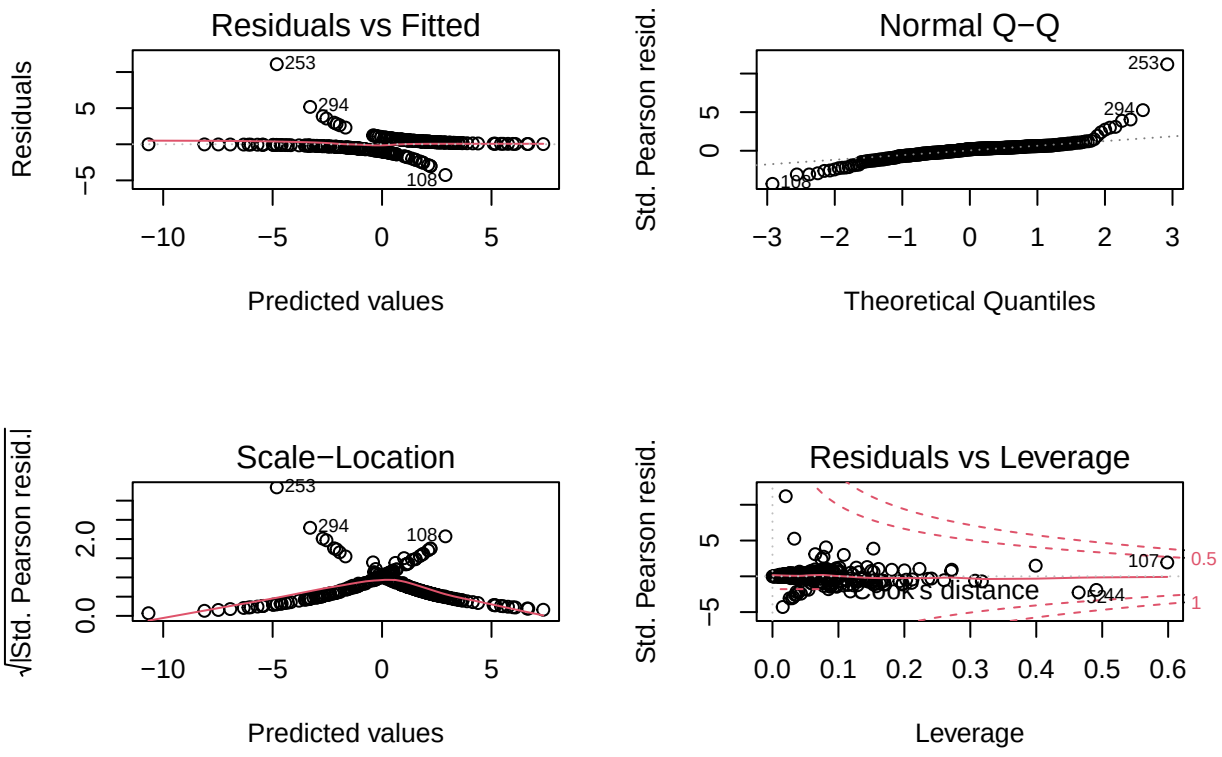
```
##
## Call:
## glm(formula = status ~ 1, family = "binomial", data = chol_updated)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.304  -1.304   1.055   1.055   1.055
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2938     0.1191   2.466   0.0137 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 393.11  on 287  degrees of freedom
## Residual deviance: 393.11  on 287  degrees of freedom
## AIC: 395.11
##
## Number of Fisher Scoring iterations: 4
```

```
summary(chol.log.fit)
```

```
##
## Call:
## glm(formula = status ~ ., family = "binomial", data = chol_updated)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4289  -0.4794   0.2625   0.5948   3.1043
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.772e+01  4.123e+00   4.298 1.72e-05 ***
## n_days         7.561e-04  2.046e-04   3.697 0.000219 ***
## drugPlacebo    3.645e-01  3.563e-01   1.023 0.306309
## age           -3.926e-02  1.845e-02  -2.128 0.033324 *
## sexM          -9.376e-01  6.050e-01  -1.550 0.121202
## ascitesY      -1.477e+00  1.221e+00  -1.209 0.226555
## hepatomegalyY -4.056e-01  3.996e-01  -1.015 0.310087
## spidersY      -2.218e-01  4.224e-01  -0.525 0.599575
## edemaS         1.501e-01  6.835e-01   0.220 0.826159
## edemaY        -6.197e-01  1.483e+00  -0.418 0.676066
## bilirubin     -5.833e-02  8.058e-02  -0.724 0.469149
## cholesterol   -8.217e-04  1.052e-03  -0.781 0.434715
## albumin       -2.999e-01  5.225e-01  -0.574 0.565975
## copper        -2.359e-03  2.929e-03  -0.805 0.420720
## alk_phos      -3.056e-04  9.364e-05  -3.263 0.001102 **
## sgot          -6.187e-03  3.357e-03  -1.843 0.065326 .
## tryglicerides -3.164e-03  3.517e-03  -0.900 0.368321
## platelets     -2.385e-04  2.053e-03  -0.116 0.907520
## prothrombin   -9.465e-01  2.323e-01  -4.076 4.59e-05 ***
## stage2        -2.879e+00  1.568e+00  -1.836 0.066335 .
```

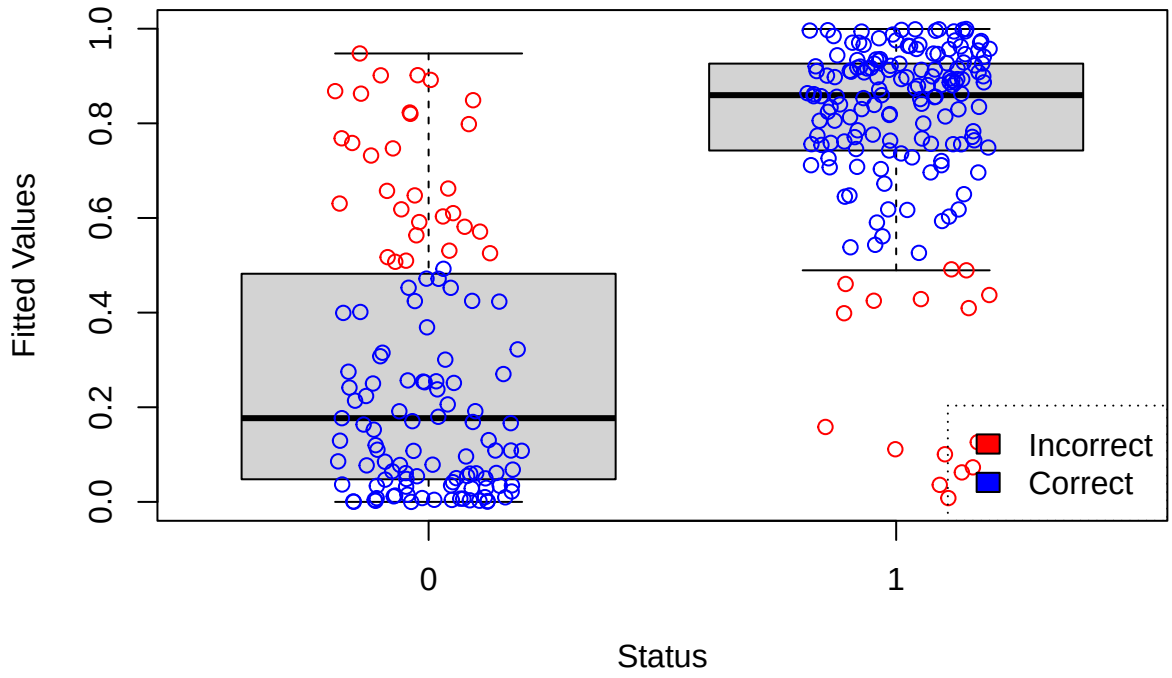
```
## stage3        -3.186e+00  1.567e+00  -2.033 0.042025 *
## stage4        -2.954e+00  1.557e+00  -1.897 0.057803 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 393.11  on 287  degrees of freedom
## Residual deviance: 221.40  on 266  degrees of freedom
## AIC: 265.4
##
## Number of Fisher Scoring iterations: 6
```

```
par(mfrow=c(2,2))
plot(chol.log.fit)
```




```
#changing levels of status to (0,1) where 0 is dead and 1 is alive
chol_updated <- na.omit(cholangitis)
chol_updated <- subset(chol_updated, select = -id)
invisible(droplevels(chol_updated$status))
chol_updated$status <- as.numeric(chol_updated$status == "C")
correct <- chol_updated$status == (chol.log.fit$fitted.values >= 0.5)
boxplot(fitted(chol.log.fit) ~ chol_updated$status, at = c(0, 1),
        outline = F, ylab = "Fitted Values", xlab = "Status")
points(x = jitter(chol_updated$status), fitted(chol.log.fit),
```

```
       col = c("red", "blue")[factor(correct)])
legend("bottomright", legend = c("Incorrect","Correct"), fill = c("red", "blue"), box.lty = 3)
```



The boxplot visualization shows us that our model is good at predicting at the threshold of 0.5. We changed the factor variable `Status` to a numeric variable for ease of plotting. Our inference remains the same.

Although the Residual Deviance (RD) measures "goodness" of fit, it cannot be used for variable selection because the full model will have the smallest RD. So we use the AIC as a criterion instead (smaller AIC reflects a better model).

```
anova(chol.int.fit, chol.log.fit, test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: status ~ 1
## Model 2: status ~ n_days + drug + age + sex + ascites + hepatomegaly +
##     spiders + edema + bilirubin + cholesterol + albumin + copper +
##     alk_phos + sgot + tryglicerides + platelets + prothrombin +
##     stage
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       287     393.11
## 2       266     221.40 21    171.7 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC(chol.int.fit)
```

```
## [1] 395.1059
```

```
AIC(chol.log.fit)
```

```
## [1] 265.4005
```

We improved our model as the Residual Deviance decreased, as did AIC. So, our model is better than just using the intercept.

```
summary(step(chol.log.fit, direction = "both", trace = F))
```

```
##
## Call:
## glm(formula = status ~ n_days + age + sex + ascites + alk_phos +
##     sgot + prothrombin + stage + bilirubin, family = "binomial",
##     data = chol_updated)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3272  -0.4947   0.2939   0.6228   2.9560
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.526e+01  3.380e+00   4.514 6.36e-06 ***
## n_days       7.614e-04  1.894e-04   4.020 5.83e-05 ***
## age         -3.767e-02  1.740e-02  -2.166 0.030344 *
## sexM        -1.186e+00  5.435e-01  -2.183 0.029019 *
## ascitesY    -1.467e+00  1.105e+00  -1.328 0.184146
## alk_phos    -3.304e-04  8.582e-05  -3.850 0.000118 ***
## sgot        -6.197e-03  3.094e-03  -2.003 0.045181 *
## prothrombin -8.761e-01  2.186e-01  -4.008 6.12e-05 ***
## stage2      -2.962e+00  1.517e+00  -1.952 0.050894 .
## stage3      -3.442e+00  1.503e+00  -2.290 0.022036 *
## stage4      -3.211e+00  1.458e+00  -2.203 0.027607 *
## bilirubin   -1.462e-01  7.556e-02  -1.935 0.052965 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 393.11  on 287  degrees of freedom
## Residual deviance: 227.43  on 276  degrees of freedom
## AIC: 251.43
##
## Number of Fisher Scoring iterations: 6
```

```
step(chol.int.fit, scope = formula(chol.log.fit), direction = "forward", trace = F)
```

```
##
## Call:  glm(formula = status ~ bilirubin + prothrombin + n_days + alk_phos +
##     age + sgot + sex + stage + ascites, family = "binomial",
##     data = chol_updated)
##
## Coefficients:
## (Intercept)    bilirubin  prothrombin       n_days     alk_phos          age
##  15.2584796   -0.1462272   -0.8761077    0.0007614   -0.0003304   -0.0376747
##        sgot         sexM       stage2       stage3       stage4     ascitesY
##  -0.0061970   -1.1864857   -2.9618456   -3.4424207   -3.2113840   -1.4670833
##
## Degrees of Freedom: 287 Total (i.e. Null);  276 Residual
## Null Deviance:       393.1
## Residual Deviance: 227.4     AIC: 251.4
```

Running the `step` function models (forward and both) shows us the discrepancies in using this function to compute the optimal model. Even though the values for AIC and Residual Deviance are almost equal, and the size of the models is equal, the variables are in different order. Running regsubsets with categorical variables is too complicated, so we just stick with our current results.

Step ("both") gives us `n_days` + `age` + `sex` + `ascites` + `alk_phos` + `sgot` + `prothrombin` + `stage` + `bilirubin`.

On the other hand, `bilirubin` + `prothrombin` + `n_days` + `alk_phos` + `age` + `sgot` + `sex` + `stage` + `ascites`.

Conclusion

Without domain knowledge, it is difficult to make inference on the variables or make predictions. But, overall, we see a glimpse of the important numerical variables and the trends in the data. Our application of various methods helps us understand the data and fulfills the job of a statistician (i.e. to do statistical analysis).