Project Report - Predicting Liver Disease

Bhathiya Maneendra Pilanawithana

2023-04-07

1. Introduction

The project is aimed at developing a predictive algorithm for liver disease using patient records from India, which are publicly available on Kaggle.com through the link "https://www.kaggle.com/datasets/uciml/indian-liver-patient-records"".

Liver disease is a significant health problem worldwide, and its early detection is crucial because it allows for prompt treatment, which can prevent the progression of the disease and potentially save lives. The liver is a vital organ that performs numerous functions in the body, such as filtering toxins from the blood, producing bile, and metabolizing drugs. Liver disease can develop over time and may not present any symptoms until it has progressed to an advanced stage. Early detection can help identify the disease before symptoms become severe and irreversible damage has occurred. Furthermore, some types of liver disease, such as viral hepatitis, can be highly contagious, making early detection even more important in preventing the spread of the disease to others. Additionally, early detection of liver disease can enable healthcare professionals to closely monitor and manage the condition, which can help reduce the risk of complications and improve overall health outcomes. This may involve lifestyle changes, medication, and in some cases, surgery or liver transplantation.

In this project, various machine learning will be applied and fine tuned to predict the likelihood of liver disease in patients.

2. Dataset Description

This is extracted from Kaggle.The data set contains 416 liver patient records and 167 non liver patient records collected from North East of Andhra Pradesh, India. Any patient whose age exceeded 89 is listed as being of age "90".

Columns:

1. Age of the patient
2. Gender of the patient
3. Total Bilirubin
4. Direct Bilirubin
5. Alkaline Phosphotase
6. Alamine Aminotransferase
7. Aspartate Aminotransferase
8. Total Protiens
9. Albumin
10. Albumin and Globulin Ratio
11. Dataset: field used to split the data into two sets (patient with liver disease, or no disease)

3. Data Wrangling

First, the csv file stored in "Raw-Dataset" subfolder is loaded in to a dataframe using following code.

```
dat <- read.csv("./Raw-Dataset/indian_liver_patient.csv")
```

It was observed that presence of liver disease is stated in the "Dataset" column of the Dataframe and the existence of liver disease is denoted by 1 and non-existence is denoted by 2. Since this notation is counter intuitive, "Dataset" Column is renamed to "Disease" and existence and non-existence of liver disease is denoted by 1 and 0 respectively. Apart from this, the "Gender" column of the Dataframe is a character vector with on "Male" and "Female" entries. This should also be converted to a factor vector. Both of these can be achieved by the following code chunk.

```
dat <- dat %>% mutate(Disease = ifelse(Dataset==1,1,0), Gender=as.factor(Gender)) %>%
  select(-Dataset)
```

Now, the existence of liver disease is noted in "Disease" column intuitively using 1 and 0.

By running the following code it was observed that there are missing values in the Dataframe and they are all in the "Albumin_and_Globulin_Ratio" column.

```
sum(is.na(dat %>% select(-Gender)))
```

```
## [1] 4
```

```
colSums(is.na(dat %>% select(-Gender)))
```

```
##                          Age             Total_Bilirubin
##                            0                           0
##            Direct_Bilirubin       Alkaline_Phosphotase
##                            0                           0
##   Alamine_Aminotransferase Aspartate_Aminotransferase
##                            0                           0
##              Total_Protiens                     Albumin
##                            0                           0
## Albumin_and_Globulin_Ratio                     Disease
##                            4                           0
```

Since there are 583 total observations, omission of the 4 rows with missing values was considered to have a negligible effect. The omission was carried out by the following code.

```
dat <- na.omit(dat)
```

The Dataframe is now considered ready for further statistical analysis to find out which variable to use for prediction of liver disease.

4. Analysis

First, we will analyse the gender influence in liver disease. The portion of people with liver disease for male and female genders can be extracted from the following code.

```
mean((dat %>% filter(Gender=="Male"))$Disease==1)
```

```
## [1] 0.7357631
```

```
mean((dat %>% filter(Gender=="Female"))$Disease==1)
```

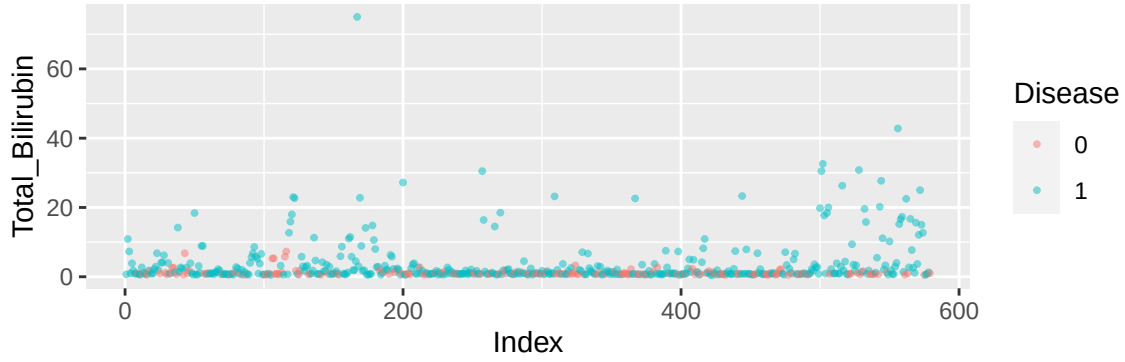
```
## [1] 0.65
```

It was observed, that according to the sample in the dataset both genders have more than 50% chance of having liver disease, which might not be case for the populations. However, gender male has higher likelyhood for liver disease, which suggests that the column gender may play an important part in liver disease prediction.

4.1 Checking for Outliers and Visible Patterns of Indivugual Numerical Variables

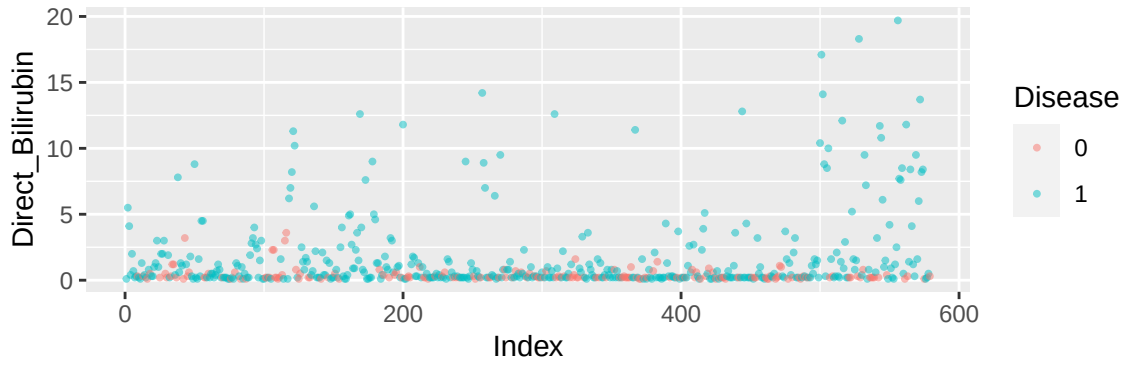
Each variable is plotted individually to identify any outliers.

```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Total_Bilirubin)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5, size = 0.7) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```

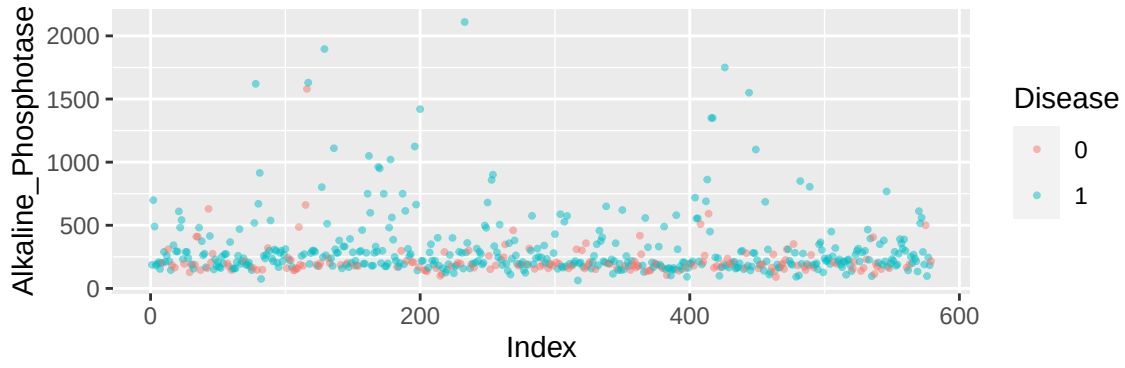


Although the plot of "Total Bilirubin" appears to have one element with very high value, it is not extreme enough to be considered an outlier.

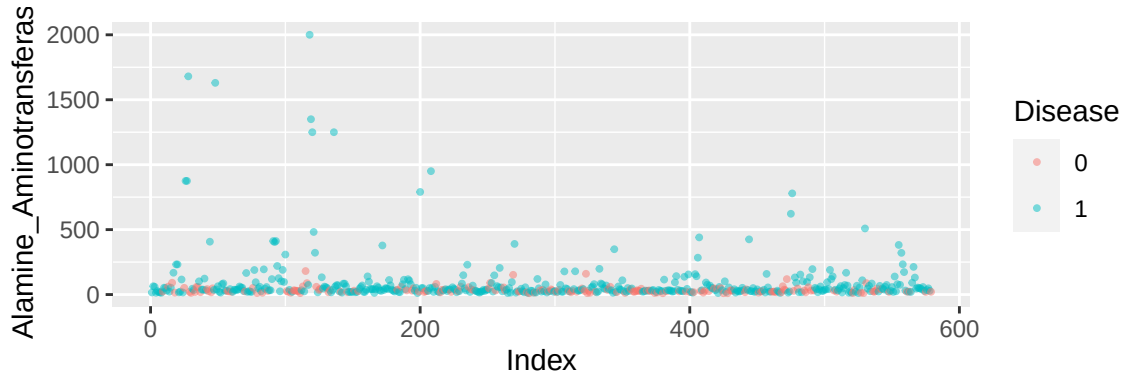
```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Direct_Bilirubin)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5, size = 0.7) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```



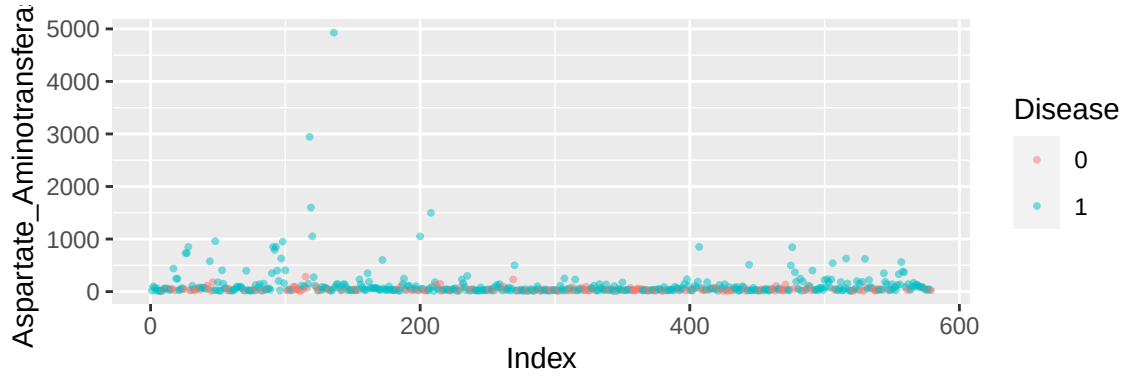
```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Alkaline_Phosphotase)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5, size = 0.7) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```



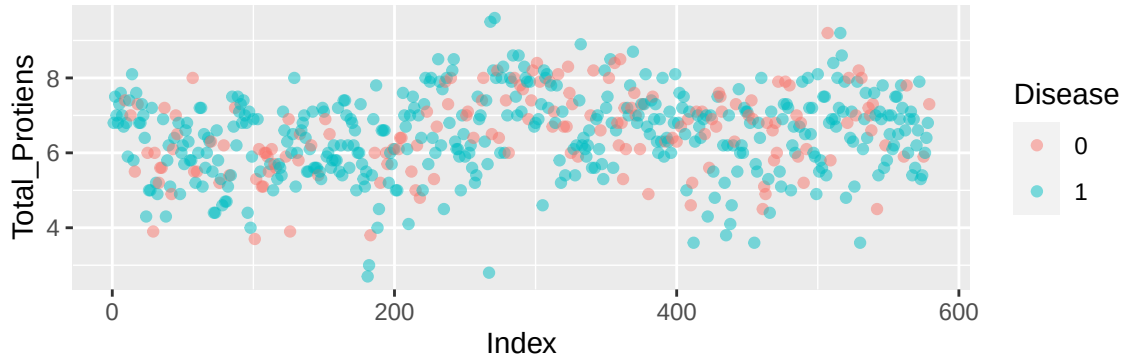

```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Alamine_Aminotransferase)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5, size = 0.7) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```



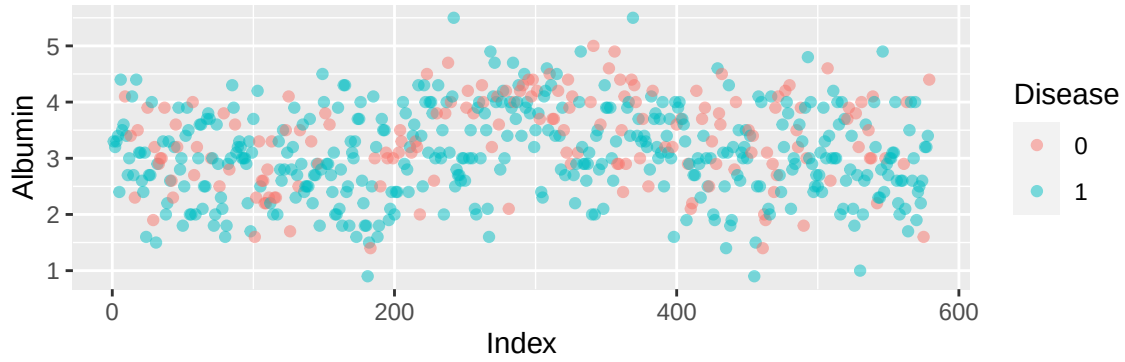

```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Aspartate_Aminotransferase)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5, size = 0.7) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```



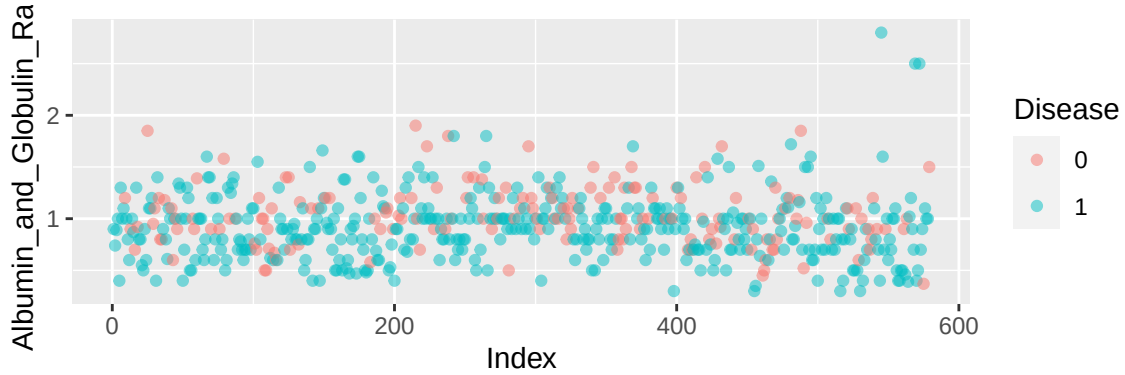
```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Total_Protiens)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```




```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Albumin)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```




```
dat %>% ggplot(aes(x = seq(1,nrow(dat),1), y = Albumin_and_Globulin_Ratio)) +
  geom_point(aes(col=as.factor(Disease)), alpha = 0.5) +
  guides(color = guide_legend(title = "Disease")) +
  xlab("Index")
```



Individual plots do not give away clues of visible patterns of individual variables. However, from the above plots it can be observed that following variables have large range, but have more points close to zero than

away from zero.

```
Range_Total_Bilirubin <- range(dat$Total_Bilirubin)
Range_Total_Bilirubin
```

```
## [1]  0.4 75.0
```

```
Range_Direct_Bilirubin <- range(dat$Direct_Bilirubin)
Range_Direct_Bilirubin
```

```
## [1]  0.1 19.7
```

```
Range_Alkaline_Phosphotase <- range(dat$Alkaline_Phosphotase)
Range_Alkaline_Phosphotase
```

```
## [1]   63 2110
```

```
Range_Alamine_Aminotransferase <- range(dat$Alamine_Aminotransferase)
Range_Alamine_Aminotransferase
```

```
## [1]   10 2000
```

```
Range_Aspartate_Aminotransferase <- range(dat$Aspartate_Aminotransferase)
Range_Aspartate_Aminotransferase
```

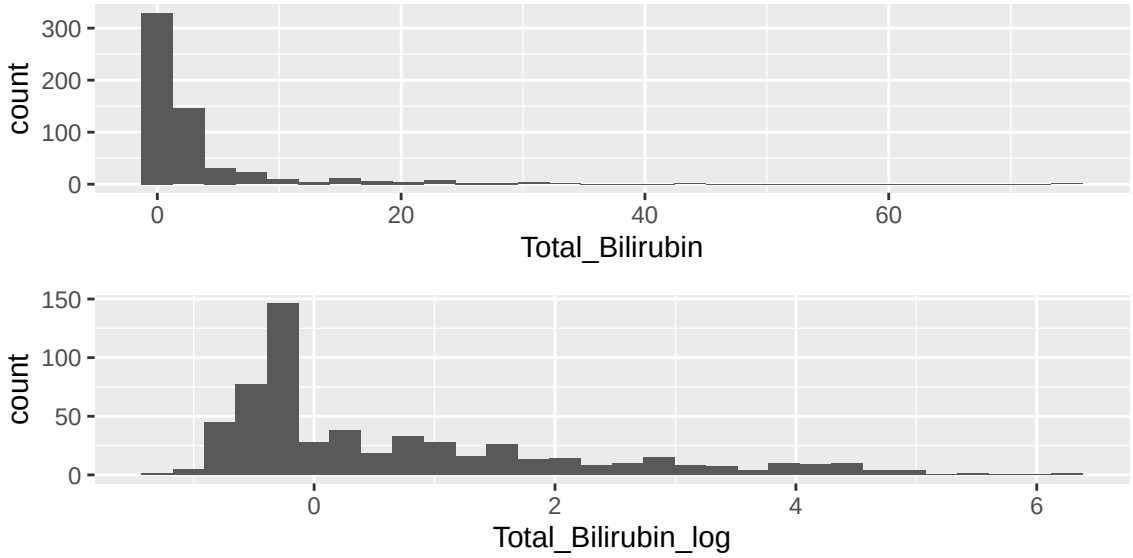
```
## [1]   10 4929
```

This suggests that log transformations applied to each of the following variables may improve the performance of a prediction algorithm. Log transformations are applied using the following code. The new Dataframe with log transformations is named data_log.

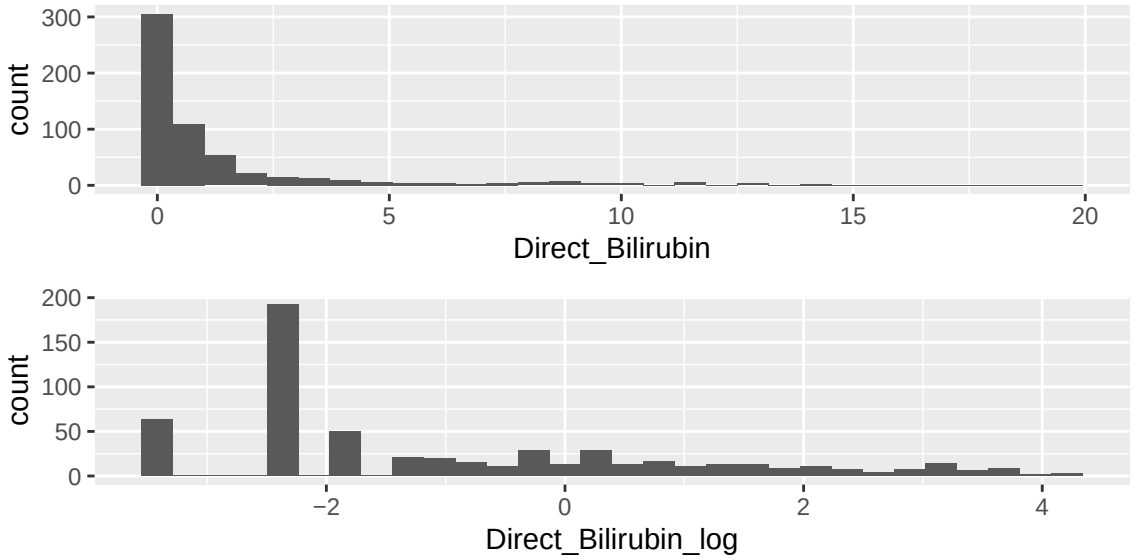
```
dat_log <- dat %>%
  mutate(Total_Bilirubin_log = log2(Total_Bilirubin),
         Direct_Bilirubin_log = log2(Direct_Bilirubin),
         Alkaline_Phosphotase_log = log10(Alkaline_Phosphotase),
         Alamine_Aminotransferase_log = log10(Alamine_Aminotransferase),
         Aspartate_Aminotransferase_log = log10(Aspartate_Aminotransferase)) %>%
  select(-Total_Bilirubin, -Direct_Bilirubin, -Alkaline_Phosphotase,
         -Alamine_Aminotransferase, -Aspartate_Aminotransferase)
```

The following histograms show a comparison of histograms for each high range variable before applying log transformation and after, respectively.

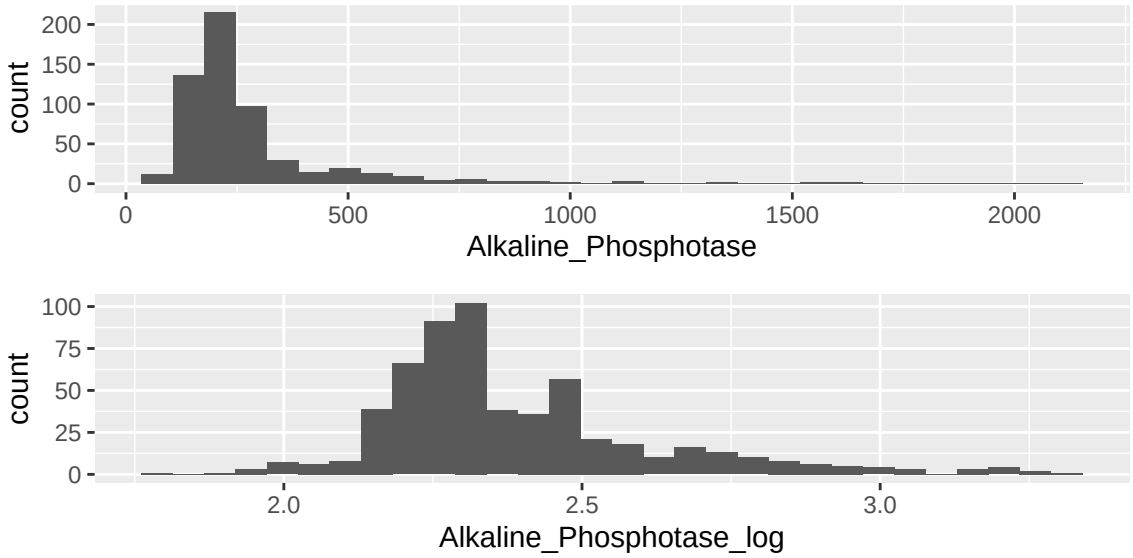
Total Bilirubin:



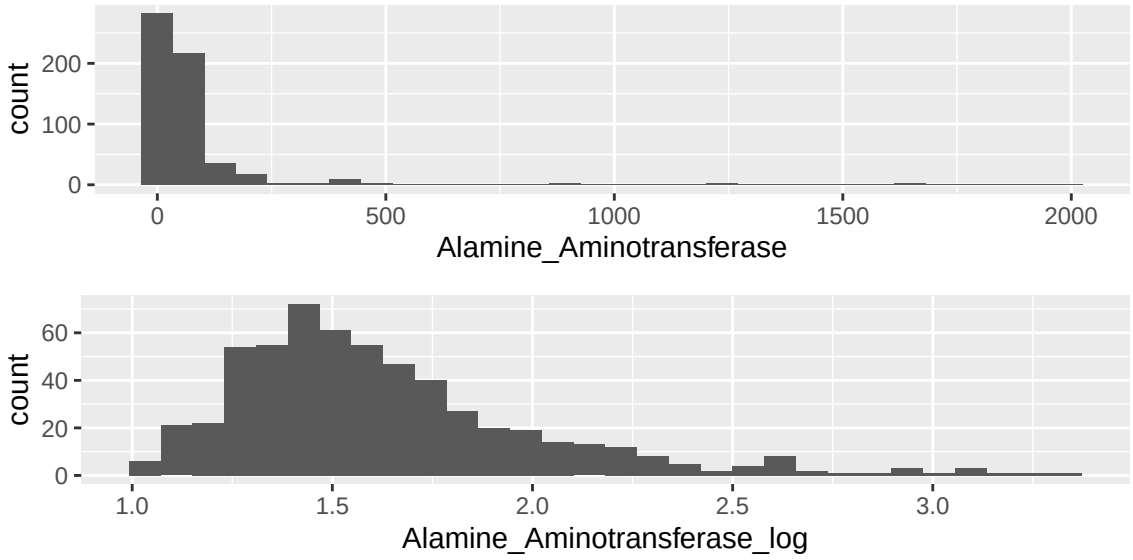
Direct Bilirubin:



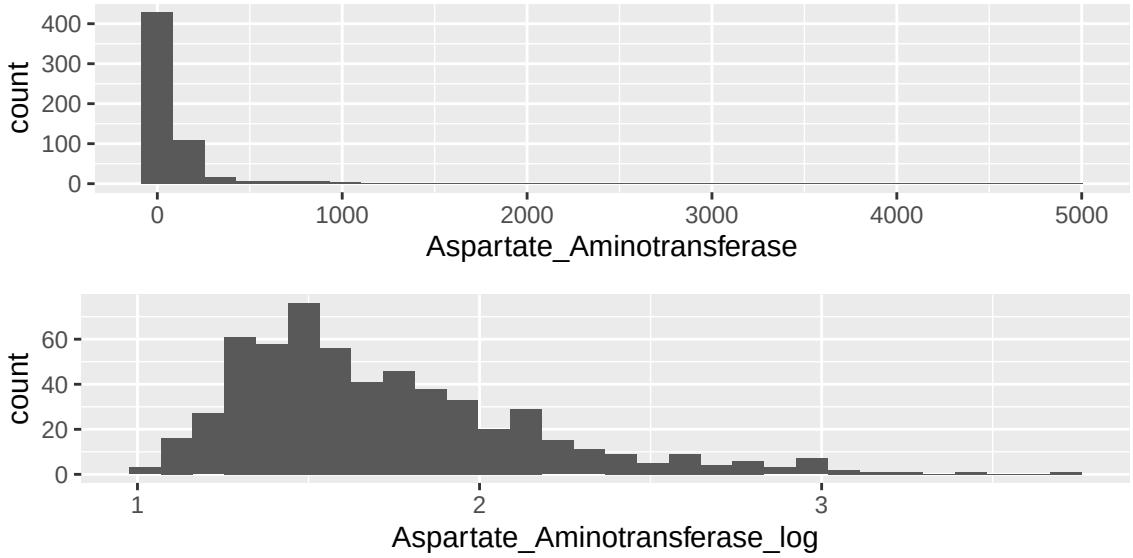
Alkaline Phosphotase:



Alamine Aminotransferase:



Aspartate Aminotransferase:



4.2 Correlation Between Numerical Variables

The correlation matrix for the numerical variables can be obtained using cor() function. However, the variable names are too long an effective presentation. Therefore, a temporary Dataframe called dat_for_corr is created with name abbreviations.

```
dat_log_for_corr <- dat_log %>% select(-Age,-Gender,-Disease)
colnames(dat_log_for_corr) <-
  c("Tot Pr","Alb", "Alb Glb", "Tot Bil", "Dir Bil", "ALP", "ALA", "ASA")
cr <- round(cor(dat_log_for_corr),4)
cr
```

```
##          Tot Pr     Alb Alb Glb Tot Bil Dir Bil     ALP     ALA     ASA
## Tot Pr   1.0000  0.7831  0.2349 -0.0514 -0.0354  0.0021 -0.0133 -0.0638
## Alb      0.7831  1.0000  0.6896 -0.2901 -0.2703 -0.1679 -0.0415 -0.1780
## Alb Glb  0.2349  0.6896  1.0000 -0.2706 -0.2631 -0.2822 -0.0609 -0.1448
## Tot Bil -0.0514 -0.2901 -0.2706  1.0000  0.9657  0.3585  0.4424  0.5406
## Dir Bil -0.0354 -0.2703 -0.2631  0.9657  1.0000  0.3568  0.4297  0.5300
## ALP      0.0021 -0.1679 -0.2822  0.3585  0.3568  1.0000  0.3396  0.3195
## ALA     -0.0133 -0.0415 -0.0609  0.4424  0.4297  0.3396  1.0000  0.8416
## ASA     -0.0638 -0.1780 -0.1448  0.5406  0.5300  0.3195  0.8416  1.0000
```

It can be observed from the correlation matrix that following pairs of variables have correlations close to 1.

1. Total_Bilirubin - Direct_Bilirubin = 0.9657
2. Alamine Aminotransferase - Aspartate Aminotransferase = 0.8416
3. Total Protiens - Albumin = 0.7831
4. Albumin - Albumin and Globulin Ratio = 0.6896

This means that it may be possible to remove one variable from each pair without sacrificing much performance from the prediction algorithm.

4.3 Applying Machine Learning

We intends to apply kNN, Decision-Tree, Random-Forest and fine tune their parameters using cross-validation. We will compare the performance of each method and select the best on for prediction. For this purpose we

first partition the Dataset in to training-set 60% and test-set 40% using the following code.

```
set.seed(1)
test_index <- createDataPartition(dat_log$Disease, times=1, p=0.4, list=FALSE)
test_set <- dat_log[test_index,]
train_set <- dat_log[-test_index,]
```