Liver Disease Capstone

Nishaal Ajmera

05/06/2020

Introduction

Over the years, patients with liver disease have been on the rise. Factors contributing to this include increased alcohol consumption, drugs consumption, unhealthy and fatty foods and inhalation of toxins. This dataset was obtained patients records in the North East of Andhra Pradesh, India. The original dataset has 416 liver disease patients and 167 non-liver disease patients. The data has been modified to remove NA's and it contains 414 liver disease patients and 165 non-liver disease patients. The dataset contains 11 variables (Age,Gender,Total Bilirubin, Direct Bilirubin,Alkaline Phosphotase,Alamine Aminotransferase, Aspartate Aminotransferase, Total Proteins, Albumin, Albumin to Globulin Ratio). The final "Dataset" variable is used to distinguish between liver disease patients (1) and non-liver disease patients (2).
The key goals of this project are:

- to help doctors diagnose patients with liver disease
- to investigate best machine learning model to predict patients with liver disease accurately

Obtaining Data

This section shows the required packages and how the data can be obtained

```
#Packages
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require(gridExtra)) install.packages("gridExtra",repos="https://cran.rstudio.com/bin/macosx/el-capit
```

```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

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

```
if(!require(rpart)) install.packages("rpart",repos= "http://cran.us.r-project.org")
```

```
## Loading required package: rpart
```

```
#Getting the dataset
liverurl="https://raw.githubusercontent.com/nishaalajmera/Indian-Liver-Disease-Capstone-/master/indian_l
liverdis<-read_csv(url(liverurl)) #reading and saving file from url into workable format
```

```
## Parsed with column specification:
## cols(
##   Age = col_double(),
##   Gender = col_character(),
##   Total_Bilirubin = col_double(),
##   Direct_Bilirubin = col_double(),
##   Alkaline_Phosphotase = col_double(),
##   Alamine_Aminotransferase = col_double(),
##   Aspartate_Aminotransferase = col_double(),
##   Total_Protiens = col_double(),
##   Albumin = col_double(),
##   Albumin_and_Globulin_Ratio = col_double(),
##   Dataset = col_double()
## )
```

Data Modification

The Dataset is modified for smooth analysis

```r
head(liverdis) #Looking at first few rows of the dataset
```

```
## # A tibble: 6 x 11
##     Age Gender Total_Bilirubin Direct_Bilirubin Alkaline_Phosph~
##   <dbl> <chr>            <dbl>            <dbl>            <dbl>
## 1    65 Female             0.7              0.1              187
## 2    62 Male              10.9              5.5              699
## 3    62 Male               7.3              4.1              490
## 4    58 Male               1                0.4              182
## 5    72 Male               3.9              2                195
## 6    46 Male               1.8              0.7              208
## # ... with 6 more variables: Alamine_Aminotransferase <dbl>,
## #   Aspartate_Aminotransferase <dbl>, Total_Protiens <dbl>, Albumin <dbl>,
## #   Albumin_and_Globulin_Ratio <dbl>, Dataset <dbl>
```

```r
dim(liverdis) #checking dimensions of the dataset
```

```
## [1] 583  11
```

```r
liverdis<- na.omit(liverdis) #remove any rows with missing data
dim(liverdis) #now there is a total of 579 patients
```

```
## [1] 579  11
```

```r
liver_patients<- liverdis %>% filter(Dataset=="1") %>% count() #count liver patients
non_liver_patients<- liverdis %>% filter(Dataset=="2") %>% count() #count non-liver patients
liver_patients
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   414
```

```r
non_liver_patients
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   165
```

```r
#Convert Dataset to 1 and 2 to factors for easier analysis
liverdis$Dataset<- as.factor(liverdis$Dataset)
class(liverdis$Dataset) #checking the class of Dataset column
```

```
## [1] "factor"
```

```r
#Analysing basic summary statistics of the dataset
summary(liverdis)
```

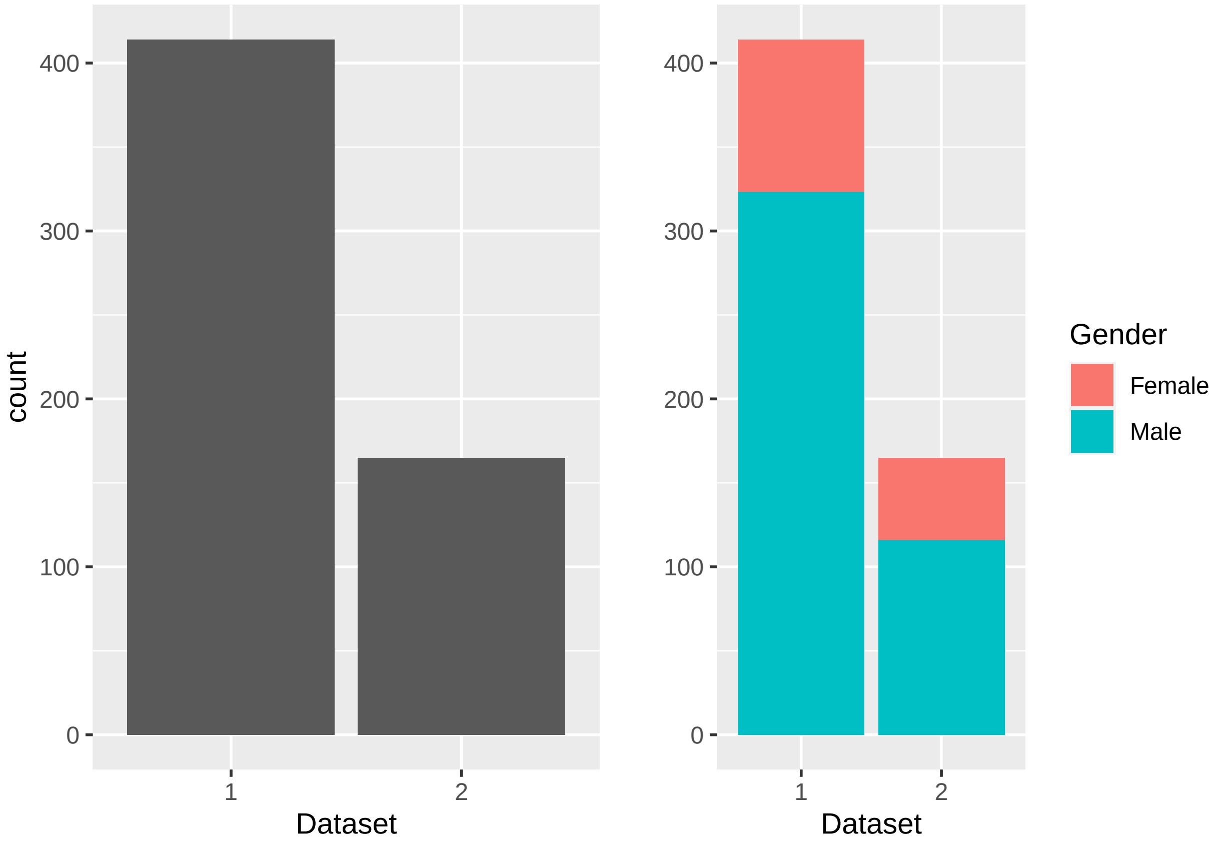
```
##       Age           Gender          Total_Bilirubin  Direct_Bilirubin
##  Min.   : 4.00   Length:579         Min.   : 0.400   Min.   : 0.100
##  1st Qu.:33.00   Class :character   1st Qu.: 0.800   1st Qu.: 0.200
##  Median :45.00   Mode  :character   Median : 1.000   Median : 0.300
##  Mean   :44.78                      Mean   : 3.315   Mean   : 1.494
##  3rd Qu.:58.00                      3rd Qu.: 2.600   3rd Qu.: 1.300
##  Max.   :90.00                      Max.   :75.000   Max.   :19.700
##  Alkaline_Phosphotase Alamine_Aminotransferase Aspartate_Aminotransferase
##  Min.   :  63.0       Min.   :  10.00          Min.   :  10.0
##  1st Qu.: 175.5       1st Qu.:  23.00          1st Qu.:  25.0
##  Median : 208.0       Median :  35.00          Median :  42.0
##  Mean   : 291.4       Mean   :  81.13          Mean   : 110.4
##  3rd Qu.: 298.0       3rd Qu.:  61.00          3rd Qu.:  87.0
##  Max.   :2110.0       Max.   :2000.00          Max.   :4929.0
##  Total_Protiens     Albumin      Albumin_and_Globulin_Ratio Dataset
##  Min.   :2.700   Min.   :0.900   Min.   :0.3000            1:414
##  1st Qu.:5.800   1st Qu.:2.600   1st Qu.:0.7000            2:165
##  Median :6.600   Median :3.100   Median :0.9300
##  Mean   :6.482   Mean   :3.139   Mean   :0.9471
##  3rd Qu.:7.200   3rd Qu.:3.800   3rd Qu.:1.1000
##  Max.   :9.600   Max.   :5.500   Max.   :2.8000
```

Exploratory Analysis

Liver patients and non-liver patients segregated by Gender

```
#Number of people with liver disease and no liver disease
b1<- liverdis %>% ggplot(aes(Dataset)) + geom_bar()
#Number of people with liver disease and no liver diseases according to gender
b2<- qplot(Dataset,data=liverdis,fill=Gender)

grid.arrange(b1,b2,ncol=2)
```

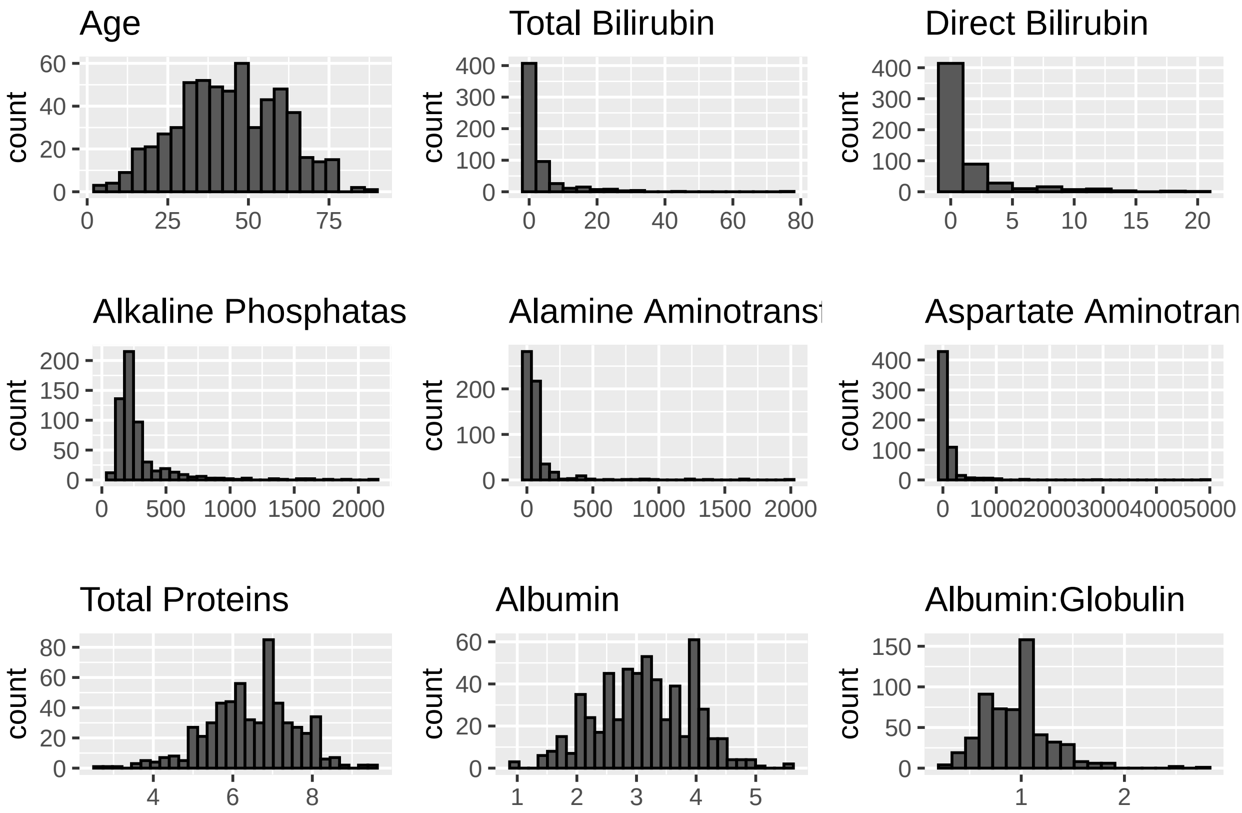


It is observed that in both groups there are less female patients compared to male patients

Distribution

The distribution of each continuous variable is shown below.

```
#Distribution
d1<- liverdis %>% ggplot(aes(x=Age))+ geom_histogram(binwidth=4, color="black") + ggtitle("Age")+xlab("
d2<- liverdis %>% ggplot(aes(x=Total_Bilirubin)) + geom_histogram(binwidth=4, colour="black") + ggtitle
d3<- liverdis %>% ggplot(aes(x=Direct_Bilirubin)) + geom_histogram(binwidth=2, colour="black") + ggtitl
d4<- liverdis %>% ggplot(aes(x=Alkaline_Phosphotase)) + geom_histogram(bins=30, colour="black") + ggtit
d5<- liverdis %>% ggplot(aes(x=Alamine_Aminotransferase)) + geom_histogram(bins=30, colour="black") + gg
d6<- liverdis %>% ggplot(aes(x=Aspartate_Aminotransferase)) + geom_histogram(bins=30, colour="black") +
d7<- liverdis %>% ggplot(aes(x=Total_Protiens)) + geom_histogram(bins=30, colour="black") + ggtitle("To
d8<-liverdis %>% ggplot(aes(x=Albumin)) + geom_histogram(bins=30, colour="black") + ggtitle("Albumin")+
d9<- liverdis %>% ggplot(aes(x=Albumin_and_Globulin_Ratio)) + geom_histogram(bins=20, colour="black") +
grid.arrange(d1,d2,d3,d4,d5,d6,d7,d8,d9)
```



Some variables such as: Total Bilirubin, Direct Bilirubin, Alkaline Phophatase, Alamine Aminotransferase and Aspartate Aminotransferase have skewed distributions. This might be due to some clustering suggesting that the levels could be higher in one of the patient groups.

Normal Distribution Test

Shapiro-Wilk test is used here to check if continuous variables follow a normal distribution.
Null Hypothesis: Continuous variable follows a distribution pattern similar to normal distribution
Alternative Hypothesis: Continuous variable does not follow normal distribution

```
#Checking for normality using Shapiro-Wilk Test for continuous variables
shapiro.test(liverdis$Age)$p.value
```

```
## [1] 0.003336382
```

```
shapiro.test(liverdis$Total_Bilirubin)$p.value
```

```
## [1] 2.207789e-38
```

```
shapiro.test(liverdis$Direct_Bilirubin)$p.value
```

```
## [1] 1.643833e-36
```

```
shapiro.test(liverdis$Alkaline_Phosphotase)$p.value
```

```
## [1] 6.985287e-35
```

```
shapiro.test(liverdis$Alamine_Aminotransferase)$p.value
```

```
## [1] 1.906054e-41
```

```
shapiro.test(liverdis$Aspartate_Aminotransferase)$p.value
```

```
## [1] 2.009734e-42
```

```
shapiro.test(liverdis$Total_Protiens)$p.value
```

```
## [1] 0.002876621
```

```
shapiro.test(liverdis$Albumin)$p.value
```

```
## [1] 0.005338014
```

```
shapiro.test(liverdis$Albumin_and_Globulin_Ratio)$p.value
```

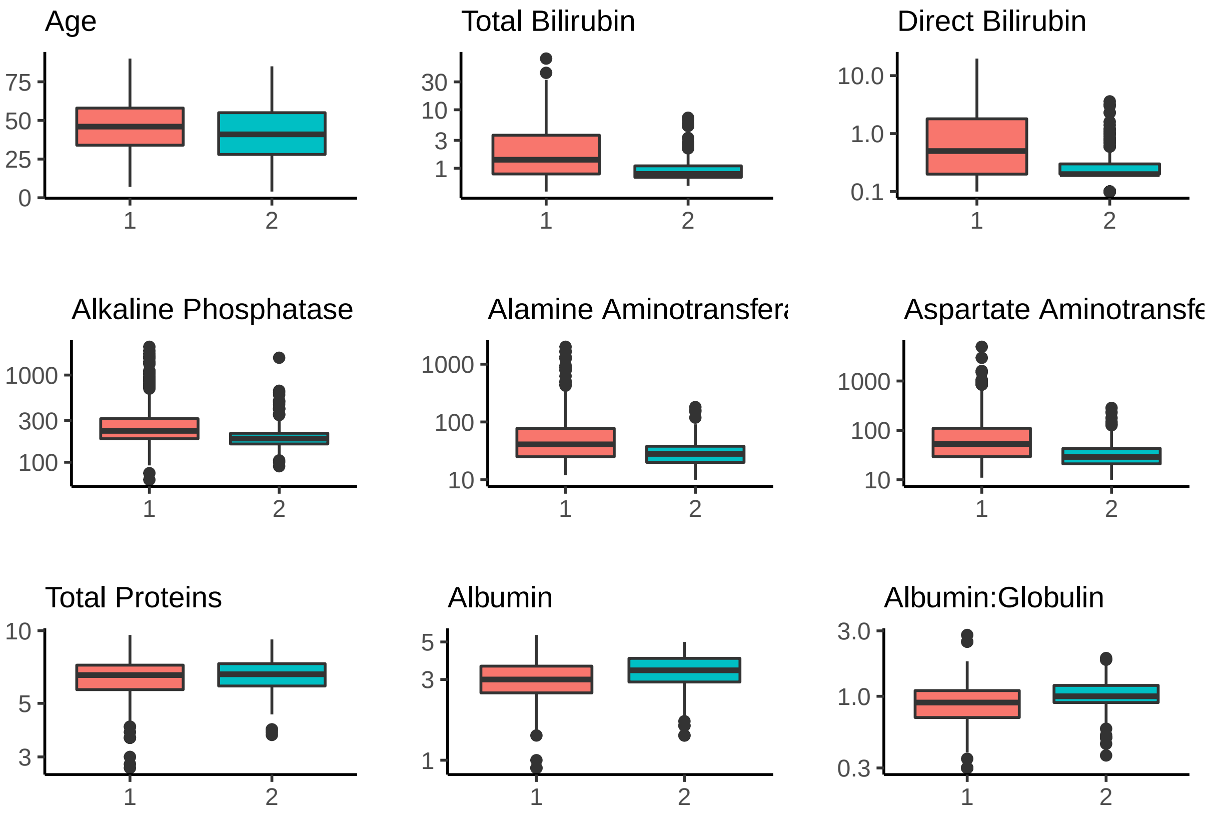
```
## [1] 1.30531e-13
```

P-values for all variables are less than 0.05 therefore the null hypothesis is rejected. Going forward, non-parametric tests will be used to assess the statistical signficance of the the data. Therefore median will be used as measure of central tendency and interquartile range will explain the variability of the data.

Data Analysis between the two groups of data; liver disease patients and non-liver disease patients

1 represents patients with liver disease and 2 represents patients with no liver disease. Below are boxplots to visualize any obvious differences. Log scale has been used to better visualize data.

```
b1<- liverdis %>% ggplot(aes(Dataset, Age)) + geom_boxplot(aes(fill = Dataset))+theme_classic()+theme(l
b2<- liverdis %>% ggplot(aes(Dataset, Total_Bilirubin)) + geom_boxplot(aes(fill = Dataset))+theme_class
b3<- liverdis %>% ggplot(aes(Dataset, Direct_Bilirubin)) + geom_boxplot(aes(fill = Dataset))+theme_clas
b4<- liverdis %>% ggplot(aes(Dataset,Alkaline_Phosphotase)) + geom_boxplot(aes(fill = Dataset))+theme_cl
b5<- liverdis %>% ggplot(aes(Dataset, Alamine_Aminotransferase)) + geom_boxplot(aes(fill = Dataset))+th
b6<- liverdis %>% ggplot(aes(Dataset, Aspartate_Aminotransferase)) + geom_boxplot(aes(fill = Dataset))+
b7<- liverdis %>% ggplot(aes(Dataset, Total_Protiens)) + geom_boxplot(aes(fill = Dataset))+theme_classi
b8<- liverdis %>% ggplot(aes(Dataset, Albumin)) + geom_boxplot(aes(fill = Dataset))+theme_classic()+the
b9<- liverdis %>% ggplot(aes(Dataset, Albumin_and_Globulin_Ratio)) + geom_boxplot(aes(fill = Dataset))+
grid.arrange(b1,b2,b3,b4,b5,b6,b7,b8,b9)
```



The median of some variables show differences in the two groups. However this has to be further assesed.

Wilcoxon Signed Ranked Test

Wilcoxon Signed Ranked Test is a non-parametric test is used to compare two related samples.

```
#Displaying non-parametric measures and carrying out Wilcoxon Signed Rank Test to test for any signific
#Age
Age<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Age),IQR=IQR(Age))
Age_pvalue<- wilcox.test(liverdis$Age~liverdis$Dataset)$p.value

#Total Bilirubin
Total_Bilirubin<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Total_Bilirubin)
Total_Bilirubin_pvalue<- wilcox.test(liverdis$Total_Bilirubin~liverdis$Dataset)$p.value

#Direct Bilirubin
Direct_Bilirubin<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Direct_Bilirubi
Direct_Bilirubin_pvalue<- wilcox.test(liverdis$Direct_Bilirubin~liverdis$Dataset)$p.value

#Alkaline Phosphatase
Alkaline_Phosphatase<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Alkaline_Pho
Alkaline_Phosphatase_pvalue<- wilcox.test(liverdis$Alkaline_Phosphotase~liverdis$Dataset)$p.value

#Alamine Aminotransferase
Alamine_Aminotransferase <- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Alamine
Alamine_Aminotransferase_pvalue <- wilcox.test(liverdis$Alamine_Aminotransferase~liverdis$Dataset)$p.val

#Aspartate Aminotransferase
```

```
Aspartate_Aminotransferase<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Aspar
Aspartate_Aminotransferase_pvalue<- wilcox.test(liverdis$Aspartate_Aminotransferase~liverdis$Dataset)$p

#Total Proteins
Total_Proteins<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Total_Protiens),I
Total_Proteins_pvalue <- wilcox.test(liverdis$Total_Protiens~liverdis$Dataset)$p.value

#Albumin
Albumin<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Albumin),IQR=IQR(Albumin
Albumin_pvalue <- wilcox.test(liverdis$Albumin~liverdis$Dataset)$p.value

#Albumin: Globulin
Albumin_Globulin_ratio<- liverdis %>% group_by(Dataset) %>% summarize(count=n(),median=median(Albumin_an
Albumin_Globulin_ratio_pvalue<- wilcox.test(liverdis$Albumin_and_Globulin_Ratio~liverdis$Dataset)$p.valu

Age
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414     46    24
## 2 2         165     41    27
```

```
Age_pvalue
```

```
## [1] 0.002731931
```

```
Total_Bilirubin
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414    1.4  2.88
## 2 2         165    0.8  0.4
```

```
Total_Bilirubin_pvalue
```

```
## [1] 2.748439e-13
```

```
Direct_Bilirubin
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414    0.5 1.6
## 2 2         165    0.2 0.100
```

```
Direct_Bilirubin_pvalue
```

```
## [1] 6.449568e-13
```

```
Alkaline_Phosphatase
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414    229  130.
## 2 2         165    187   53
```

```
Alkaline_Phosphatase_pvalue
```

```
## [1] 9.936943e-11
```

```
Alamine_Aminotransferase
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414     41  52.5
## 2 2         165     28  18
```

```
Alamine_Aminotransferase_pvalue
```

```
## [1] 3.702919e-12
```

```
Aspartate_Aminotransferase
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414     53    81
## 2 2         165     29    22
```

```
Aspartate_Aminotransferase_pvalue
```

```
## [1] 1.310704e-13
```

```
Total_Proteins
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414   6.55  1.5
## 2 2         165   6.6   1.40
```

```
Total_Proteins_pvalue
```

```
## [1] 0.446179
```

```
Albumin
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414    3     1.1
## 2 2         165    3.4   1.1
```

```
Albumin_pvalue
```

```
## [1] 5.931425e-05
```

```
Albumin_Globulin_ratio
```

```
## # A tibble: 2 x 4
##   Dataset count median   IQR
##   <fct>   <int>  <dbl> <dbl>
## 1 1         414    0.9 0.4
## 2 2         165    1   0.300
```

```
Albumin_Globulin_ratio_pvalue
```

```
## [1] 5.812072e-06
```

```
All p-values except Total Proteins show that there is a significance    between the liver disease and n
```

Generating Training and Test Samples

```
set.seed(1,sample.kind = "Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index<- createDataPartition(y=liverdis$Dataset,times=1,p=0.2,list=FALSE) #index of test set
train_set<- liverdis[-test_index,] #generating train set
test_set<- liverdis[test_index,] #generating test set
```

Model 1: Logistic Regression Model

This model uses logistic regression to predict the patient group. Here all the variables are used as predictors.

```
#Model 1: Logistic Regression Model (all predictors)
fit_glm<- glm(Dataset~.,data=train_set,family="binomial") #Training algorithm
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
p_hat_glm<- predict(fit_glm,test_set,type = "response")
y_hat_glm<- ifelse(p_hat_glm>0.5,"1","2")
confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1  5  9
##          2 78 24
##
##                Accuracy : 0.25
##                  95% CI : (0.1743, 0.3389)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.1304
##
##  Mcnemar's Test P-Value : 3.091e-13
##
##             Sensitivity : 0.06024
##             Specificity : 0.72727
##          Pos Pred Value : 0.35714
##          Neg Pred Value : 0.23529
##              Prevalence : 0.71552
##          Detection Rate : 0.04310
##    Detection Prevalence : 0.12069
##       Balanced Accuracy : 0.39376
##
##        'Positive' Class : 1
##
```

```r
m1 <- confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- tibble(model = "Logistic Regression with all predictors", Accuracy = m1 ) #saving m
overall_accuracy
```

```
## # A tibble: 1 x 2
##   model                                   Accuracy
##   <chr>                                      <dbl>
## 1 Logistic Regression with all predictors     0.25
```

It is seen that this model gives very poor accuracy. We wil try improving the model by removing some variables.

Model 2: Logistic Regression Model

In this model we are only using continuous variables to predict the dataset.

```r
#Model 2: Logistic Regression Model (continuous variables)
fit_glm<- glm(Dataset~Age+ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_Aminotra
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
p_hat_glm<- predict(fit_glm,test_set,type = "response")
y_hat_glm<- ifelse(p_hat_glm>0.5,"1","2")
confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1  5 10
##          2 78 23
##
##                Accuracy : 0.2414
##                  95% CI : (0.1668, 0.3296)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.1498
##
##  Mcnemar’s Test P-Value : 9.183e-13
##
##             Sensitivity : 0.06024
##             Specificity : 0.69697
##          Pos Pred Value : 0.33333
##          Neg Pred Value : 0.22772
##              Prevalence : 0.71552
##          Detection Rate : 0.04310
##    Detection Prevalence : 0.12931
##       Balanced Accuracy : 0.37861
##
##        ’Positive’ Class : 1
##
```

```
m2<- confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- bind_rows(overall_accuracy, tibble(model = "Logistic Regression with 9 predictors",
overall_accuracy
```

```
## # A tibble: 2 x 2
##   model                                   Accuracy
##   <chr>                                      <dbl>
## 1 Logistic Regression with all predictors    0.25
## 2 Logistic Regression with 9 predictors      0.241
```

This model has reduced the accuracy.

Model 3: Logistic Regression

In this model, we will use the variables that have a skewed distribution. It is proposed that some the levels of some variables might be higher in a one group of patient.

```
#Model 3: Logistic Regression using variables that have a skewed distribution
fit_glm<- glm(Dataset~ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_Aminotransfe
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
p_hat_glm<- predict(fit_glm,test_set,type = "response")
y_hat_glm<- ifelse(p_hat_glm>0.5,"1","2")
confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1  1  1
##          2 82 32
##
##                Accuracy : 0.2845
##                  95% CI : (0.2046, 0.3757)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0105
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.012048
##             Specificity : 0.969697
##          Pos Pred Value : 0.500000
##          Neg Pred Value : 0.280702
##              Prevalence : 0.715517
##          Detection Rate : 0.008621
##    Detection Prevalence : 0.017241
##       Balanced Accuracy : 0.490873
##
##        'Positive' Class : 1
##
```

```
m3<- confusionMatrix(relevel(as.factor(y_hat_glm),"1"),test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- bind_rows(overall_accuracy, tibble(model = "Logistic Regression with 5 predictors",
overall_accuracy
```

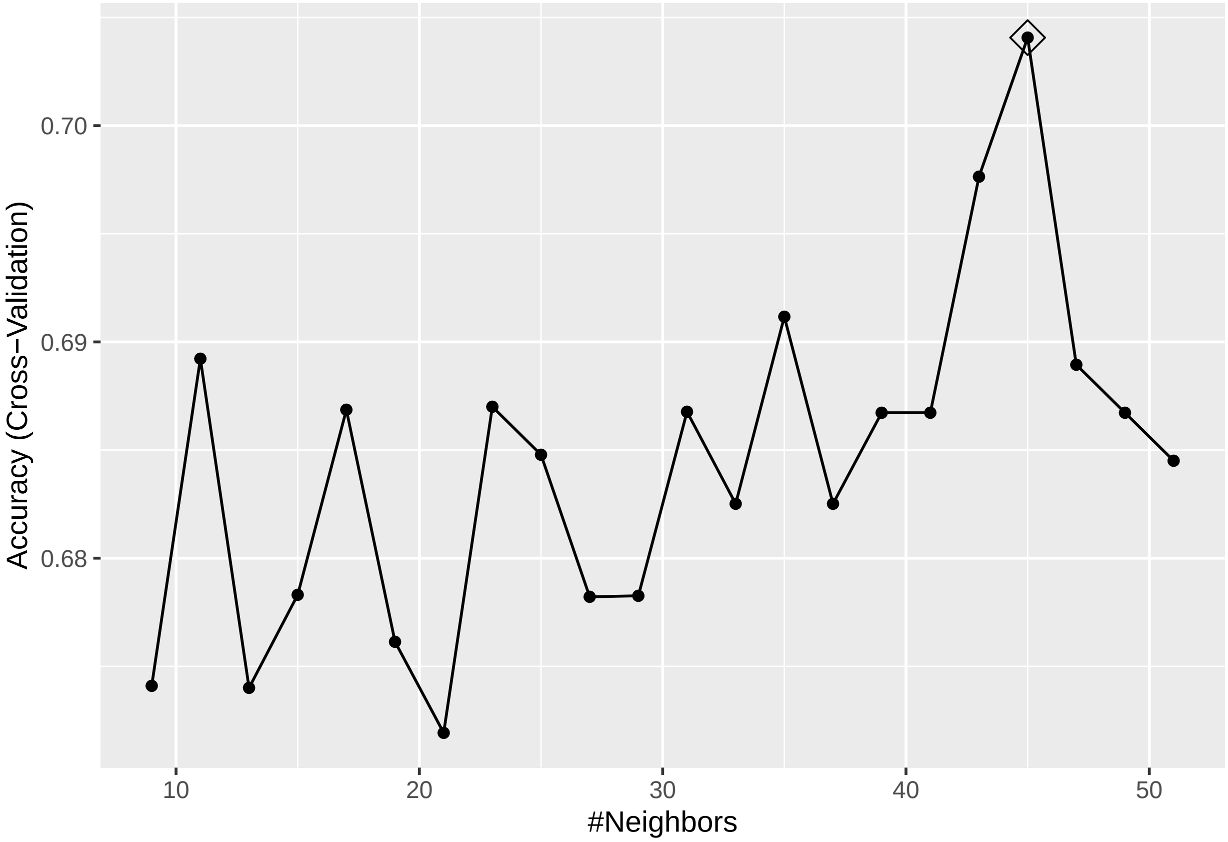
```
## # A tibble: 3 x 2
##   model                                  Accuracy
##   <chr>                                     <dbl>
## 1 Logistic Regression with all predictors    0.25
## 2 Logistic Regression with 9 predictors      0.241
## 3 Logistic Regression with 5 predictors      0.284
```

There a has been a slight improvement of 13% compared to the first logistic regression model in the accuracy. We will try using a different model to improve the predictions.

Model 4: KNN model 1 (continuous variables)

The K-nearest neighbours model will be applied here to all the continuous variables. It is a non-parametric machine learning algorithm that is easy to apply to multiple dimensions.

```
#Model 4: KNN Model 1
#Used all continuous variables
control<- trainControl("cv",number=10,p=.9)
train_knn<- train( Dataset ~ Age+ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_A
                   data = train_set, method = "knn",
                   tuneGrid= data.frame(k=seq(9,51,2)),
                   trControl = control)
ggplot(train_knn,highlight=TRUE)
```



```
train_knn$bestTune
```

```
##     k
## 19 45
```

```
p_hat_knn <- train_knn %>% predict(test_set)
confusionMatrix(p_hat_knn,test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction  1  2
##          1 73 24
##          2 10  9
##
##                Accuracy : 0.7069
##                  95% CI : (0.6152, 0.7877)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.62623
##
##                   Kappa : 0.1746
##
##  Mcnemar's Test P-Value : 0.02578
##
##             Sensitivity : 0.8795
##             Specificity : 0.2727
##          Pos Pred Value : 0.7526
##          Neg Pred Value : 0.4737
##              Prevalence : 0.7155
##          Detection Rate : 0.6293
##    Detection Prevalence : 0.8362
##       Balanced Accuracy : 0.5761
##
##        'Positive' Class : 1
##
```

```
m4<-confusionMatrix(p_hat_knn,test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- bind_rows(overall_accuracy, tibble(model = "KNN with 9 predictors", Accuracy = m4 )
overall_accuracy
```

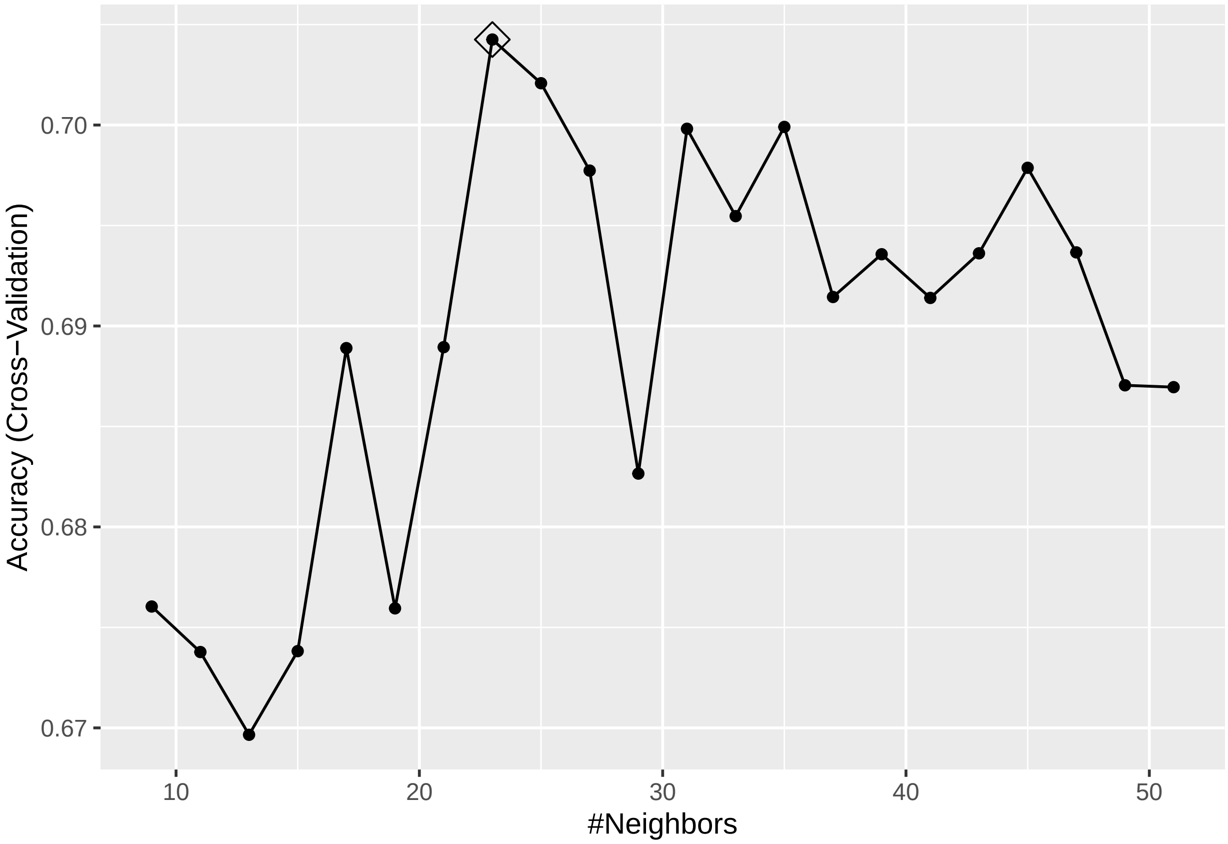
```
## # A tibble: 4 x 2
##   model                                   Accuracy
##   <chr>                                      <dbl>
## 1 Logistic Regression with all predictors    0.25
## 2 Logistic Regression with 9 predictors      0.241
## 3 Logistic Regression with 5 predictors      0.284
## 4 KNN with 9 predictors                      0.707
```

It is seen that the accuracy improves greatly. However will try and get the accuracy as close to 100%.

Model 5: KNN Model 2

In this KNN model we have used on the variables that we significant in the Wilcoxon Signed Rank Test.

```
#Model 5: KNN Model 2
#Used significant variables (removed Total_Protiens)
control<- trainControl("cv",number=10,p=.9)
train_knn<- train( Dataset ~ Age+ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_A
                   data = train_set, method = "knn",
                   tuneGrid= data.frame(k=seq(9,51,2)),
                   trControl = control)
ggplot(train_knn,highlight=TRUE)
```



```
train_knn$bestTune
```

```
##    k
## 8 23
```

```
p_hat_knn <- train_knn %>% predict(test_set)
confusionMatrix(p_hat_knn,test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 72 26
##          2 11  7
##
##                Accuracy : 0.681
##                  95% CI : (0.5881, 0.7645)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.82338
##
##                   Kappa : 0.0922
##
##  Mcnemar's Test P-Value : 0.02136
##
##             Sensitivity : 0.8675
##             Specificity : 0.2121
##          Pos Pred Value : 0.7347
##          Neg Pred Value : 0.3889
##              Prevalence : 0.7155
##          Detection Rate : 0.6207
##    Detection Prevalence : 0.8448
##       Balanced Accuracy : 0.5398
##
##        'Positive' Class : 1
##
```

```
m5<- confusionMatrix(p_hat_knn,test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- bind_rows(overall_accuracy, tibble(model = "KNN with 8 predictors", Accuracy = m5 )
overall_accuracy
```

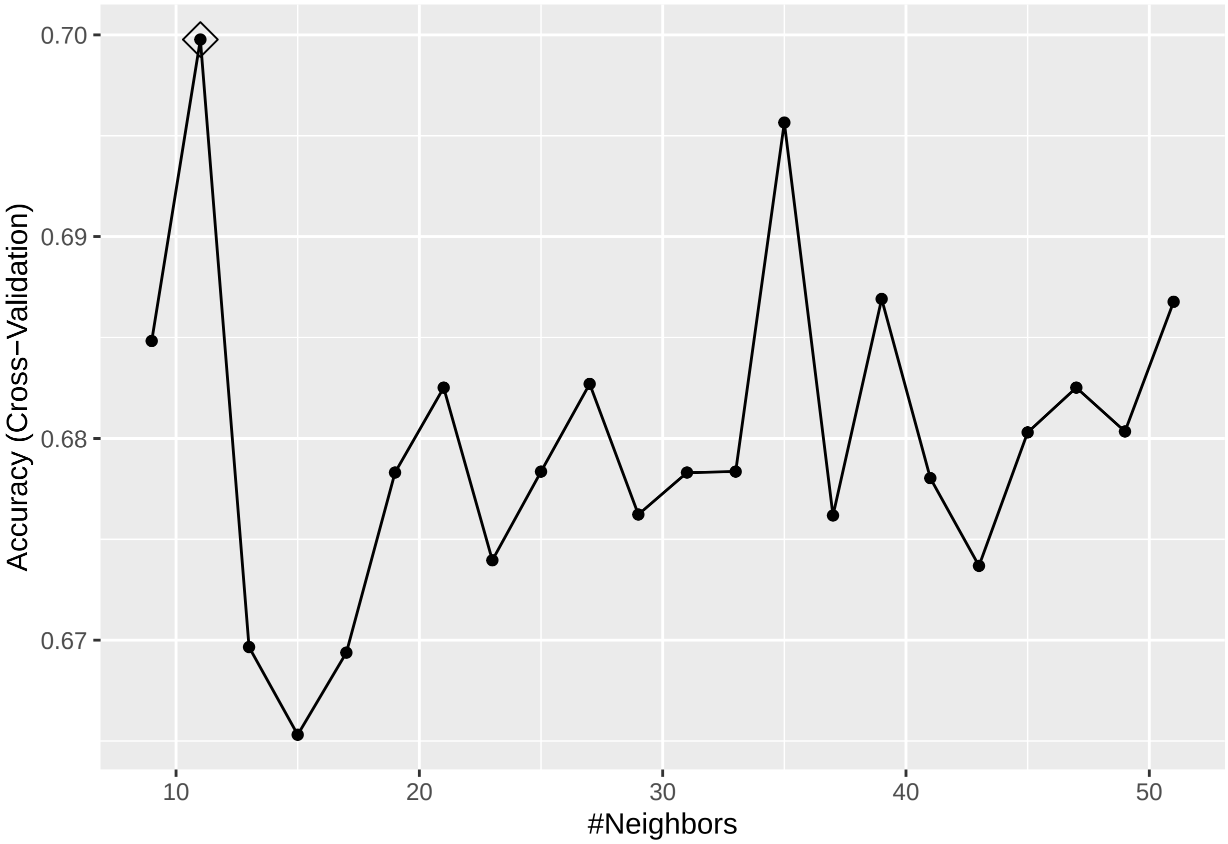
```
## # A tibble: 5 x 2
##   model                                   Accuracy
##   <chr>                                      <dbl>
## 1 Logistic Regression with all predictors    0.25
## 2 Logistic Regression with 9 predictors      0.241
## 3 Logistic Regression with 5 predictors      0.284
## 4 KNN with 9 predictors                      0.707
## 5 KNN with 8 predictors                      0.681
```

The accuracy remains the same. Therefore, we will further add some change to improve it.

Model 6: KNN Model 3

In this KNN model we will use the variables that have a skewed distribution.

```
#Model 6: KNN Model 3
#Used significant variables (using skewed distribution variables)
control<- trainControl("cv",number=10,p=.9)
train_knn<- train( Dataset ~ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_Aminot
                   data = train_set, method = "knn",
                   tuneGrid= data.frame(k=seq(9,51,2)),
                   trControl = control)
ggplot(train_knn,highlight=TRUE)
```



```
train_knn$bestTune
```

```
##    k
## 2 11
```

```
p_hat_knn <- train_knn %>% predict(test_set)
confusionMatrix(p_hat_knn,test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 71 23
##          2 12 10
##
##                Accuracy : 0.6983
##                  95% CI : (0.6061, 0.78)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.70025
##
##                   Kappa : 0.1761
##
##  Mcnemar's Test P-Value : 0.09097
##
##             Sensitivity : 0.8554
##             Specificity : 0.3030
##          Pos Pred Value : 0.7553
##          Neg Pred Value : 0.4545
##              Prevalence : 0.7155
##          Detection Rate : 0.6121
##    Detection Prevalence : 0.8103
##       Balanced Accuracy : 0.5792
##
##        ’Positive’ Class : 1
##
```

```
m6<- confusionMatrix(p_hat_knn,test_set$Dataset)$overall["Accuracy"]
overall_accuracy <- bind_rows(overall_accuracy, tibble(model = "KNN with 7 predictors", Accuracy = m6 )
overall_accuracy
```

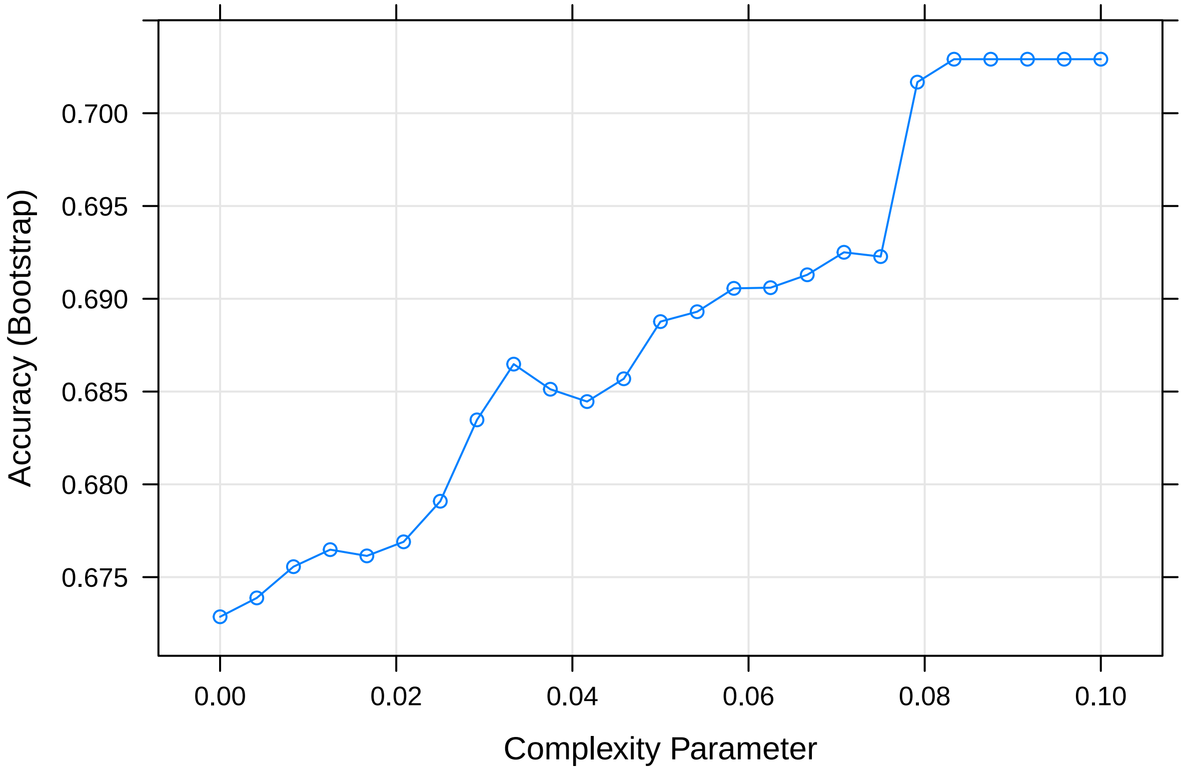
```
## # A tibble: 6 x 2
##   model                                   Accuracy
##   <chr>                                      <dbl>
## 1 Logistic Regression with all predictors    0.25
## 2 Logistic Regression with 9 predictors      0.241
## 3 Logistic Regression with 5 predictors      0.284
## 4 KNN with 9 predictors                      0.707
## 5 KNN with 8 predictors                      0.681
## 6 KNN with 7 predictors                      0.698
```

It is observed that the accuracy still remains the same.

Model 7: Classification (Decision) Trees Model

We will use a different algorithm. Since the outcome is categorical we will use the classification (decision) trees model.

```
#Model 7- Classification (Decision) Trees Model
train_rpart <- train(Dataset ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
                     data = train_set)
plot(train_rpart)
```



```
confusionMatrix(predict(train_rpart,test_set),test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 83 33
##          2  0  0
##
##                Accuracy : 0.7155
##                  95% CI : (0.6243, 0.7954)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.5468
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 2.54e-08
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7155
##          Neg Pred Value :    NaN
##              Prevalence : 0.7155
##          Detection Rate : 0.7155
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 1
##
```

```
m7<- confusionMatrix(predict(train_rpart,test_set),test_set$Dataset)$overall["Accuracy"]
overall_accuracy<- bind_rows(overall_accuracy, tibble(model="Classification Decision Trees Model with al
overall_accuracy
```

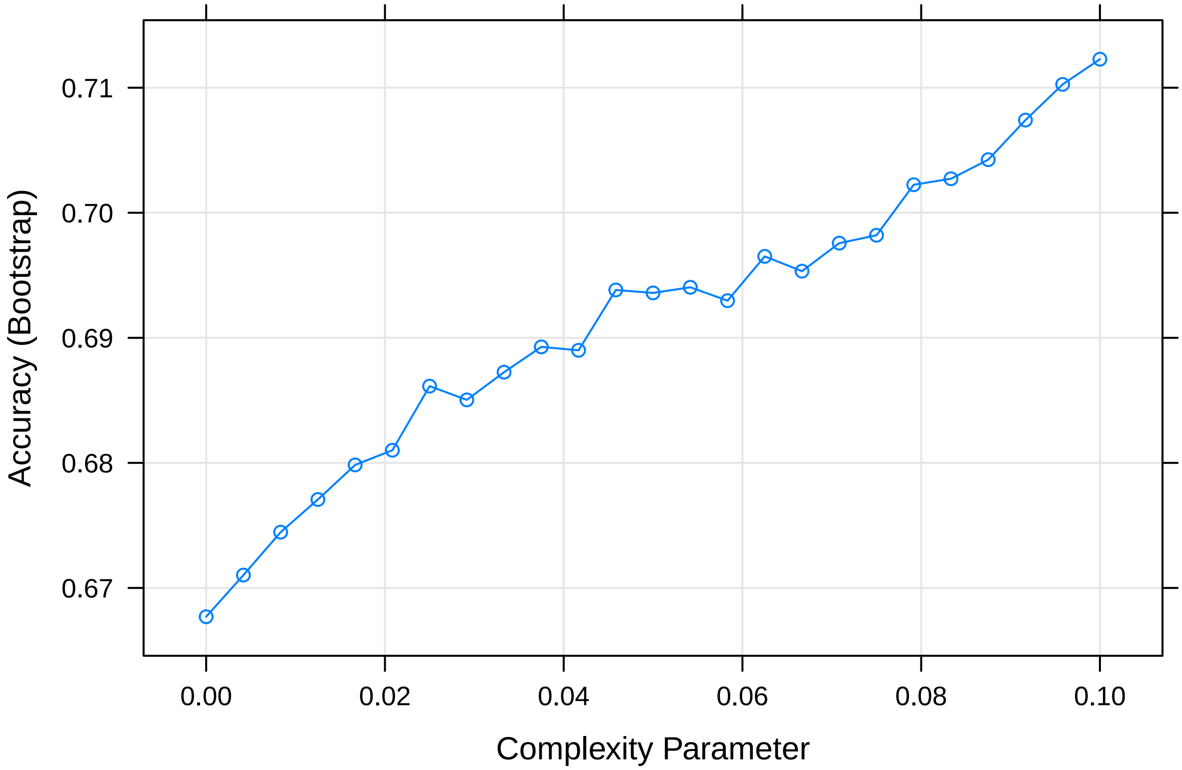
```
## # A tibble: 7 x 2
##   model                                                   Accuracy
##   <chr>                                                      <dbl>
## 1 Logistic Regression with all predictors                    0.25
## 2 Logistic Regression with 9 predictors                      0.241
## 3 Logistic Regression with 5 predictors                      0.284
## 4 KNN with 9 predictors                                      0.707
## 5 KNN with 8 predictors                                      0.681
## 6 KNN with 7 predictors                                      0.698
## 7 Classification Decision Trees Model with all predictors    0.716
```

The overall accuracy great improves. We will try improve this slightly more.

Model 8: Classification (Decision) Trees Model 2 (with significant variables)

In this model only the variables that had significant p-values in the Wilcoxon Signed Rank test will be used.

```
#Model 8- Classification (Decision) Trees Model with significant variables
train_rpart <- train(Dataset ~ Age+ Total_Bilirubin + Direct_Bilirubin + Alkaline_Phosphotase + Alamine_
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
                     data = train_set)
plot(train_rpart)
```



```r
confusionMatrix(predict(train_rpart,test_set),test_set$Dataset)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction  1  2
##          1 83 33
##          2  0  0
## 
##                Accuracy : 0.7155
##                  95% CI : (0.6243, 0.7954)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.5468
## 
##                   Kappa : 0
## 
##  Mcnemar’s Test P-Value : 2.54e-08
## 
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7155
##          Neg Pred Value :    NaN
##              Prevalence : 0.7155
##          Detection Rate : 0.7155
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
## 
##        ’Positive’ Class : 1
## 
```

```r
m8<- confusionMatrix(predict(train_rpart,test_set),test_set$Dataset)$overall["Accuracy"]
overall_accuracy<- bind_rows(overall_accuracy, tibble(model="Classification Decision Trees Model with 8
overall_accuracy
```

```
## # A tibble: 8 x 2
##   model                                                   Accuracy
##   <chr>                                                      <dbl>
## 1 Logistic Regression with all predictors                    0.25 
## 2 Logistic Regression with 9 predictors                      0.241
## 3 Logistic Regression with 5 predictors                      0.284
## 4 KNN with 9 predictors                                      0.707
## 5 KNN with 8 predictors                                      0.681
## 6 KNN with 7 predictors                                      0.698
## 7 Classification Decision Trees Model with all predictors    0.716
## 8 Classification Decision Trees Model with 8 predictors      0.716
```

Results

```r
print.data.frame(overall_accuracy)
```

```
##                                                 model  Accuracy
## 1                  Logistic Regression with all predictors 0.2500000
## 2                    Logistic Regression with 9 predictors 0.2413793
## 3                    Logistic Regression with 5 predictors 0.2844828
## 4                                   KNN with 9 predictors 0.7068966
## 5                                   KNN with 8 predictors 0.6810345
## 6                                   KNN with 7 predictors 0.6982759
## 7 Classification Decision Trees Model with all predictors 0.7155172
## 8   Classification Decision Trees Model with 8 predictors 0.7155172
```

From the results we can see that the classification trees model performed the best. The predictors used is either all or 8 significant variables.

Conclusion

In this project various models have been tested to build a prediction algorithm for doctors to diagnose liver disease. The overall accuracy results show generally all the variables can be used as predictors. However it is suggested that the 8 significant ones are used for the algorithm. Overall, the Logistic Regression model did't perform well and the Classification Trees model performed the best. A limitation of the dataset is that it comes from a very niche group of people and the observations are very few. In future a larger dataset from people accross different regions can be obtained to improve the algorithm. Some kind of clustering categorization also could be added to see if there if any of the variables tend to cluster in a certain group of patients. Additional history of the patient diet and lifestyle could also be added to the clinical variables as it might enhance the prediction.