Report on Liver Disease Prediction Model

Cao Fang

2019/6/13

Introduction

Liver disease is a broad term that covers all the potential problems that cause the liver to fail to perform its designated functions. It can be affected by many conditions. For example, excessive amounts of acetaminophen, acetaminophen combination medications,or cirrhosis and alcohol abuse. This project is designed based on the dataset of Indian Liver Patient Records to build prediction algorithm for liver disease.

This dataset contains 416 liver patient records and 167 non liver patient records collected from North East of Andhra Pradesh, India. Descriptive statistics of the cleaned data are presented through figures and multiple plots. In order to build an accurate prediction model, different methods are used to evaluate the algorithm. The final result indicates that the Generalized Linear Model works better than other methods with the accuracy of 0.724.

Analysis Methods

I.Prepare the Liver data set.

First, download the R packages and the Indian Liver Patient Records dataset. For analysis convenience, the dataset is renamed as Liver.

The Liver dataset contains 583 rows and 11 columns. Each row represents one record of a patient. The first 10 columns include age and gender of the patients, along with other 8 medical indicators. The “Dataset” variable implies if the patient has liver disease or not.

```
Liver %>% as_tibble()
```

```
## # A tibble: 583 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
##  7    26 Female            0.9             0.2             154
##  8    29 Female            0.9             0.3             202
##  9    17 Male              0.9             0.3             202
```

```
## 10    55 Male              0.7            0.2             290
## # ... with 573 more rows, and 6 more variables:
## #   Alamine_Aminotransferase <dbl>, Aspartate_Aminotransferase <dbl>,
## #   Total_Protiens <dbl>, Albumin <dbl>, Albumin_and_Globulin_Ratio <dbl>,
## #   Dataset <dbl>
```

II.Data Cleaning

First, redefine the "Dataset" variable as "Disease" and convert the original values to 0(not identified liver disease) and 1(identified liver disease).

```
str(Liver$Disease)
```

```
##  Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
```

Then, detect all the NAs and remove them from the dataset. Now, there are 579 rows kept in Liver.

```
colSums(sapply(Liver, is.na))
```

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

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

```
dim(Liver)
```

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

III.Descriptive statistic of the Liver Disease dataset

The following descriptive statistics show the basic features of the cleaned dataset. Meanwhile, details of variables are presented through visualization tools below.

```
Liver %>% as_tibble()
```

```
## # A tibble: 579 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
```

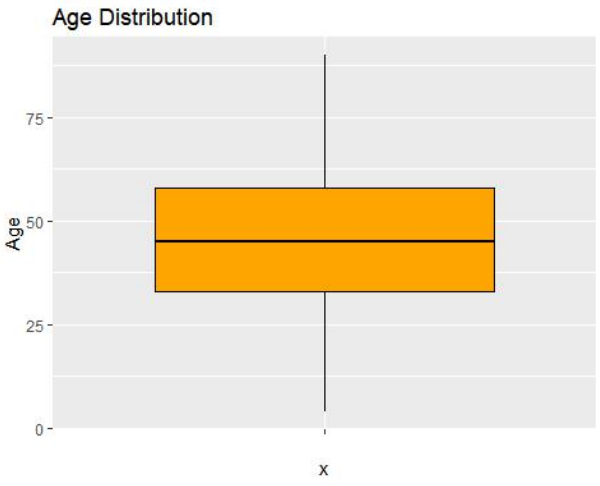
```
##  6    46 Male              1.8             0.7             208
##  7    26 Female            0.9             0.2             154
##  8    29 Female            0.9             0.3             202
##  9    17 Male              0.9             0.3             202
## 10    55 Male              0.7             0.2             290
## # ... with 569 more rows, and 6 more variables:
## #   Alamine_Aminotransferase <dbl>, Aspartate_Aminotransferase <dbl>,
## #   Total_Protiens <dbl>, Albumin <dbl>, Albumin_and_Globulin_Ratio <dbl>,
## #   Disease <fct>
```

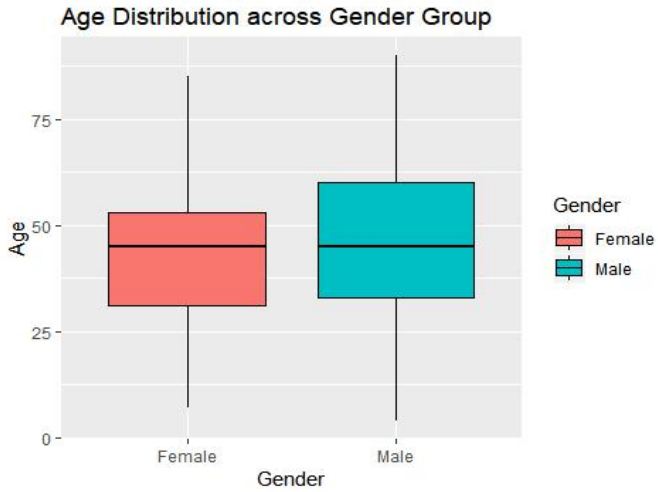
1.Age.

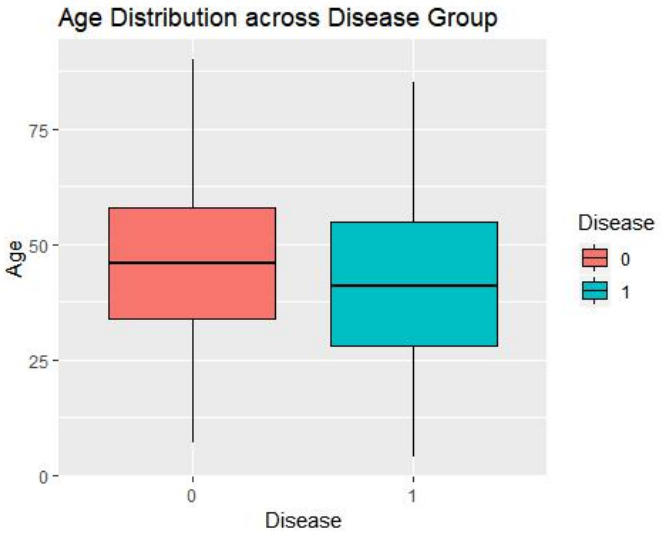
Any patient whose age exceeded 89 is listed as being of age "90". According to the summary, the youngest patient is only 4 years old. And the mean age is 44.8, which is close to the median age of 45.

```
summary(Liver$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00   33.00   45.00   44.78   58.00   90.00
```







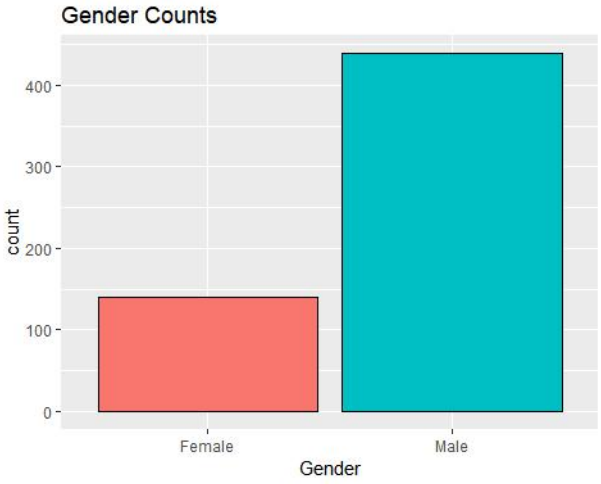
Females and males have similar median age, while males have wider range of age than females. Compared to those who do not identified with liver disease, patients with liver disease tend to have lower median age.

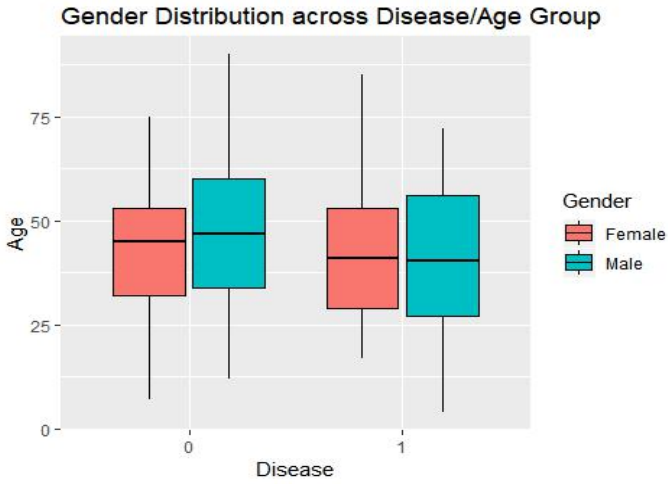
2. Gender

More males than females are included in this dataset. Similar with the general age distribution trend across the disease group, males and females who do not identified liver disease have higher median age. One notable feature of female group is that the minimum and maximum age of those who have liver disease are higher than who do not, which is different than the general trend and trend in the male group.

```
summary(Liver$Gender)

##    Length     Class      Mode
##       579 character character
```





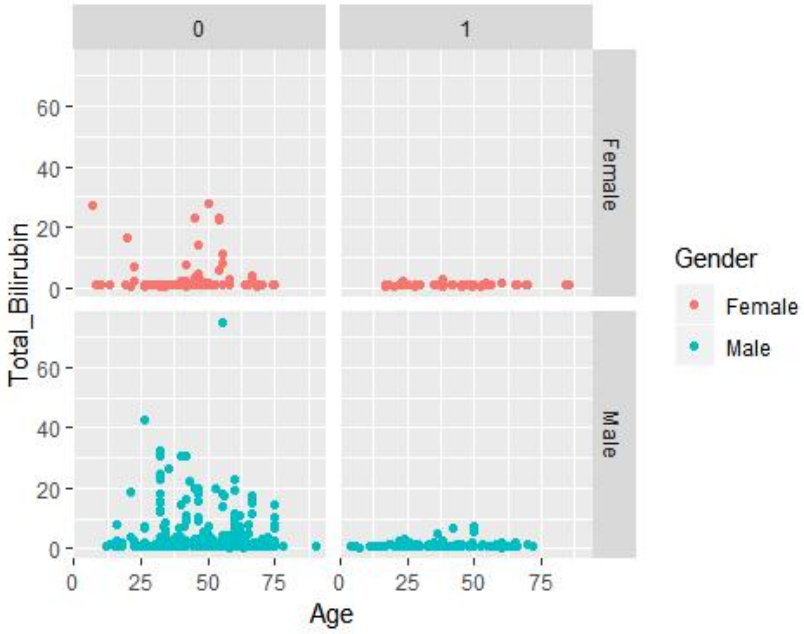
3. Total_Bilirubin

The Total_Bilirubin variable has minimum value of 0.4, median value of 1 and mean value of 3.31. The summary data shows a wide range of Total_Bilirubin distribution, but according to the plot, the value of max 75 is clearly one outlier. Moreover, the plot presents a major difference of the distribution between patients who have liver disease and who do not.

```
summary(Liver$Total_Bilirubin)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.400   0.800   1.000   3.315   2.600  75.000
```

Total_Bilirubin Distribution across Disease/Gender Group



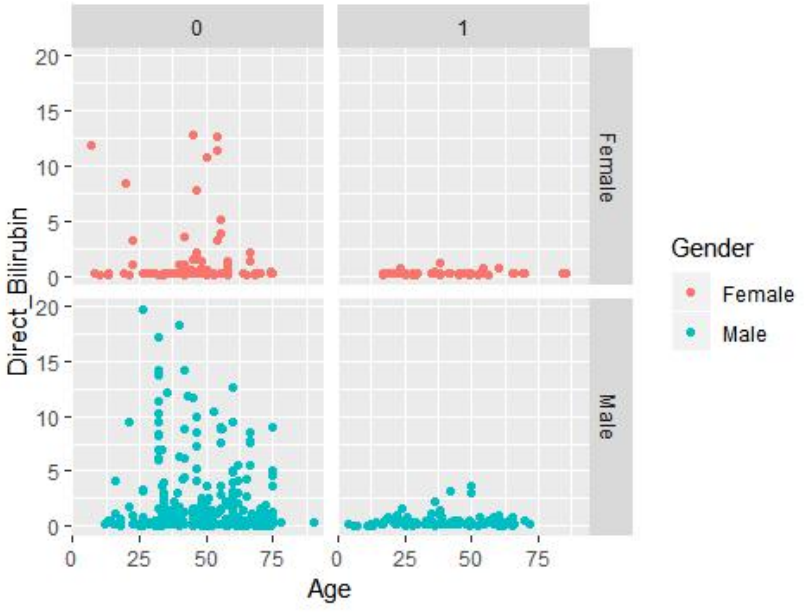
4. Direct_Bilirubin

The Direct_Bilirubin variable distributed unevenly across the scale. It has minimum value of 0.1, median value of 0.3, third quarter value of 1.3, however the maximum value is up to 19.7. And the Direct_Bilirubin is generally low among patients who identified with liver disease.

```
summary(Liver$Direct_Bilirubin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.100   0.200   0.300   1.494   1.300  19.700
```

Direct_Bilirubin Distribution across Disease/Gender Group



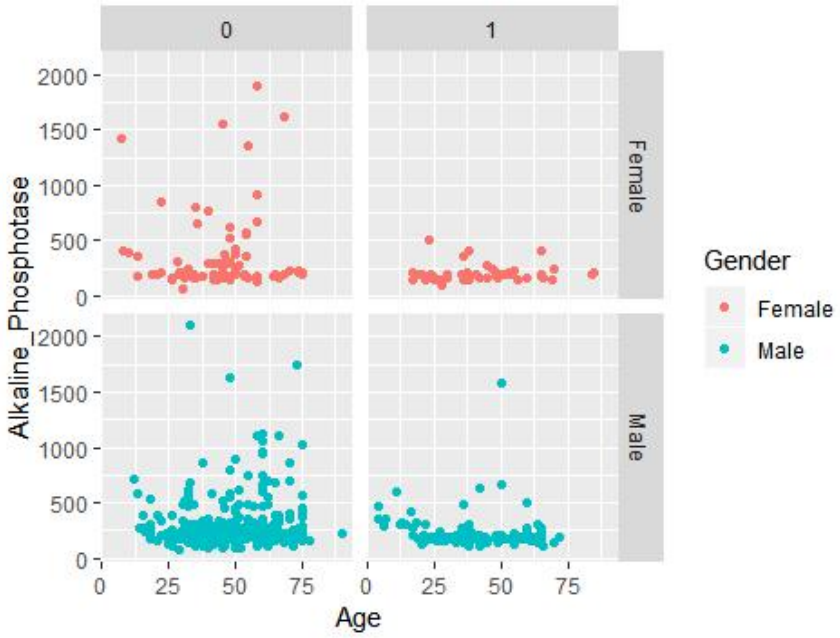
5. Alkaline_Phosphotase

The range for the Alkaline Phosphotase variable is also wide, with a minimum value of 63, median value 208 and maximum value of 2110. The distribution of Alkaline_Phosphotase is skewed, especially among patients who do not have liver disease.

```
summary(Liver$Alkaline_Phosphotase)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    63.0   175.5   208.0   291.4   298.0  2110.0
```

Alkaline_Phosphotase Distribution across Disease/Gender Group



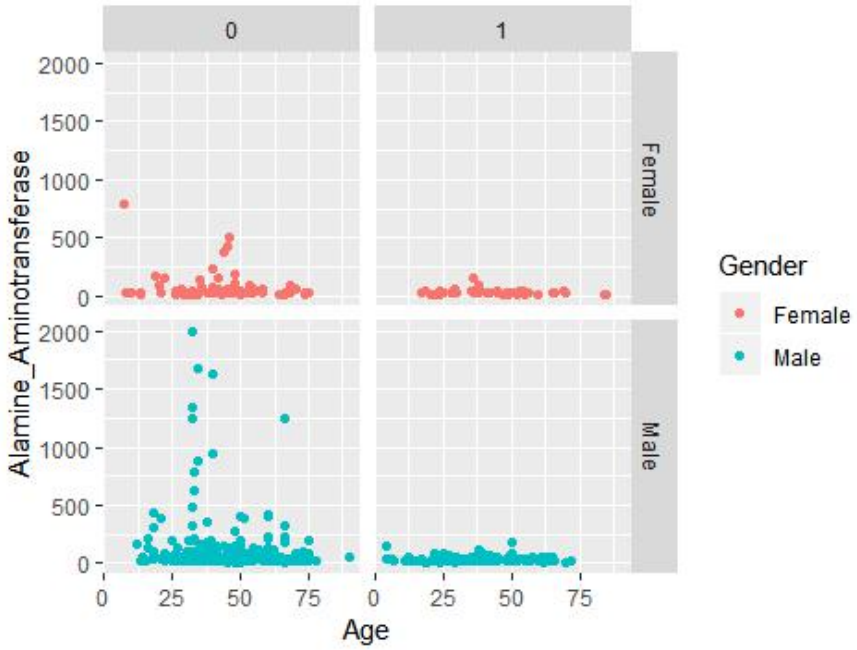
6. Alamine_Aminotransferase

The wide range of the Alamine_Aminotransferase variable can be attributed to males who do not have liver disease. Values above 1000 are all from this subgroup. Significant difference of distribution is showed in among males across the disease groups, but not much among females.

```
summary(Liver$Alamine_Aminotransferase)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.00   23.00   35.00   81.13   61.00 2000.00
```

Alamine_Aminotransferase Distribution across Disease/Gender



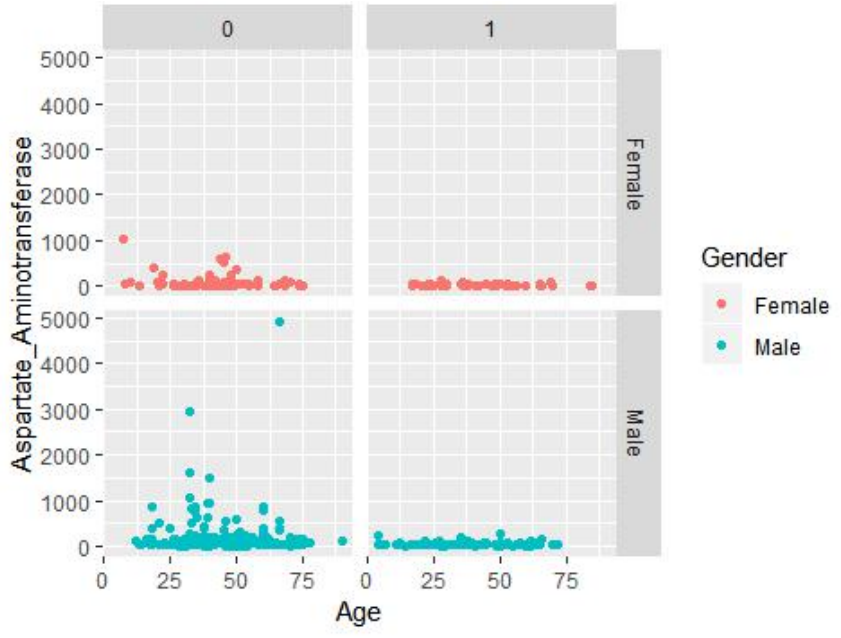
7. Aspartate_Aminotransferase

The Aspartate_Aminotransferase variable has minimum value of 10, median value of 42, but mean value of 110, which is more than twice of the median. The plot shows two outliers that contribute to it, as well as to the wide range of distribution.

```
summary(Liver$Aspartate_Aminotransferase)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.0    25.0    42.0   110.4    87.0  4929.0
```

Aspartate_Aminotransferase Distribution across Disease/Gender

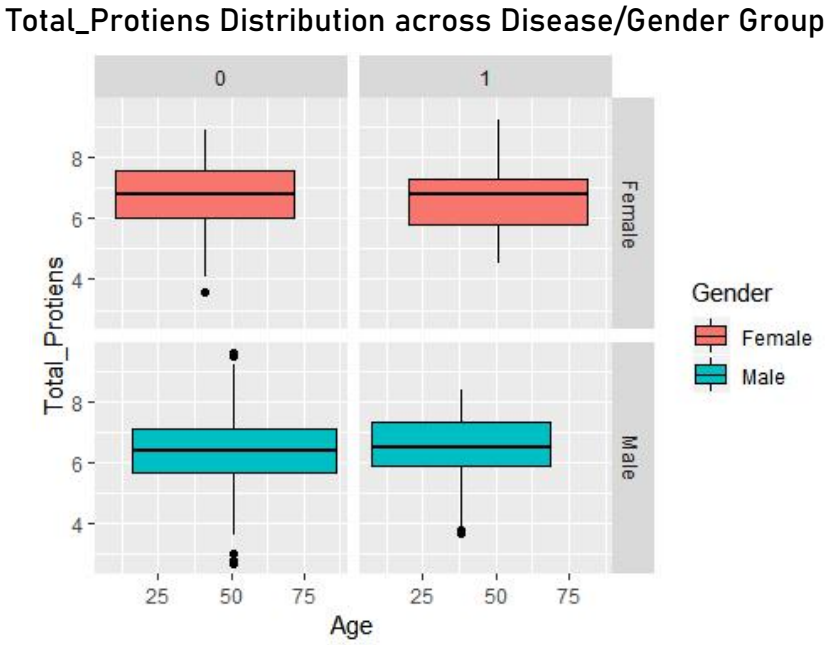


8. Total_Protiens

Box plot is used here due to the close distribution of Total_Protiens. The Total_Protiens variable has minimum value of 2.7, maximum value of 9.6 and mean value of 6.48. Among the four subgroups, males who do not have liver disease have the wildest range of Total_Protiens distribution.

```
summary(Liver$Total_Protiens)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.700   5.800   6.600   6.482   7.200   9.600
```



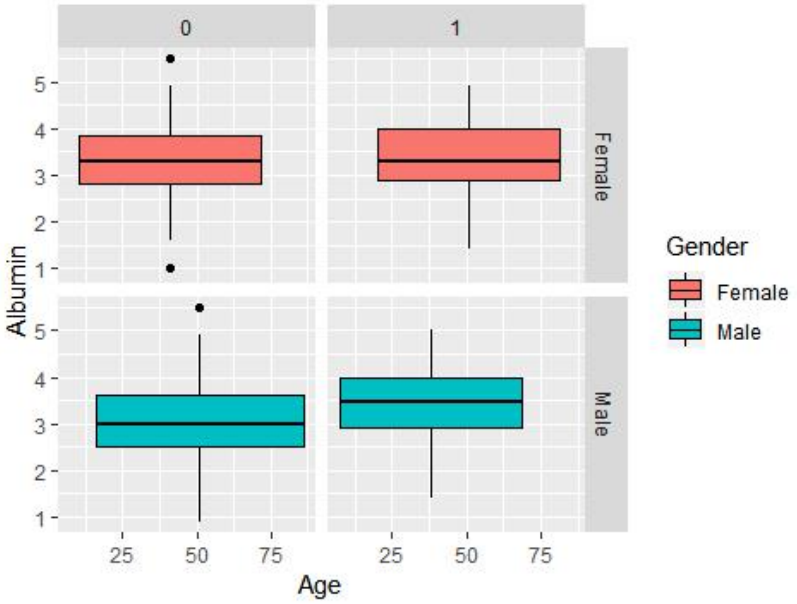
9. Albumin

The Albumin variable has minimum value of 0.9, maximum value of 5.5 and mean value of 3.14. The distributions do not present significant differences among the four subgroups.

```
summary(Liver$Albumin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.900   2.600   3.100   3.139   3.800   5.500
```

Albumin Distribution across Disease/Gender Group



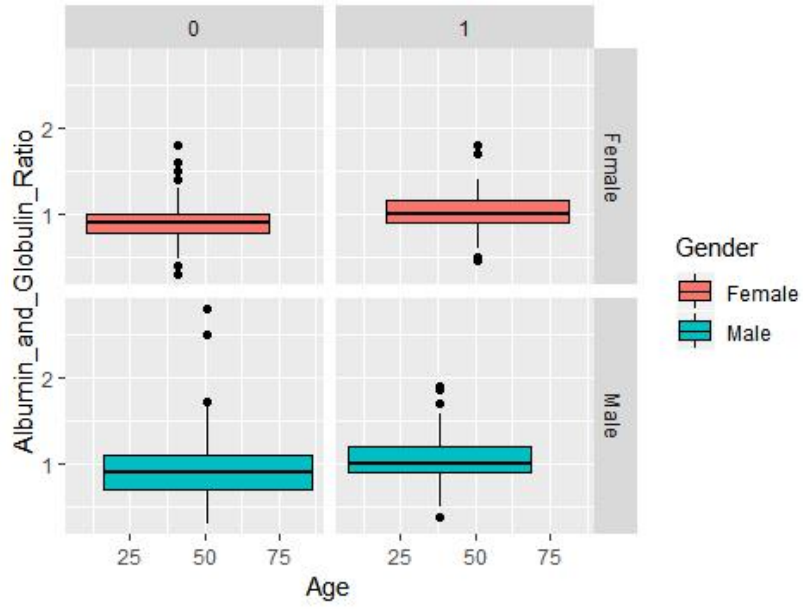
10. Albumin_and_Globulin_Ratio

The Albumin_and_Globulin_Ratio variable has minimum value of 0.3, maximum value of 2.8 and mean value of 0.947. Similar distribution features are presented across gender and disease groups.

```
summary(Liver$Albumin_and_Globulin_Ratio)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3000  0.7000  0.9300  0.9471  1.1000  2.8000
```

Albumin_and_Globulin_Ratio Distribution across Disease/Gender



V.Building the prediction model

1. Remove the highly correlated predictors

Before building the prediction model, highly correlated variables need to be removed to ensure all the predictors in the model are independent. Correlated predictors cannot contribute to the prediction model. The plot below shows the correlation between variables except two non-numeric columns – “Gender” and “Disease”.

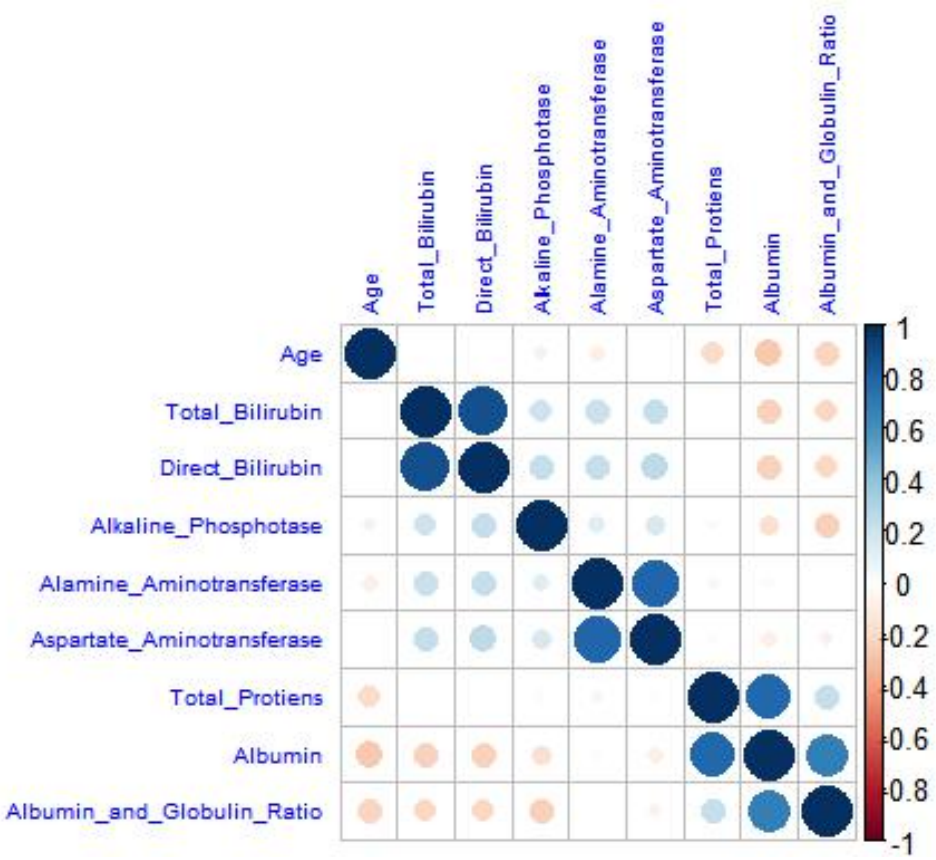
```
##                                    Age     Total_Bilirubin  Direct_Bilirubin
## Age                         1.000000000    0.011000374     6.784303e-03
## Total_Bilirubin             0.011000374    1.000000000     8.744810e-01
## Direct_Bilirubin            0.006784303    0.874480969     1.000000e+00
## Alkaline_Phosphotase        0.078878350    0.205739173     2.340076e-01
## Alamine_Aminotransferase   -0.087799162    0.213375493     2.331801e-01
## Aspartate_Aminotransferase -0.020498946    0.237323055     2.570224e-01
## Total_Protiens             -0.186248122   -0.007905923     3.270877e-05
## Albumin                    -0.264210935   -0.222086570    -2.284092e-01
## Albumin_and_Globulin_Ratio -0.216408346   -0.206267186    -2.001247e-01


##                           Alkaline_Phosphotase     Alamine_Aminotransferase
## Age                                 0.07887835             -0.08779916
## Total_Bilirubin                     0.20573917              0.21337549
## Direct_Bilirubin                    0.23400757              0.23318008
## Alkaline_Phosphotase                1.00000000              0.12477671
## Alamine_Aminotransferase            0.12477671              1.00000000
## Aspartate_Aminotransferase          0.16657999              0.79186215
## Total_Protiens                     -0.02706202             -0.04243210
## Albumin                            -0.16341865             -0.02865750
## Albumin_and_Globulin_Ratio         -0.23416650             -0.00237499


##                           Aspartate_Aminotransferase  Total_Protiens
## Age                                 -0.02049895         -1.862481e-01
## Total_Bilirubin                      0.23732305         -7.905923e-03
## Direct_Bilirubin                     0.25702239          3.270877e-05
## Alkaline_Phosphotase                 0.16657999         -2.706202e-02
## Alamine_Aminotransferase             0.79186215         -4.243210e-02
## Aspartate_Aminotransferase           1.00000000         -2.575101e-02
## Total_Protiens                      -0.02575101          1.000000e+00
## Albumin                             -0.08491457          7.831122e-01
## Albumin_and_Globulin_Ratio          -0.07003983          2.348872e-01


##                               Albumin          Albumin_and_Globulin_Ratio
## Age                          -0.26421094               -0.21640835
## Total_Bilirubin              -0.22208657               -0.20626719
## Direct_Bilirubin             -0.22840915               -0.20012469
## Alkaline_Phosphotase         -0.16341865               -0.23416650
## Alamine_Aminotransferase     -0.02865750               -0.00237499
## Aspartate_Aminotransferase   -0.08491457               -0.07003983
## Total_Protiens                0.78311217                0.23488718
```

```
## Albumin                     1.00000000              0.68963234
## Albumin_and_Globulin_Ratio  0.68963234              1.00000000
```



Setting the cutoff at 0.5, three variables are detected to be highly correlated. After removing them, only 8 variables are kept in the dataset.

```
high_Cor <- findCorrelation(correlations, cutoff = 0.5, names=TRUE)
high_Cor
```

```
## [1] "Albumin"                  "Direct_Bilirubin"
## [3] "Aspartate_Aminotransferase"
```

```
Liver <- Liver[, !(names(Liver) %in% high_Cor)]

dim(Liver)
```

```
## [1] 579   8
```

2.Create train set and test set

To ensure the consistence of the results, the seed will be set at 1 throughout the project. The train set and test set are created through createDataPartition function.

The train set account for 50% of the whole dataset. The algorithm will be evaluated in the test set.

```
set.seed(1, sample.kind = "Rounding")
test_index <- createDataPartition(Liver$Disease, times = 1, p = 0.5, list = FALSE)
train_set <- Liver[-test_index,]
test_set <- Liver[test_index,]
```

3.Model 1 - Generalized Linear Model

In the Generalized Linear Model, logistic regression is applied for prediction through "glm" function. And the model is specified with the "family" argument as "binomial". According to the confusion matrix, the Generalized Linear Model has the accuracy of 0.724.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 191  64
##         1  16  19
##
##               Accuracy : 0.7241
##                 95% CI : (0.6689, 0.7748)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.3759
##
##                  Kappa : 0.1834
##
##  Mcnemar's Test P-Value : 1.482e-07
##
##            Sensitivity : 0.9227
##            Specificity : 0.2289
##         Pos Pred Value : 0.7490
##         Neg Pred Value : 0.5429
##             Prevalence : 0.7138
##         Detection Rate : 0.6586
##   Detection Prevalence : 0.8793
##      Balanced Accuracy : 0.5758
##
##       'Positive' Class : 0
##
```

For the purpose of comparison with other models, values of accuracy, sensitivity and specificity are stored independently.

4.Model 2 - KNN Model

Compared to the previous Generalized Linear Model, k-nearest neighbors algorithm comes up with a much lower accuracy of 0.648 when k is equal to 5.

```
## Confusion Matrix and Statistics
##
```

```
##          Reference
## Prediction   0   1
##         0 166  61
##         1  41  22
##
##               Accuracy : 0.6483
##                 95% CI : (0.5903, 0.7032)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.99361
##
##                  Kappa : 0.0722
##
##  Mcnemar's Test P-Value : 0.05993
##
##            Sensitivity : 0.8019
##            Specificity : 0.2651
##         Pos Pred Value : 0.7313
##         Neg Pred Value : 0.3492
##             Prevalence : 0.7138
##         Detection Rate : 0.5724
##   Detection Prevalence : 0.7828
##      Balanced Accuracy : 0.5335
##
##       'Positive' Class : 0
##
```

However, this model can be improved through parameter k with the following process.

```
set.seed(1, sample.kind = "Rounding")
fit.knn_k <- train(Disease ~ ., data = train_set, method = "knn", tuneGrid =
 data.frame(k = seq(1,100,1)))
fit.knn_k$bestTune
```

```
##    k
## 62 62
```

When setting the k to 62, the KNN model accuracy is improved to 0.714

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 184  60
##         1  23  23
##
##               Accuracy : 0.7138
##                 95% CI : (0.658, 0.7651)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.5296
##
##                  Kappa : 0.1916
##
##  Mcnemar's Test P-Value : 7.766e-05
##
```

```
##            Sensitivity : 0.8889
##            Specificity : 0.2771
##         Pos Pred Value : 0.7541
##         Neg Pred Value : 0.5000
##             Prevalence : 0.7138
##         Detection Rate : 0.6345
##   Detection Prevalence : 0.8414
##      Balanced Accuracy : 0.5830
##
##       'Positive' Class : 0
##
```

5.Model 3 - Linear Discriminant Analysis Model

Linear Discriminant Analysis Model works well in accuracy and sensitivity. However, the specificity is only 0.07, which is much lower than the previous models.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 197  77
##         1  10   6
##
##               Accuracy : 0.7
##                 95% CI : (0.6437, 0.7522)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.7226
##
##                  Kappa : 0.0316
##
##  Mcnemar's Test P-Value : 1.484e-12
##
##            Sensitivity : 0.95169
##            Specificity : 0.07229
##         Pos Pred Value : 0.71898
##         Neg Pred Value : 0.37500
##             Prevalence : 0.71379
##         Detection Rate : 0.67931
##   Detection Prevalence : 0.94483
##      Balanced Accuracy : 0.51199
##
##       'Positive' Class : 0
##
```

6.Model 4 - Random Forest Model 1

In the first Random Forest Model, “rf” function is used for prediction. The model set the nodesize to 50, maxnodes to 25 and ntree to 1000. The accuracy of 0.676 is not very good.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
```

```
##         0 172  59
##         1  35  24
##
##               Accuracy : 0.6759
##                 95% CI : (0.6187, 0.7294)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.93106
##
##                  Kappa : 0.1315
##
##  Mcnemar's Test P-Value : 0.01768
##
##            Sensitivity : 0.8309
##            Specificity : 0.2892
##         Pos Pred Value : 0.7446
##         Neg Pred Value : 0.4068
##             Prevalence : 0.7138
##         Detection Rate : 0.5931
##   Detection Prevalence : 0.7966
##      Balanced Accuracy : 0.5600
##
##       'Positive' Class : 0
##
```

The mtry values can be tuned to improve the performance of the model.

```
## Random Forest
##
## 289 samples
##   7 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 260, 260, 261, 260, 260, 261, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    1    0.7196880  0.03518328
##    2    0.7296880  0.21139908
##    3    0.7155337  0.20510006
##    4    0.7189819  0.21739512
##    5    0.7087521  0.19244265
##    6    0.7191051  0.22333175
##    7    0.7089901  0.21092121
##    8    0.7226601  0.23100795
##    9    0.7192200  0.23011622
##   10    0.7157718  0.22195726
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

With the best mtry value of 2, the accuracy is improved to 0.7.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 180  60
##         1  27  23
##
##               Accuracy : 0.7
##                 95% CI : (0.6437, 0.7522)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.7226163
##
##                  Kappa : 0.1665
##
##  Mcnemar's Test P-Value : 0.0006019
##
##            Sensitivity : 0.8696
##            Specificity : 0.2771
##         Pos Pred Value : 0.7500
##         Neg Pred Value : 0.4600
##             Prevalence : 0.7138
##         Detection Rate : 0.6207
##   Detection Prevalence : 0.8276
##      Balanced Accuracy : 0.5733
##
##       'Positive' Class : 0
##
```

7.Model 5 - Random Forest Model 2

Different with the first Random Forest Model, this model uses “Rborist” function to improve the prediction by making the estimates smoother. Before tuning the parameter, the accuracy is only 0.666.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 171  61
##         1  36  22
##
##               Accuracy : 0.6655
##                 95% CI : (0.608, 0.7196)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.96875
##
##                  Kappa : 0.1002
##
##  Mcnemar's Test P-Value : 0.01482
##
##            Sensitivity : 0.8261
```

```
##            Specificity : 0.2651
##         Pos Pred Value : 0.7371
##         Neg Pred Value : 0.3793
##             Prevalence : 0.7138
##         Detection Rate : 0.5897
##   Detection Prevalence : 0.8000
##      Balanced Accuracy : 0.5456
##
##       'Positive' Class : 0
##
```

The following process detects the most fitted predFixed and minNode to maximize the results.

```
## Random Forest
##
## 289 samples
##   7 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 289, 289, 289, 289, 289, 289, ...
## Resampling results across tuning parameters:
##
##   predFixed  minNode  Accuracy   Kappa
##    1          5      0.7083067  0.04095090
##    1        105      0.7072808  0.01125428
##    1        205      0.7068960  0.00000000
##    2         15      0.7047549  0.20567685
##    2        115      0.7023984  0.04016657
##    2        215      0.7068960  0.00000000
##    3         25      0.6953327  0.18328481
##    3        125      0.7015931  0.03868017
##    3        225      0.7072696  0.01404073
##    4         35      0.6890256  0.17506508
##    4        135      0.7007439  0.03861806
##    4        235      0.7026311  0.03012542
##    5         45      0.6954810  0.19432066
##    5        145      0.6966087  0.03957741
##    5        245      0.6981156  0.03952105
##    6         55      0.6879672  0.18674522
##    6        155      0.6942643  0.04032261
##    6        255      0.6947509  0.04447658
##    7         65      0.6858073  0.20401457
##    7        165      0.6878843  0.03711525
##    7        265      0.6882218  0.04053675
##    8         75           NaN        NaN
##    8        175           NaN        NaN
##    8        275           NaN        NaN
##    9         85           NaN        NaN
##    9        185           NaN        NaN
##    9        285           NaN        NaN
##   10         95           NaN        NaN
##   10        195           NaN        NaN
```

```
##   10        295           NaN        NaN
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were predFixed = 1 and minNode = 5.
```

With the prefFixed equal to 1 and minNode equal to 5, the accuracy of Random Forest Model 2 is improved a lot to 0.714.

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction   0   1
##         0 206  82
##         1   1   1
##
##               Accuracy : 0.7138
##                 95% CI : (0.658, 0.7651)
##    No Information Rate : 0.7138
##    P-Value [Acc > NIR] : 0.5296
##
##                  Kappa : 0.0102
##
##  Mcnemar's Test P-Value : <2e-16
##
##            Sensitivity : 0.99517
##            Specificity : 0.01205
##         Pos Pred Value : 0.71528
##         Neg Pred Value : 0.50000
##             Prevalence : 0.71379
##         Detection Rate : 0.71034
##   Detection Prevalence : 0.99310
##      Balanced Accuracy : 0.50361
##
##       'Positive' Class : 0
##
```

Results

The results of all five models are showed below.

```
##                  Method                  Accuracy     Sensitivity    Specificity
## 1 Generalized Linear Model               0.7241379     0.9227053      0.22891566
## 2 KNN Model                              0.7137931     0.8888889      0.27710843
## 3 Linear Discriminant Analysis Model     0.7000000     0.9516908      0.07228916
## 4 Random Forest Model 1                  0.7000000     0.8695652      0.27710843
## 5 Random Forest Model 2                  0.7137931     0.9951691      0.01204819
```

After tuning the parameter, all the models achieve an accuracy above 0.7. However, it is notable that the Linear Discriminant Analysis Model and Random Forest Model 2 have extremely low specificity, despite their high sensitivity. The model with best accuracy is the Generalized Linear Model with 0.724.

Conclusion

Generalized Linear Model with highest accuracy works best compared to other models. However, in this particular medical context, accuracy may not be the only indicator. Specificity means the accurate diagnosis when the patient is actually affected with liver disease. Thus, although the Random Forest Model 2 has the same accuracy with KNN model, and even higher sensitivity, it cannot be applied in practice due to its high rate of misdiagnosis of true patients. With this consideration, it seems that the KNN Model works better than Generalized Linear Model, since the former specificity is 21% higher than the latter one.