Data Science Capstone: Liver Disease Prediction Project

Shannon Hakkal

4/8/2020

Contents

1 Introduction

In order to aid in doctors' diagnoses, this project was designed to evaluate various machine learning algorithms that could help predict the presence of liver disease. A large dataset was obtained from the Kaggle website and the UCI Machine Learning Repository (Lichman, 2013). This data was collected from an area northeast of Andhra Pradesh in India in 2013.

The goal of this project was to develop multiple machine learning models and then determine which one gives the best prediction results (i.e. the highest overall accuracy). After testing five different methods, including Logistic Regression, Naive Bayes, Linear Discriminant Analysis, K-Nearest Neighbors, and Random Forest, we concluded that the Random Forest model elicited the best predictive accuracy.

2 Methods/Analysis

To complete this project, we accessed and imported the Indian Liver Patient Records dataset from the Kaggle website: https://www.kaggle.com/uciml/indian-liver-patient-records/data. We then used R to split the dataset into two subsets called "modeling" and "validation." The modeling set was used to train and test each of the five algorithms we proposed. The validation set was only used to test the best-performing, final model at the end of the analysis.

2.1 Materials

The materials used for this project are listed below.

- Indian Liver Patient Records Dataset This can be downloaded here: https://www.kaggle.com/uciml/indian-liver-patient-records/data
- R and RStudio

2.2 Preparations

Install packages and load libraries:

```
if(!require(readr)) install.packages("readr", repos = "http://cran.us.r-project.org")
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(gridExtra)) install.packages("gridExtra", repos = "http://cran.us.r-project.org")
if(!require(klaR)) install.packages("klaR", repos = "http://cran.us.r-project.org")
if(!require(rminer)) install.packages("rminer", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")

library("readr")
library("ggplot2")
library("dplyr")
library("caret")
library("tidyverse")
library("data.table")
library("gridExtra")
library("klaR")
library("rminer")
library("rpart")
library("randomForest")
```

Load dataset:

```
indian_liver_patient <-
read_csv("~/Downloads/indian_liver_patient.csv")
```

or

```
dl <- tempfile()
 download.file("https://www.kaggle.com/uciml/indian-liver-patient-records/data", dl)
```

Examine the structure of the data: Indian Liver Patient Records is a dataset with 583 observations (rows) and 11 variables (columns).

```
str(indian_liver_patient)
```

```
## tibble [583 x 11] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Age                       : num [1:583] 65 62 62 58 72 46 26 29 17 55 ...
##  $ Gender                    : chr [1:583] "Female" "Male" "Male" "Male" ...
##  $ Total_Bilirubin           : num [1:583] 0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
##  $ Direct_Bilirubin          : num [1:583] 0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ Alkaline_Phosphotase      : num [1:583] 187 699 490 182 195 208 154 202 202 290 ...
##  $ Alamine_Aminotransferase  : num [1:583] 16 64 60 14 27 19 16 14 22 53 ...
##  $ Aspartate_Aminotransferase: num [1:583] 18 100 68 20 59 14 12 11 19 58 ...
##  $ Total_Protiens            : num [1:583] 6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ Albumin                   : num [1:583] 3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ Albumin_and_Globulin_Ratio: num [1:583] 0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ Dataset                   : num [1:583] 1 1 1 1 1 1 1 1 2 1 ...
##  - attr(*, "spec")=
##   .. 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()
##   .. )
```

Notice that the "Dataset" column represents the outcomes: "1" = liver disease, "2" = no liver disease.

2.3 Procedure

2.3.1 1. Exploratory Data Analysis

The first step was to explore and visualize the Indian Liver Patient Records dataset.

```
head(indian_liver_patient)
```

```
## # 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>
```

Compute summary statistics of the variables.

```
summary(indian_liver_patient)
```

```
##       Age           Gender          Total_Bilirubin  Direct_Bilirubin
##  Min.   : 4.00   Length:583         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   :44.75                      Mean   : 3.299   Mean   : 1.486
##  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   : 290.6       Mean   :  80.71          Mean   : 109.9
##  3rd Qu.: 298.0       3rd Qu.:  60.50          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             Min.   :1.000
##  1st Qu.:5.800   1st Qu.:2.600   1st Qu.:0.7000             1st Qu.:1.000
##  Median :6.600   Median :3.100   Median :0.9300             Median :1.000
##  Mean   :6.483   Mean   :3.142   Mean   :0.9471             Mean   :1.286
##  3rd Qu.:7.200   3rd Qu.:3.800   3rd Qu.:1.1000             3rd Qu.:2.000
##  Max.   :9.600   Max.   :5.500   Max.   :2.8000             Max.   :2.000
##                                  NA's   :4
```

Remove NA entries from the dataset.

```
indian_liver_patient <- indian_liver_patient %>%
    filter(!is.na(Albumin_and_Globulin_Ratio))
```

Change "Gender" and "Dataset" variables to factors.

```
indian_liver_patient$Gender <- as.factor(indian_liver_patient$Gender)
indian_liver_patient$Dataset <- as.factor(indian_liver_patient$Dataset)
```

Count the number of males and females in the dataset.

```
sum(indian_liver_patient$Gender == "Male")
```

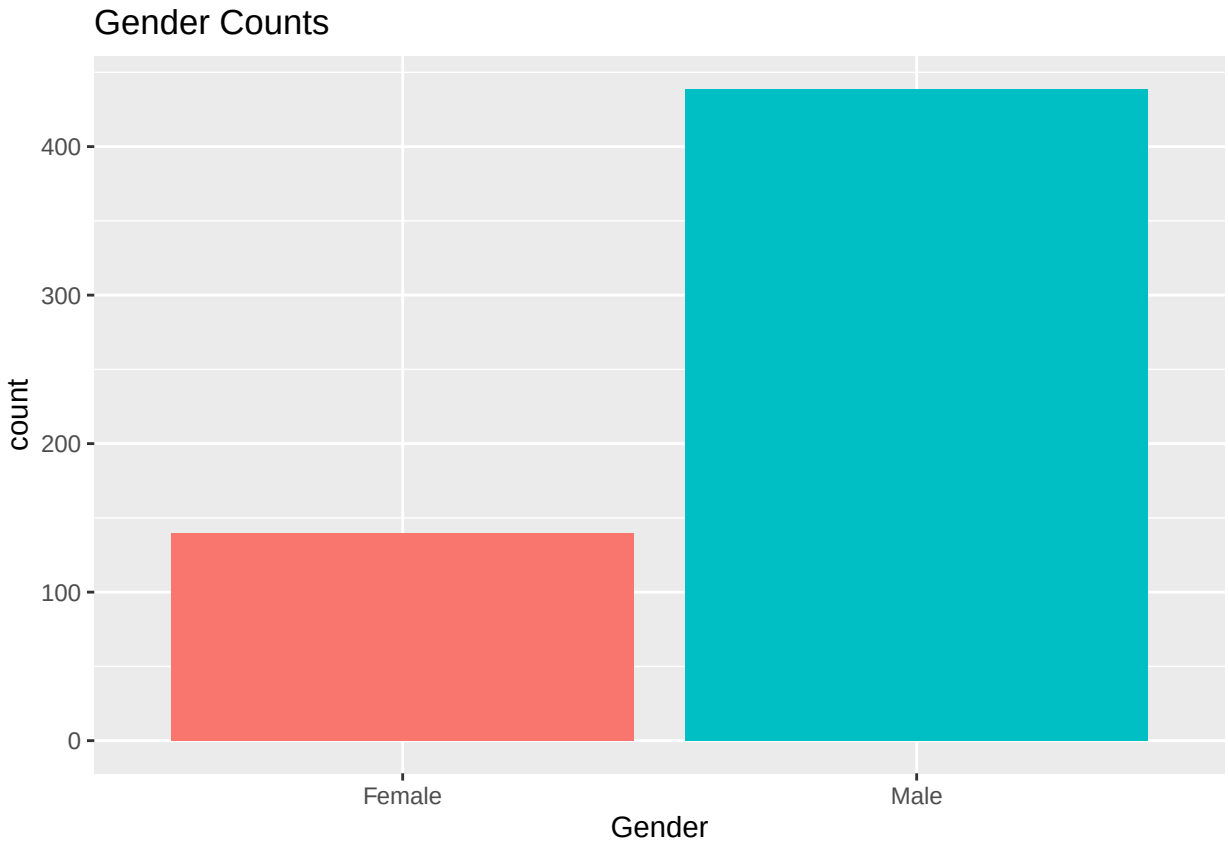
```
## [1] 439
```

```r
sum(indian_liver_patient$Gender == "Female")
```

```
## [1] 140
```

Plot the Gender Counts.

```r
ggplot(indian_liver_patient, aes(Gender)) +
    geom_bar(aes(fill = Gender)) +
    ggtitle("Gender Counts") +
    theme(legend.position="none")
```



Plot the age distribution across genders.

```r
ggplot(indian_liver_patient, aes(x=Age, color=Gender)) +
    geom_density(alpha=.5) +
    ggtitle("Age Distribution Across Genders") +
    theme(legend.position="none")
```

Count the number of positive and negative liver disease cases in the dataset.

```
sum(indian_liver_patient$Dataset == "1")
```

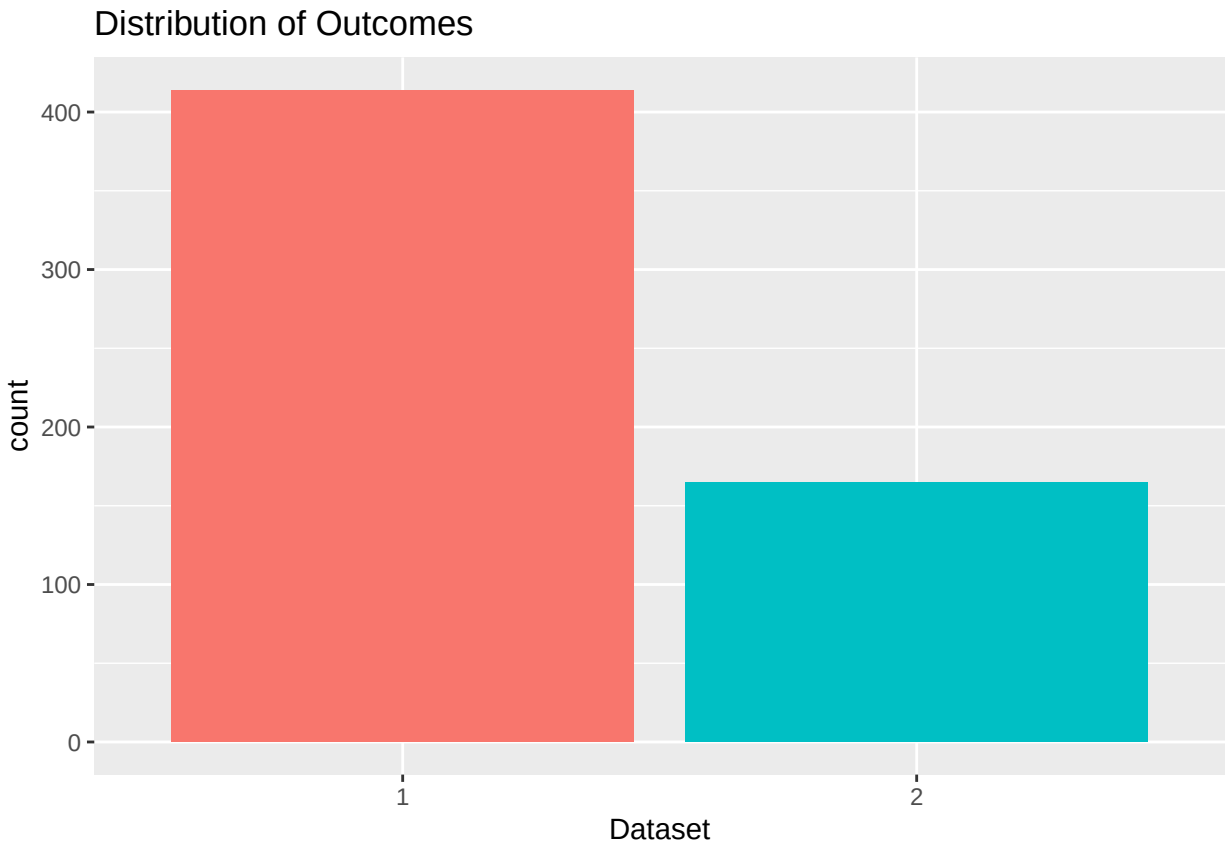
```
## [1] 414
```

```
sum(indian_liver_patient$Dataset == "2")
```

```
## [1] 165
```

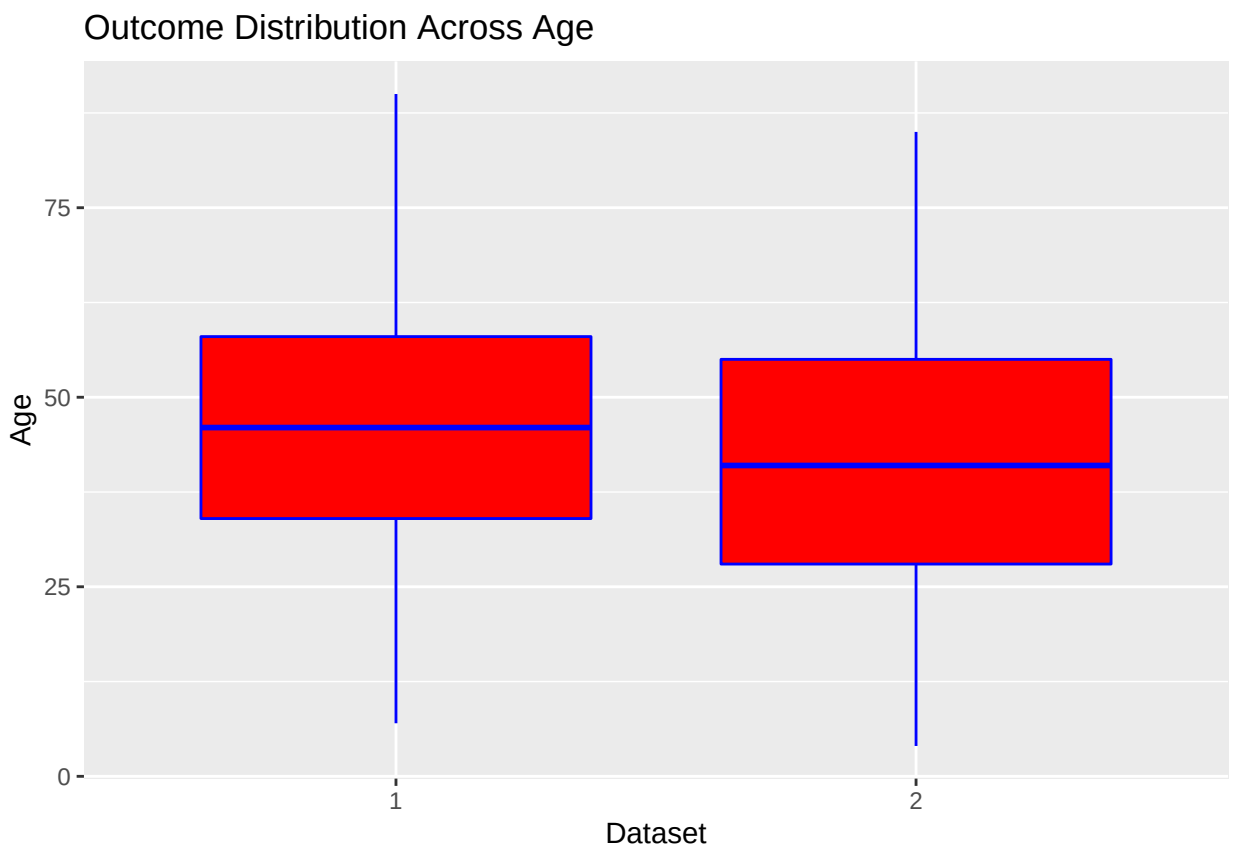
Plot the outcome distribution ("1" = liver disease or "2" = no liver disease).

```
ggplot(indian_liver_patient, aes(Dataset)) +
    geom_bar(stat = "count", aes(fill = Dataset)) +
    ggtitle("Distribution of Outcomes") +
    theme(legend.position = "none")
```



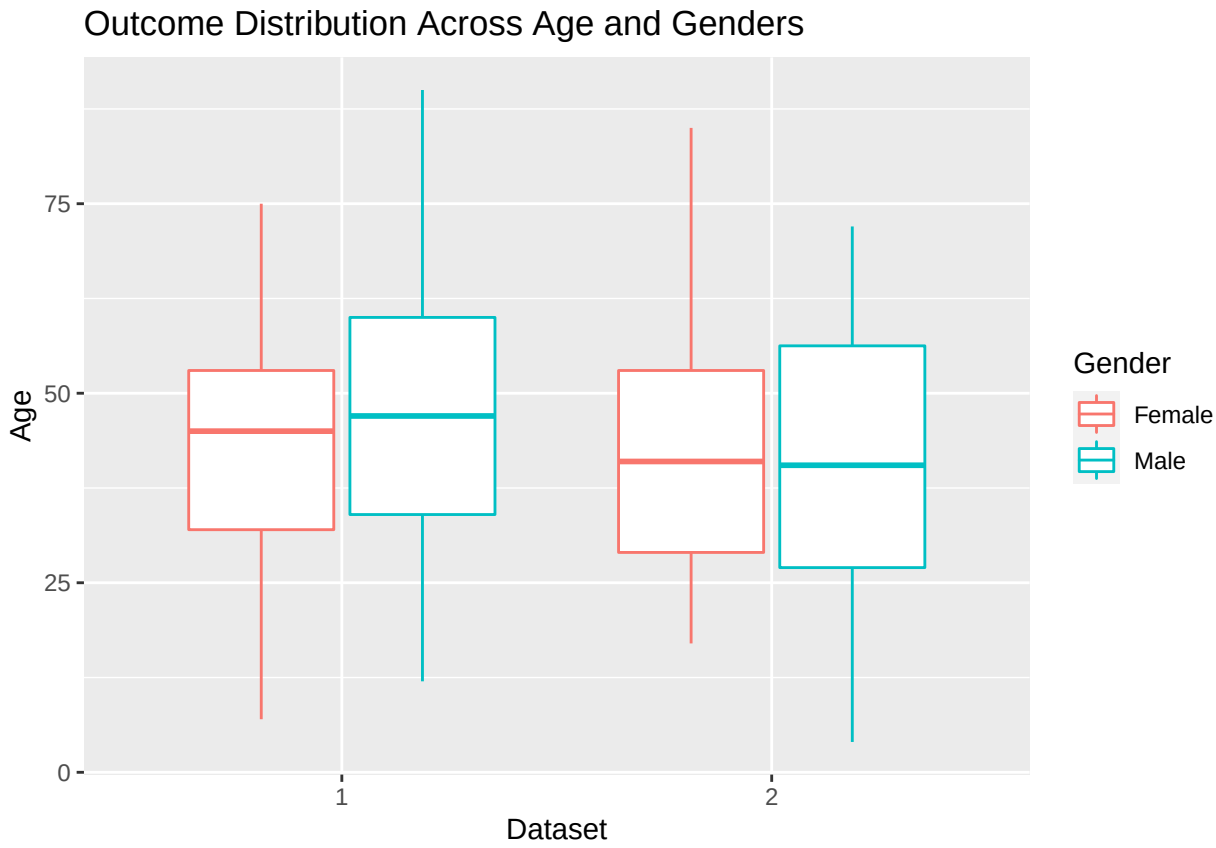
Plot the outcome distribution ("1" = liver disease or "2" = no liver disease) across age.

```
ggplot(indian_liver_patient, aes(x = Dataset, y = Age, group = Dataset)) +
   geom_boxplot(color = "blue", fill = "red") +
   ylab("Age") +
   ggtitle("Outcome Distribution Across Age")
```



Plot the outcome distribution ("1" = liver disease or "2" = no liver disease) across age and genders.

```
indian_liver_patient %>% group_by(Dataset, Gender) %>%
  ggplot(aes(x = Dataset, y = Age)) +
  geom_boxplot(aes(color = Gender)) +
  ylab("Age") +
  ggtitle("Outcome Distribution Across Age and Genders")
```

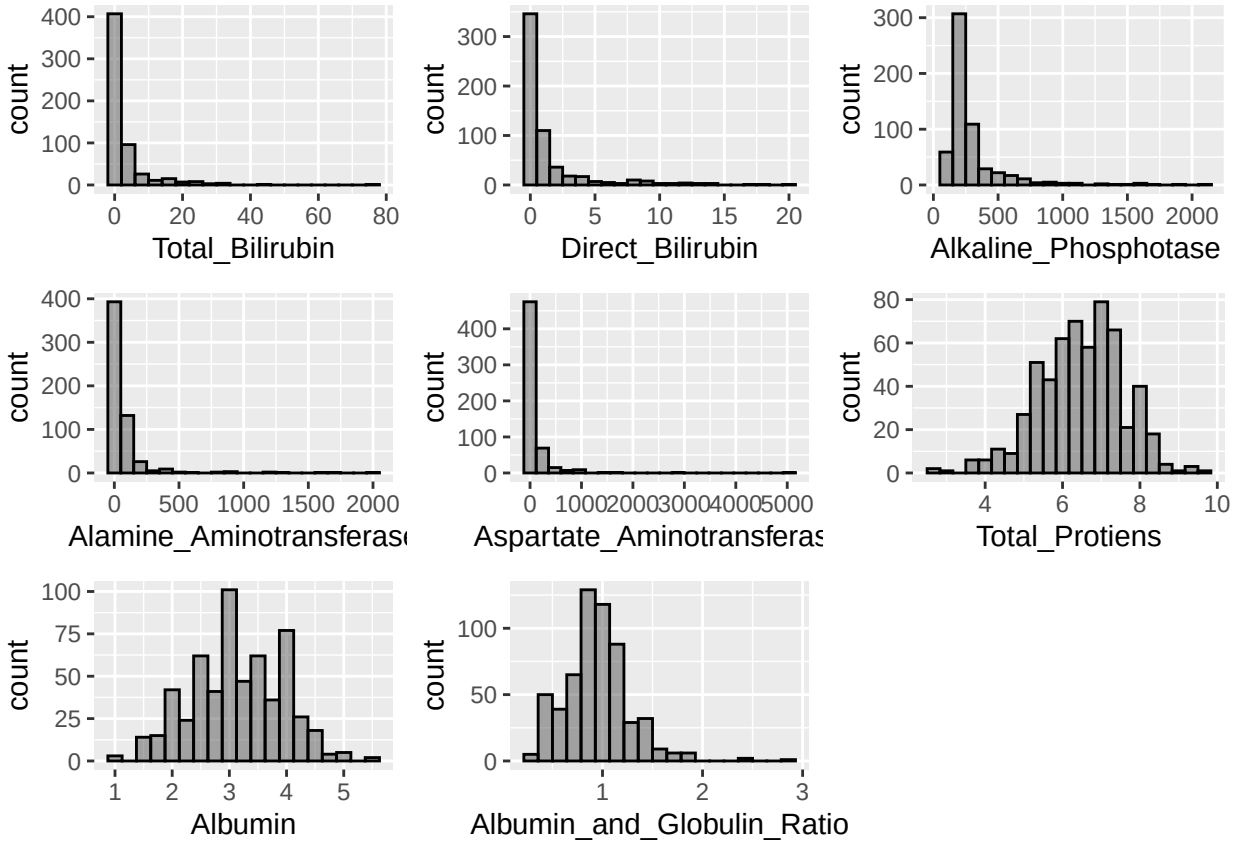


The age range of the patients in the dataset is 4 to 90, with a median age of 45. There are 439 males and 140 females in the dataset. The age variable is fairly normally distributed for both genders. There is a slightly higher mean for age in the Dataset = 1 (liver disease) group than in the Dataset = 2 (no liver disease) group. Also, there is a higher mean age for males in the Dataset = 1 group than for females, but there is no significant difference between the mean ages for the genders in the Dataset = 2 group. Overall, there were 414 positive cases of liver disease and 165 negative cases of liver disease in this dataset.

Plot the frequency distributions (histograms) of the remaining predictors.

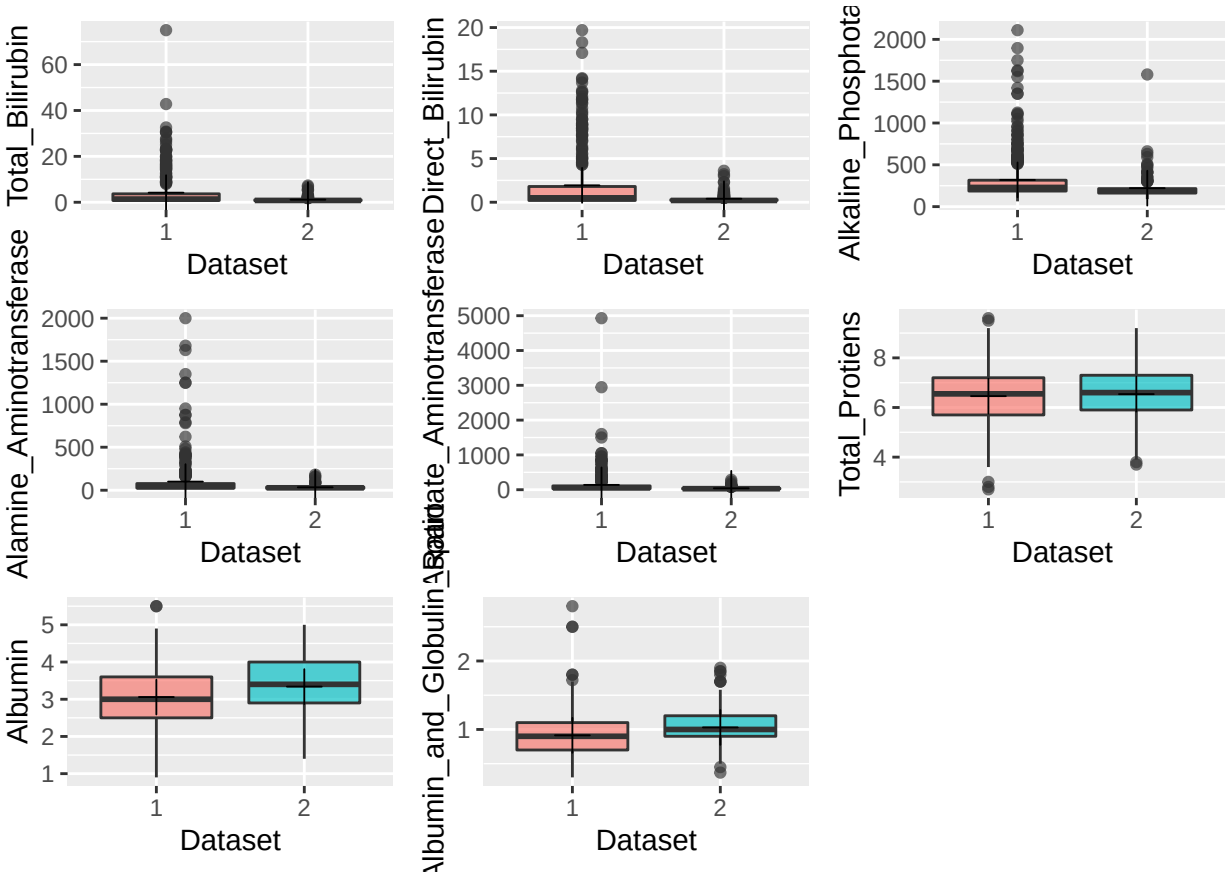
```
ph1 <- ggplot(indian_liver_patient, aes(x=Total_Bilirubin)) +
    geom_histogram(binwidth=4, colour="black", alpha=.5)

ph2 <- ggplot(indian_liver_patient, aes(x=Direct_Bilirubin)) +
    geom_histogram(binwidth=1, colour="black", alpha=.5)

ph3 <- ggplot(indian_liver_patient, aes(x=Alkaline_Phosphotase)) +
    geom_histogram(binwidth=100, colour="black", alpha=.5)

ph4 <- ggplot(indian_liver_patient, aes(x=Alamine_Aminotransferase)) +
    geom_histogram(binwidth=100, colour="black", alpha=.5)

ph5 <- ggplot(indian_liver_patient, aes(x=Aspartate_Aminotransferase)) + geom_histogram(binwidth=240, c

ph6 <- ggplot(indian_liver_patient, aes(x=Total_Protiens)) +
    geom_histogram(binwidth=1/3, colour="black", alpha=.5)

ph7 <- ggplot(indian_liver_patient, aes(x=Albumin)) +
    geom_histogram(binwidth=1/4, colour="black", alpha=.5)
```

```
ph8 <- ggplot(indian_liver_patient, aes(x=Albumin_and_Globulin_Ratio)) + geom_histogram(binwidth=1/7, co

grid.arrange(ph1, ph2, ph3, ph4, ph5, ph6, ph7, ph8, ncol=3)
```



Plot the outcome distributions (boxplots) of the remaining predictors.

```
pb1 <- ggplot(indian_liver_patient, aes(Dataset, Total_Bilirubin)) + geom_boxplot(aes(fill = Dataset), a

pb2 <- ggplot(indian_liver_patient, aes(Dataset, Direct_Bilirubin)) + geom_boxplot(aes(fill = Dataset),

pb3 <- ggplot(indian_liver_patient, aes(Dataset, Alkaline_Phosphotase)) + geom_boxplot(aes(fill = Datas

pb4 <- ggplot(indian_liver_patient, aes(Dataset, Alamine_Aminotransferase)) + geom_boxplot(aes(fill = Da

pb5 <- ggplot(indian_liver_patient, aes(Dataset, Aspartate_Aminotransferase)) + geom_boxplot(aes(fill =

pb6 <- ggplot(indian_liver_patient, aes(Dataset, Total_Protiens)) + geom_boxplot(aes(fill = Dataset), al

pb7 <- ggplot(indian_liver_patient, aes(Dataset, Albumin)) + geom_boxplot(aes(fill = Dataset), alpha = 1

pb8 <- ggplot(indian_liver_patient, aes(Dataset, Albumin_and_Globulin_Ratio)) + geom_boxplot(aes(fill =

grid.arrange(pb1, pb2, pb3, pb4, pb5, pb6, pb7, pb8, ncol=3)
```



These plots illustrate that some of the variables have skewed distributions, and some of them appear to be better predictors of liver disease than others.

2.3.2 2. Model Development

As mentioned earlier, before we began testing machine learning algorithms, we first divided the Indian Liver Patient Records dataset into two subsets: "modeling" and "validation." The modeling set would include 90% of the data, and the validation set would include the remaining 10%.

Split the dataset into "modeling" and "validation" subsets.

```
dat <- as.data.frame(indian_liver_patient)
dat <-  na.omit(dat)
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = dat$Dataset, times = 1, p = 0.1, list = FALSE)
modeling <- dat[-test_index,]
validation <- dat[test_index,]
```

The next step was to further split the modeling set into training and testing subsets (90%/10% again) in order to implement and evaluate the proposed algorithms. This secondary split ensured that we did not overtrain or overfit the data because the validation set remained off limits until the conclusion of the analysis.

Divide the modeling data into train and test sets in order to explore different training models.

```
set.seed(1, sample.kind="Rounding")
test_index2 <- createDataPartition(y = modeling$Dataset, times = 1, p = 0.1, list = FALSE)
```

```
train_set <- modeling[-test_index2,]
test_set <- modeling[test_index2,]
```

2.3.2.1 Logistic Regression Model

The first algorithm we tested on the modeling data was Logistic Regression (LR). We fitted the LR model to the training set, examined the most important predictors, and then used the model to make predictions on the testing set.

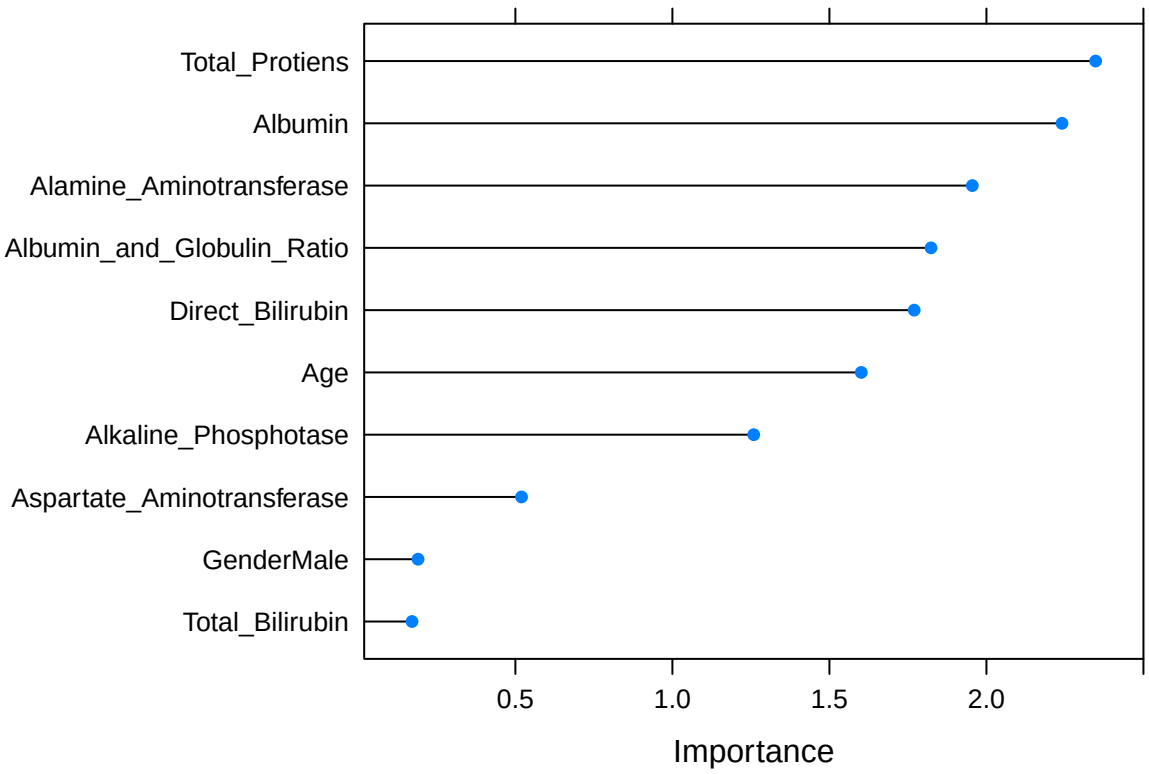
```
modelFit_LR = train(Dataset ~ ., data = train_set, method = "glm", family = "binomial", na.action=na.om
modelFit_LR
```

```
## Generalized Linear Model
##
## 467 samples
##  10 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 467, 467, 467, 467, 467, 467, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.705191  0.1279335
```

```
LR_imp = varImp(modelFit_LR, scale = FALSE)
LR_imp
```

```
## glm variable importance
##
##                            Overall
## Total_Protiens              2.3477
## Albumin                     2.2407
## Alamine_Aminotransferase    1.9550
## Albumin_and_Globulin_Ratio  1.8236
## Direct_Bilirubin            1.7700
## Age                         1.6014
## Alkaline_Phosphotase        1.2590
## Aspartate_Aminotransferase  0.5195
## GenderMale                  0.1900
## Total_Bilirubin             0.1708
```

```
plot(LR_imp)
```




```
pred_LR = predict(modelFit_LR, test_set)
confusionMatrix(pred_LR, test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 37 11
##          2  1  4
##
##                Accuracy : 0.7736
##                  95% CI : (0.6379, 0.8772)
##     No Information Rate : 0.717
##     P-Value [Acc > NIR] : 0.225885
##
##                   Kappa : 0.3011
##
##  Mcnemar's Test P-Value : 0.009375
##
##             Sensitivity : 0.9737
##             Specificity : 0.2667
##          Pos Pred Value : 0.7708
##          Neg Pred Value : 0.8000
##              Prevalence : 0.7170
##          Detection Rate : 0.6981
##    Detection Prevalence : 0.9057
##       Balanced Accuracy : 0.6202
##
##        'Positive' Class : 1
```

```
##
```

The Logistic Regression model predicted the presence of liver disease fairly well with an accuracy of 0.7736. The two most important predictors were Total Proteins and Albumin.

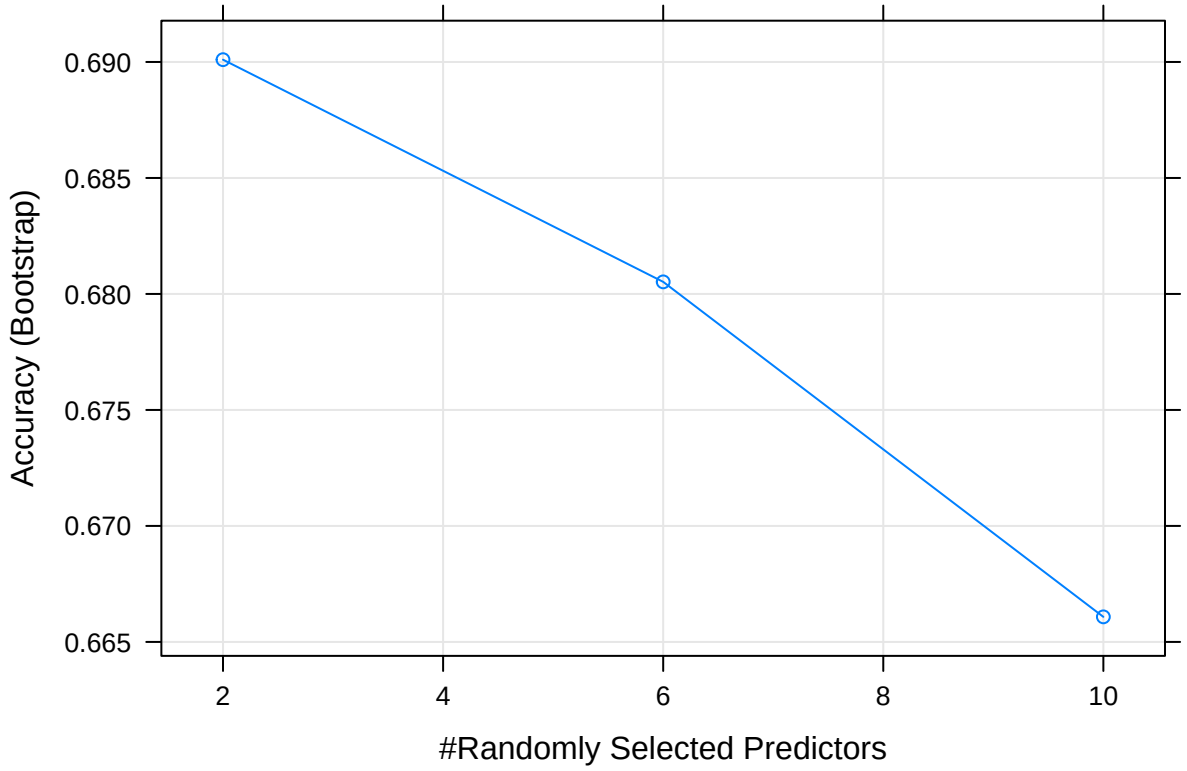
2.3.2.2 Random Forest Model

The second algorithm we tested on the modeling data was Random Forest (RF). We fitted the RF model to the training set, examined the most important predictors, and then used the model to make predictions on the testing set.

```
modelFit_RF = train(Dataset ~ ., method = "rf", data = train_set, prox = TRUE, na.action=na.omit)
modelFit_RF
```

```
## Random Forest
##
## 467 samples
##  10 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 467, 467, 467, 467, 467, 467, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.6900989  0.10680160
##    6    0.6805207  0.11297226
##   10    0.6660802  0.08822685
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

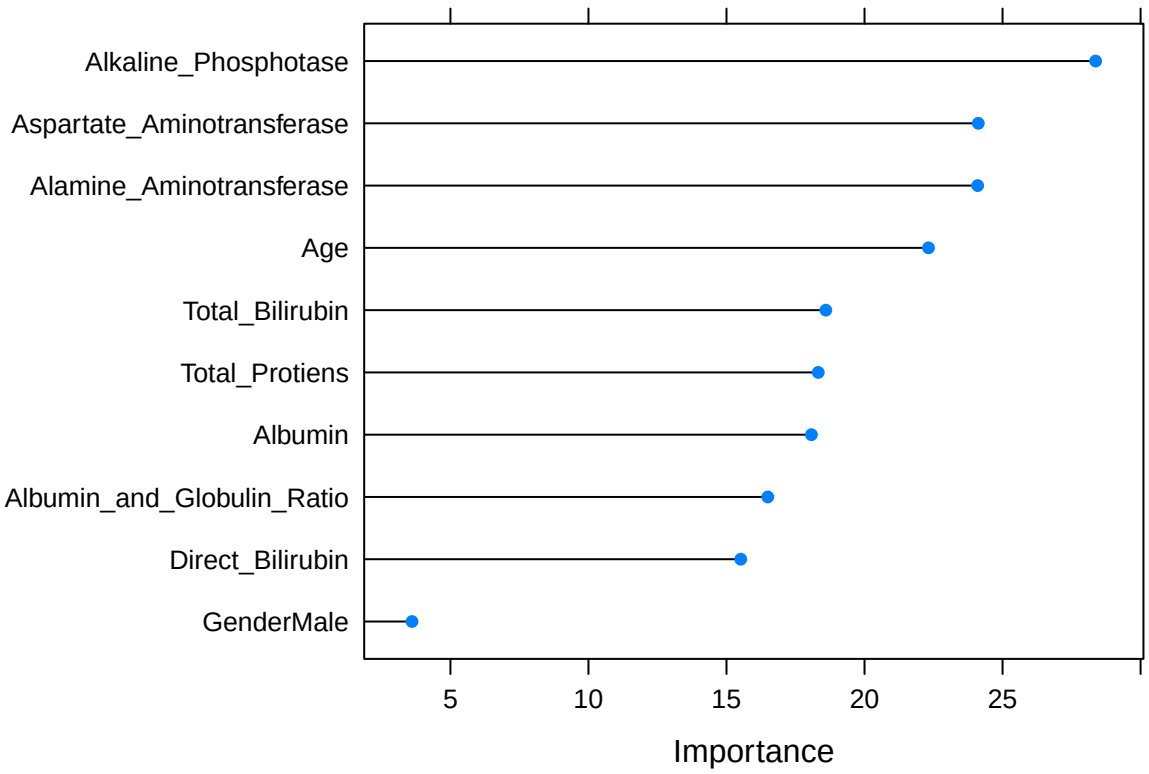
```
plot(modelFit_RF)
```




```
RF_imp = varImp(modelFit_RF, scale = FALSE)
RF_imp
```

```
## rf variable importance
##
##                            Overall
## Alkaline_Phosphotase         28.37
## Aspartate_Aminotransferase   24.12
## Alamine_Aminotransferase     24.10
## Age                          22.32
## Total_Bilirubin              18.60
## Total_Protiens               18.32
## Albumin                      18.08
## Albumin_and_Globulin_Ratio   16.50
## Direct_Bilirubin             15.52
## GenderMale                    3.61
```

```
plot(RF_imp)
```




```
pred_RF = predict(modelFit_RF, test_set)
confusionMatrix(pred_RF, test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 36  9
##          2  2  6
##
##                Accuracy : 0.7925
##                  95% CI : (0.6589, 0.8916)
##     No Information Rate : 0.717
##     P-Value [Acc > NIR] : 0.14199
##
##                   Kappa : 0.4045
##
##  Mcnemar's Test P-Value : 0.07044
##
##             Sensitivity : 0.9474
##             Specificity : 0.4000
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.7500
##              Prevalence : 0.7170
##          Detection Rate : 0.6792
##    Detection Prevalence : 0.8491
##       Balanced Accuracy : 0.6737
##
##        'Positive' Class : 1
```

```
##
```

The Random Forest model predicted the presence of liver disease better than LR with an accuracy of 0.7925. This accuracy was optimized by using just two variables. The two most important predictors were Alkaline Phosphotase and Alamine Aminotransferase, while Aspartate Aminotransferase and Age followed close behind.

2.3.2.3 Naive Bayes Model

The third algorithm we tested on the modeling data was Naive Bayes (NB). We fitted the NB model to the training set, and then used the model to make predictions on the testing set.

```
modelFit_nb = train(Dataset ~., "nb", data = train_set, trControl = trainControl(method ="cv", number =
modelFit_nb
```

```
## Naive Bayes
##
## 467 samples
##  10 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 421, 420, 420, 420, 420, 421, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.5480188  0.2360884
##    TRUE      0.6234120  0.2645224
##
## Tuning parameter 'fL' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 0, usekernel = TRUE and adjust
##  = 1.
```

```
pred_nb <- predict(modelFit_nb, test_set)
confusionMatrix(pred_nb, test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 25  3
##          2 13 12
##
##                Accuracy : 0.6981
##                  95% CI : (0.5566, 0.8166)
##     No Information Rate : 0.717
##     P-Value [Acc > NIR] : 0.68238
##
```

```
##                  Kappa : 0.381
##
##  Mcnemar's Test P-Value : 0.02445
##
##             Sensitivity : 0.6579
##             Specificity : 0.8000
##          Pos Pred Value : 0.8929
##          Neg Pred Value : 0.4800
##              Prevalence : 0.7170
##          Detection Rate : 0.4717
##    Detection Prevalence : 0.5283
##       Balanced Accuracy : 0.7289
##
##        'Positive' Class : 1
##
```

The Naive Bayes model predicted the presence of liver disease not very well with an accuracy of only 0.6981.

2.3.2.4 Linear Discriminant Analysis Model

The fourth algorithm we tested on the modeling data was Linear Discriminant Analysis (LDA). We fitted the LDA model to the training set, and then used the model to make predictions on the testing set.

```
modelFit_lda <- train(Dataset ~ ., data = train_set, method = "lda", na.action=na.omit)
modelFit_lda
```

```
## Linear Discriminant Analysis
##
## 467 samples
##  10 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 467, 467, 467, 467, 467, 467, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6960434  0.002921524
```

```
pred_lda <- predict(modelFit_lda, test_set)
confusionMatrix(pred_lda, test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 38 14
##          2  0  1
##
##                Accuracy : 0.7358
##                  95% CI : (0.5967, 0.8474)
```

```
##     No Information Rate : 0.717
##     P-Value [Acc > NIR] : 0.448136
##
##                   Kappa : 0.0929
##
##  Mcnemar's Test P-Value : 0.000512
##
##             Sensitivity : 1.00000
##             Specificity : 0.06667
##          Pos Pred Value : 0.73077
##          Neg Pred Value : 1.00000
##              Prevalence : 0.71698
##          Detection Rate : 0.71698
##    Detection Prevalence : 0.98113
##       Balanced Accuracy : 0.53333
##
##        'Positive' Class : 1
##
```

The Linear Discriminant Analysis model predicted the presence of liver disease better than NB, but not as well as LR or RF, with an accuracy of only 0.7358.

2.3.2.5 K-Nearest Neighbors Model

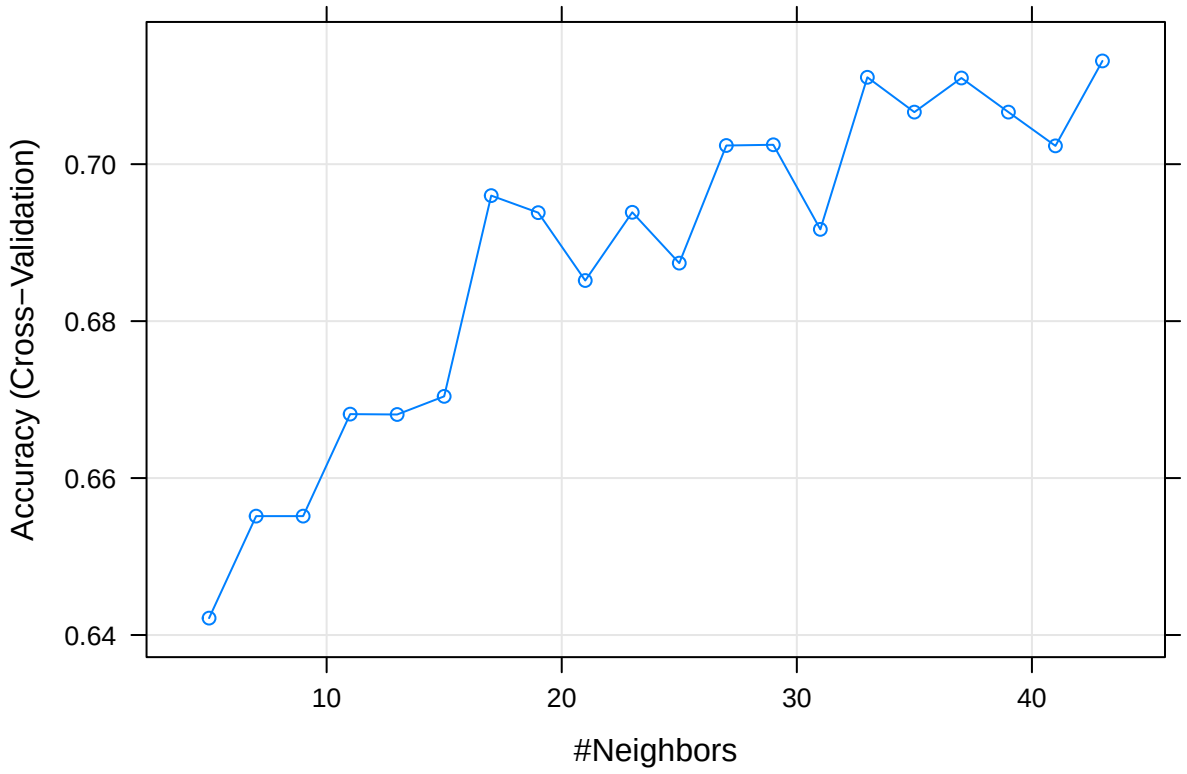
The fifth algorithm we tested on the modeling data was K-Nearest Neighbors (KNN). We fitted the KNN model to the training set, and then used the model to make predictions on the testing set.

```
modelFit_KNN <- train(
  Dataset ~ ., data = train_set, method = "knn",
  trControl = trainControl("cv", number = 10),
  preProcess = c("center","scale"),
  tuneLength = 20
  )
modelFit_KNN
```

```
## k-Nearest Neighbors
##
## 467 samples
##  10 predictor
##   2 classes: '1', '2'
##
## Pre-processing: centered (10), scaled (10)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 420, 421, 421, 419, 420, 421, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.6421523   0.060714722
##    7  0.6551496   0.035222515
##    9  0.6551496   0.023845974
##   11  0.6681487   0.039715291
##   13  0.6681025   0.050503873
##   15  0.6704113   0.049341488
```

```
##    17  0.6959952   0.096871419
##    19  0.6938232   0.067932403
##    21  0.6851738   0.039695911
##    23  0.6938676   0.056730712
##    25  0.6873921   0.030244413
##    27  0.7023801   0.046065493
##    29  0.7024726   0.041223403
##    31  0.6916936  -0.008030672
##    33  0.7110739   0.034045280
##    35  0.7066335   0.027301418
##    37  0.7109813   0.028420285
##    39  0.7066335   0.008068923
##    41  0.7023319  -0.006060124
##    43  0.7131553   0.014872865
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 43.
```

```
plot(modelFit_KNN)
```




```
pred_KNN <- predict(modelFit_KNN, test_set)
confusionMatrix(pred_KNN, test_set$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 38 14
```

```
##          2  0  1
##
##                Accuracy : 0.7358
##                  95% CI : (0.5967, 0.8474)
##     No Information Rate : 0.717
##     P-Value [Acc > NIR] : 0.448136
##
##                   Kappa : 0.0929
##
##  Mcnemar's Test P-Value : 0.000512
##
##             Sensitivity : 1.00000
##             Specificity : 0.06667
##          Pos Pred Value : 0.73077
##          Neg Pred Value : 1.00000
##              Prevalence : 0.71698
##          Detection Rate : 0.71698
##    Detection Prevalence : 0.98113
##       Balanced Accuracy : 0.53333
##
##        'Positive' Class : 1
##
```

The K-Nearest Neighbors model predicted the presence of liver disease better than NB and LDA, but not as well as LR or RF, with a fairly good accuracy of 0.7547.

3 Results

After training and analyzing five different models of machine learning algorithms, we determined that the best and final model for prediction was the Random Forest Model.

```
summaryTable <- data.frame("Model" = c("Naive Bayes", "Linear Discriminant Analysis", "K-Nearest Neighb
 "Accuracy" = c(0.6981, 0.7358, 0.7547, 0.7736, 0.7925))
summaryTable
```

```
##                          Model Accuracy
## 1                  Naive Bayes   0.6981
## 2 Linear Discriminant Analysis   0.7358
## 3          K-Nearest Neighbors   0.7547
## 4          Logistic Regression   0.7736
## 5                Random Forest   0.7925
```

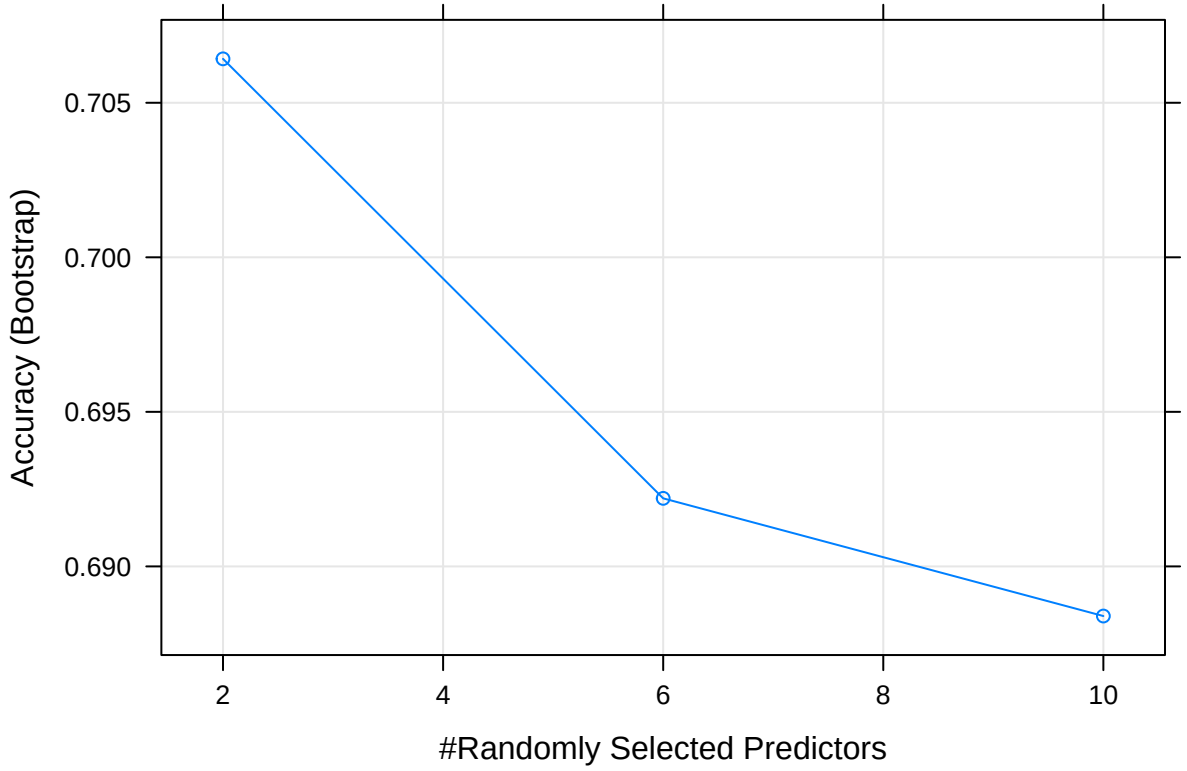
Finally, we returned to the modeling and validation sets created at the beginning of this project and tested the final Random Forest model using those original data subsets.

```
FinalmodelFit_RF = train(Dataset ~ ., method = "rf", data = modeling, prox = TRUE, na.action=na.omit)
FinalmodelFit_RF
```

```
## Random Forest
##
## 520 samples
```

```
##  10 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 520, 520, 520, 520, 520, 520, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.7064209  0.1633366
##    6    0.6922015  0.1508938
##   10    0.6883939  0.1473674
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
plot(FinalmodelFit_RF)
```

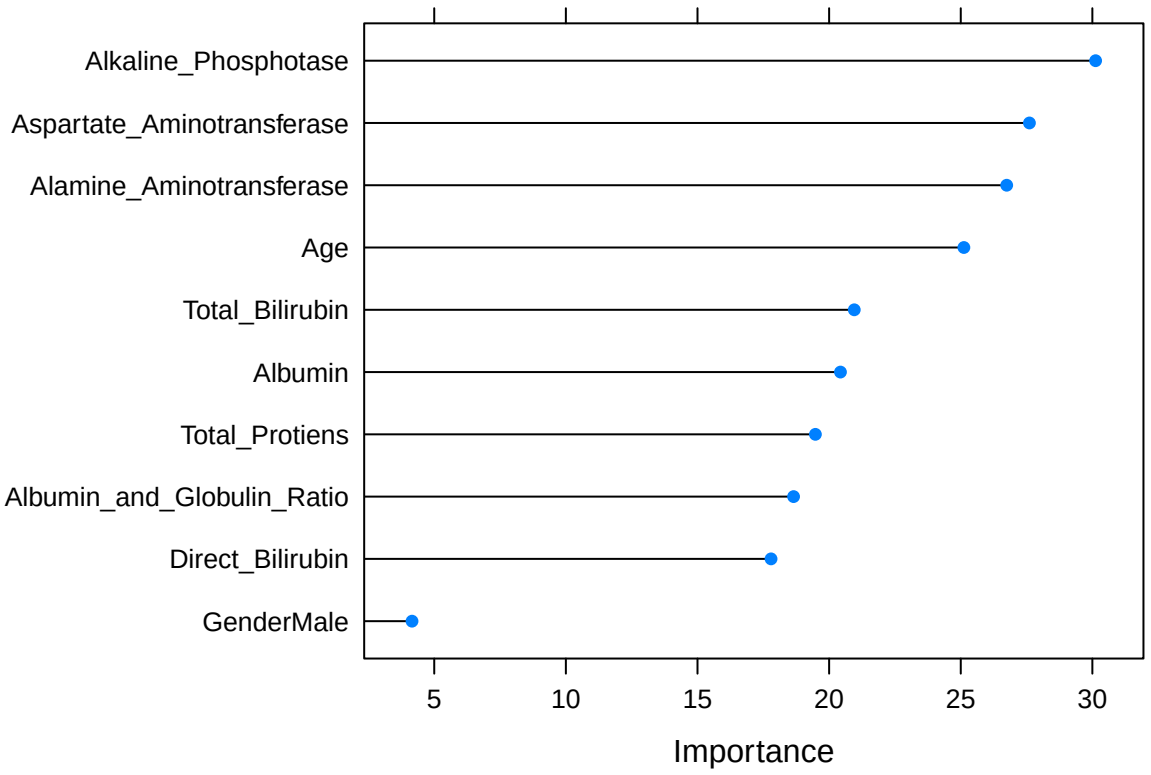



```
FinalmodelRF_imp = varImp(FinalmodelFit_RF, scale = FALSE)
FinalmodelRF_imp
```

```
## rf variable importance
##
##                            Overall
## Alkaline_Phosphotase        30.123
## Aspartate_Aminotransferase  27.609
## Alamine_Aminotransferase    26.746
## Age                         25.119
```

```
## Total_Bilirubin             20.956
## Albumin                     20.431
## Total_Protiens              19.479
## Albumin_and_Globulin_Ratio  18.650
## Direct_Bilirubin            17.794
## GenderMale                   4.156
```

```
plot(FinalmodelRF_imp)
```




```
Finalmodelpred_RF = predict(FinalmodelFit_RF, validation)
confusionMatrix(Finalmodelpred_RF, validation$Dataset)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 35 10
##          2  7  7
##
##                Accuracy : 0.7119
##                  95% CI : (0.5792, 0.8224)
##     No Information Rate : 0.7119
##     P-Value [Acc > NIR] : 0.5650
##
##                   Kappa : 0.2587
##
##  Mcnemar's Test P-Value : 0.6276
##
```

```
##             Sensitivity : 0.8333
##             Specificity : 0.4118
##          Pos Pred Value : 0.7778
##          Neg Pred Value : 0.5000
##              Prevalence : 0.7119
##          Detection Rate : 0.5932
##    Detection Prevalence : 0.7627
##       Balanced Accuracy : 0.6225
##
##        'Positive' Class : 1
##
```

The prediction model that we built using the Random Forest machine learning algorithm predicted the outcome of liver disease in the validation set with an accuracy of 0.7119. The most important predictors were determined to be Alkaline Phosphotase and Aspartate Aminotransferase. The RF model also had a better balance of sensitivity and specificity than the other models.

4 Conclusion

The goal of this project was to develop multiple machine learning models and then determine which one gives the best prediction results (i.e. the highest overall accuracy). We approached this task by implementing and examining the results of five different machine learning algorithms on a subset of the Indian Liver Patient Records dataset. After testing the five different methods of Logistic Regression, Naive Bayes, Linear Discriminant Analysis, K-Nearest Neighbors, and Random Forest, we concluded that the Random Forest model elicited the best predictive accuracy.

Of the five models we developed, the Random Forest Model was determined to be the best model overall. When tested on the validation set at the end of the analysis, the Random Forest Model produced an accuracy of 0.7119, which is quite effective. It also had the best balance of sensitivity and specificity of all the models tested.

Even though we developed an efficient predictive algorithm, there are still many ways to improve upon our final model. Transforming the data through scaling, standardization, or removing highly-correlated variables could make a big difference in the results. Furthermore, the accuracy of the predictions could be increased through the utilization of other machine learning techniques, both supervised and unsupervised, such as K-Means Clustering, Classification and Regression Trees (CART), Principle Component Analysis (PCA), or ensemble methods. Applying different types of algorithms to this data could reveal further insights to enhance the predictive efficacy.

5 References

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.