Data-driven modelling for predicting liver disease

Pravik Solanki

06/11/2020

Contents

1 Introduction

Liver disease is a significant driver of morbidity and mortality worldwide. To assist in diagnosis and screening, we will produce data-driven models that predict the presence of liver disease. The dataset contains demographic data and blood test results from 416 liver disease patients and 167 non liver disease patients (n=583 total) in Andhra Pradesh, India. This dataset is publicly accessible from Kaggle at https://www.kaggle.com/uciml/indian-liver-patient-records, and originates from the University of California Machine Learning Repository (Lichman M., 2013).

1.1 Objective

The objective of this project is to identify the best model(s) that can predict the presence of liver disease from demographic data and blood test results. We will use three different techniques (logistic regression, classification trees, and random forests), applied firstly to original data and secondly to log-transformed data, producing a total of six models.

1.2 Importing & preparing the dataset

We begin by loading relevant packages and importing the dataset into the R environment.

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

library(tidyverse)
library(data.table)
library(gridExtra)
library(caret)
library(rpart)
library(randomForest)

# Importing the liver disease dataset and checking column names
liver <- read.csv(file="https://storage.googleapis.com/kagglesdsdata/datasets/2607/4342/indian_liver_pa
names(liver)
```

```
##  [1] "Age"                        "Gender"
##  [3] "Total_Bilirubin"            "Direct_Bilirubin"
##  [5] "Alkaline_Phosphotase"       "Alamine_Aminotransferase"
##  [7] "Aspartate_Aminotransferase" "Total_Protiens"
##  [9] "Albumin"                    "Albumin_and_Globulin_Ratio"
## [11] "Dataset"
```

To make the dataset easy to work with, we rename variables based on commonly-used medical shorthand notation, and recode our outcome variable 'disease' as 0 (no liver disease) or 1 (liver disease).

```r
# Renaming variables based on medical shorthand notation
colnames(liver) <- c("age", "sex", "t_bilirubin", "d_bilirubin",
                     "ALP", "ALT", "AST", "protein", "albumin", "ag_ratio", "disease")
liver$sex[liver$sex=="Male"] <- "M"
liver$sex[liver$sex=="Female"] <- "F"

# Recoding 'disease' into '0' (no liver disease) and 1 (liver disease)
liver$disease[liver$disease==2] <- 0
```

2 Methods

2.1 Exploratory Data Analysis (EDA)

We seek to visualise our dataset, assess the distribution of variables, and compare them between those who have liver disease and those who do not. This will indicate whether this dataset is appropriate for the purposes of predictive modelling.

We begin with a statistical overview of our dataset.

```
# Overview of dataset
summary(liver)
```

```
##       age             sex             t_bilirubin       d_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
##
##       ALP              ALT               AST              protein
##  Min.   :  63.0   Min.   :  10.00   Min.   :  10.0   Min.   :2.700
##  1st Qu.: 175.5   1st Qu.:  23.00   1st Qu.:  25.0   1st Qu.:5.800
##  Median : 208.0   Median :  35.00   Median :  42.0   Median :6.600
##  Mean   : 290.6   Mean   :  80.71   Mean   : 109.9   Mean   :6.483
##  3rd Qu.: 298.0   3rd Qu.:  60.50   3rd Qu.:  87.0   3rd Qu.:7.200
##  Max.   :2110.0   Max.   :2000.00   Max.   :4929.0   Max.   :9.600
##
##     albumin          ag_ratio          disease
##  Min.   :0.900   Min.   :0.3000   Min.   :0.0000
##  1st Qu.:2.600   1st Qu.:0.7000   1st Qu.:0.0000
##  Median :3.100   Median :0.9300   Median :1.0000
##  Mean   :3.142   Mean   :0.9471   Mean   :0.7136
##  3rd Qu.:3.800   3rd Qu.:1.1000   3rd Qu.:1.0000
##  Max.   :5.500   Max.   :2.8000   Max.   :1.0000
##                  NA's   :4
```

```
# Binary variables
table(liver$sex)
```

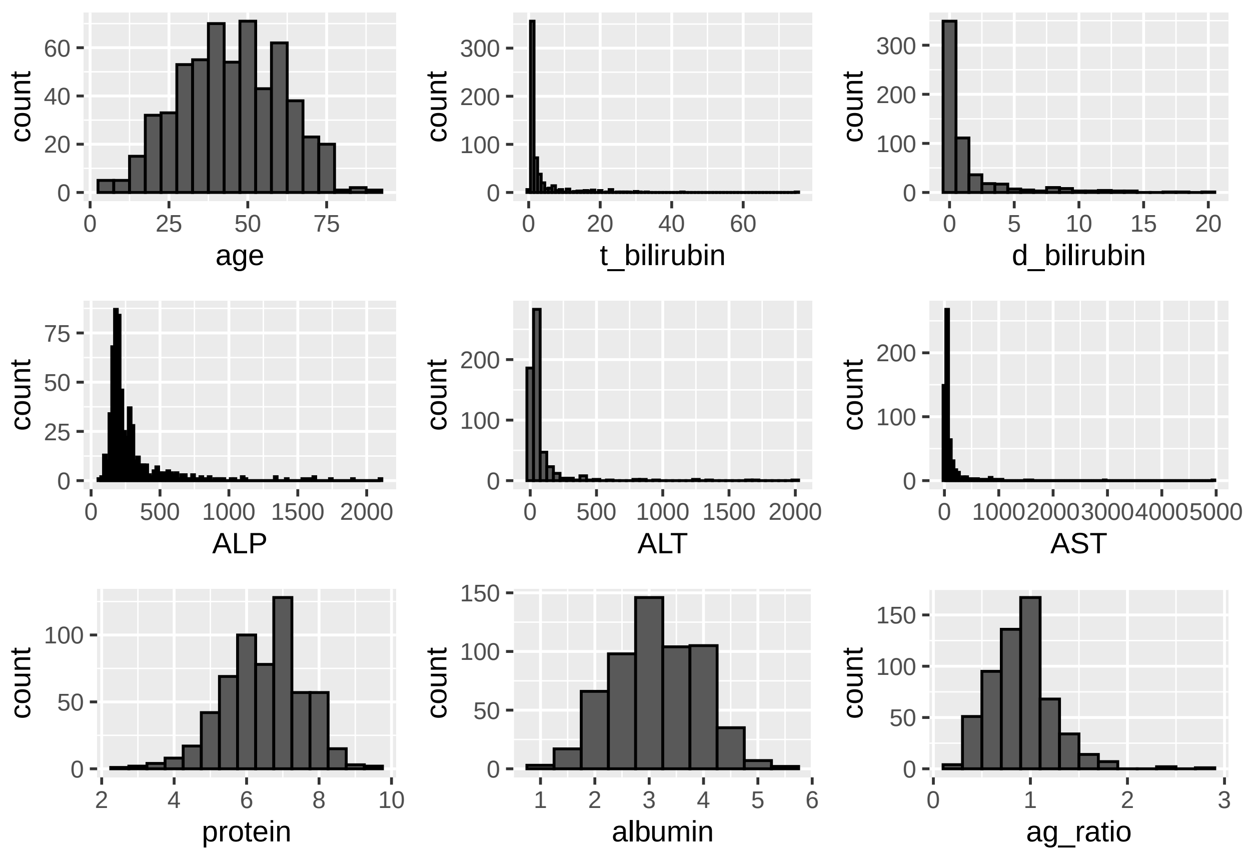
```
##
##   F   M
## 142 441
```

```
table(liver$disease)
```

```
##
##   0   1
## 167 416
```

We now visualise variables to better appreciate their distribution.

```r
# Histograms of all other variables
graph1 <- ggplot(liver, aes(x=age)) +
  geom_histogram(binwidth=5, colour="black")

graph2 <- ggplot(liver, aes(x=t_bilirubin)) +
  geom_histogram(binwidth=1, colour="black")

graph3 <- ggplot(liver, aes(x=d_bilirubin)) +
  geom_histogram(binwidth=1, colour="black")

graph4 <- ggplot(liver, aes(x=ALP)) +
  geom_histogram(binwidth=20, colour="black")

graph5 <- ggplot(liver, aes(x=ALT)) +
  geom_histogram(binwidth=50, colour="black")

graph6 <- ggplot(liver, aes(x=AST)) +
  geom_histogram(binwidth=50, colour="black")

graph7 <- ggplot(liver, aes(x=protein)) +
  geom_histogram(binwidth=0.5, colour="black")

graph8 <- ggplot(liver, aes(x=albumin)) +
  geom_histogram(binwidth=0.5, colour="black")

graph9 <- ggplot(liver, aes(x=ag_ratio)) +
  geom_histogram(binwidth=0.2, colour="black")

grid.arrange(graph1, graph2, graph3, graph4, graph5, graph6, graph7,
             graph8, graph9, ncol=3)
```



We find that many variables have outliers. We also find that some variables - namely total bilirubin, direct bilirubin, ALP, ALT and AST - are right skewed, which could influence the accuracy of models if these variables are used as predictors directly. Hence, we create a ‘log’ version of our dataset, in which we apply the natural logarithm to these variables to make their distribution less skewed. Some patients have 0 for their bilirubin results (a value that cannot be log-transformed); to bypass this issue, +1 is applied across all bilirubin results prior to their transformation.

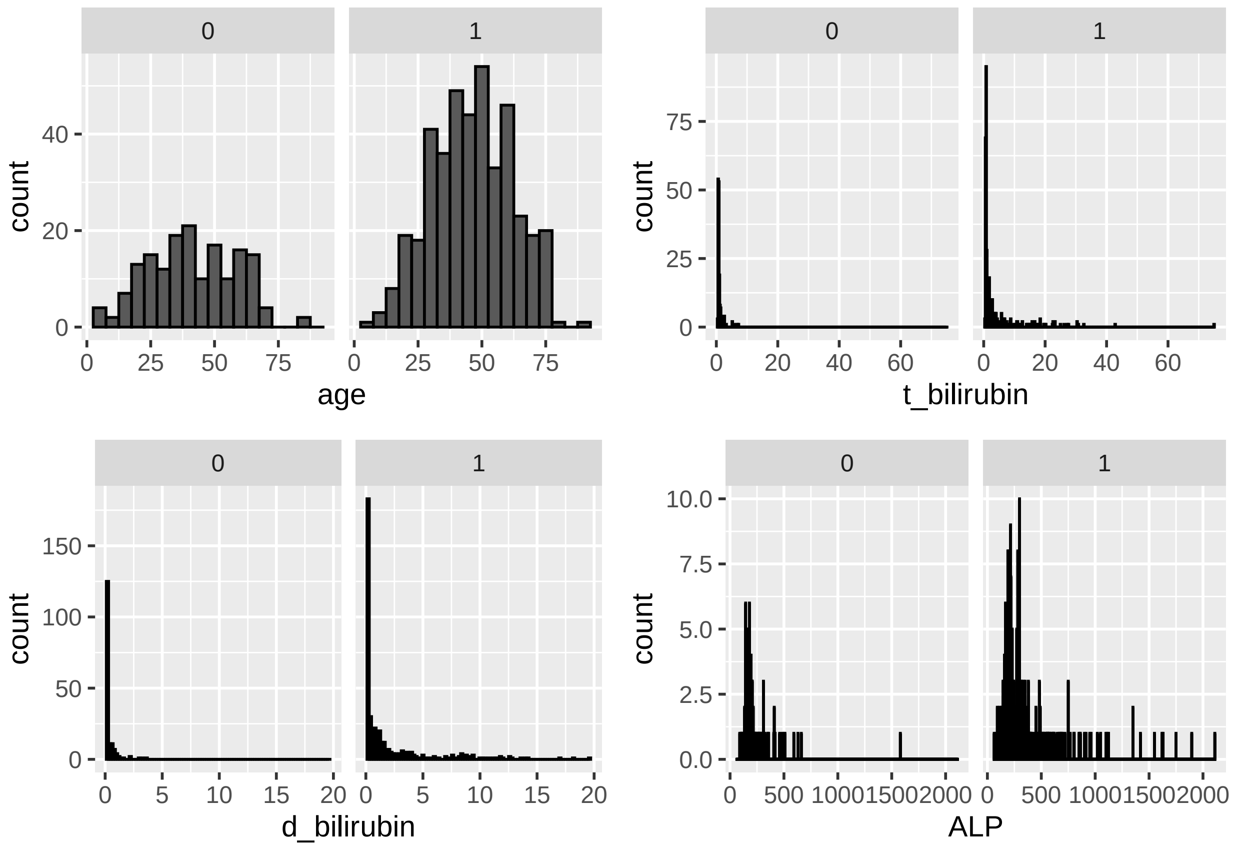
```
liver_log <- liver
liver_log$t_bilirubin <- log(1+ liver_log$t_bilirubin)
liver_log$d_bilirubin <- log(1+ liver_log$d_bilirubin)
liver_log$ALP <- log(liver_log$ALP)
liver_log$ALT <- log(liver_log$ALT)
liver_log$AST <- log(liver_log$AST)
```

This log dataset will not be explored further. However, we will later train all our models on both original and log-transformed data and assess for any differences in predictive performance.

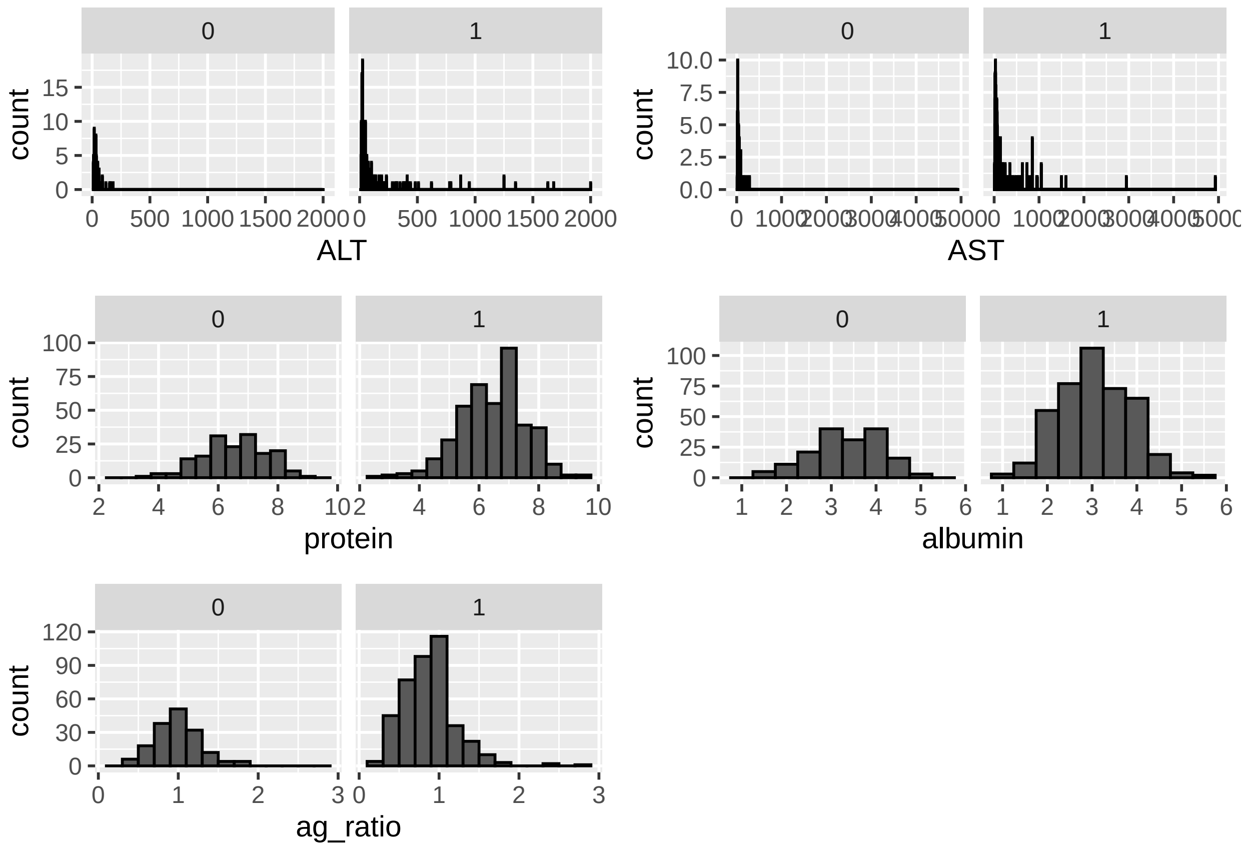
Returning to the original dataset, our next step is to assess for any direct differences between those without (0) and those with (1) liver disease.

```
graph1 <- ggplot(liver, aes(x=age)) +
  geom_histogram(binwidth=5, colour="black") +
  facet_grid(.~disease)

graph2 <- ggplot(liver, aes(x=t_bilirubin)) +
  geom_histogram(binwidth=0.2, colour="black") +
```

```r
  facet_grid(.~disease)

graph3 <- ggplot(liver, aes(x=d_bilirubin)) +
  geom_histogram(binwidth=0.2, colour="black") +
  facet_grid(.~disease)

graph4 <- ggplot(liver, aes(x=ALP)) +
  geom_histogram(binwidth=0.5, colour="black") +
  facet_grid(.~disease)

graph5 <- ggplot(liver, aes(x=ALT)) +
  geom_histogram(binwidth=0.5, colour="black") +
  facet_grid(.~disease)

graph6 <- ggplot(liver, aes(x=AST)) +
  geom_histogram(binwidth=0.5, colour="black") +
  facet_grid(.~disease)

graph7 <- ggplot(liver, aes(x=protein)) +
  geom_histogram(binwidth=0.5, colour="black") +
  facet_grid(.~disease)

graph8 <- ggplot(liver, aes(x=albumin)) +
  geom_histogram(binwidth=0.5, colour="black") +
  facet_grid(.~disease)

graph9 <- ggplot(liver, aes(x=ag_ratio)) +
  geom_histogram(binwidth=0.2, colour="black") +
  facet_grid(.~disease)

grid.arrange(graph1, graph2, graph3, graph4, ncol=2)
```



```r
grid.arrange(graph5, graph6, graph7, graph8, graph9, ncol=2)
```



There may be differences, but it is imprecise to assess this visually - we want to compare variables between the two groups with formal statistical tests. For our categorical variable sex, the appropriate test is the Chi square test. For other variables, the appropriate test is either the t-test (if the distribution follows a normal distribution) or the Mann-Whitney U test (if the distribution is non-parametric).

To assess the normality of variables, we apply the Shapiro-Wilk test:

```
shapiro.test(liver$age)
shapiro.test(liver$t_bilirubin)
shapiro.test(liver$d_bilirubin)
shapiro.test(liver$ALP)
shapiro.test(liver$ALT)
shapiro.test(liver$AST)
shapiro.test(liver$protein)
shapiro.test(liver$albumin)
shapiro.test(liver$ag_ratio)
```

Results are not displayed here (for ease of reading), but all variables differed significantly ($p<0.05$) to the normal distribution, indicating that they are non-parametric and should therefore be compared between groups with the Mann-Whitney U test. Variables are now compared between groups with appropriate tests:

```
liverdisease <- liver %>% filter(disease==1)
livernodisease <- liver %>% filter(disease==0)

wilcox.test(liverdisease$age, livernodisease$age)
wilcox.test(liverdisease$t_bilirubin, livernodisease$t_bilirubin)
wilcox.test(liverdisease$d_bilirubin, livernodisease$d_bilirubin)
wilcox.test(liverdisease$ALP, livernodisease$ALP)
wilcox.test(liverdisease$ALT, livernodisease$ALT)
wilcox.test(liverdisease$AST, livernodisease$AST)
wilcox.test(liverdisease$protein, livernodisease$protein)
wilcox.test(liverdisease$albumin, livernodisease$albumin)
wilcox.test(liverdisease$ag_ratio, livernodisease$ag_ratio)

table(liverdisease$sex)
table(livernodisease$sex)
chisq.test(as.table(matrix(c(92, 324, 50, 117),ncol=2)))
```

Results are not displayed here (for ease of reading), but we found all variables to differ significantly ($p<0.05$) between the two groups, except for protein ($p=0.44$) and sex ($p=0.06$). This gives us a high degree of confidence that a model sufficiently trained on all these predictor variables would be able to predict the presence of liver disease with reasonable accuracy.

2.2 Generating models

We now seek to build models that can predict the presence of liver disease. To do this fairly, we randomly partition our dataset into a training set (80% of patients) and a test set (20% of patients). We partition our log-transformed dataset in the same way.

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

index <- createDataPartition(y = liver$disease, times = 1, p = 0.2, list = FALSE)
train <- liver[-index,]
test <- liver[index,]

index <- createDataPartition(y = liver_log$disease, times = 1, p = 0.2, list = FALSE)
train_log <- liver_log[-index,]
test_log <- liver_log[index,]
```

We will use only the training set to generate our models, and reserve the test set as a means to evaluate the predictive accuracy of models once they have been built. We will generate models on both original and log-transformed data, but will discuss only the former in this section. Nonetheless, the predictive performance of both approaches (across all three model types) is evaluated and discussed further in this report.

2.2.1 Logistic regression

The simplest method for predicting a binary outcome using multiple predictors is multivariable logistic regression. We begin by creating logistic regression models on our training dataset (inputting original and log-transformed variables respectively).

```
logistic <- glm(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP + ALT + AST +
                  protein + albumin + ag_ratio, data=train, family=binomial)

summary(logistic)
```

```
##
## Call:
```

```
## glm(formula = disease ~ age + sex + t_bilirubin + d_bilirubin +
##     ALP + ALT + AST + protein + albumin + ag_ratio, family = binomial,
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2179  -1.0903   0.4075   0.9219   1.5565
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.1236846  1.4827409  -2.107   0.0351 *
## age          0.0163180  0.0072872   2.239   0.0251 *
## sexM         0.1893581  0.2544065   0.744   0.4567
## t_bilirubin -0.4516088  0.4176013  -1.081   0.2795
## d_bilirubin  1.2105139  0.8391076   1.443   0.1491
## ALP          0.0010245  0.0008296   1.235   0.2169
## ALT          0.0114886  0.0057760   1.989   0.0467 *
## AST          0.0044225  0.0040189   1.100   0.2712
## protein      0.8313627  0.4044830   2.055   0.0398 *
## albumin     -1.6127488  0.7967298  -2.024   0.0429 *
## ag_ratio     1.8382176  1.2442022   1.477   0.1396
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 559.96  on 461  degrees of freedom
## Residual deviance: 464.65  on 451  degrees of freedom
##   (4 observations deleted due to missingness)
## AIC: 486.65
##
## Number of Fisher Scoring iterations: 7
```

```r
exp(coef(logistic)) # Adjusted Odds ratios
```

```
## (Intercept)         age        sexM t_bilirubin d_bilirubin         ALP
##  0.04399477  1.01645190  1.20847360  0.63660319  3.35520853  1.00102500
##         ALT         AST     protein     albumin    ag_ratio
##  1.01155481  1.00443227  2.29644608  0.19933892  6.28532561
```

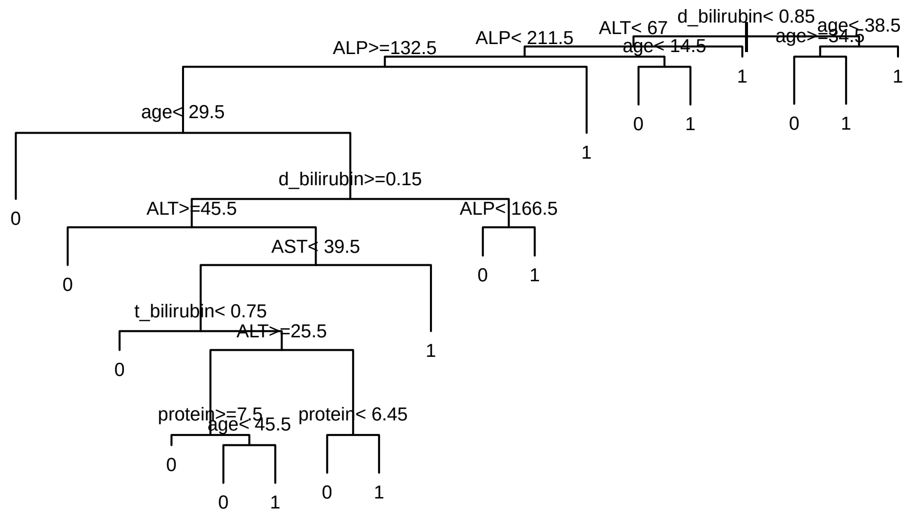
```r
# Log-transformed version
logistic_log <- glm(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP + ALT + AST +
                  protein + albumin + ag_ratio, data=train_log, family=binomial)
```

In this model, age, ALT, protein, and albumin have an independent, statistically significant association with liver disease. The adjusted odds ratios indicate that age and ALT have a mild positive association, protein has a moderate positive association, and albumin has a strong negative association with liver disease. The highest p-value was, by far, the p-value for sex, consistent with the lack of significant difference when comparing this variable directly between those with and without liver disease earlier. Despite protein having no appreciable difference earlier when compared directly between those with and without liver disease, it had a relatively strong independent association with liver disease in this model, since regression controls for the interfering effects of other variables.

2.2.2 Classification tree

An alternative method for predicting a binary outcome is a classification tree, a method that uses branching logic (spanning multiple decision nodes) to produce a final prediction.

```r
ctree <- rpart(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP + ALT + AST +
        protein + albumin + ag_ratio, data=train, method="class")

plot(ctree, margin=0.03)
text(ctree, cex=0.6)
```



```r
ctree$variable.importance # Importance of variables
```

```
## d_bilirubin t_bilirubin         ALT         age         ALP         AST
##  20.9650144  20.2717246  20.2099185  20.1114388  17.1500076  15.0181109
##     protein     albumin    ag_ratio         sex
##   6.3182455   4.4447548   2.7760826   0.5789239
```

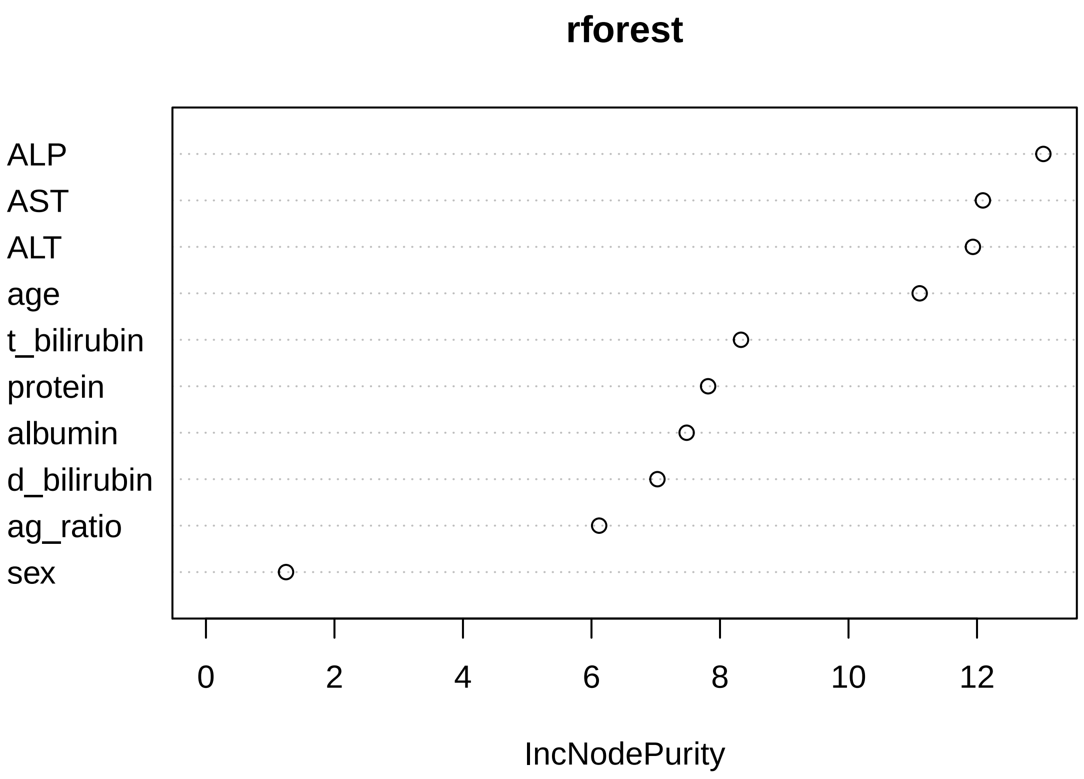
```r
# Log-transformed version
ctree_log <- rpart(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP + ALT + AST +
                  protein + albumin + ag_ratio, data=train_log, method="class")
```

In this model, the most important variables are direct bilirubin, total bilirubin, ALT, and age. Consistent with previous findings, the least important variable was sex, which does not feature at any of the decisional nodes. Because this model is relatively simple to apply (as one needs only to follow the branching chart, reading from top to bottom), it has the benefit of being easy for a healthcare worker to use in a clinical setting.

2.2.3 Random forest

The final model we will generate is a random forest, which predicts the most frequent estimate from a multitude of classification trees generated with a bootstrapping technique.

```r
# The randomForest function rejects NA values, and hence we explicitly remove them
trainforest <- na.omit(train)

set.seed(1, sample.kind="Rounding")
rforest <- randomForest(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP + ALT + AST +
                  protein + albumin + ag_ratio, data=trainforest)

varImpPlot(rforest)
```



```r
# Log-transformed version
trainforest_log <- na.omit(train_log)

set.seed(1, sample.kind="Rounding")
rforest_log <- randomForest(disease ~ age + sex + t_bilirubin + d_bilirubin + ALP +
                              ALT + AST + protein + albumin + ag_ratio, data=trainforest_log)
```

In this model, the most important variables are ALP, AST, ALT and age. Once again, the least important variable was sex. Having been produced from a bootstrapping technique, this model is inherently more difficult to visualise and interpret, but it should (in theory) protect against the overfitting that is common with classification trees.

3 Results

Three techniques have been used to generate predictive models: logistic regression, classification trees, and random forests. For each model, two versions have been generated: one trained on original variables, and one trained on log-transformed variables. All models have been trained only using the appropriate 'train' dataset, and have not yet interacted with the 'test' dataset.

To evaluate the success of these six models, we will now use them to predict the presence of liver disease in the 'test' dataset. We generate these predictions as follows:

```r
# Logistic regression, original variables
prob1 <- logistic %>% predict(test, type = "response")
logisticpred <- ifelse(prob1 > 0.5, 1, 0)

# Logistic regression, log-transformed variables
prob1log <- logistic_log %>% predict(test_log, type = "response")
logisticpred_log <- ifelse(prob1log > 0.5, 1, 0)

# Classification tree, original variables
prob2 <- ctree %>% predict(test) %>% as.matrix()
ctreepred <- ifelse(prob2[,1] < 0.5, 1, 0)

# Classification tree, log-transformed variables
prob2log <- ctree_log %>% predict(test_log) %>% as.matrix()
ctreepred_log <- ifelse(prob2log[,1] < 0.5, 1, 0)

# Random forest, original variables
prob3 <- rforest %>% predict(test)
rforestpred <- ifelse(prob3 > 0.5, 1, 0)

# Random forest, log-transformed variables
prob3log <- rforest_log %>% predict(test_log)
rforestpred_log <- ifelse(prob3log > 0.5, 1, 0)
```

We will now assess the accuracy of models by comparing these predictions with the actual disease outcomes in the test set.

3.1 Logistic regression models

```r
# Ensuring disease variable is in factor form (for confusionMatrix function)
test$disease <- as.factor(test$disease)
test_log$disease <- as.factor(test_log$disease)

# Logistic regression models
logisticpred <- as.factor(logisticpred)
logisticpred_log <- as.factor(logisticpred_log)
```

```r
# Trained on original variables:
confusionMatrix(data=logisticpred, reference=test$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  9 11
##          1 20 77
##
##                Accuracy : 0.735
##                  95% CI : (0.6455, 0.8123)
##     No Information Rate : 0.7521
##     P-Value [Acc > NIR] : 0.7081
##
##                   Kappa : 0.2069
##
##  Mcnemar's Test P-Value : 0.1508
##
##             Sensitivity : 0.8750
##             Specificity : 0.3103
##          Pos Pred Value : 0.7938
##          Neg Pred Value : 0.4500
##              Prevalence : 0.7521
##          Detection Rate : 0.6581
##    Detection Prevalence : 0.8291
##       Balanced Accuracy : 0.5927
##
##        'Positive' Class : 1
##
```

```r
# Trained on log-transformed variables:
confusionMatrix(data=logisticpred_log, reference=test_log$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 11 15
##          1 17 74
##
##                Accuracy : 0.7265
##                  95% CI : (0.6364, 0.8048)
##     No Information Rate : 0.7607
##     P-Value [Acc > NIR] : 0.8356
##
##                   Kappa : 0.2299
##
##  Mcnemar's Test P-Value : 0.8597
##
##             Sensitivity : 0.8315
##             Specificity : 0.3929
##          Pos Pred Value : 0.8132
##          Neg Pred Value : 0.4231
##              Prevalence : 0.7607
##          Detection Rate : 0.6325
##    Detection Prevalence : 0.7778
##       Balanced Accuracy : 0.6122
##
##        'Positive' Class : 1
##
```

The logistic regression model trained on original variables had an overall accuracy of **73.5%**, a sensitivity of 87.5%, and a specificity of 31.0%. The logistic regression model trained on log-transformed variables had an overall accuracy of **72.7%**, a sensitivity of 83.2%, and a specificity of 39.3%.

The logistic regression models have moderately high accuracy and sensitivities (making them good for ruling out liver disease confidently), but low specificities (making them poor for ruling in liver disease confidently). The logistic regression model trained on original variables had marginally better overall performance, but the logistic regression model trained on log-transformed variables had an appreciably higher specificity. Since a regression model involves multiplying predictor variables by their respective coefficients, we would indeed expect log-transformation (a non-linear adjustment) to produce appreciably different results.

3.2 Classification trees

```
# Classification trees
ctreepred <- as.factor(ctreepred)
ctreepred_log <- as.factor(ctreepred_log)

# Trained on original variables:
confusionMatrix(data=ctreepred, reference=test$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 10 17
##          1 19 71
##
##                Accuracy : 0.6923
##                  95% CI : (0.6003, 0.7743)
##     No Information Rate : 0.7521
##     P-Value [Acc > NIR] : 0.9431
##
##                   Kappa : 0.1552
##
##  Mcnemar's Test P-Value : 0.8676
##
##             Sensitivity : 0.8068
##             Specificity : 0.3448
##          Pos Pred Value : 0.7889
##          Neg Pred Value : 0.3704
##              Prevalence : 0.7521
##          Detection Rate : 0.6068
##    Detection Prevalence : 0.7692
##       Balanced Accuracy : 0.5758
##
##        'Positive' Class : 1
##
```

```r
# Trained on log-transformed variables:
confusionMatrix(data=ctreepred_log, reference=test_log$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 14 23
##          1 14 66
##
##                Accuracy : 0.6838
##                  95% CI : (0.5913, 0.7666)
##     No Information Rate : 0.7607
##     P-Value [Acc > NIR] : 0.9777
##
##                   Kappa : 0.2176
##
##  Mcnemar's Test P-Value : 0.1884
##
##             Sensitivity : 0.7416
##             Specificity : 0.5000
##          Pos Pred Value : 0.8250
##          Neg Pred Value : 0.3784
##              Prevalence : 0.7607
##          Detection Rate : 0.5641
##    Detection Prevalence : 0.6838
##       Balanced Accuracy : 0.6208
##
##        'Positive' Class : 1
##
```

The classification tree trained on original variables had an overall accuracy of **69.2%**, a sensitivity of 80.7%, and a specificity of 34.5%. The classification tree trained on log-transformed variables had an overall accuracy of **68.4%**, a sensitivity of 74.2%, and a specificity of 50.0%.

The classification trees had slightly poorer predictive performance than logistic regression models. The classification tree trained on original variables had marginally higher accuracy and appreciably higher sensitivity than the classification tree trained on log-transformed variables. However, the classification tree trained on log-transformed variables had a markedly higher specificity, exceeding all models evaluated thus far.

3.3 Random forests

```r
# Random forests
rforestpred <- as.factor(rforestpred)
rforestpred_log <- as.factor(rforestpred_log)
```

```r
# Trained on original variables:
confusionMatrix(data=rforestpred, reference=test$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  6 14
##          1 23 74
##
##                Accuracy : 0.6838
##                  95% CI : (0.5913, 0.7666)
##     No Information Rate : 0.7521
##     P-Value [Acc > NIR] : 0.9628
##
##                   Kappa : 0.0534
##
##  Mcnemar's Test P-Value : 0.1884
##
##             Sensitivity : 0.8409
##             Specificity : 0.2069
##          Pos Pred Value : 0.7629
##          Neg Pred Value : 0.3000
##              Prevalence : 0.7521
##          Detection Rate : 0.6325
##    Detection Prevalence : 0.8291
##       Balanced Accuracy : 0.5239
##
##        'Positive' Class : 1
##
```

```r
# Trained on log-transformed variables:
confusionMatrix(data=rforestpred_log, reference=test_log$disease, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0  7 21
##          1 21 68
##
##                Accuracy : 0.641
##                  95% CI : (0.5471, 0.7276)
##     No Information Rate : 0.7607
##     P-Value [Acc > NIR] : 0.9987
##
##                   Kappa : 0.014
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.7640
##             Specificity : 0.2500
##          Pos Pred Value : 0.7640
##          Neg Pred Value : 0.2500
##              Prevalence : 0.7607
##          Detection Rate : 0.5812
##    Detection Prevalence : 0.7607
##       Balanced Accuracy : 0.5070
##
##        'Positive' Class : 1
##
```

The random forest trained on original variables had an overall accuracy of **68.4%**, a sensitivity of 84.1%, and a specificity of 20.7%. The random forest trained on log-transformed variables had an overall accuracy of **64.1%**, a sensitivity of 76.4%, and a specificity of 25.0%.

The random forests also had poorer predictive performance than logistic regression models. The random forest trained on original variables had appreciably higher accuracy and sensitivity than the random forest trained on log-transformed variables. Conversely, the random forest trained on log-transformed variables had appreciably higher specificity.

3.4 Which models have the best performance?

Overall, **the logistic regression model trained on original variables** had the highest accuracy (73.5%) and sensitivity (87.5%) across all six models. Models trained on log-transformed variables had appreciably lower accuracies and sensitivities than their corresponding models trained on original variables. However, they had appreciably higher specificities, with **the classification tree trained on log-transformed variables** having the highest specificity (50.0%) across all six models.

4 Conclusion

We sought to identify the **best model(s) for predicting liver disease** from demographic data and blood test results, utilising a dataset of 583 patients in Andhra Pradesh, India. After an exploratory analysis of the dataset, we partitioned the dataset into a training set (80%) for training models, and a test set (20%) for evaluating their performance. We then used three techniques - logistic regression, classification trees, and random forests - on original and log-transformed variables, generating a total of six predictive models. The best models were identified as **the logistic regression model trained on original variables**, which had the highest accuracy (73.5%) and sensitivity (87.5%) of all models, and **the classification tree trained on log-transformed variables**, which had the highest specificity (50.0%) of all models.

This begs the question - is it more important to identify the presence of liver disease if it is there, or to exclude the presence of liver disease if it is not there? Since a test with high sensitivity is good for ruling out disease whilst a test with high specificity is good for ruling in disease, **the 'best' model depends on what is most important for the specific clinical situation in question**. Nonetheless, the relatively low specificities produced by our models was of particular concern, indicating the need for further modelling before a predictive approach could feasibly be applied in the real world. An ideal model would have a high sensitivity and specificity, but such a model could not be produced with our methods.

Our approach had three key strengths - we carefully explored our dataset before modelling, at least partly corrected for variables having a skewed distribution, and trialed various modelling techniques. However, it also had three key limitations - we did not correct for outliers (despite exploratory analyses indicating their likely presence), applied only a limited selection of modelling techniques (without a clear justification for our selection), and made no attempt to improve our models by optimising a tuning parameter. Future analyses should account for outliers whilst utilising additional modelling techniques and/or tuning parameters to **produce models with better predictive performance**, and in particular higher specificities.