# Predict Liver Disease

Harcharan Kabbay

12/12/2020

## Introduction

This report analyzes a data set of health records to find patterns and build models that can be used to predict liver disease in a person. The data set used for analysis is available at Kaggle Indian-Liver-Patients. The approach is to find any correlations among columns in the data set. At a high level, following steps are performed to finalize a prediction model.
- Data Analysis: Inspect the data and perform any cleanup required.
- Data Visualization
- Feature selection
- Identify patterns and correlations
- Predict whether a patient has liver disease or not.

## Data Analysis

The data set contains 416 records for persons with liver disease and 167 for those without a liver disease. The data set contains records for 441 males and 142 females, which age range from 4 years to 90 years.

Here are the data set columns and related information: -
Columns:
- Age: Age of the patient
- Gender: Gender of the patient
- Total_Bilirubin: Bilirubin is an orange-yellow pigment that occurs normally when part of your red blood cells break down.
- Direct_Bilirubin: This is the bilirubin once it reaches the liver and undergoes a chemical change.
- Alkaline_Phosphotase: Alkaline phosphatase is an enzyme you have in your liver, bile ducts, and bone.
- Alamine_Aminotransferase: Your body uses (Alamine Aminotransferase) ALT to break down food into energy.
- Aspartate_Aminotransferase: (Aspartate Aminotransferase) AST is an enzyme your liver makes
- Total_Protiens: The total protein test measures the total amount of two classes of proteins i.e. albumin and globulin.
- Albumin: Albumin's a building block that helps your body heal.
- Albumin_and_Globulin_Ratio: Ratio of protiens albumin and globulin
- Dataset: Split the data into two sets (patient with liver disease, or no disease)

**Note**: Information on columns have been gathered from WebMD

```
##       Age           Gender          Total_Bilirubin
##  Min.   : 4.00   Length:583         Min.   : 0.400
##  1st Qu.:33.00   Class :character   1st Qu.: 0.800
##  Median :45.00   Mode  :character   Median : 1.000
##  Mean   :44.75                      Mean   : 3.299
##  3rd Qu.:58.00                      3rd Qu.: 2.600
##  Max.   :90.00                      Max.   :75.000
##
```

```
##  Direct_Bilirubin Alkaline_Phosphotase Alamine_Aminotransferase
##  Min.   : 0.100   Min.   :  63.0       Min.   :  10.00
##  1st Qu.: 0.200   1st Qu.: 175.5       1st Qu.:  23.00
##  Median : 0.300   Median : 208.0       Median :  35.00
##  Mean   : 1.486   Mean   : 290.6       Mean   :  80.71
##  3rd Qu.: 1.300   3rd Qu.: 298.0       3rd Qu.:  60.50
##  Max.   :19.700   Max.   :2110.0       Max.   :2000.00
##
##  Aspartate_Aminotransferase Total_Protiens     Albumin
##  Min.   :  10.0             Min.   :2.700   Min.   :0.900
##  1st Qu.:  25.0             1st Qu.:5.800   1st Qu.:2.600
##  Median :  42.0             Median :6.600   Median :3.100
##  Mean   : 109.9             Mean   :6.483   Mean   :3.142
##  3rd Qu.:  87.0             3rd Qu.:7.200   3rd Qu.:3.800
##  Max.   :4929.0             Max.   :9.600   Max.   :5.500
##
##  Albumin_and_Globulin_Ratio    Dataset
##  Min.   :0.3000             Min.   :1.000
##  1st Qu.:0.7000             1st Qu.:1.000
##  Median :0.9300             Median :1.000
##  Mean   :0.9471             Mean   :1.286
##  3rd Qu.:1.1000             3rd Qu.:2.000
##  Max.   :2.8000             Max.   :2.000
##  NA's   :4
```

**Data Cleaning**: Four records have missing value for column Albumin_and_Globulin_Ratio, we'll remove these records from the data set so that our results remain meaningful.

```
## Identify missing data
colSums(sapply(df, 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
##                     Dataset
##                           0
```

```
## Remove 4 NA rows
df <- na.omit(df)
```

Number of records by liver disease

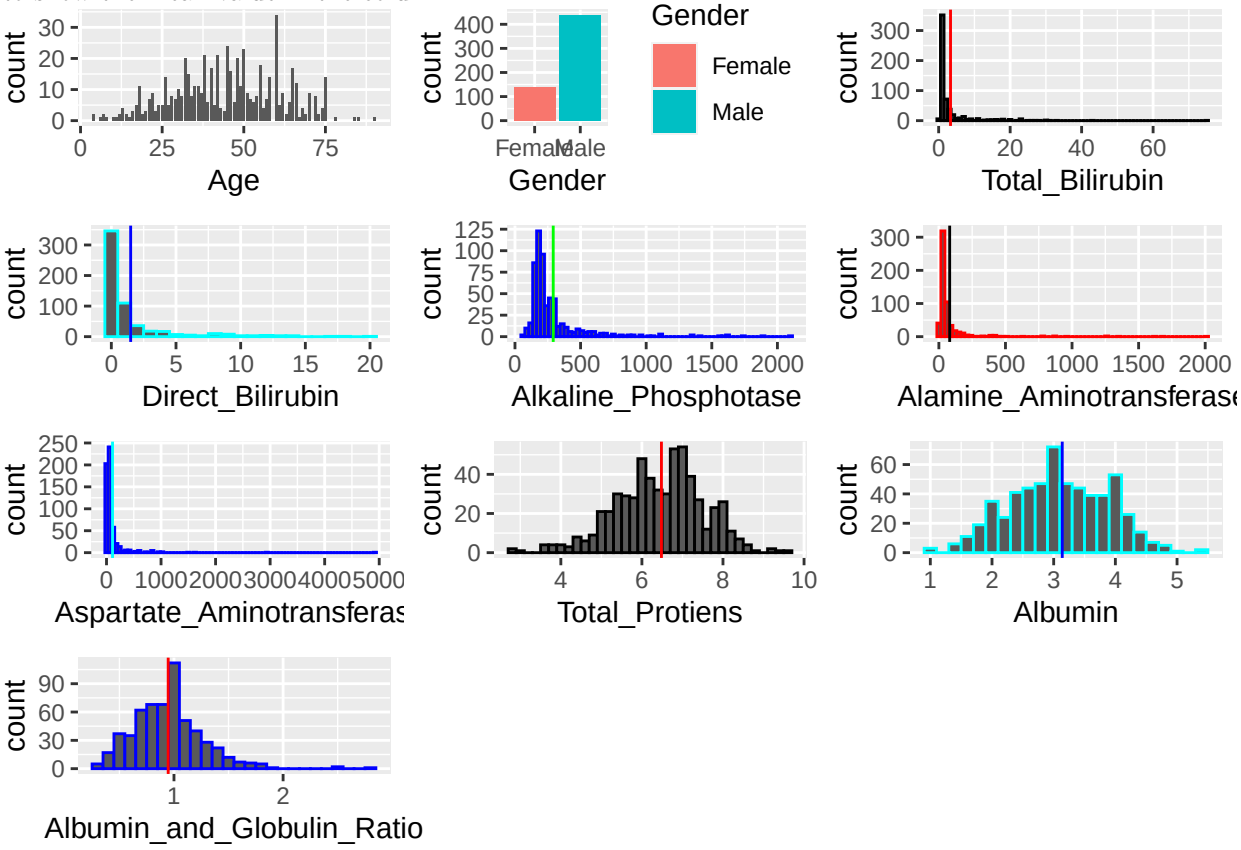
```
table(df$Dataset)
```

```
##
##   1   2
## 414 165
```

For simplicity, we will subtract "1" from column data set to make the values 0 (for liver disease) and 1 (for no liver disease), followed by conversion to a factor vector.

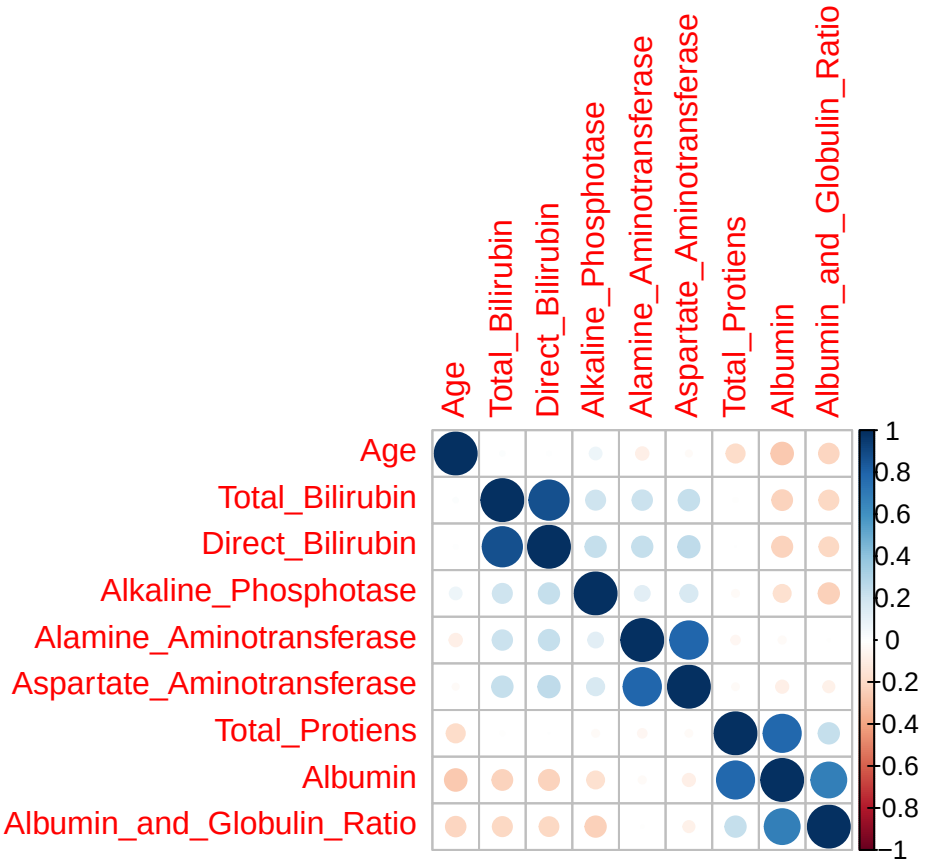
### Data Visualization

Next, we start visualizing the data. These are the graphs for values in each column by the no. of occurences, just to understand the range and spread of data. Graphs other than Age and Gender contains a vertical line to show the mean value in the column.




### Correlations

Data visualization provides us some insight into what data looks like, however, we still do not know how this data is related. We will create a correlation matrix to find the strength and direction of the correlation. This can be achieved by using `cor` function to find the correlation and then plot the graph using `corrplot`.




Correlation graph explains the relationship between different fields, and also shows some high correlation pairs. We identified that columns "Albumin", "Direct_Bilirubin" and "Aspartate_Aminotransferase" are in high correlation pairs and were hence removed from further analysis.

At this stage, we have a cleaned data set with the final number of features.

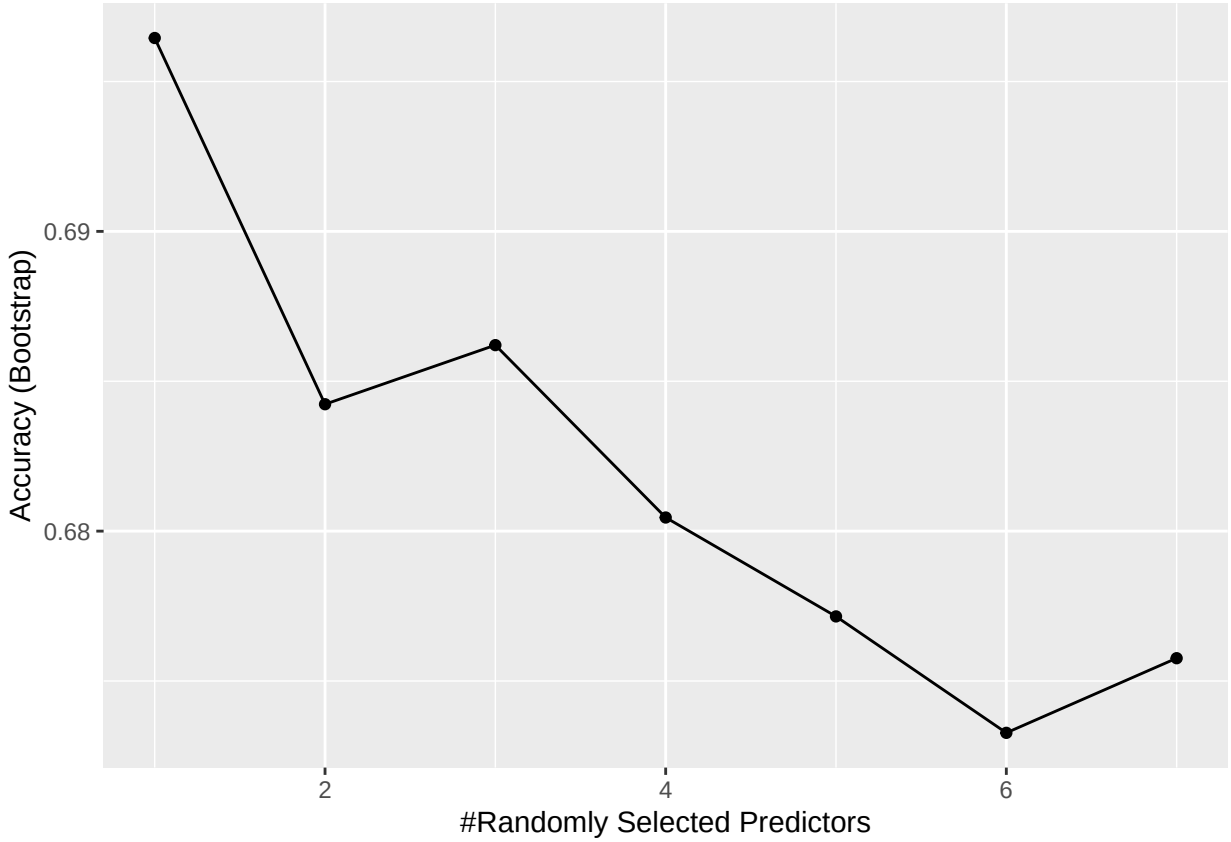
## Methods

In order to test different prediction algorithms/models, we split the data set into train and test sets with 70% to 30% ratio respectively. We will start with building the models on `train set` and then test those models using `test set`.

### Random Forest Model

Random forest classification model run with a range of mtry values so as to identify the best tune mtry(no. of variables randomly sampled as candidate at each split). The best tune mtry value was identified as 1. Once the model is built based on train data, we predicted the values from test set and caluclated the accuracy using `confusionMatrix` or just finding the `mean(pred_rf == df$Disease)`, which comes out to be **71.4%**.

```
set.seed(14, sample.kind = "Rounding")    # simulate R 3.5
train_rf <- train(Disease ~ .,
                  data = train_set,
                  method = "rf",
                  ntree = 100,
                  tuneGrid = data.frame(mtry = seq(1:7)))
ggplot(train_rf)
```




```
accuracy_rf <- confusionMatrix(predict(train_rf, test_set,
                        type = "raw"),
                test_set$Disease)$overall["Accuracy"]
```

Random Forest also gives the list of variables sorted by their importance. We can understand the data in a better way by looking at it e.g. Alamine_Aminotransferase is the most important feature in determining the liver-disease in a person while gender makes no difference.

```
imp <- varImp(train_rf)
imp
```

```
## rf variable importance
##
##                             Overall
## Alamine_Aminotransferase     100.00
## Total_Bilirubin               91.00
## Alkaline_Phosphotase          90.30
## Age                           70.52
## Total_Protiens                60.35
## Albumin_and_Globulin_Ratio    56.85
## GenderMale                     0.00
```

### GLM

Generalized Linear Model or GLM is chosen as a second method to create model on train set and perform the prediction on test set. First run of the `glm` was performed for all the available features using simple formula `train(Disease ~ . , method = "glm", data = train_set)`. Accuracy of glm comes out to be **70.3%**. Next, we removed Gender and Albumin_and_Globulin_Ratio from the features to update

the model as `train(Disease ~ . - Gender - Albumin_and_Globulin_Ratio , method = "glm", data = train_set)`and re-run. Predictions performed by updated model have an improved accuracy of **72%**.

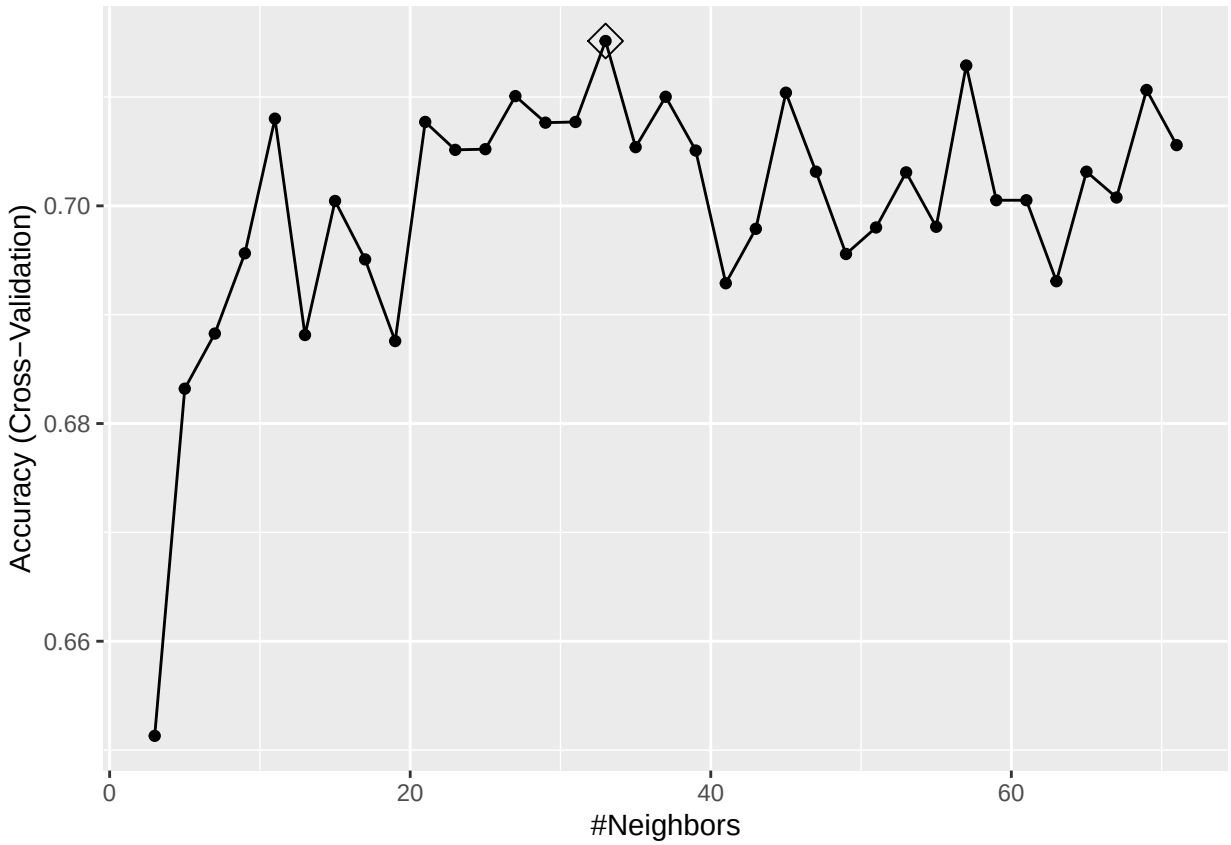
```
## Train GLM by all predictors
set.seed(1, sample.kind = "Rounding")
train_glm <- train(Disease ~ . , method = "glm", data = train_set)
pred_glm <- predict(train_glm, test_set)
accuracy_glm <- confusionMatrix(data = pred_glm,
                                reference = test_set$Disease)$overall["Accuracy"]

## Train GLM on reduced features
set.seed(1, sample.kind = "Rounding")
train_glm1 <- train(Disease ~ . - Gender - Albumin_and_Globulin_Ratio ,
                    method = "glm", data = train_set)
pred_glm1 <- predict(train_glm1, test_set)
accuracy_glm1 <- confusionMatrix(data = pred_glm1,
                                reference = test_set$Disease)$overall["Accuracy"]
```

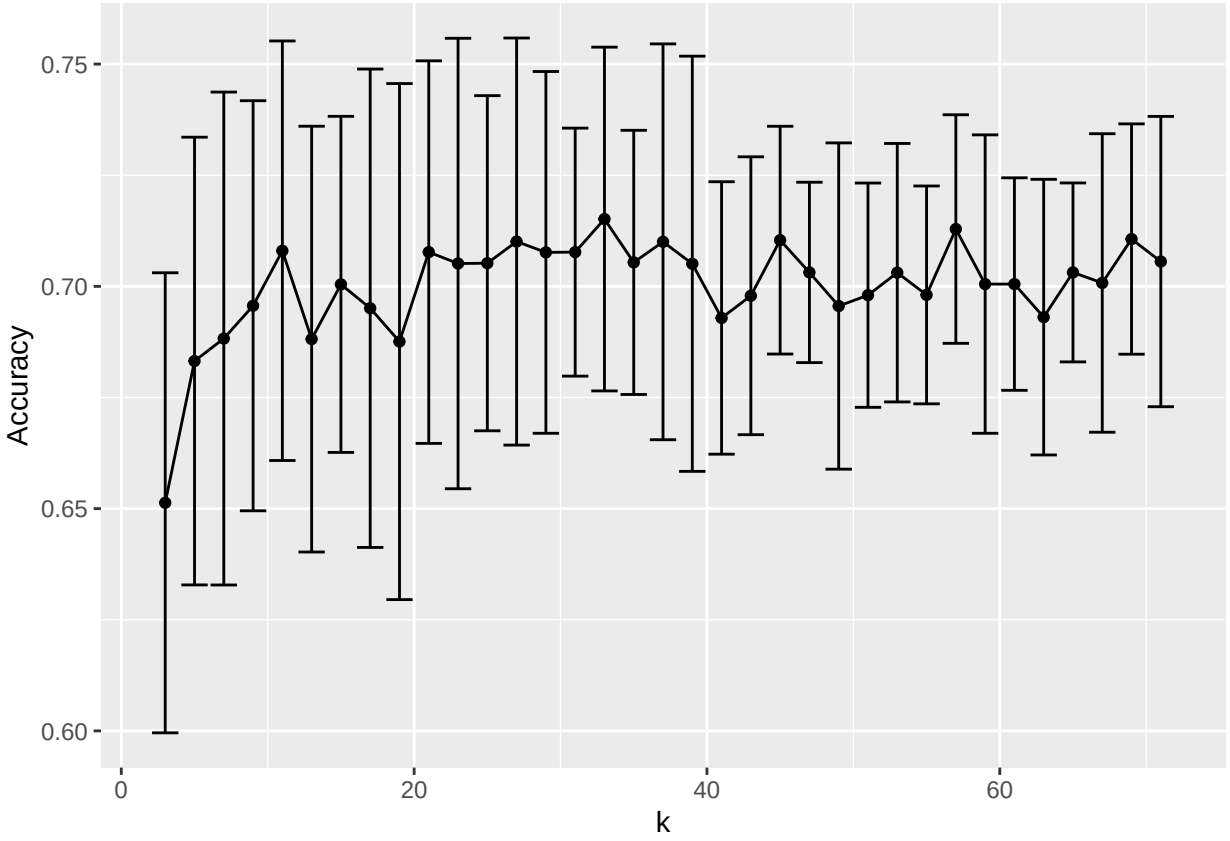
### KNN

Final method for the analysis is KNN. We ran the model using a sequence of values for k so as to find the most optimized **k** which comes out to be 33.

```
set.seed(6, sample.kind = "Rounding")
control <- trainControl(method = "cv", number = 10, p = .9)
train_knn <- train(Disease ~ ., method = "knn",
                       data = train_set,
                       tuneGrid = data.frame(k = seq(3,71,2)),
                       trControl = control)
ggplot(train_knn, highlight = TRUE)
```




We can also plot the accuracy graph for the sequence of values of **k** used in our knn model.



We found an accuracy of **69.7%** on test set using the KNN method.

## Results

So far, we trained models Random forest, GLM and KNN on train set, and then performed the predictions on test set. Here are the final results of accuracy observed for each model.

| Method | Accuracy |
|---|---|
| Random Forest | 0.7142857 |
| GLM - All features | 0.7028571 |
| GLM - Reduced features | 0.7200000 |
| KNN | 0.6971429 |

GLM model with reduced features has the highest accuracy score of **72%**, means it can predict the liver disease with this accuracy for the given set of data. Random forest is the second most accurate in the group with accuracy of **71.4%**.

## Conclusion

The report covers a high level methodology to understand and analyze the patient data to find correlations and build models. Correlations are often used as a first step before exploring models, which helps us reduce the number of features. We also observed that reducing the most un-reliable features helps improve the accuracy score (e.g. in GLM where the score improved to **72%**.
Though, this is a very tiny data set of 579 records to build and predict models, still the method can be leveraged to build a good prediction system to aid the medical specialists to predict disease in patients. There is definitely a scope to improve the accuracy if the methods are performed on a bigger set of data. This reports exhibits a generalized method, that can be used to predict other diseases on similar sets of data.