Comparison of selected algorithms to classify the subjects under study and determination of optimal dose of a drug.

Amruta S. Vasantgadkar

Abstract

BACKGROUND: The liver plays an important role in many bodily functions from protein production, blood clotting and several metabolisms. There seems to be continuous increase in the patients with liver diseases due to excessive consumption of alcohol, air pollution, drug usage etc. To treat a liver disease, levels of several blood components like aspartate aminotransferase, alanine aminotransferase, alkaline phosphatase are considered.

METHOD: 3 selected algorithms were applied on one dataset for classification of subjects into patients and non-patients and the algorithms' performance was evaluated. Additionally, exploratory data analysis was done, to find out which age group and which gender is most likely prone to liver diseases. Another dataset was also analyzed, which contained range of dose of a drug. Optimal dose of a drug was determined by comparing the response of the doses of that drug. Lastly, one of the dimentionality reduction techniques was used on both of these datasets to check if the results match.

RESULTS: For the ILPD dataset, confusion matrices of algorithms KNN and LDA show good classification of "patients" but poor classification of "non-patients". SOM algorithm performed well in classifying the variables through codes plot as well as through heat map. For liver.csv dataset, the blood component levels were within normal range for patients treated with dose A of "Astrazeneca" drug.

CONCLUSION: SOM algorithm performed well in classification of ILPD data set. Dose A of "Astrazeneca" drug was found to be the most optimal dose for the liver.csv dataset.

Introduction

Classification algorithms are widely used in various medical diagnosis tools. Diagnosis of liver diseases can be accomplished by analyzing several blood components like like aspartate aminotransferase, alanine aminotransferase, alkaline phosphatase, bilirubin, albumin etc. These automatic classification algorithms can be useful for people in medical fields like endocrinologists to classify the subject under study as a patient or a non-patient. Classification algorithms, for example K-nearest neighbours (KNN), Back propagation Neural Network algorithm, Naive Bayes classifier, C 4.5 algorithm, Support Vector Machines (SVM) algorithm have been used for classification of liver patients [1] (Bendi Venkata Ramana, May 2011).

In this paper, 3 algorithms, K-Nearest Neighbours (KNN), Linear Discriminant Analysis (LDA) and Self Organizing Maps (SOM) have been considered and their performance is evaluated, by using the dataset called ILPD (Indian Liver Patient Dataset) [2]. Thus, two new algorithms (LDA) and (SOM) have been used in this paper. ILPD dataset was collected from north east of Andhra Pradesh, India. It contains information about age, gender of the subjects, levels of various blood components, and a selector field, to divide the subjects into groups (liver patient or not). Additionally, Exploratory Data Analysis (EDA) has been carried out to find out age group and which gender is most likely prone to liver diseases. Exploratory data analysis provides a good visualization of the data through various plots.

Another dataset called liver.csv has been used[3]. It contains liver related laboratory data from a randomized, blind, parallel group clinical trial with 4 doses of the 'Astrazeneca' drug. Normal ranges of the specific blood attributed were considered. Most optimal dose of the drug was determined by referring to normal ranges of these attributed after treatment by using Exploratory Data Analysis (EDA).

Additionally, Principal Component Analysis (PCA) was carried out on both of these datasets, to see if the results match in terms of the blood components present in both of these datasets.

Materials and Methods

For applying the algorithms and exploratory data analysis, 'R' programming language was used[4]. The comments related to the specific code were shown as '#' sign. Initially, required packages were loaded as follows :

```
library(sdcMicro)

library(ggplot2)

library(MASS)

library(car)

library(psych)

library(ggbiplot)

library(reshape2)

library(gclus)

library(class)

library(kohonen)
```

First, ILPD (Indian Liver Patient Dataset) was imported into the workspace:

```
setwd("C:/Users/Sameer/Documents/Sameer/My Documents/Amruta/North Eastern/Courses_Fall2015/DSCS6030 DataMining/Module09/course material")

ILPD.raw <- read.csv ("Indian Liver Patient Dataset (ILPD).csv",header=FALSE,sep=",")

colnames(ILPD.raw) <- c("Age","Gender","TotalBilirubin","DirectBilirubin","AlkalinePhosphotase","AlanineAminotransferase","AspartateAminotransferase","TotalProtiens","Albumin","AlbuminAndGlobulinRatio","SelectorField") #column names are added as per the information provided on UCI machine learning repository.



ILPD <- na.omit(ILPD.raw) #NA values are omitted from the data.

ILPD$SelectorField <- as.factor(ILPD$SelectorField) #Last column is converted into factors

#Please note that 1 represents "patients" and 2 represents "non patients" in the last column "SelectorField".
```

The data was shuffled in order to get a mixed population of the subjects before applying algorithms:

```
shuff <- runif(nrow(ILPD))

ILPD.shuff <- ILPD[order(shuff),]
```

The data was normalized:

```
normalize<- function(x) {

  return((x-min(x))/(max(x)-min(x)))

}

ILPD.normalized <- as.data.frame(lapply(ILPD.shuff[,c(1,3,4,5,6,7,8,9,10)],normalize))

summary(ILPD.normalized)
```

To carry out sensitivity analysis on the data, varying degree of noise was added as follows:

```
ILPD1 <- addNoise(ILPD.normalized, noise = 1) #1% noise is added

ILPD1$xm

df.ILPD1 <- as.data.frame(ILPD1$xm)
```

```
ILPD5 <- addNoise(ILPD.normalized, noise =5) #5% noise is added
ILPD5$xm
df.ILPD5 <- as.data.frame(ILPD5$xm)

ILPD10 <- addNoise(ILPD.normalized, noise = 10) #10% noise is added
ILPD10$xm
df.ILPD10 <- as.data.frame(ILPD10$xm)

ILPD15 <- addNoise(ILPD.normalized, noise = 15) #15% noise is added
ILPD15$xm
df.ILPD15 <- as.data.frame(ILPD15$xm)

ILPD20 <- addNoise(ILPD.normalized, noise = 20) #20% noise is added
ILPD20$xm
df.ILPD20 <- as.data.frame(ILPD20$xm)

ILPD25 <- addNoise(ILPD.normalized, noise = 25) #25% noise is added
ILPD25$xm
df.ILPD25 <- as.data.frame(ILPD25$xm)
```

KNN algorithm on ILPD data

To apply KNN algorithm [5] on the data, 'train' and 'test' data were formed:

```
ILPD1.train <- df.ILPD1[1:490,]
```

```
ILPD1.test <- df.ILPD1[491:579,]


ILPD5.train <- df.ILPD5[1:490,]
ILPD5.test <- df.ILPD5[491:579,]


ILPD10.train <- df.ILPD10[1:490,]
ILPD10.test <- df.ILPD10[491:579,]


ILPD15.train <- df.ILPD15[1:490,]
ILPD15.test <- df.ILPD15[491:579,]


ILPD20.train <- df.ILPD20[1:490,]
ILPD20.test <- df.ILPD20[491:579,]


ILPD25.train <- df.ILPD25[1:490,]
ILPD25.test <- df.ILPD25[491:579,]


ILPD.normalized.train <- ILPD.normalized[1:490,]
ILPD.normalized.test <- ILPD.normalized[491:579,]


ILPD.train.target <- ILPD.shuff[1:490,c(11)] #Target data is constructed.
ILPD.test.target <- ILPD.shuff[491:579,c(11)]
```

The code to run KNN algorithm on the data:

```
k<-25
```

```
knn.m1 <- knn(train = ILPD1.train, test = ILPD1.test, ILPD.train.target, k) #KNN on a data having 1% noise

knn.m1



k<-25

knn.m5 <- knn(train = ILPD5.train, test = ILPD5.test, ILPD.train.target, k) #KNN on a data having 5% noise

knn.m5



k<-25

knn.m10 <- knn(train = ILPD10.train, test = ILPD10.test, ILPD.train.target, k)#KNN on a data having 10% noise

knn.m10



k<-25

knn.m15 <- knn(train = ILPD15.train, test = ILPD15.test, ILPD.train.target, k)#KNN on a data having 15% noise

knn.m15



k<-25

knn.m20 <- knn(train = ILPD20.train, test = ILPD20.test, ILPD.train.target, k)#KNN on a data having 20% noise

knn.m20



k<-25

knn.m25 <- knn(train = ILPD25.train, test = ILPD25.test, ILPD.train.target, k)#KNN on a data having 25% noise
```

```
knn.m25



k<-25

knn.m.normalized <- knn(train = ILPD.normalized.train, test = ILPD.normalized.test,
ILPD.train.target, k)

knn.m.normalized #KNN on a normalized data containing no noise
```

Confusion matrices of the data frames with varying degree of noise were plotted to evaluate the performance of the KNN algorithm. Below is the code to plot the confusion matrices:

```
cm1 <- table(ILPD.test.target, knn.m1)

cm1



cm5 <- table(ILPD.test.target, knn.m5)

cm5



cm10 <- table(ILPD.test.target, knn.m10)

cm10



cm15 <- table(ILPD.test.target, knn.m15)

cm15



cm20 <- table(ILPD.test.target, knn.m20)

cm20



cm25 <- table(ILPD.test.target, knn.m25)

cm25
```

```
cm.normalized <- table(ILPD.test.target, knn.m.normalized)

cm.normalized
```

LDA algorithm on ILPD data

To get better understanding of the data before running LDA algorithm [6], some plots were generated. Below is the code to plot a scatterplot:

```
scatterplotMatrix(ILPD[,3:10], main="scatterplot of ILPD data showing relationships among variables")
```

Below is the code to plot specific attributes against each other to see the relationship among them:

```
plot(ILPD$TotalProtiens, ILPD$Albumin, xlab="Total proteins", ylab="Albumin", main="Positive correlation between Total proteins and Albumin")

plot(ILPD$Albumin, ILPD$AlbuminAndGlobulinRatio, xlab="Albumin", ylab="Ratio of Albumin to Globulin", main="positive correlation between Albumin and Albumin to Globulin ratio")
```

Next, qplots of different variables were plotted against each other to see how well the 2 groups in variable SelectorField are separated. Following is the code to plot alkaline phosphatase against alanine aminotransferase:

```
qplot(ILPD$AlkalinePhosphotase,ILPD$AlanineAminotransferase,data=ILPD, xlab="Alkaline Phosphatase", ylab="Alanine Aminotransferase", main="Plot showing distribution of the outcome variable")+geom_point(aes(colour = factor(ILPD$SelectorField),shape = factor(ILPD$SelectorField)))
```

Following is the code to plot total proteins against total bilirubin:

```
 qplot(ILPD$TotalProtiens,ILPD$TotalBilirubin,data=ILPD ,xlab="Total Proteins", ylab="Total Bilirubin",main= "plot showing distribution of the outcome variable")+geom_point(aes(colour = factor(ILPD$SelectorField),shape = factor(ILPD$SelectorField)))
```

Following is the code to run LDA algorithm on the ILPD data containing varying degree of noise:

```
ILPD.lda1 <- lda(ILPD$SelectorField ~ df.ILPD1$TotalBilirubin + df.ILPD1$DirectBilirubin + df.ILPD1$AlkalinePhosphotase + df.ILPD1$AlanineAminotransferase + df.ILPD1$AspartateAminotransferase + df.ILPD1$TotalProtiens + df.ILPD1$Albumin + df.ILPD1$AlbuminAndGlobulinRatio )
```

```
ILPD.lda5 <- lda(ILPD$SelectorField ~ df.ILPD5$TotalBilirubin + df.ILPD5$DirectBilirubin +
df.ILPD5$AlkalinePhosphotase + df.ILPD5$AlanineAminotransferase +
df.ILPD5$AspartateAminotransferase + df.ILPD5$TotalProtiens + df.ILPD5$Albumin +
df.ILPD5$AlbuminAndGlobulinRatio )

ILPD.lda10 <- lda(ILPD$SelectorField ~ df.ILPD10$TotalBilirubin +
df.ILPD10$DirectBilirubin + df.ILPD10$AlkalinePhosphotase +
df.ILPD10$AlanineAminotransferase + df.ILPD10$AspartateAminotransferase +
df.ILPD10$TotalProtiens + df.ILPD10$Albumin + df.ILPD10$AlbuminAndGlobulinRatio )

ILPD.lda15 <- lda(ILPD$SelectorField ~ df.ILPD15$TotalBilirubin +
df.ILPD15$DirectBilirubin + df.ILPD15$AlkalinePhosphotase +
df.ILPD15$AlanineAminotransferase + df.ILPD15$AspartateAminotransferase +
df.ILPD15$TotalProtiens + df.ILPD15$Albumin + df.ILPD15$AlbuminAndGlobulinRatio )

ILPD.lda20 <- lda(ILPD$SelectorField ~ df.ILPD20$TotalBilirubin +
df.ILPD20$DirectBilirubin + df.ILPD20$AlkalinePhosphotase +
df.ILPD20$AlanineAminotransferase + df.ILPD20$AspartateAminotransferase +
df.ILPD20$TotalProtiens + df.ILPD20$Albumin + df.ILPD20$AlbuminAndGlobulinRatio )

ILPD.lda25 <- lda(ILPD$SelectorField ~ df.ILPD25$TotalBilirubin +
df.ILPD25$DirectBilirubin + df.ILPD25$AlkalinePhosphotase +
df.ILPD25$AlanineAminotransferase + df.ILPD25$AspartateAminotransferase +
df.ILPD25$TotalProtiens + df.ILPD25$Albumin + df.ILPD25$AlbuminAndGlobulinRatio )
```

Confusion matrices of the data frames with varying degree of noise were plotted to evaluate the performance of the LDA algorithm. Following is the code to plot the confusion matrices:

```
ILPD.lda.values1 <- predict(ILPD.lda1, df.ILPD1)

cm.m1 <- table(ILPD.lda.values1$class,ILPD[,c(11)])

cm.m1

ILPD.lda.values5 <- predict(ILPD.lda5, df.ILPD5)

cm.m5 <- table(ILPD.lda.values5$class,ILPD[,c(11)])

cm.m5

ILPD.lda.values10 <- predict(ILPD.lda10, df.ILPD10)

cm.m10 <- table(ILPD.lda.values10$class,ILPD[,c(11)])

cm.m10
```

```
ILPD.lda.values15 <- predict(ILPD.lda15, df.ILPD15)

cm.m15 <- table(ILPD.lda.values15$class,ILPD[,c(11)])

cm.m15

ILPD.lda.values20 <- predict(ILPD.lda20, df.ILPD20)

cm.m20 <- table(ILPD.lda.values20$class,ILPD[,c(11)])

cm.m20

ILPD.lda.values25 <- predict(ILPD.lda25, df.ILPD25)

cm.m25 <- table(ILPD.lda.values25$class,ILPD[,c(11)])

cm.m25
```

SOM algorithm on ILPD data

Following is the code to plot the "codes" plot in SOM algorithm [7] on the ILPD data containing varying degree of noise:

```
training1 <- sample(nrow(df.ILPD1), 450)

Xtraining1 <- scale(df.ILPD1[training1, ])

# 5, 5, "hexagonal"

fit.som1 <- som(Xtraining1, somgrid(5, 5, "hexagonal"))

map.som1 <- map(fit.som1,

            scale(df.ILPD1[-training1, ],

                center=attr(Xtraining1, "scaled:center"),

                scale=attr(Xtraining1, "scaled:scale")))

training5 <- sample(nrow(df.ILPD5), 450)

Xtraining5 <- scale(df.ILPD5[training5, ])

# 5, 5, "hexagonal"

fit.som5 <- som(Xtraining5, somgrid(5, 5, "hexagonal"))

map.som5 <- map(fit.som5,
```

```
            scale(df.ILPD5[-training5, ],
                center=attr(Xtraining5, "scaled:center"),
                scale=attr(Xtraining5, "scaled:scale")))
training10 <- sample(nrow(df.ILPD10), 450)
Xtraining10 <- scale(df.ILPD10[training10, ])
# 5, 5, "hexagonal"
fit.som10 <- som(Xtraining10, somgrid(5, 5, "hexagonal"))
map.som10 <- map(fit.som10,
            scale(df.ILPD10[-training10, ],
                center=attr(Xtraining10, "scaled:center"),
                scale=attr(Xtraining10, "scaled:scale")))
training15 <- sample(nrow(df.ILPD15), 450)
Xtraining15 <- scale(df.ILPD15[training15, ])
# 5, 5, "hexagonal"
fit.som15 <- som(Xtraining15, somgrid(5, 5, "hexagonal"))
map.som15 <- map(fit.som15,
            scale(df.ILPD15[-training15, ],
                center=attr(Xtraining15, "scaled:center"),
                scale=attr(Xtraining15, "scaled:scale")))
training20 <- sample(nrow(df.ILPD20), 450)
Xtraining20 <- scale(df.ILPD20[training20, ])
# 5, 5, "hexagonal"
fit.som20 <- som(Xtraining20, somgrid(5, 5, "hexagonal"))
map.som20 <- map(fit.som20,
            scale(df.ILPD20[-training20, ],
```

```
                center=attr(Xtraining20, "scaled:center"),
                scale=attr(Xtraining20, "scaled:scale")))
training25 <- sample(nrow(df.ILPD25), 450)
Xtraining25 <- scale(df.ILPD25[training25, ])
# 5, 5, "hexagonal"
fit.som25 <- som(Xtraining25, somgrid(5, 5, "hexagonal"))
map.som25 <- map(fit.som25,
          scale(df.ILPD25[-training25, ],
                center=attr(Xtraining25, "scaled:center"),
                scale=attr(Xtraining25, "scaled:scale")))
plot(fit.som1,"codes", main="Code plot of ILPD data containing 1% noise")
plot(fit.som5,"codes", main="Code plot of ILPD data containing 5% noise")
plot(fit.som10,"codes", main="Code plot of ILPD data containing 10% noise")
plot(fit.som15,"codes", main="Code plot of ILPD data containing 15% noise")
plot(fit.som20,"codes", main="Code plot of ILPD data containing 20% noise")
plot(fit.som25,"codes", main="Code plot of ILPD data containing 25% noise")
```

Following is the code to plot the heat map of ILPD data [8]:

```
ILPD.new <- ILPD.raw[,c(1,3,4,5,6,7,8,9,10,11)]
ILPD.m <- as.matrix(ILPD.new)
heatmap(ILPD.m, distfun=dist, scale="row",keep.dendro = FALSE)
```

EDA on ILPD data

To check which gender is more likely to have liver disease, 2 plots were generated. Following is the code to plot male and female distribution in the ILPD data using EDA [9].

```
qplot(x=Gender, data=ILPD)
```

Following is the code to plot "liver patient" and "non liver patient" distribution according to gender:

```
qplot(SelectorField,  data=ILPD,  fill=Gender, position="dodge")
```

Next, to check which age group is most likely prone to liver diseases, 2 plots were generated. Following is the code to plot age distribution within the ILPD data:

```
qplot(x=Age, data=ILPD)
```

Following is the code to plot a boxplot showing gender age distribution into two groups "liver patient" and "non liver patient".

```
qplot(SelectorField,Age, data =ILPD, geom="boxplot")
```

Determining optimal dosage of an "Astrazeneca" drug using EDA

First the liver.csv dataset was imported into R.

The data was divided into subsets so that each subset contains the data related to that particular dose of the drug. The code to subset the data is as follows:

```
liver.subA <- subset(liver, dose=="A")

liver.subB <- subset(liver, dose=="B")

liver.subC <- subset(liver, dose=="C")

liver.subD <- subset(liver, dose=="D")
```

Range of each given blood components, which are ALP (Alkaline phosphatase), ALT (Alanine aminotransferase), AST (Aspartate aminotransferase) and TBL (Total Bilirubin) after treatment are checked. The code to check the range of these blood components is as follows:

Code to check range of ALP:

```
range(liver.subA$ALP.M)

range(liver.subB$ALP.M)

range(liver.subC$ALP.M)

range(liver.subD$ALP.M)
```

Code to check range of ALT:

```
range(liver.subA$ALT.M)

range(liver.subB$ALT.M)
```

```
range(liver.subC$ALT.M)

range(liver.subD$ALT.M)
```

Code to check range of AST:

```
range(liver.subA$AST.M)

range(liver.subB$AST.M)

range(liver.subC$AST.M)

range(liver.subD$AST.M)
```

Code to check range of TBL:

```
range(liver.subA$TBL.M)

range(liver.subB$TBL.M)

range(liver.subC$TBL.M)

range(liver.subD$TBL.M)
```

PCA on ILPD data:

The data frame ILPD was modified so as to keep only the columns with numeric values. The new data frame was named as newILPD. The code is as follows:

```
newILPD <- ILPD[,c(1,3,4,5,6,7,8,9,10)]
```

Following is the pairs plot showing correlations of variables among them.

```
my.abs <- abs(cor(newILPD))

dta.col <- dmat.color(my.abs) # get colors

cpairs(newILPD, panel.colors=dta.col, gap=.5)
```

Principal component analysis was carried out using princomp function, on the ILPD data set:

```
ILPDpca <- princomp((newILPD), cor = TRUE)

ILPDpca

summary(ILPDpca)
```

Next, principal component analysis was carried out using prcomp function, on the ILPD data set:

```
ILPDprc <- prcomp(na.omit(newILPD), retx=TRUE, center=TRUE, scale.=TRUE)

ILPDprc
```

Following is the code to plot the screeplot:

```
plot(ILPDpca)

screeplot(ILPDpca, type="lines")
```

Following is the code to plot the barplot and the biplot showing PCA on ILPD data:

```
barplot(ILPDpca$sdev/ILPDpca$sdev[1])

biplot(ILPDpca)
```

PCA on liver.csv data

Following is the code showing PCA on liver.csv data.

```
new.liver <- liver[,c(2,3,4,5,6,7,8,9)]# the data is modified to remove the first column containing index and to remove the last column which has outcome variables.
```

Following is the code to plot pairs plot showing correlation of the variables among them.

```
my.abs1 <- abs(cor(new.liver))

dta.col1 <- dmat.color(my.abs1) # get colors

cpairs(new.liver, panel.colors=dta.col1, gap=.5)
```

Principal Component Analysis was done on the liver.csv data:

```
liver.prc <- prcomp(new.liver, retx=TRUE, center=TRUE, scale.=TRUE)

liver.prc

summary(liver.prc)
```

The screeplot was plotted as follows:

```
plot(liver.prc)

screeplot(liver.prc, type="lines")
```

Following is the code to plot barplot and biplot after carrying out PCA on liver.csv data

```
barplot(liver.prc$sdev/liver.prc$sdev[1])

biplot(liver.prc)
```

Results

Following is the summary of ILPD data:

```
      Age           Gender    TotalBilirubin   DirectBilirubin
 Min.   : 4.00   Female:140   Min.   : 0.400   Min.   : 0.100
 1st Qu.:33.00   Male  :439   1st Qu.: 0.800   1st Qu.: 0.200
 Median :45.00                Median : 1.000   Median : 0.300
 Mean   :44.78                Mean   : 3.315   Mean   : 1.494
 3rd Qu.:58.00                3rd Qu.: 2.600   3rd Qu.: 1.300
 Max.   :90.00                Max.   :75.000   Max.   :19.700
 AlkalinePhosphotase AlanineAminotransferase AspartateAminotransferase
 Min.   :  63.0      Min.   :  10.00         Min.   :  10.0
 1st Qu.: 175.5      1st Qu.:  23.00         1st Qu.:  25.0
 Median : 208.0      Median :  35.00         Median :  42.0
 Mean   : 291.4      Mean   :  81.13         Mean   : 110.4
 3rd Qu.: 298.0      3rd Qu.:  61.00         3rd Qu.:  87.0
 Max.   :2110.0      Max.   :2000.00         Max.   :4929.0
 TotalProtiens      Albumin      AlbuminAndGlobulinRatio SelectorField
 Min.   :2.700   Min.   :0.900   Min.   :0.3000          1:414
 1st Qu.:5.800   1st Qu.:2.600   1st Qu.:0.7000          2:165
 Median :6.600   Median :3.100   Median :0.9300
 Mean   :6.482   Mean   :3.139   Mean   :0.9471
 3rd Qu.:7.200   3rd Qu.:3.800   3rd Qu.:1.1000
 Max.   :9.600   Max.   :5.500   Max.   :2.8000
```

Results of KNN algorithm application on ILPD data

Following is the confusion matrix when KNN was applied on the ILPD data containing 1% noise:

```
                knn.m1
ILPD.test.target  1  2
               1 57  7
               2 21  4
```

Following is the confusion matrix when KNN was applied on the ILPD data containing 5% noise:

```
                knn.m5
ILPD.test.target  1  2
               1 64  0
               2 24  1
```

Following is the confusion matrix when KNN was applied on the ILPD data containing 10% noise:

```
                knn.m10
ILPD.test.target  1  2
               1 64  0
               2 24  1
```

Following is the confusion matrix when KNN was applied on the ILPD data containing 15% noise:

```
                knn.m15
ILPD.test.target  1  2
               1 64  0
               2 25  0
```

Following is the confusion matrix when KNN was applied on the ILPD data containing 20% noise:

```
                knn.m20
ILPD.test.target  1  2
               1 63  1
               2 25  0
```

Following is the confusion matrix when KNN was applied on the ILPD data containing 25% noise:

```
                knn.m25
ILPD.test.target  1  2
               1 64  0
               2 25  0
```

Following is the confusion matrix when KNN was applied on the ILPD data without noise:

```
                knn.m.normalized
ILPD.test.target  1  2
               1 55  9
               2 19  6
```

Results of LDA algorithm application on ILPD data

To get better understanding of the data before running LDA algorithm, some plots were generated as follows:

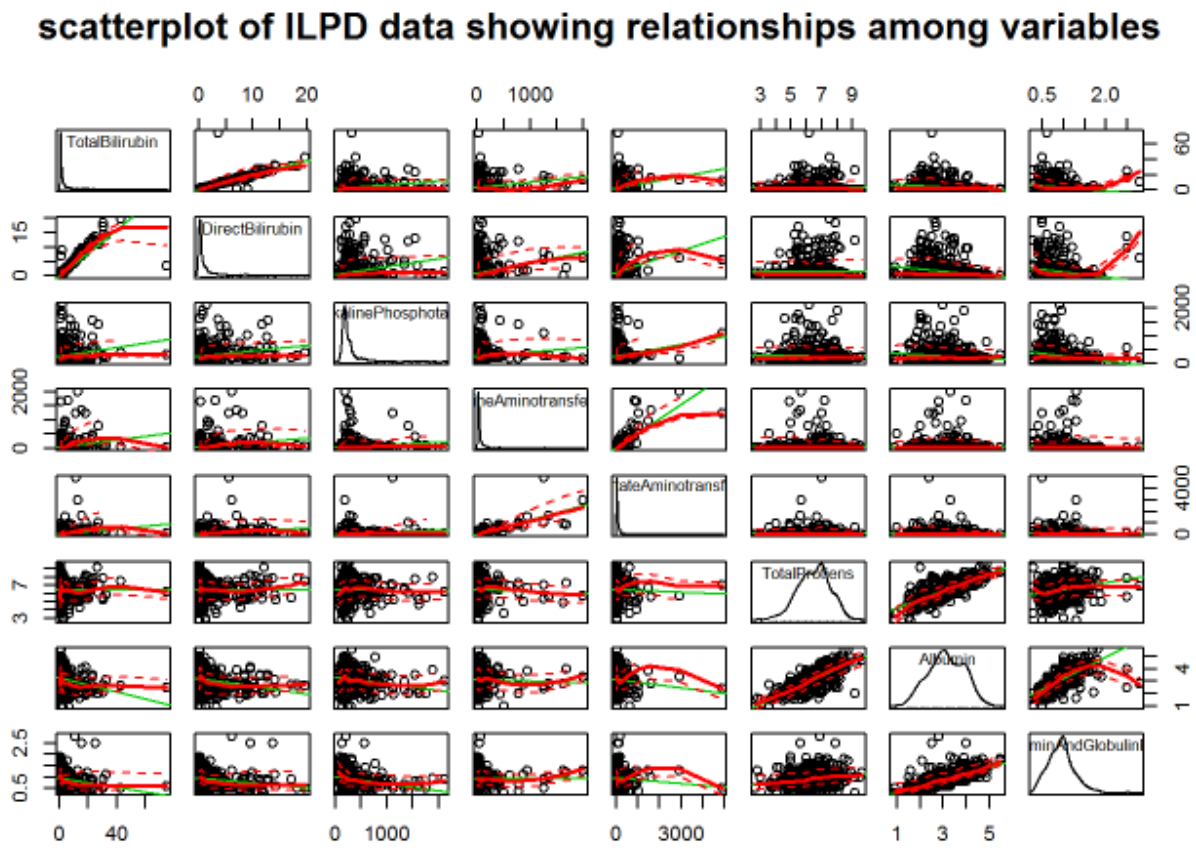

Fig.1: The scatterplot shows how different variables are related to each other, few of them showing positive correlation.

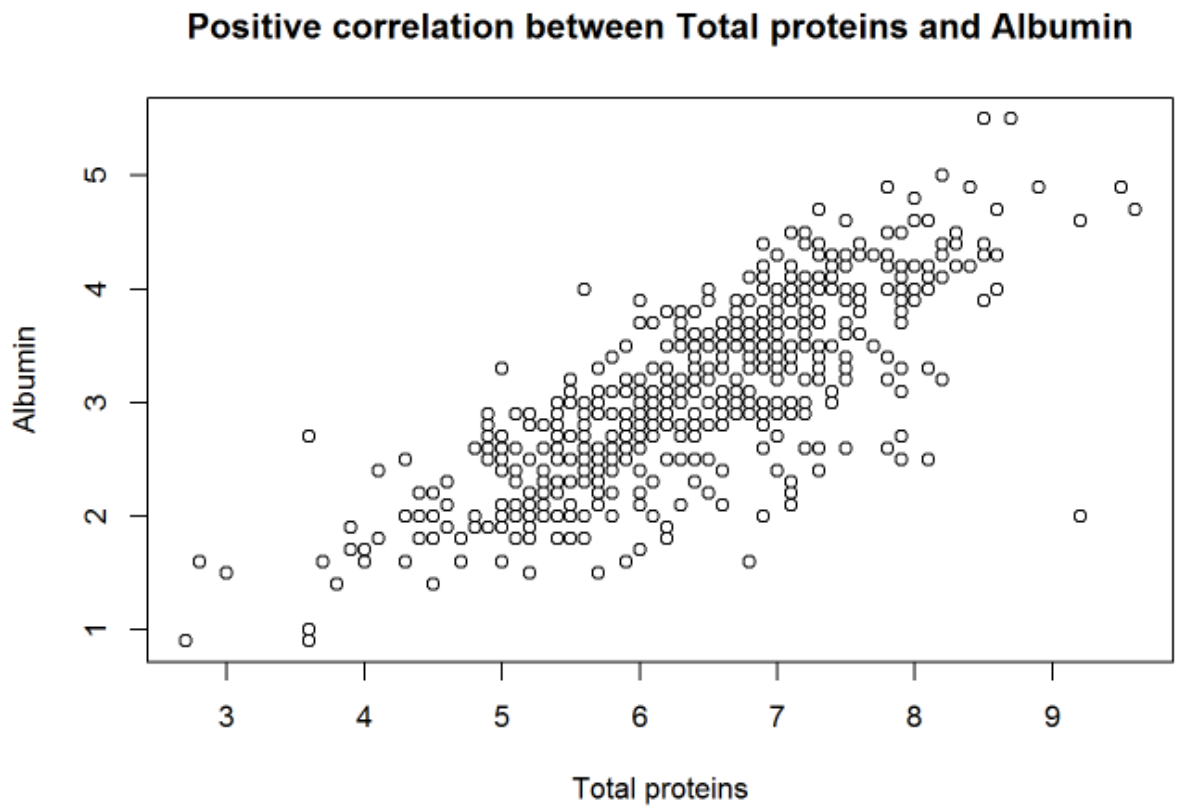

Fig.2: The above plot shows strong positive correlation between Total proteins and Albumin.

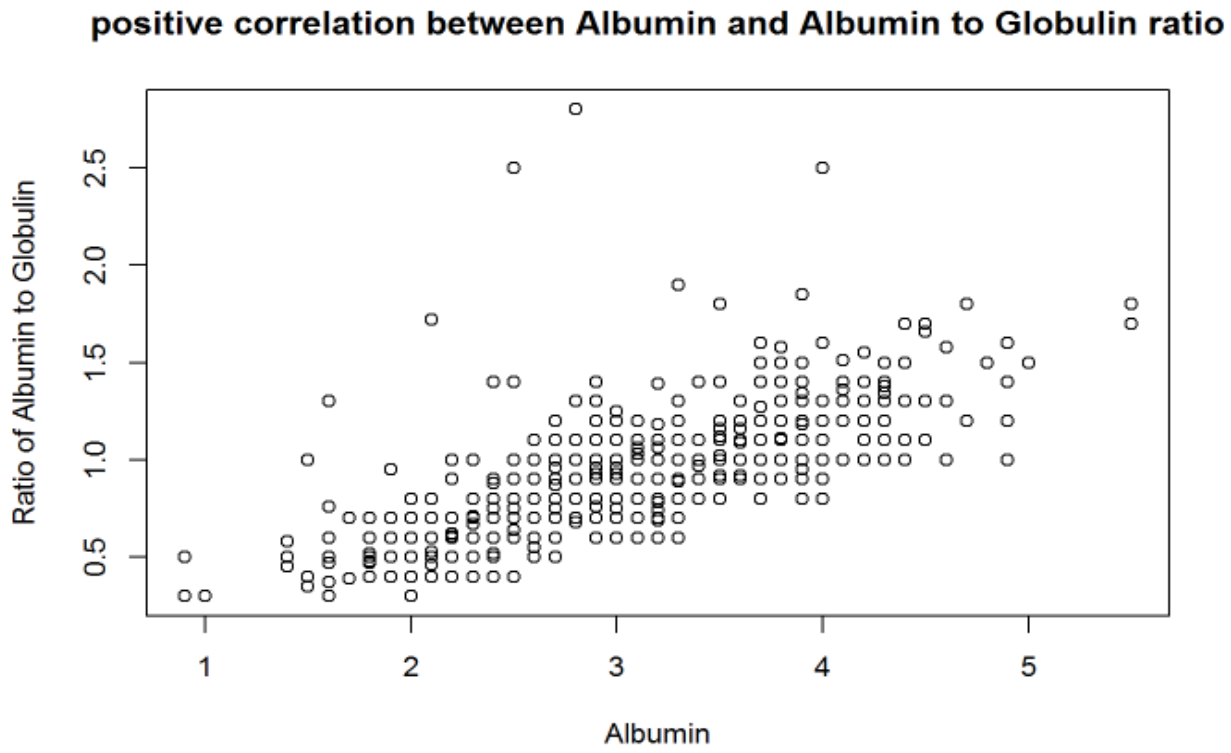

Fig.3: The above plot shows positive correlation between Albumin and ratio of Albumin to Globulin.

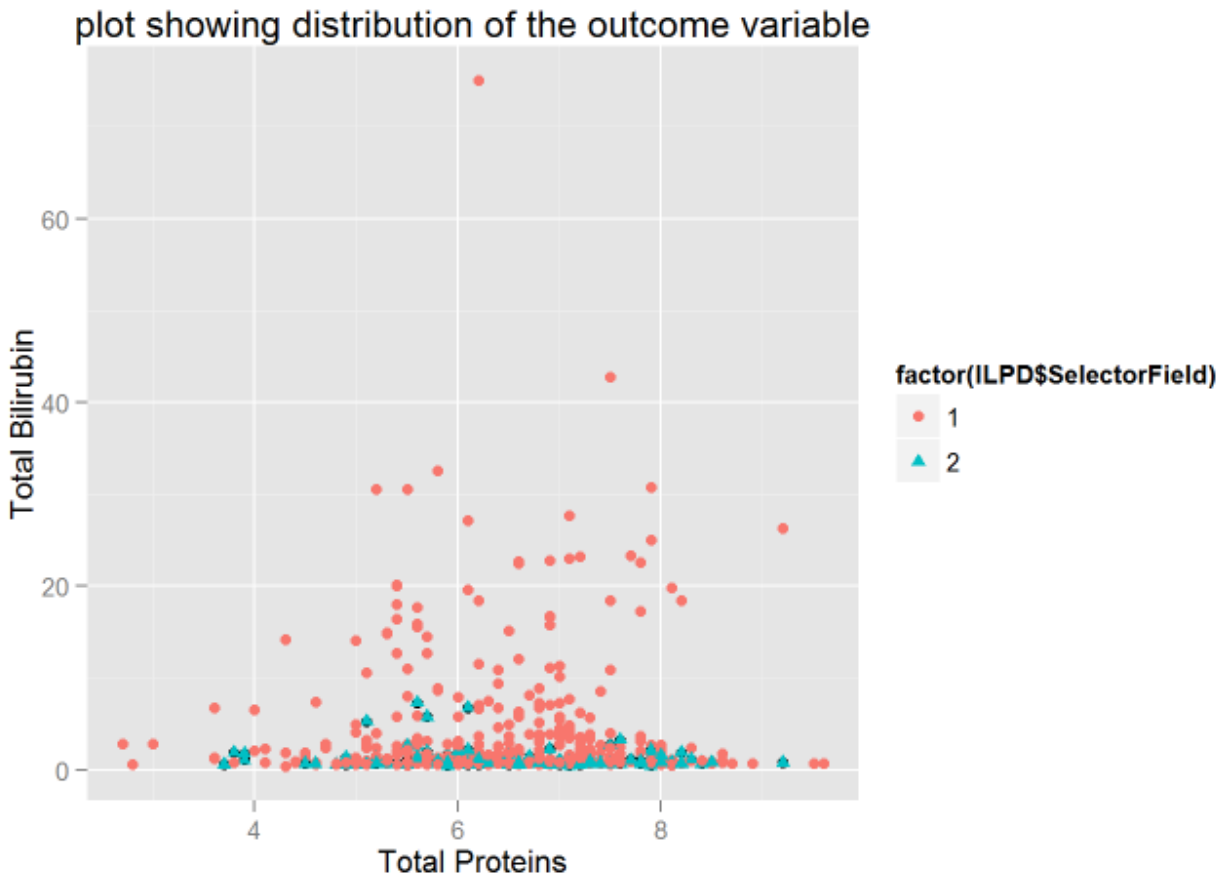

Fig.4: The above plot shows distribution of outcome variable "1" being "Patient" and "2" being "non-patient". There is some intermixing of the 2 variables near x-axis.

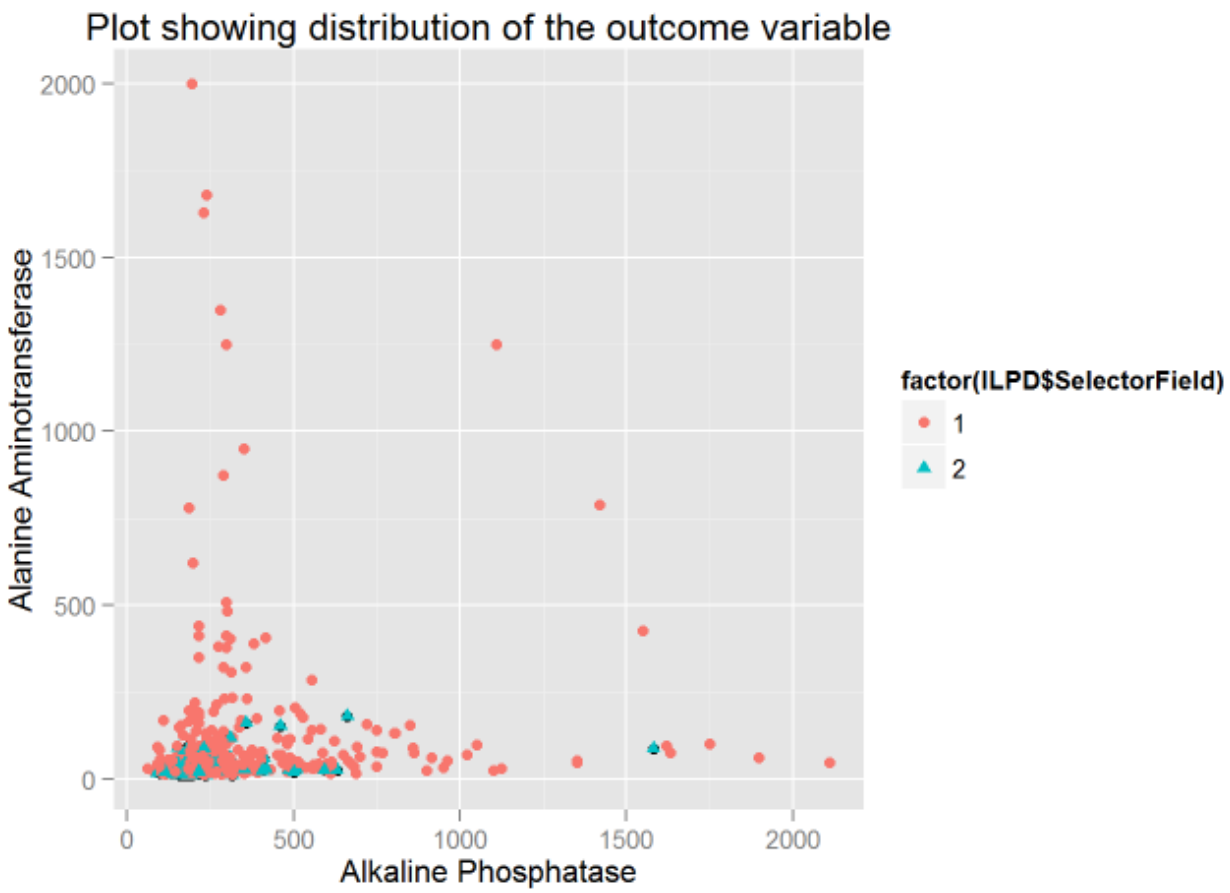

Fig.5: The plot shows the distribution of outcome variable, "1" being "patient" and "2" being "non-patient". There is intermixing in the distribution of the 2 outcome variables.

Results of LDA algorithm application on ILPD data, showing confusion matrices:

Following is the confusion matrix when LDA was applied on the ILPD data containing 1% noise:

```
cm.m1
      1   2
1 410 164
2   4   1
```

Following is the confusion matrix when LDA was applied on the ILPD data containing 5% noise:

```
cm.m5

      1   2
1 414 165
2   0   0
```

Following is the confusion matrix when LDA was applied on the ILPD data containing 10% noise:

```
cm.m10
```

```
      1   2
1 413 165
2   1   0
```

Following is the confusion matrix when LDA was applied on the ILPD data containing 15% noise:

```
cm.m15
```

```
      1   2
1 414 165
2   0   0
```

Following is the confusion matrix when LDA was applied on the ILPD data containing 20% noise:

```
cm.m20
```

```
      1   2
1 414 164
2   0   1
```

Following is the confusion matrix when LDA was applied on the ILPD data containing 25% noise:

```
cm.m25
```

```
      1   2
1 413 165
2   1   0
```

Results of SOM algorithm on ILPD data:

Following are the "code plots" showing distribution of variables on ILPD data containing varying degree of noise.

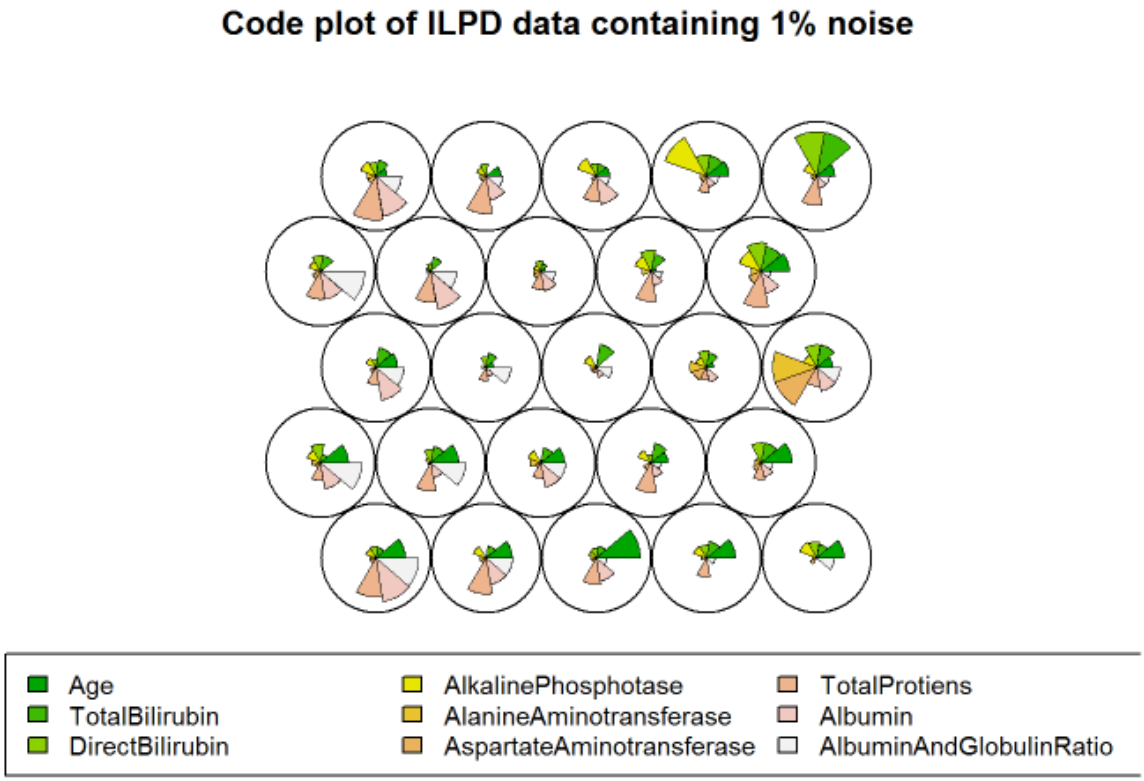

Fig.6: Above plot shows a good distribution of variables in the ILPD data with 1% added noise. On the left hand side, "Albumin and Globulin ratio" is seen. On the bottom left corner, "Total Proteins" and "Albumin" are separated. On the bottom side of the plot, "Age" seems to be distributed. On the top right corner, "Total Bilirubin" and "Direct Bilirubin" are seen. Top right hand side has "Alkaline phosphatase" and on the right side of the plot, there are "Alanine Aminotransferase" and "Aspartate Aminotransferase".

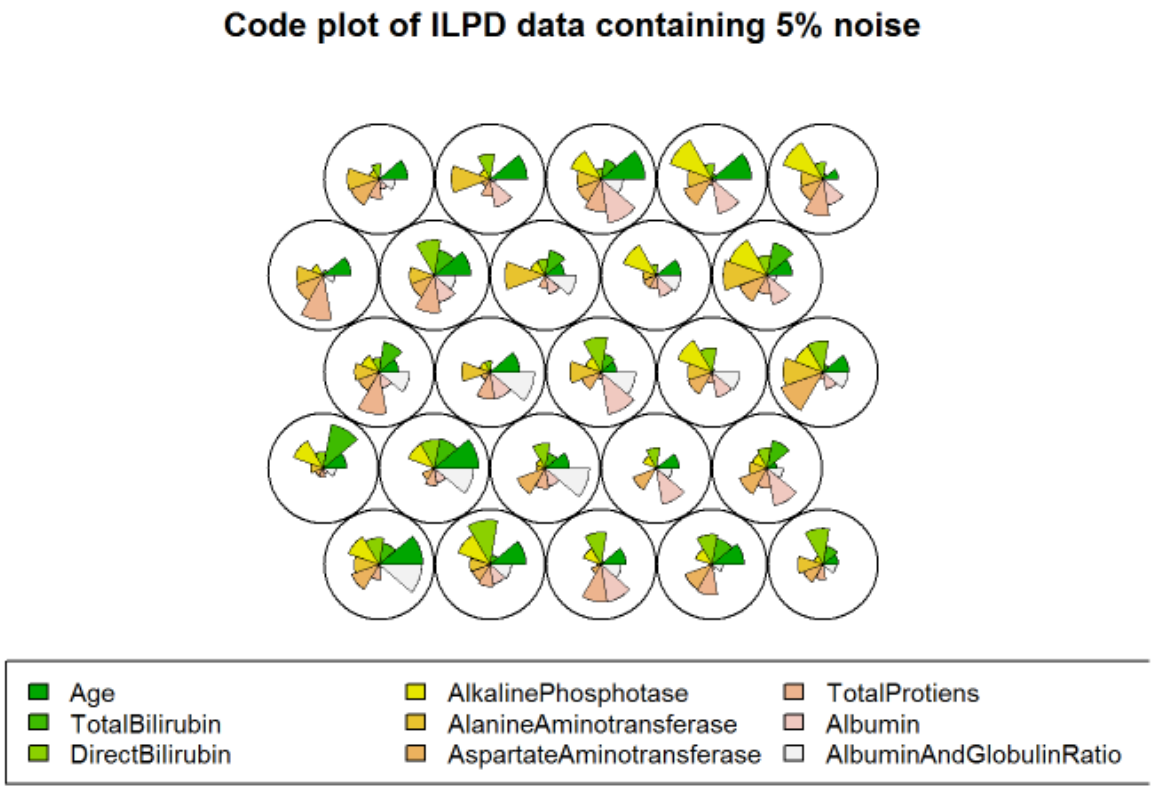

Fig.7: The above codes plot of ILPD data with 5% noise is not showing very clear distribution of variables. On bottom left corner, there are "Total Bilirubin" and "Direct Bilirubin". On top right side, there is "Alkaline Phosphatase". On the bottom middle portion, there are "Total Proteins" and "Albumin".

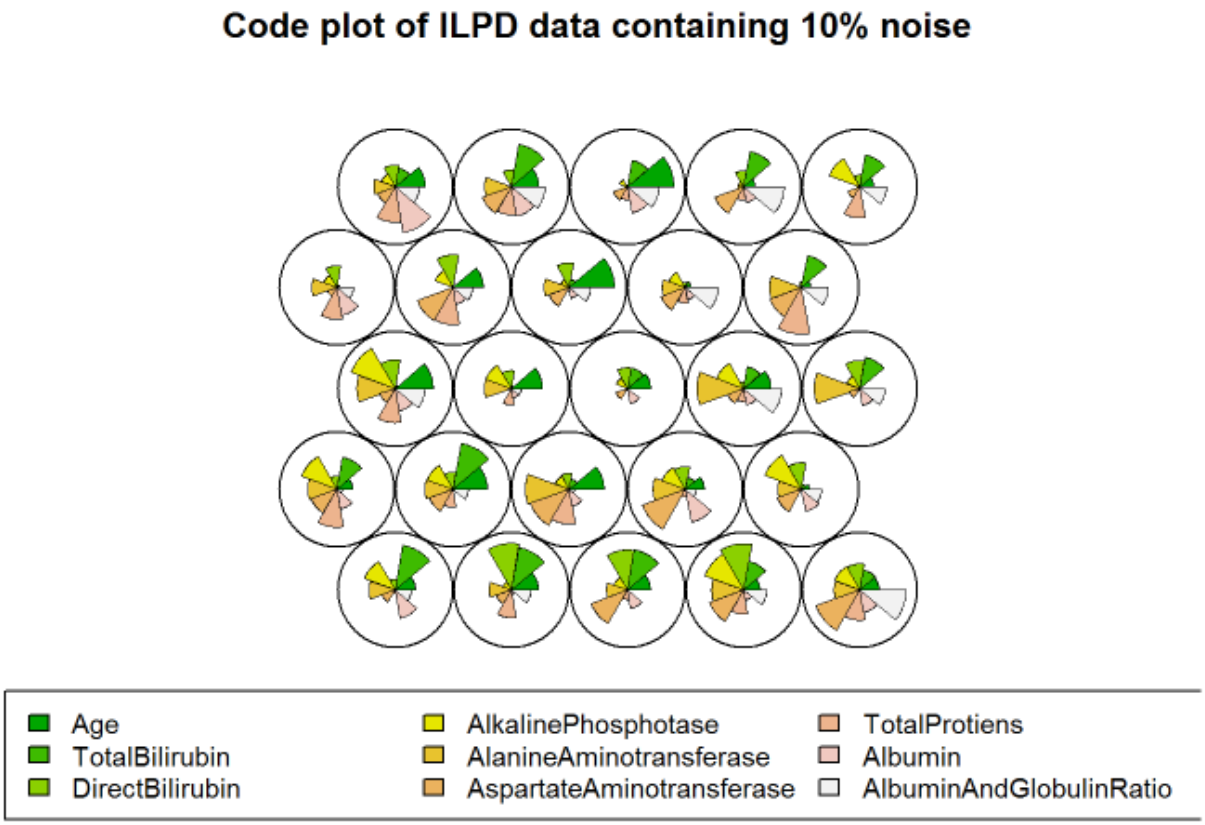

Fig.8: The above code plot shows clear distribution of some variables. "Total Bilirubin" is on bottom left side. "Albumin and Globulin ratio" and "Total Proteins" are on bottom right side. "Total Proteins" and "Albumin" are on top left side. "Age" is distributed on top middle position. "Alkaline Phosphatase" is on the left middle side.

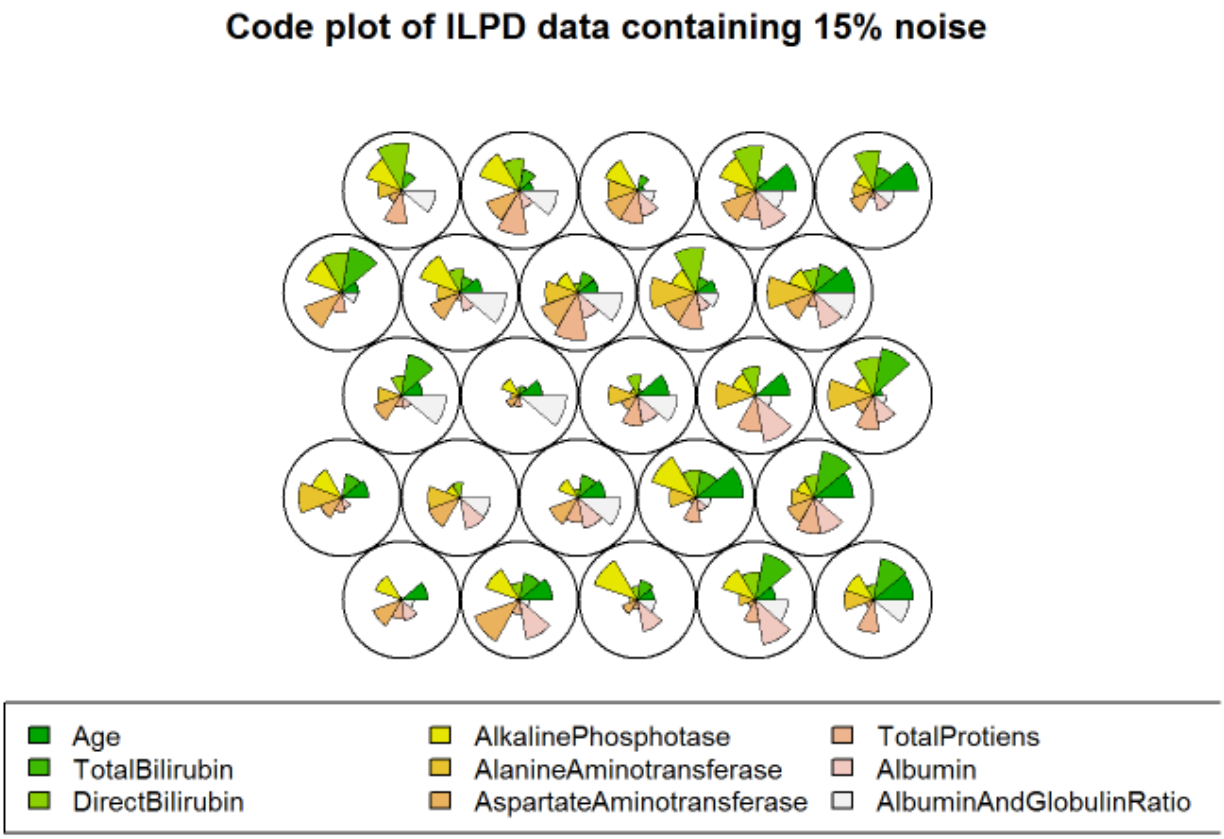

Fig.9: The plot shows distribution of variables in ILPD data when 15% noise is added. "Age" and "Total Bilirubin" are on the right side of the plot. "Total Proteins" and "Aspartate Aminotransferase" are on the bottom side. Left side of the plot show "Alanine Aminotransferase" and "Aspartate Aminotransferase".

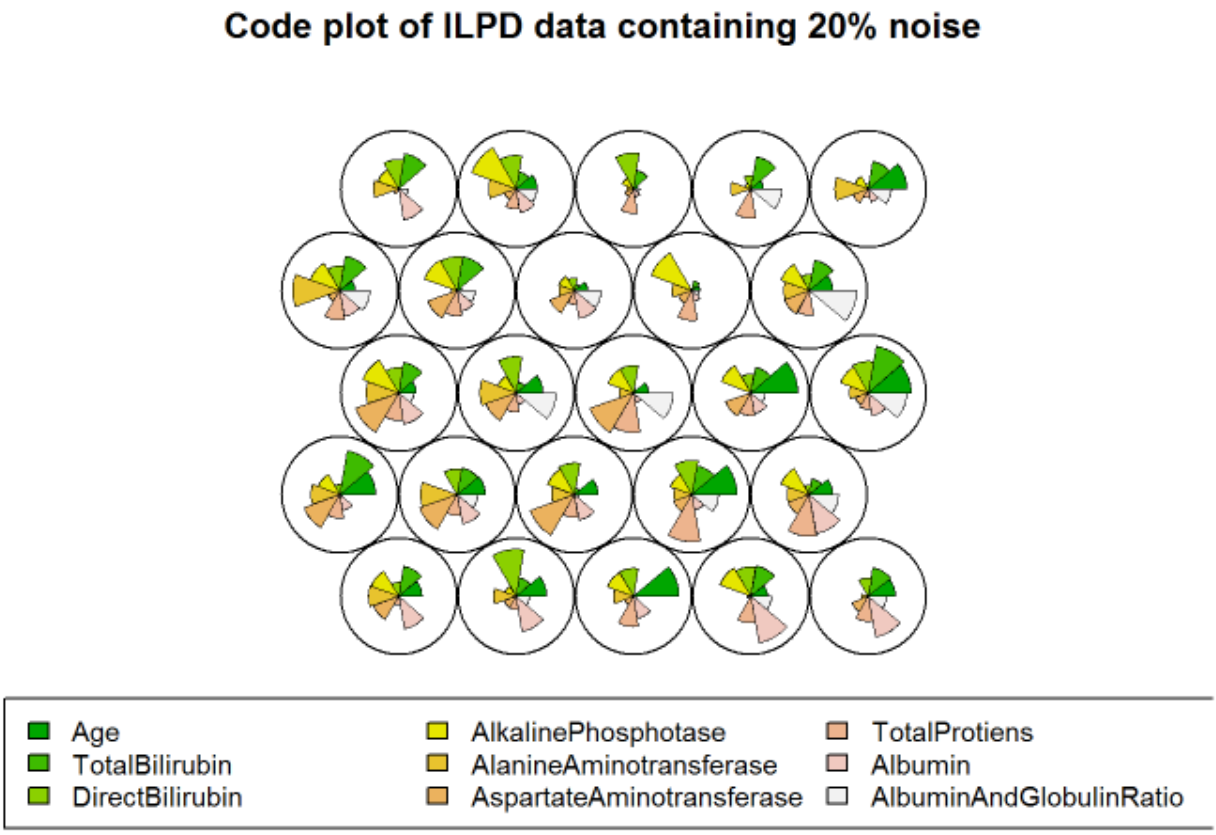

Fig.10: Above is the codes plot showing distribution of ILPD data containing 20% noise. On the left side of the plot, there are "Alanine Aminotransferase" and "Aspartate Aminotransferase". On the bottom side, there are "Total Proteins" and "Albumin".

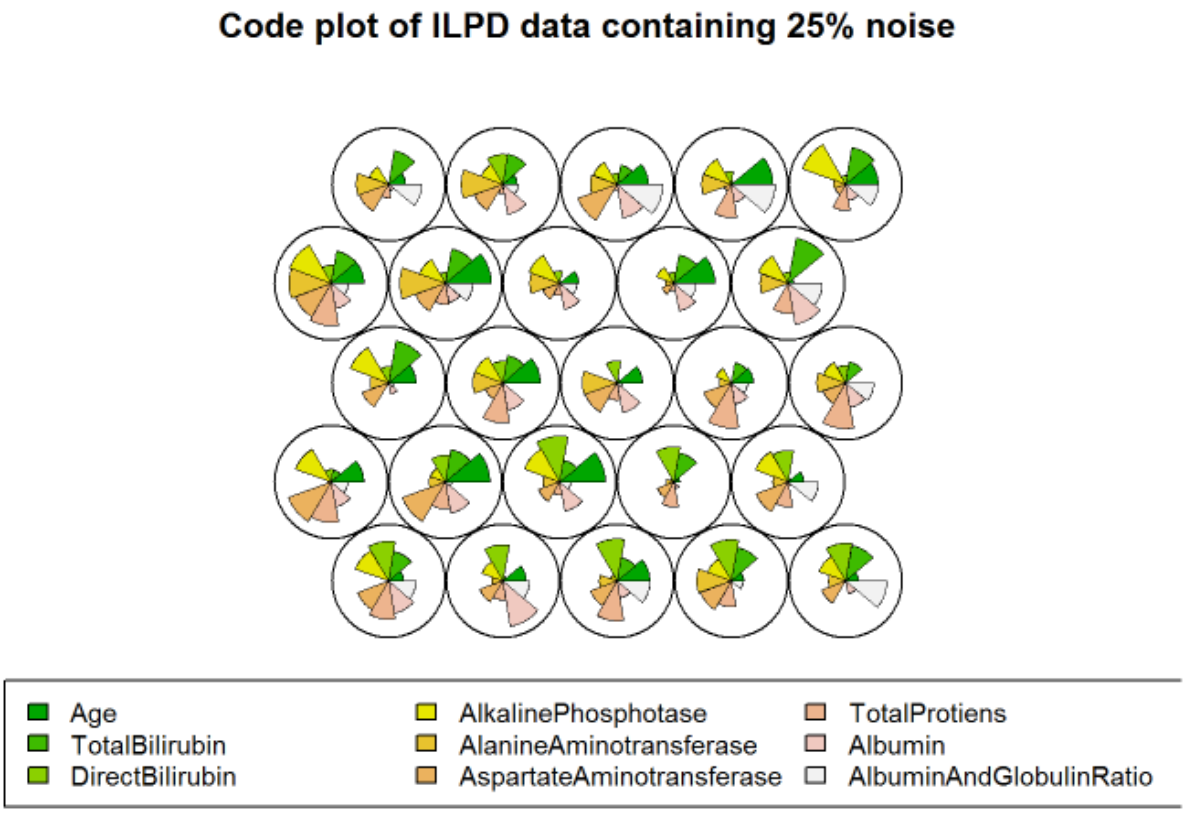

Fig.11: This Code plot shows variables' distribution in the ILPD data containing 25% noise. It does not show a good distribution of the data variables.

Following is the heat map of variables in ILPD data

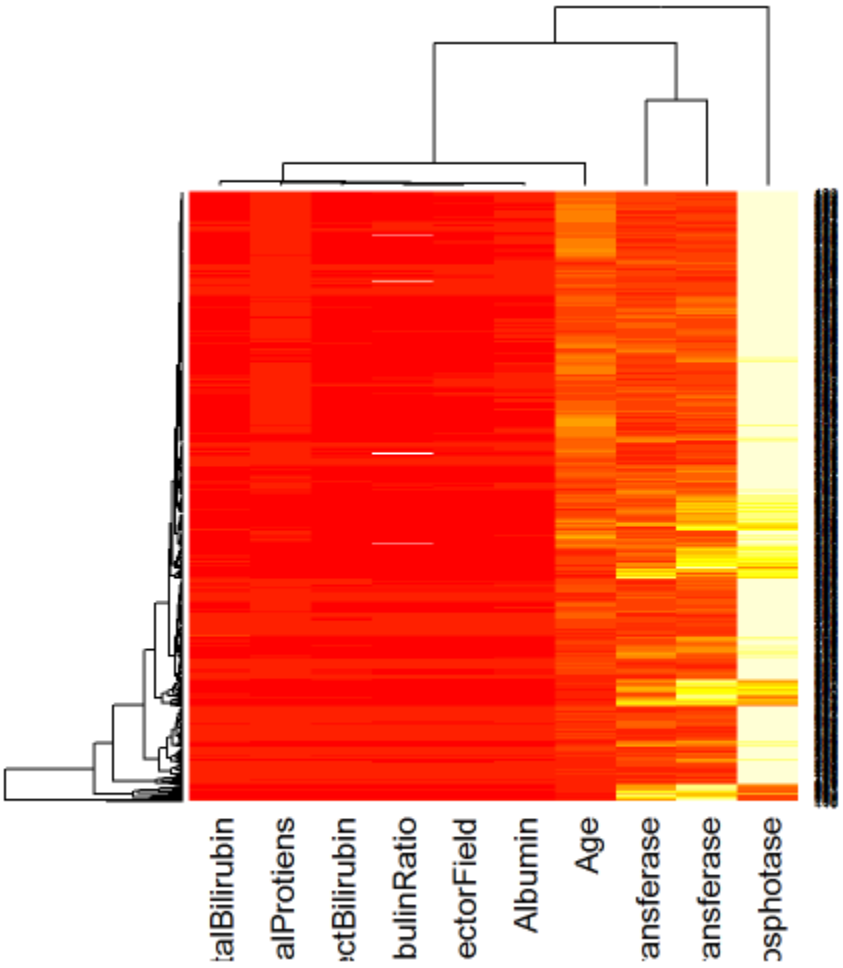

Fig.12: The above heat map shows gradient distribution of variables in the ILPD data. The gradient is prominently seen in case of variables Age, Alanine Aminotransferase, Aspartate Aminotransferase and Alkaline Phosphatase. It does show the distribution SelectorField, which is the outcome variable having categories "patients" and "non-patients".

Results of EDA on the ILPD data

First, gender distribution is considered:

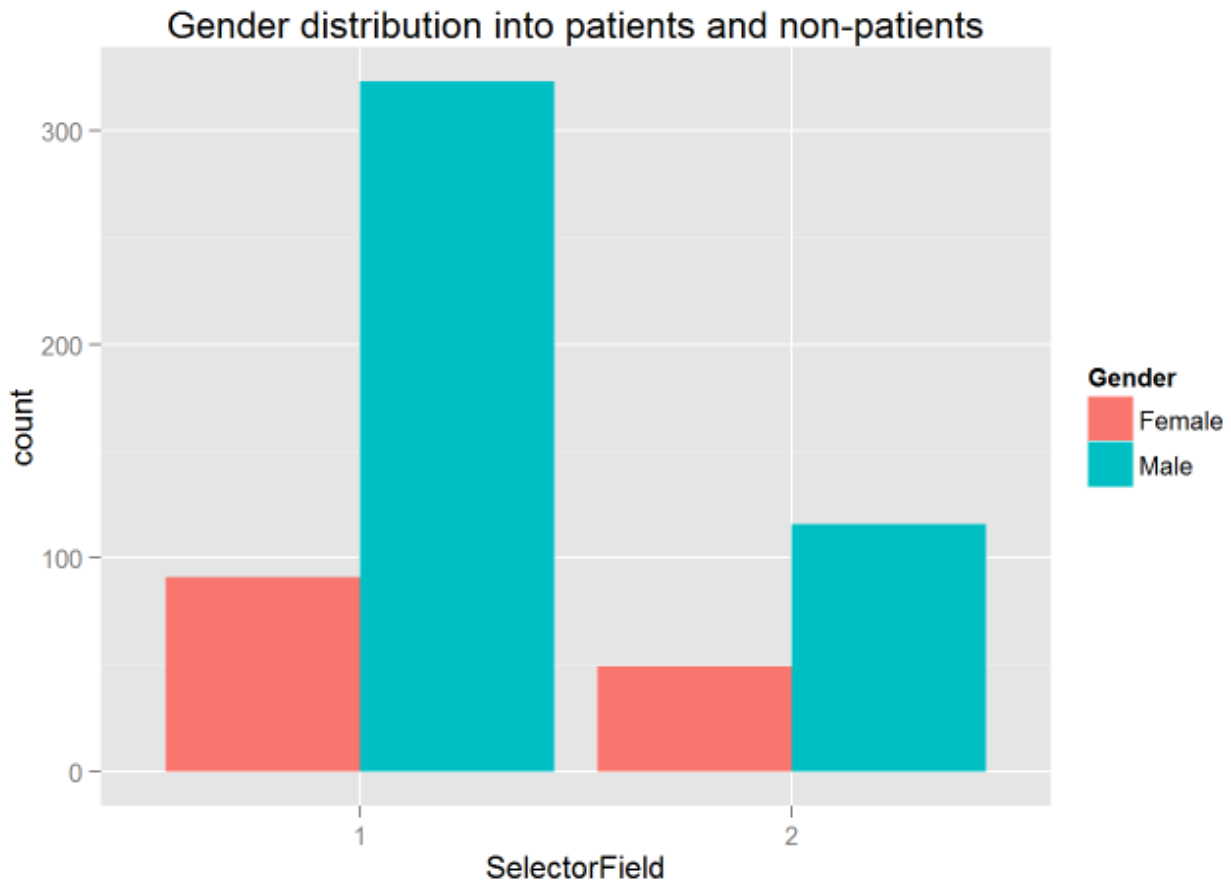

Fig.13: The above plot shows gender distribution into the two categories 1-patients, and 2-non patients.

Next, age distribution is considered:

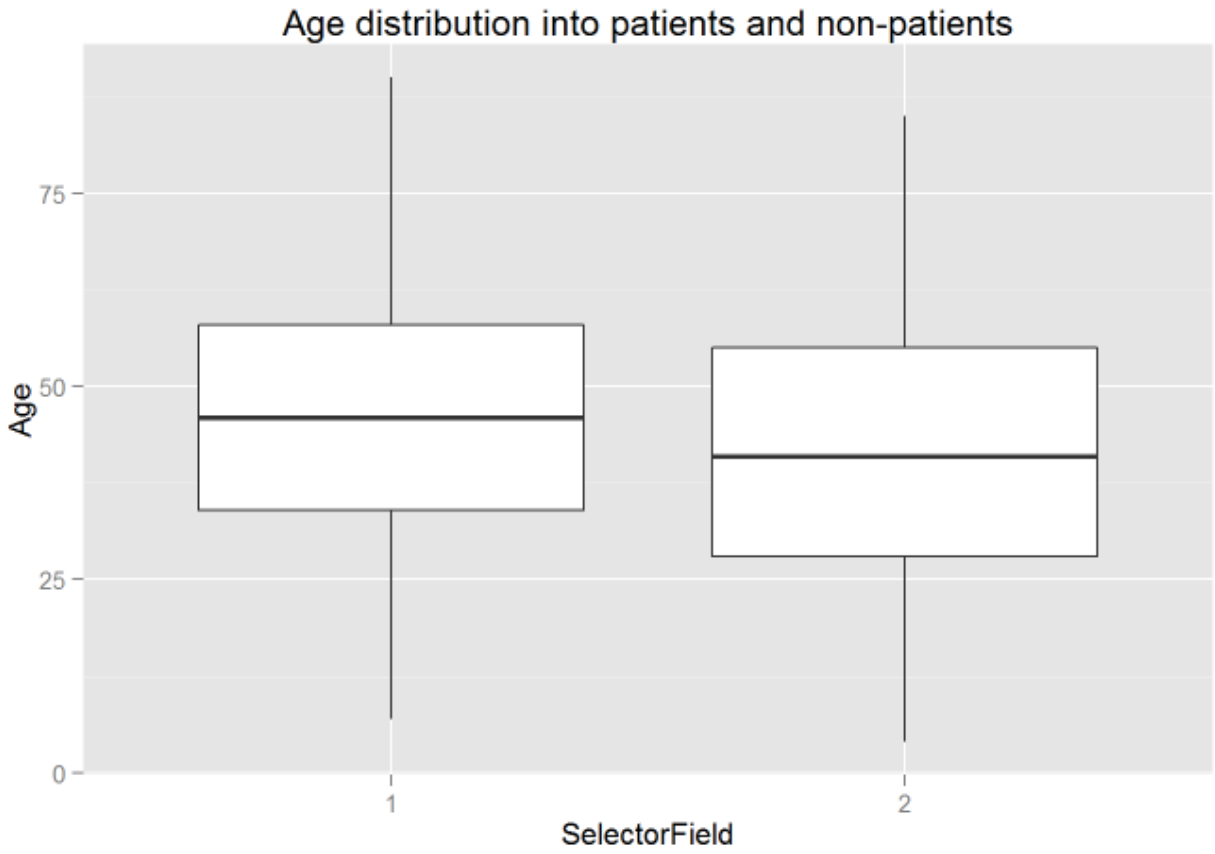

Fig. 14: The above boxplot shows age distribution into 2 categories 1-patients, 2-non patients.

Result of EDA on liver.csv data

Range of ALP after giving dose A = 31-105

Range of ALP after giving dose B = 1-179

Range of ALP after giving dose C = 22-164

Range of ALP after giving dose D = 1-341

Range of ALT after giving dose A = 6-54

Range of ALT after giving dose B = 2-78

Range of ALT after giving dose C = 6-71

Range of ALT after giving dose D = 7-324

Range of AST after giving dose A = 9-37

Range of AST after giving dose B = 6-56

Range of AST after giving dose C = 10-51

Range of AST after giving dose D =10-250

Range of TBL after giving dose A = 3.249 23.085

Range of TBL after giving dose B = 3.420 25.992

Range of TBL after giving dose C = 4.275 42.750

Range of TBL after giving dose D = 3.933 39.843

Result of PCA on ILPD data

Following is the pairs plot, showing correlations of variables among themselves:

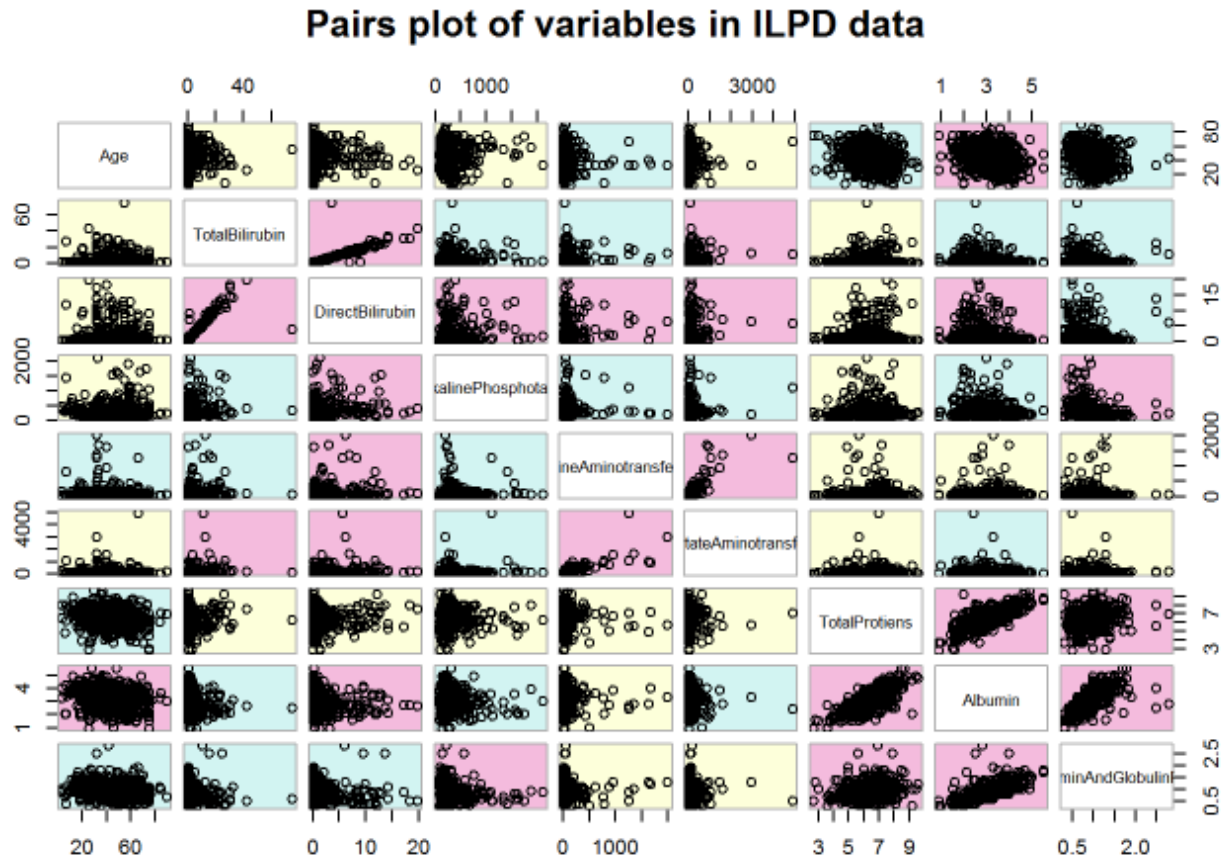

Fig.15: The above plot shows relationship of variables in ILPD data. Those which are strongly correlated are shown in dark color and those having weak correlation are shown in a light color.

Following is the summary of applying PCA on ILPD data:

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
## Standard deviation     1.6594914 1.4236792 1.1685377 0.9787747 0.91905411
## Proportion of Variance 0.3059902 0.2252069 0.1517200 0.1064444 0.09385116
## Cumulative Proportion  0.3059902 0.5311971 0.6829171 0.7893616 0.88321272
##                            Comp.6     Comp.7     Comp.8      Comp.9
## Standard deviation     0.81632677 0.45105044 0.35470439 0.235445205
## Proportion of Variance 0.07404327 0.02260517 0.01397947 0.006159383
## Cumulative Proportion  0.95725598 0.97986115 0.99384062 1.000000000
```

Following is the screeplot of PCA on ILPD data:

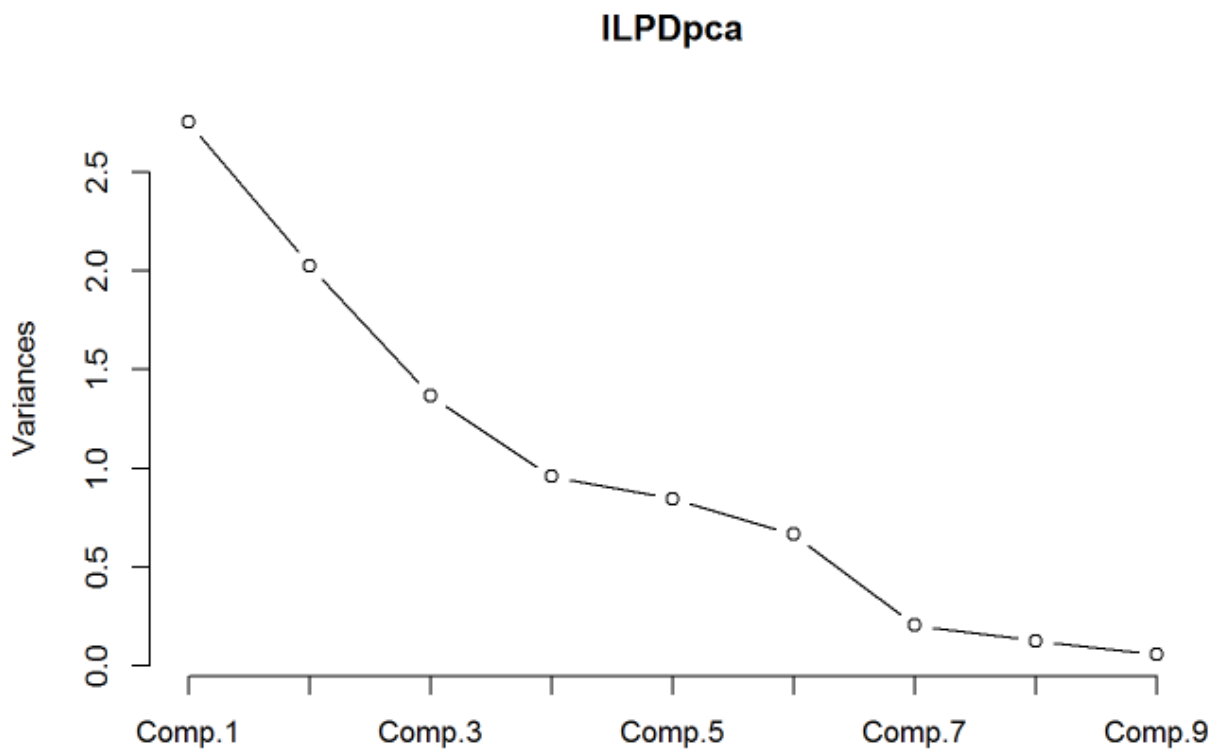

Fig.16: This is the screeplot showing distribution of 9 principal components. There is a sharp decrease in line after component 6.

Next is the biplot:

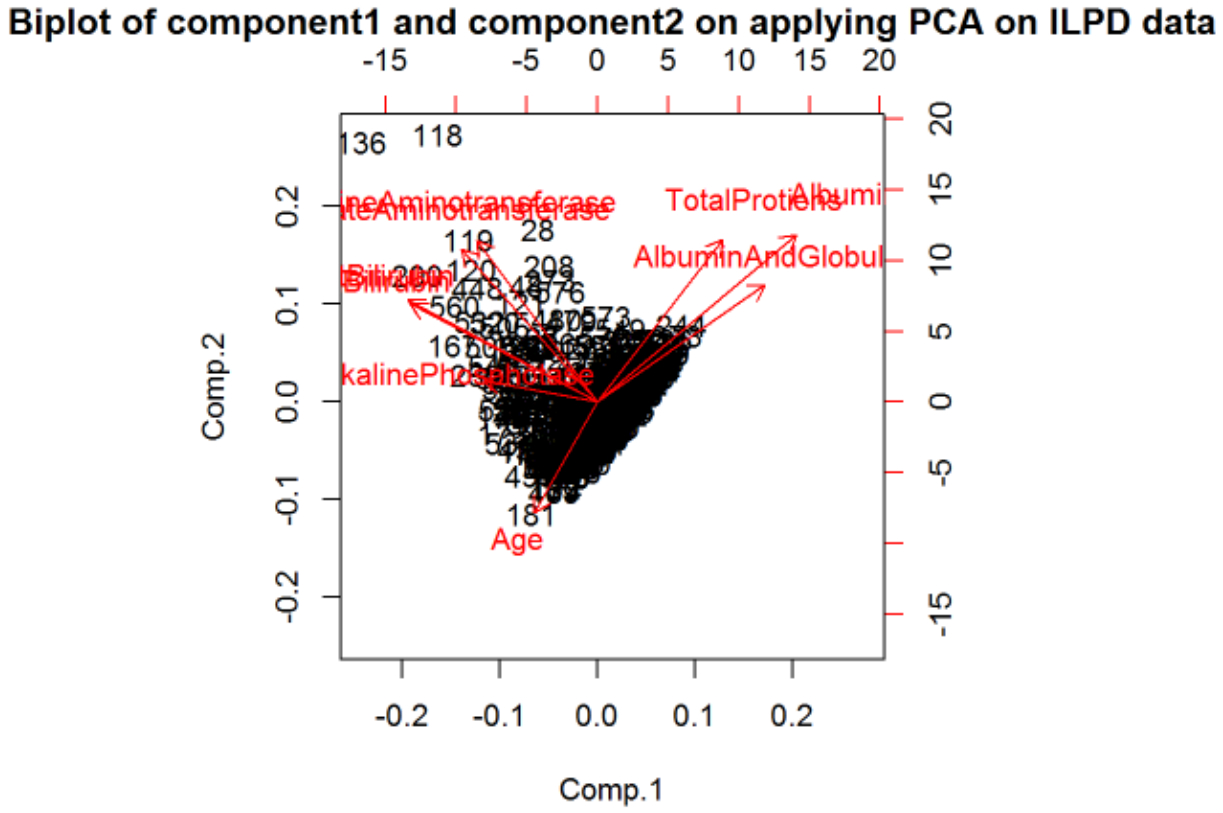

Fig.17: The above biplot shows distribution of variables of ILPD data, between component 1 and component 2.

Application PCA on liver.csv data

Following is the pairs plot:

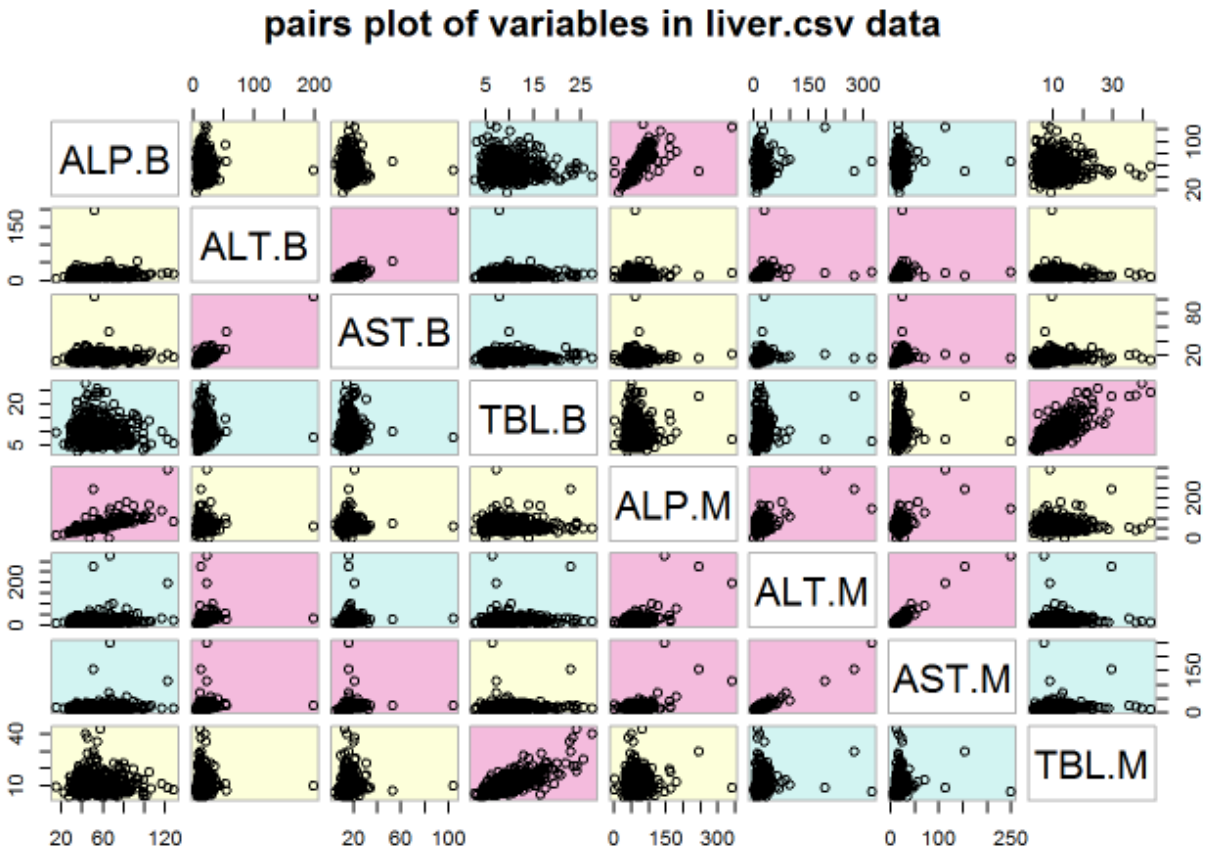

Fig.18: The above plot shows relationship of variables in liver.csv data. Those which are strongly correlated are shown in dark color and those having weak correlation are shown in a light color.

Following is the summary of applying PCA on liver.csv data

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5    PC6     PC7
## Standard deviation     1.6278 1.3741 1.2721 1.0986 0.48443 0.4630 0.39418
## Proportion of Variance 0.3312 0.2360 0.2023 0.1509 0.02933 0.0268 0.01942
## Cumulative Proportion  0.3312 0.5672 0.7695 0.9204 0.94971 0.9765 0.99593
##                           PC8
## Standard deviation     0.18046
## Proportion of Variance 0.00407
## Cumulative Proportion  1.00000
```

Following is the screeplot after applying PCA on liver.csv

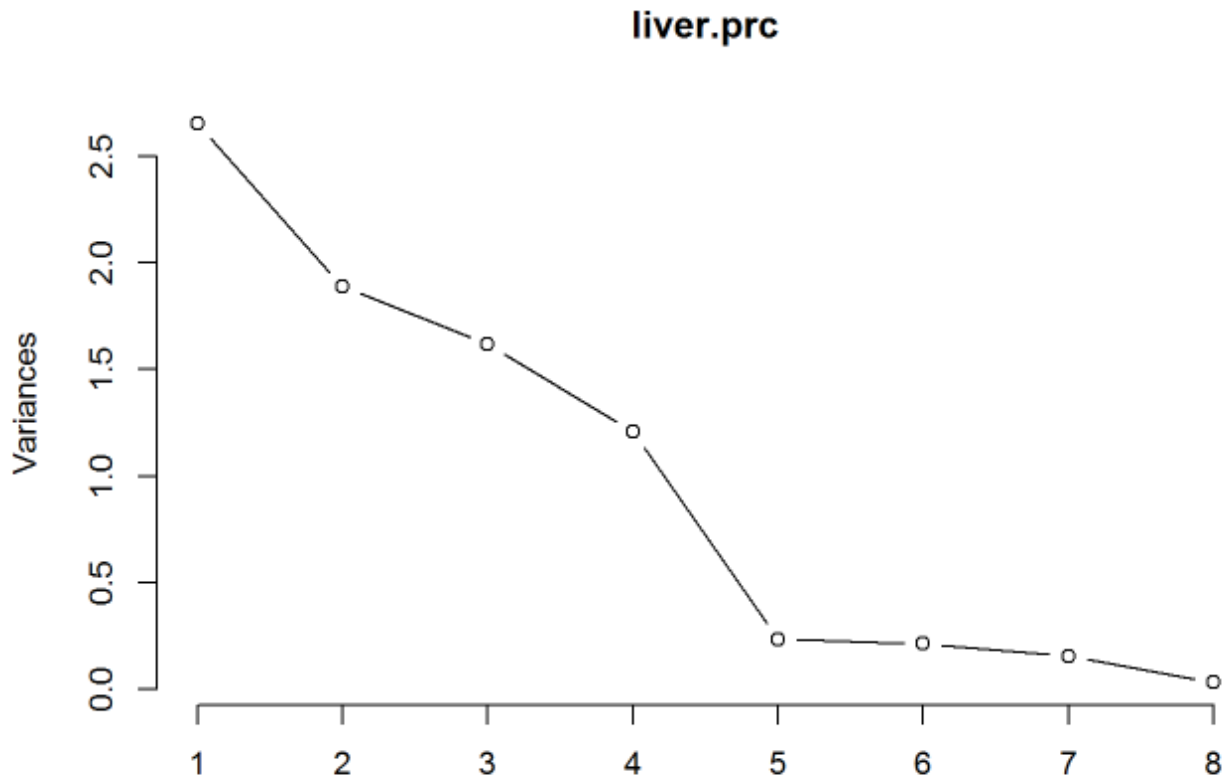

Fig.19: This is the screeplot showing distribution of 8 principal components. There is a sharp decrease in the line after component 4.

Next, is the biplot:

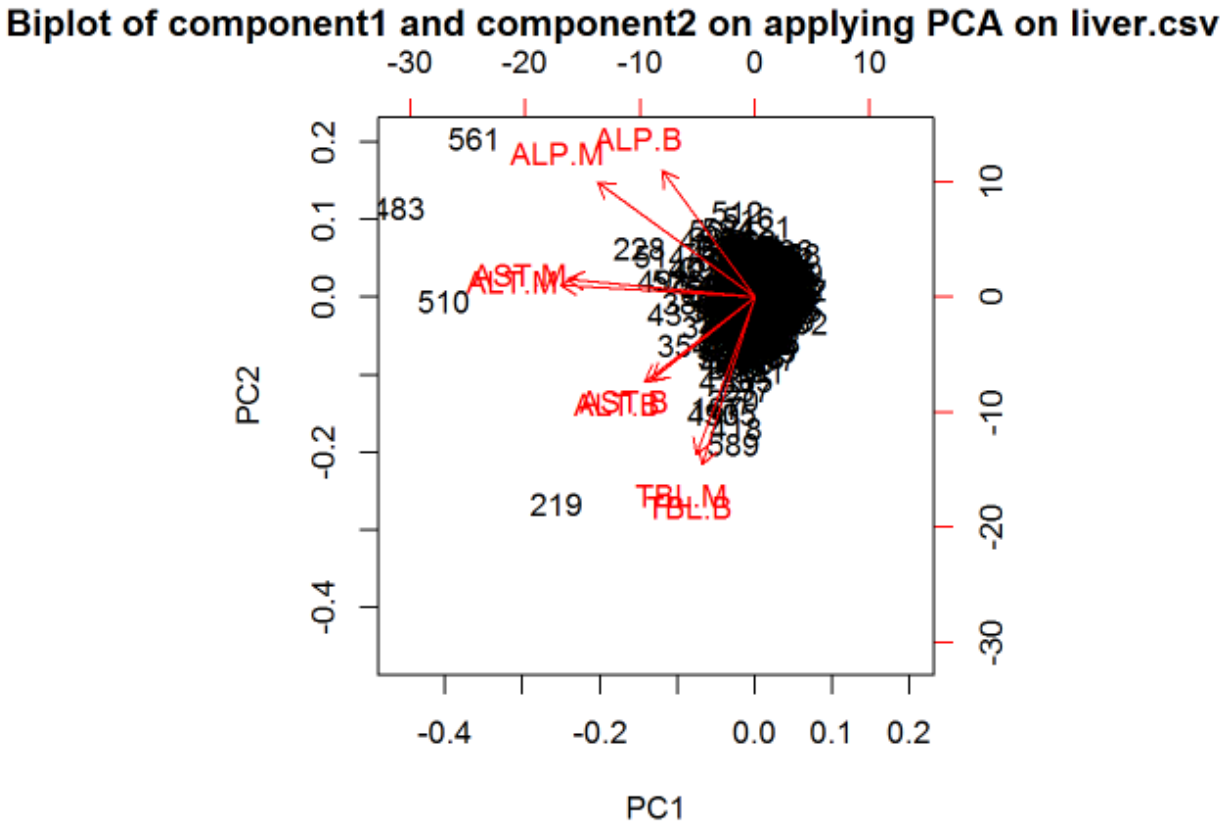

Fig.20: The above biplot shows distribution of variables of liver.csv data, between component 1 and component 2.

Discussions:

Discussion of applying KNN algorithm on ILPD data:

Pros of KNN application on ILPD data:

1. The assumptions implied by KNN algorithm were implied by just the distance function.

2. It was easy to form training data set and update the same.

3. The algorithm performed well in classifying "patients" in the data set ILPD.

Cons of applying KNN algorithm on ILPD data set:

1. It needed handling of missing data.

2. It needed to normalize the data since it is sensitive to outliers.

3. It performed poorly in classifying "non-patients" in the data set ILPD.

4. It did not respond to the range of noise added to perform sensitivity analysis.

Discussion of applying LDA algorithm on ILPD data:

Pros of LDA application on ILPD data:

1. It was possible to add or remove variables while applying the LDA algorithm. It was not necessary to pre-clean the data.

2. It made possible to consider the loadings in the discriminant function, to derive the contrast among the concentrations of different variables.

3. The algorithm performed well in classifying "patients" in the ILPD data.

Cons of LDA algorithm:

1. It performed poorly in classifying "non-patients" in the ILPD data.

2. It did not respond to the range of noise added to perform sensitivity analysis.

The reason behind poor classification of "non-patients", by KNN and LDA algorithms seems to be higher number of "patients" (414 patients) than that of "non-patients" (165 non-patients). Another reason seems to be the use of all the variables in the data while performing KNN and LDA algorithm. Instead, combination of different variables which are most crucial in liver diseases would yield better results.

Discussion of applying SOM algorithm on ILPD data:

Pros of using SOM algorithm:

1. Relatively simple algorithm, so that results can be explained to a non data scientist [10].

2. It provided good visualization of the variables in ILPD data.

3. It showed varied response to the data containing varying degree of noise.

4. The heat map showed fair distribution of variables in the form of a color gradient.

Cons of applying SOM algorithm on ILPD data:

1. It requires clean, numeric data [10].

2. It is difficult to represent too many variables in two dimentional space [10].

Discussion of performing EDA on ILPD data:

From the plots in EDA, there are approximately 90 female and 320 male liver patients. Thus the percentage of female liver patients is approximately (90/142) x 100 = 63.38%. Whereas that of male liver patients is (320/441) x 100 = 72.56%.

So, according to ILPD data set, male patients are most likely prone to liver disease.

In case of age distribution, according to results of boxplot, the median age for liver patients is approximately 47 years and that of non-liver patients is 40 years. Therefore there is not much difference in the age group distribution of patients and non-patients.

Discussion of applying EDA on liver.csv to determine optimal dose of a drug:

It was observed that dose A produced best results in terms of levels of blood components after treatment with the drug. Therefore, dose A is the optimal dose of the given Astrazeneca drug. The optimal dose was determined by referring to the normal values of blood components.

The normal ranges of blood components are as follows:

Range of ALT in males : 10-40 U/L

Range of ALT in females : 7-35 U/L[11]

Range of ALP: The normal range is 44 to 147 IU/L[12]

Range of AST in males : 14-20 U/L

Range of AST in females : 10-36 U/L [13]

Range of TBL : 5.1 to 17 umol/L (SI units)[14]

Discussion of applying PCA on ILPD and liver.csv data sets:

In case of liver.csv data, considering the first principal component results of rotation and biplot, there is not much contrast between the variables. However in second PCA, variables ALP.B (Alkaline phosphatase at baseline), ALP.M (Alkaline phosphatase after treatment), AST.M (Aspartate aminotransferase after treatment), ALT.M (Alanine aminotransferase after treatment) are contrasted with AST.B (Aspartate aminotransferase at baseline), ALT.b (Alanine aminotransferase at baseline), TBL.M (Total bilirubin after treatment), TBL.B (Total bilirubin at baseline).

This is a good indication since in most cases the base levels of variables are different than the levels after treatment. Therefore the drug seems to be effective.

Also the results of PCA on ILPD data and liver data match to a greater extent since in both the datasets, Direct Bilirubin, Alkaline phosphatase, Alanine Aminotransferase at the baselines, are on one side and are not contrasted (in case of pca1 in ILPD data and in case of pc2 in liver data).

References

[1] Bendi Venkata Ramana, Prof. M Surendra Prasad Babu, Prof. N. B. Venkateswarlu, A critical study of selected classification algorithms for liver disease diagnosis, May 2011, Vol. 3, No. 2

[2] Indian liver patient data set - https://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset)

[3] liver.csv - https://vincentarelbundock.github.io/Rdatasets/datasets.html

[4] R programming language - https://en.wikipedia.org/wiki/R_(programming_language)

[5]KNN algorithm - https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[6] LDA algorithm - https://en.wikipedia.org/wiki/Linear_discriminant_analysis

[7] SOM algorithm - https://en.wikipedia.org/wiki/Self-organizing_map

[8] Heat map - https://stat.ethz.ch/R-manual/R-devel/library/stats/html/heatmap.html

[9] EDA - https://en.wikipedia.org/wiki/Exploratory_data_analysis

[10] Pros and cons of SOM - http://www.r-bloggers.com/self-organising-maps-for-customer-segmentation-using-r/

[11] http://www.webmd.com/digestive-disorders/alanine-aminotransferase-alt?page=2

[12] https://www.nlm.nih.gov/medlineplus/ency/article/003470.htm

[13] http://www.webmd.com/digestive-disorders/aspartate-aminotransferase-ast?page=2

[14] http://www.sw.org/HealthLibrary?page=Bilirubin