Title: PH125X9 Capston Project II: Prediction of liver disease

by JING JING SUN

OCT. 19, 2021

I.Introduction

Prediction of the human disease based on clinic laboratory data is possible as data science developing. It will play an important role in public health, estimate the risk and diagnosis of disease.

I downloaded a India Liver Patient Dataset(ILPD) from the University of California , Irvine (UCI) machine learning repository(https://archive.ics.uci.edu/ml/machine-learning-databases/00225/Indian%20Liver%20Patient%20Dataset%20(ILPD).csv).

This data set is collected from North East of Andhra Pradesh, India. There are 416 liver patient records and 167 non-liver patient records. Data set columns include Age of the patient, Gender of the patient, total Bilirubin, Direct Bilirubin, Alkaline phosphatase, Alamine Aminotransferase(ALT), Aspatate Aminotransferase(AST), Total protein, Albumin and Albumin and Globulin ratio.

The goal of this project is to create a disease prediction system use patient laboratory test result to create a disease prediction system, with the knowledge we have learned from R course series and help us get the ability to contribute to real world.

II. Method/Analysis

loading needed packages

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.3     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dslabs)
library(matrixStats)
```

```
##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
##
##     count
```

```r
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(dplyr)
```

** data clean and organization ## load dataset**

I go to the University of California , Irvine website, read the a India Liver Patient Dataset(ILPD).

```r
liver_data <- read.csv("https://archive.ics.uci.edu/ml/machine-learning-databases/00225/Indian%20Liver%
                       header = FALSE)
head(liver_data)
```

```
##   V1     V2   V3  V4  V5 V6  V7  V8  V9  V10 V11
## 1 65 Female  0.7 0.1 187 16  18 6.8 3.3 0.90   1
## 2 62   Male 10.9 5.5 699 64 100 7.5 3.2 0.74   1
## 3 62   Male  7.3 4.1 490 60  68 7.0 3.3 0.89   1
## 4 58   Male  1.0 0.4 182 14  20 6.8 3.4 1.00   1
## 5 72   Male  3.9 2.0 195 27  59 7.3 2.4 0.40   1
## 6 46   Male  1.8 0.7 208 19  14 7.6 4.4 1.30   1
```

give each column a name

I noticed there are no column name in the data set, so I gave each column a name

```r
colnames(liver_data) <- c("Age", "Sex", "Tot_Bil", "Dir_Bil", "Alkphos", "Alamine",
                          "Aspartate", "Tot_Prot", "Albumin", "A_G_Ratio", "Disease")
```

convert value in column disease into 0 (no disease) and 1 (liver disease)

In order to process data later, I use zero and one to represent no disease and with liver disease.

```
liver_data$Disease <- as.numeric(ifelse(liver_data$Disease == 2, 0, 1))
head(liver_data)
```

```
##   Age    Sex Tot_Bil Dir_Bil Alkphos Alamine Aspartate Tot_Prot Albumin
## 1  65 Female     0.7     0.1     187      16        18      6.8     3.3
## 2  62   Male    10.9     5.5     699      64       100      7.5     3.2
## 3  62   Male     7.3     4.1     490      60        68      7.0     3.3
## 4  58   Male     1.0     0.4     182      14        20      6.8     3.4
## 5  72   Male     3.9     2.0     195      27        59      7.3     2.4
## 6  46   Male     1.8     0.7     208      19        14      7.6     4.4
##   A_G_Ratio Disease
## 1      0.90       1
## 2      0.74       1
## 3      0.89       1
## 4      1.00       1
## 5      0.40       1
## 6      1.30       1
```

add column "liverdisease" as.factor "0" represent "nodisease", "1" represent "liverdisease"

I also add a column to as.factor zero and one to represent no disease and liver disease.

```
liver_data<- liver_data %>% mutate ( liver_data, liverdisease= as.factor (Disease))
head(liver_data)
```

```
##   Age    Sex Tot_Bil Dir_Bil Alkphos Alamine Aspartate Tot_Prot Albumin
## 1  65 Female     0.7     0.1     187      16        18      6.8     3.3
## 2  62   Male    10.9     5.5     699      64       100      7.5     3.2
## 3  62   Male     7.3     4.1     490      60        68      7.0     3.3
## 4  58   Male     1.0     0.4     182      14        20      6.8     3.4
## 5  72   Male     3.9     2.0     195      27        59      7.3     2.4
## 6  46   Male     1.8     0.7     208      19        14      7.6     4.4
##   A_G_Ratio Disease liverdisease
## 1      0.90       1            1
## 2      0.74       1            1
## 3      0.89       1            1
## 4      1.00       1            1
## 5      0.40       1            1
## 6      1.30       1            1
```

data set exploration

```
str(liver_data)
```

```
## 'data.frame':    583 obs. of  12 variables:
##  $ Age         : int  65 62 62 58 72 46 26 29 17 55 ...
##  $ Sex         : chr  "Female" "Male" "Male" "Male" ...
##  $ Tot_Bil     : num  0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
```

```
##  $ Dir_Bil     : num  0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ Alkphos     : int  187 699 490 182 195 208 154 202 202 290 ...
##  $ Alamine     : int  16 64 60 14 27 19 16 14 22 53 ...
##  $ Aspartate   : int  18 100 68 20 59 14 12 11 19 58 ...
##  $ Tot_Prot    : num  6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ Albumin     : num  3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ A_G_Ratio   : num  0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ Disease     : num  1 1 1 1 1 1 1 1 0 1 ...
##  $ liverdisease: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 1 2 ...
```

```
dim(liver_data)
```

```
## [1] 583  12
```

This data set have 583 row and 12 column.

remove those rows that have missing data

```
liver_data<-na.omit(liver_data)
dim(liver_data)
```

```
## [1] 579  12
```

Now this data set have 579 row and 12 column. We removed 4 rows from the dataset.

know our cleaned data

```
summary(liver_data)
```

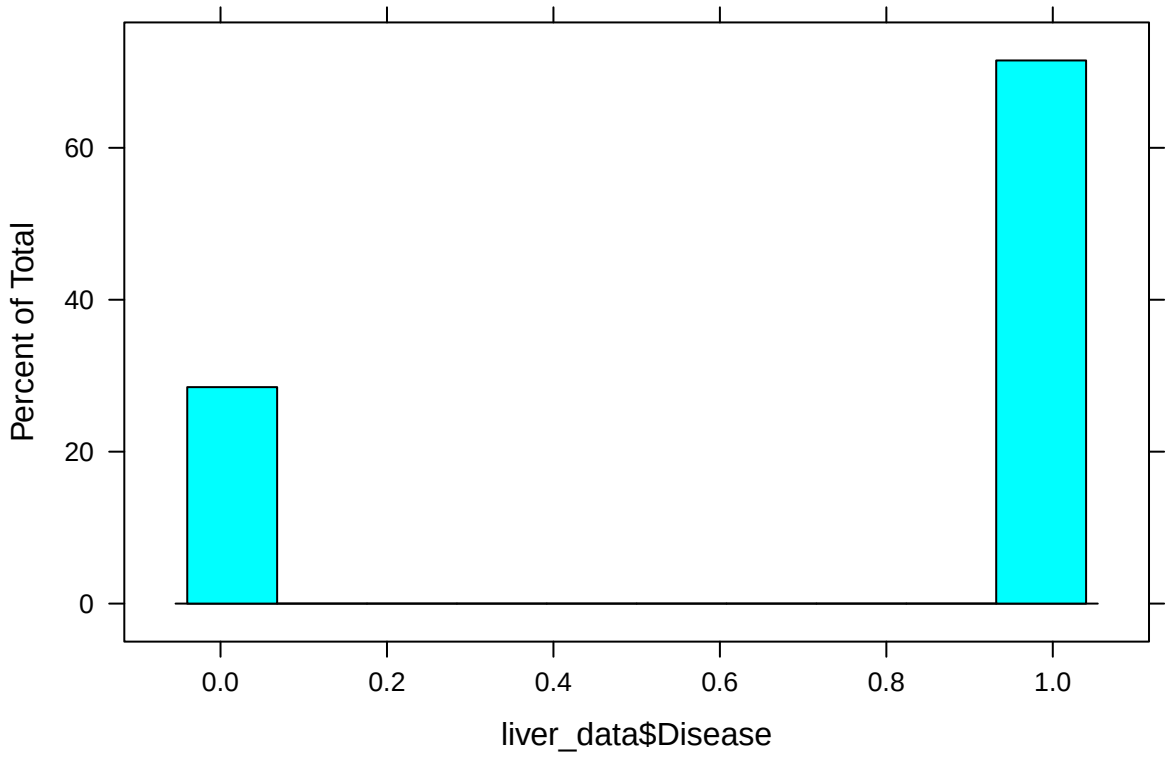
```
##       Age            Sex               Tot_Bil          Dir_Bil
##  Min.   : 4.00   Length:579         Min.   : 0.400   Min.   : 0.100
##  1st Qu.:33.00   Class :character   1st Qu.: 0.800   1st Qu.: 0.200
##  Median :45.00   Mode  :character   Median : 1.000   Median : 0.300
##  Mean   :44.78                      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
##     Alkphos          Alamine          Aspartate         Tot_Prot
##  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   : 291.4   Mean   :  81.13   Mean   : 110.4   Mean   :6.482
##  3rd Qu.: 298.0   3rd Qu.:  61.00   3rd Qu.:  87.0   3rd Qu.:7.200
##  Max.   :2110.0   Max.   :2000.00   Max.   :4929.0   Max.   :9.600
##     Albumin        A_G_Ratio         Disease      liverdisease
##  Min.   :0.900   Min.   :0.3000   Min.   :0.000   0:165
##  1st Qu.:2.600   1st Qu.:0.7000   1st Qu.:0.000   1:414
##  Median :3.100   Median :0.9300   Median :1.000
##  Mean   :3.139   Mean   :0.9471   Mean   :0.715
##  3rd Qu.:3.800   3rd Qu.:1.1000   3rd Qu.:1.000
##  Max.   :5.500   Max.   :2.8000   Max.   :1.000
```

how many kind of liver disease in this data set

```
n_distinct(liver_data$Disease)
```

```
## [1] 2
```

```
histogram(liver_data$Disease)
```



There are only 2 situations: no disease(0), liver disease(1).

how many liver disease case

```
count(liver_data$Disease == "1")
```

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

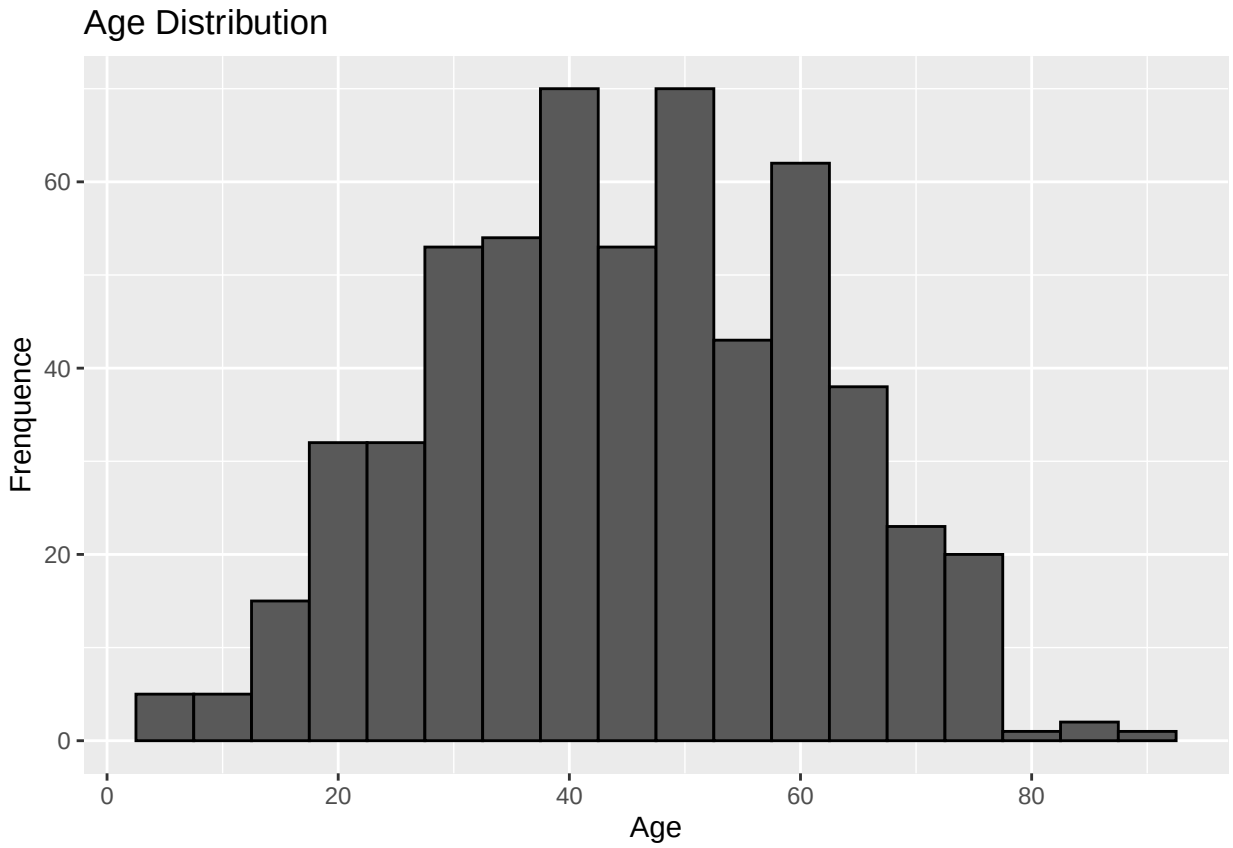
how many no disease case

```
sum(liver_data$Disease == "0")
```

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

distribution of Age

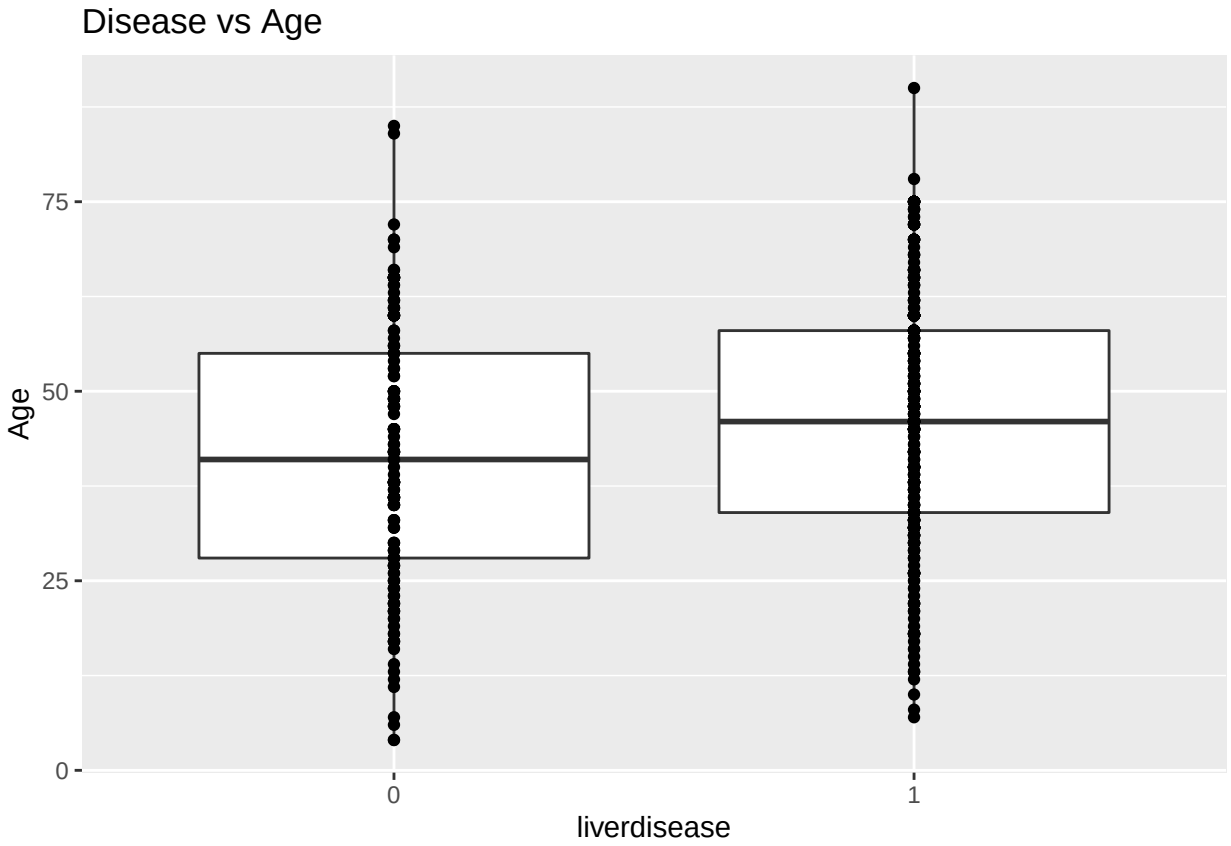
```
liver_data%>% ggplot(aes(Age)) +
  geom_histogram( binwidth = 5,color = "black") +
  scale_x_continuous(breaks=seq(0, 100, by= 20)) +
  labs(x="Age", y="Frenquence") +
  ggtitle("Age Distribution")
```



Liver disease is high incidence between 35-60 years old. liver disese is related to age.

Is liver disease related to Age

```
liver_data %>%
  ggplot(aes(liverdisease, Age))+
  geom_boxplot()+ geom_point()+
  ggtitle("Disease vs Age") +
  xlab("liverdisease") + ylab("Age")
```



People with liver disease are older than those without disease.

Sex distribution in this data set

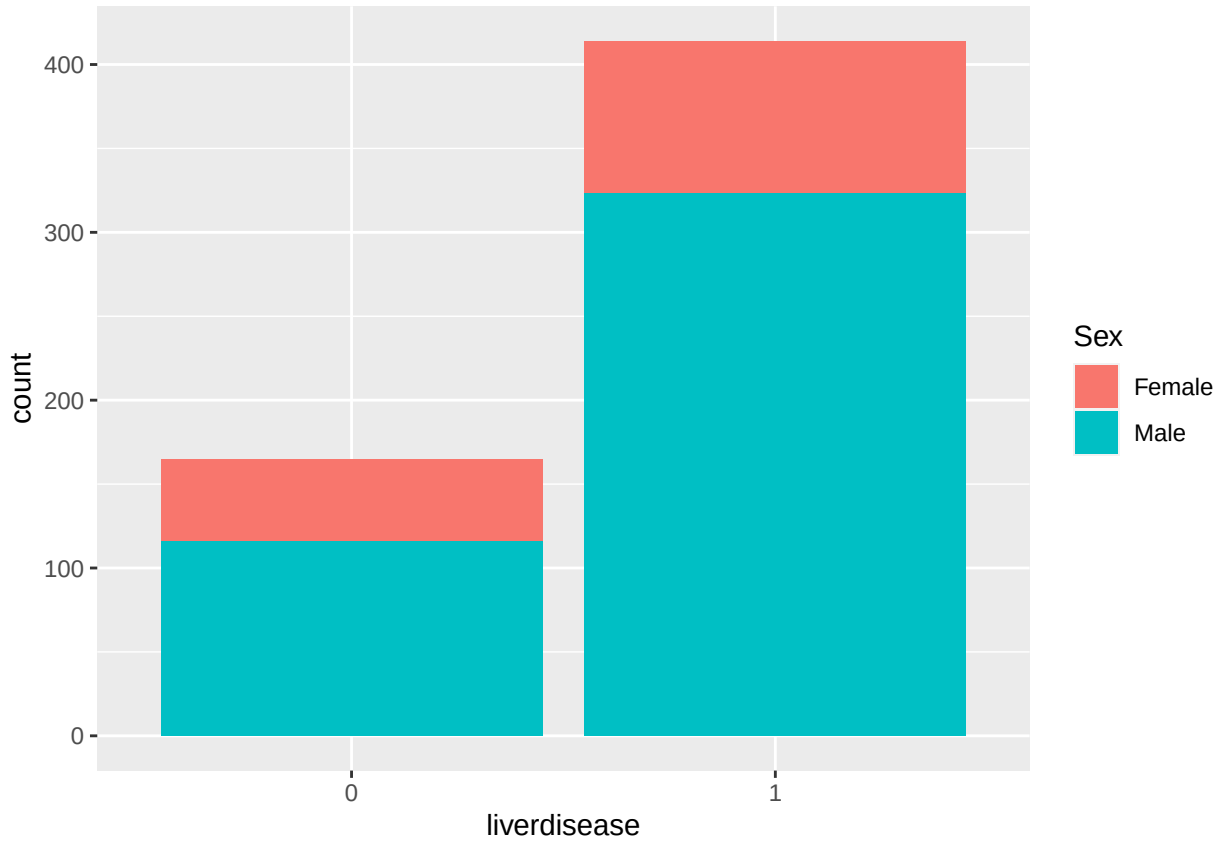
```
liver_data %>% group_by(Sex)%>%
  summarize(n=n())
```

```
## # A tibble: 2 x 2
##   Sex        n
##   <chr>  <int>
## 1 Female   140
## 2 Male     439
```

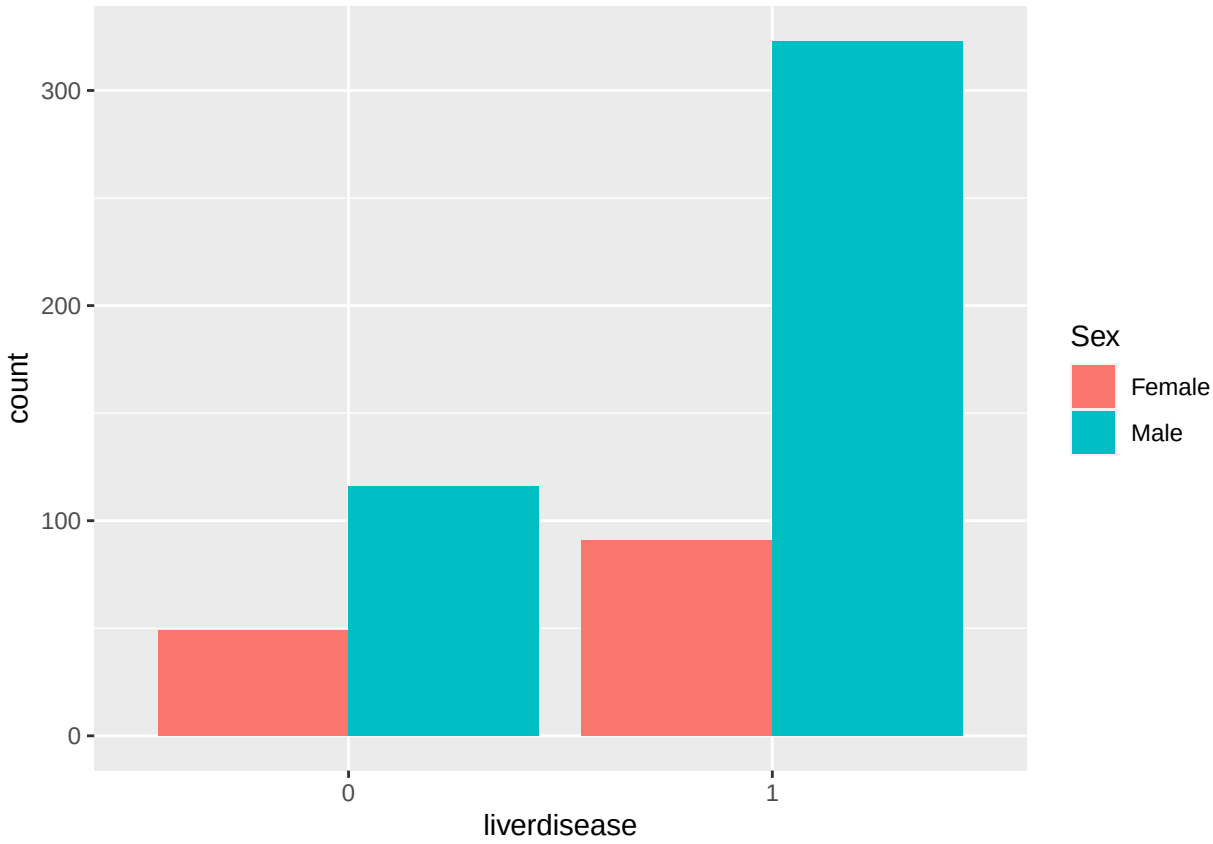
More Male in this data set

liver disease summary by Sex

```
liver_data%>% ggplot(aes( liverdisease, fill= Sex)) +
  geom_bar()
```




```
liver_data%>% ggplot(aes( liverdisease, fill= Sex)) +
  geom_bar(position = position_dodge())
```



Based on the graph, male has high liver disease incident than female.

Male with liver disease

```
liver_dataM<-liver_data %>% filter(Sex %in% 'Male')
Male_liverdiseace<-count(liver_dataM$Disease == "1")
Male_liverdiseace
```

```
## [1] 323
```

Female with liver disease

```
liver_dataF<-liver_data %>% filter(Sex %in% 'Female')
Female_liverdiseace <-count(liver_dataF$Disease == "1")
Female_liverdiseace
```

```
## [1] 91
```

ratio of Male/Female in liver disease

```
Male_liverdiseace/Female_liverdiseace
```

```
## [1] 3.549451
```

Male with no disease

```
Male_nodiseace<-count(liver_dataM$Disease == "0")
Male_nodiseace
```

```
## [1] 116
```

Female with liver disease

```
Female_nodiseace <-count(liver_dataF$Disease == "0")
Female_nodiseace
```

```
## [1] 49
```

Male liver disease incident

```
  Male_incident <- Male_liverdiseace/(Male_liverdiseace+Male_nodiseace)
  Male_incident
```

```
## [1] 0.7357631
```

Female liver incident

```
  Female_incident <- Female_liverdiseace/(Male_liverdiseace+Male_nodiseace)
  Female_incident
```

```
## [1] 0.2072893
```

liver disease incident ratio based on sex

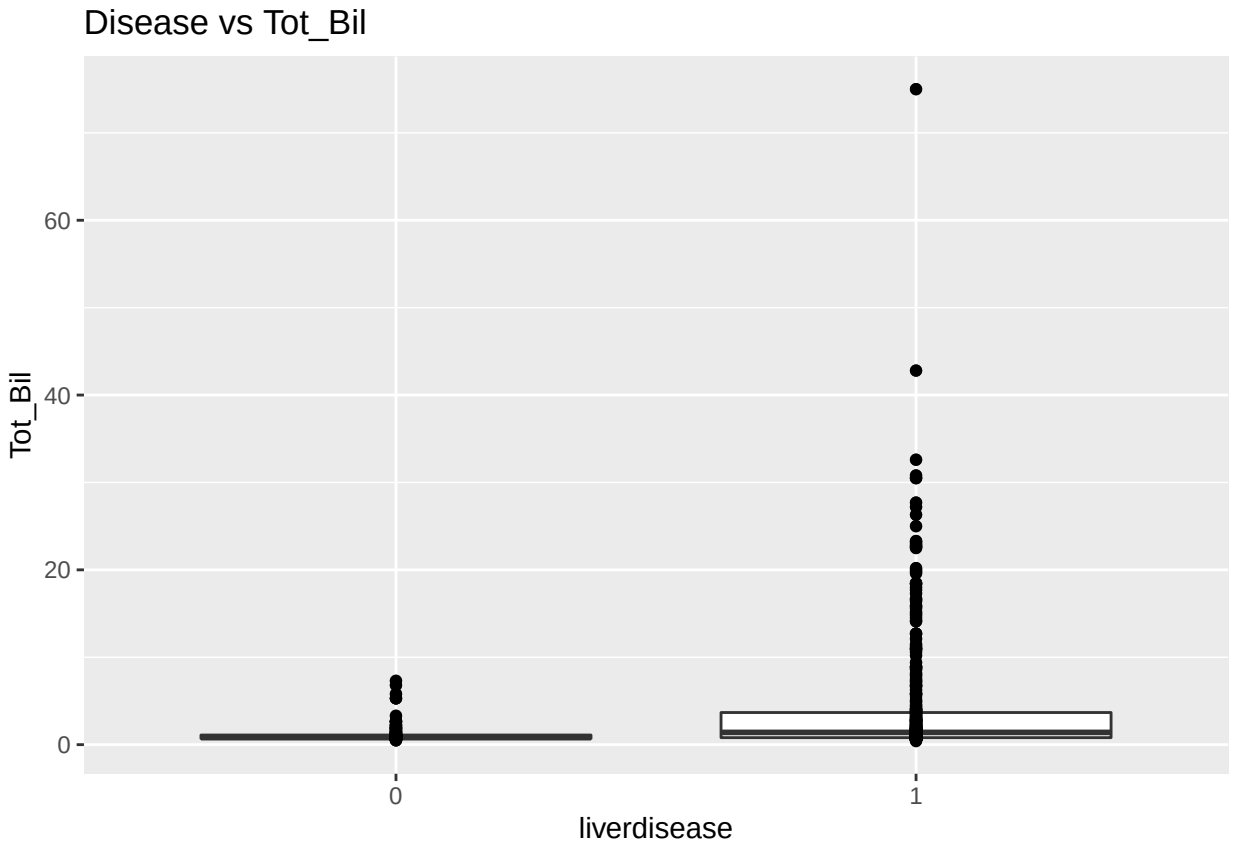
```
  Male_incident/Female_incident
```

```
## [1] 3.549451
```

liver disease incident in Male is “r Male_incident/Female_incident” times higher than in Female.

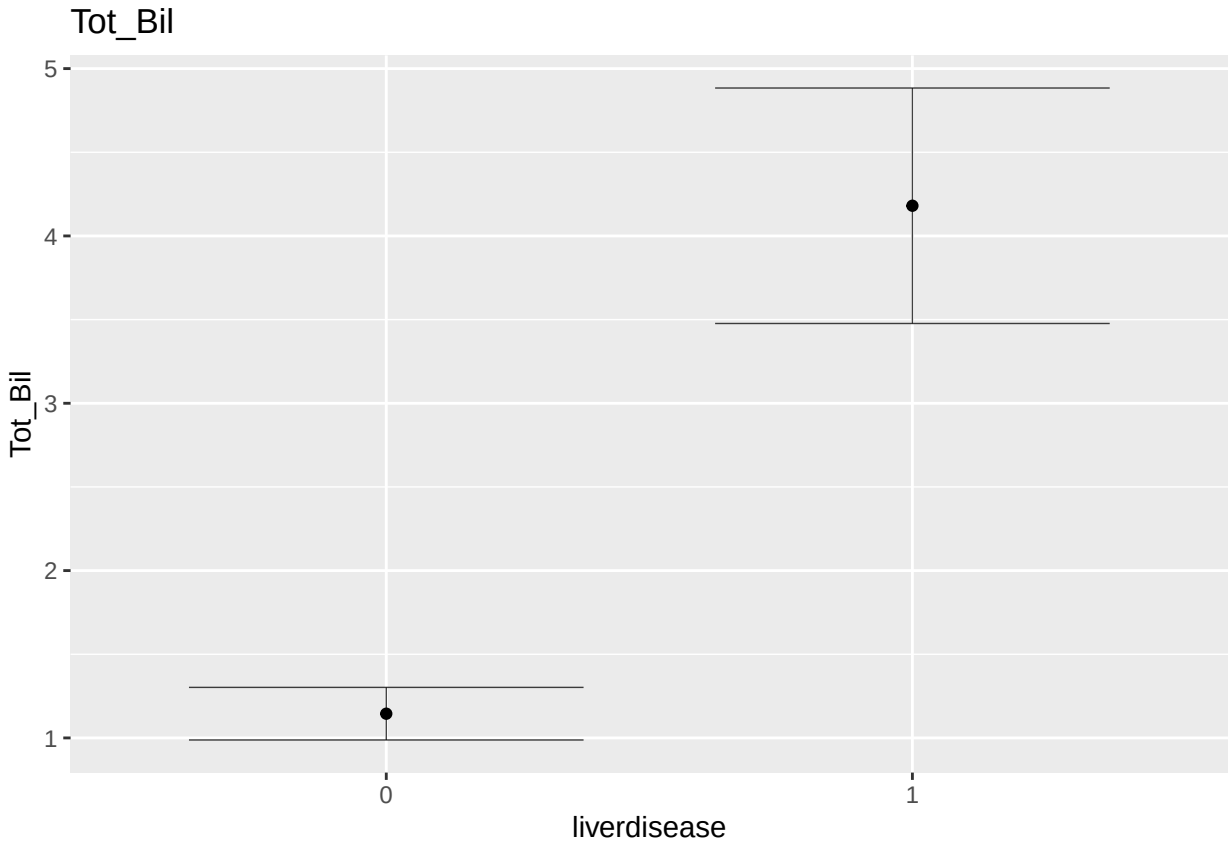
Tot_Bil express based on liverdisease

```
liver_data %>%
  ggplot(aes(liverdisease, Tot_Bil))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Tot_Bil") +
  xlab("liverdisease") + ylab("Tot_Bil")
```



Tot_Bil expression based on liver disease (mean+/-2se)

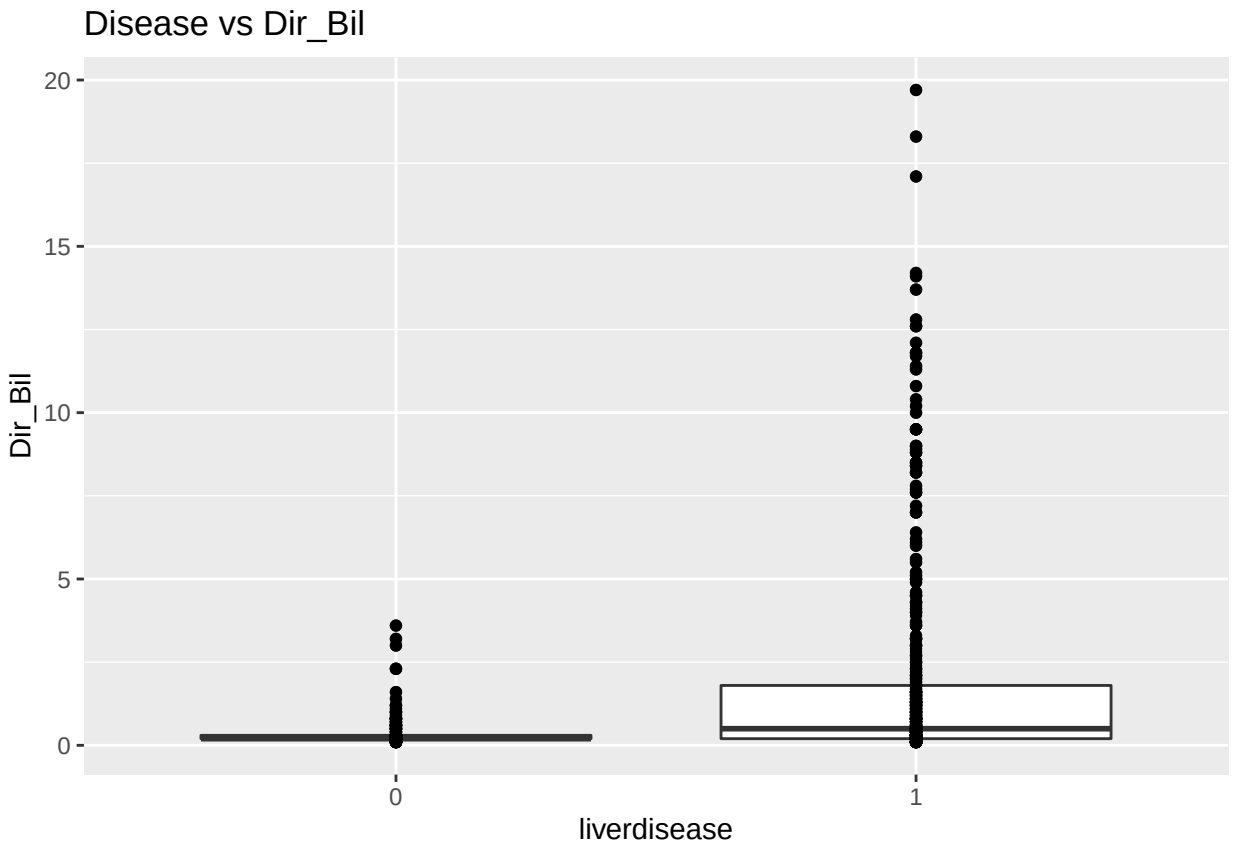
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Tot_Bil),sd=sd(Tot_Bil),se=sd(Tot_Bil)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Tot_Bil")+
  xlab("liverdisease") + ylab("Tot_Bil")
```



Tot_Bil is higher in those have liver disease.

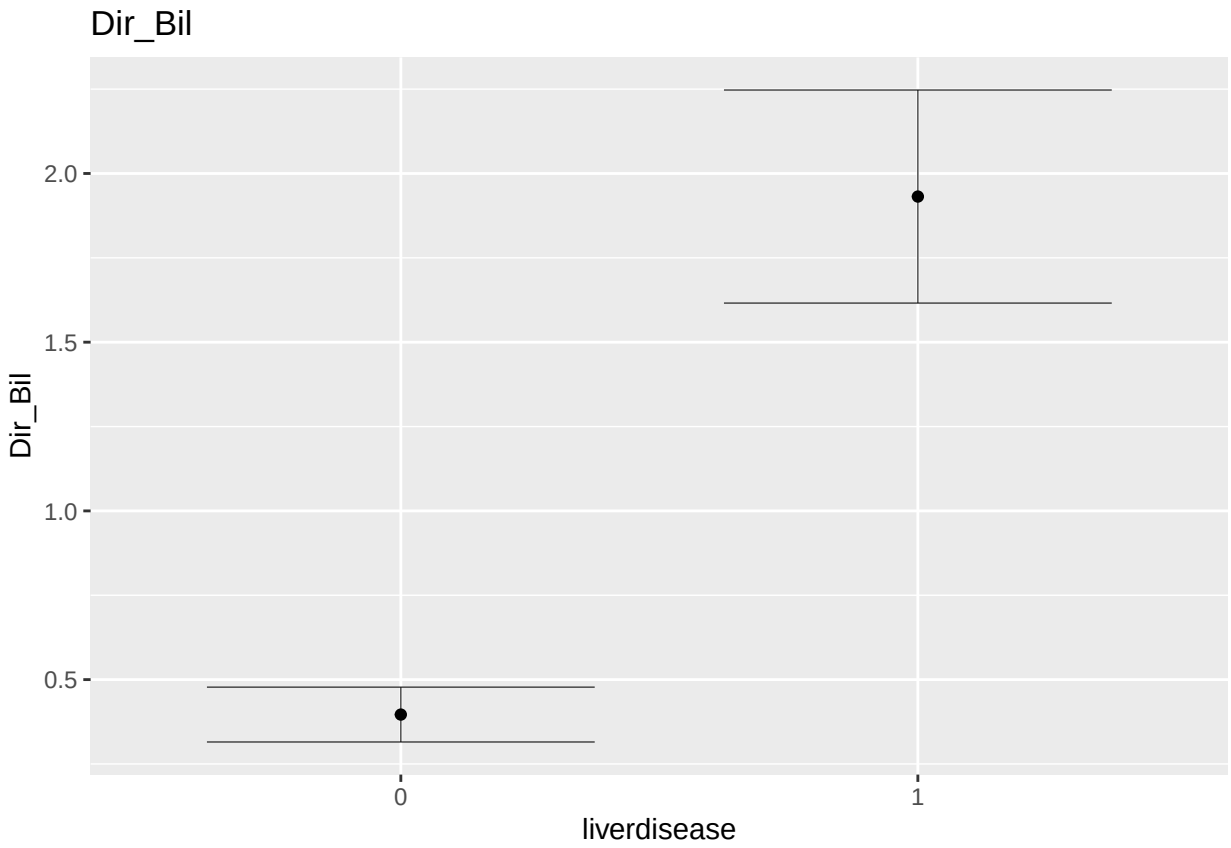
Dir_Bil expression based on liver disease—wrong

```
liver_data %>%
  ggplot(aes(liverdisease, Dir_Bil))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Dir_Bil") +
  xlab("liverdisease") + ylab("Dir_Bil")
```



Dir_Bil expression based on liver disease (mean+/-2se)

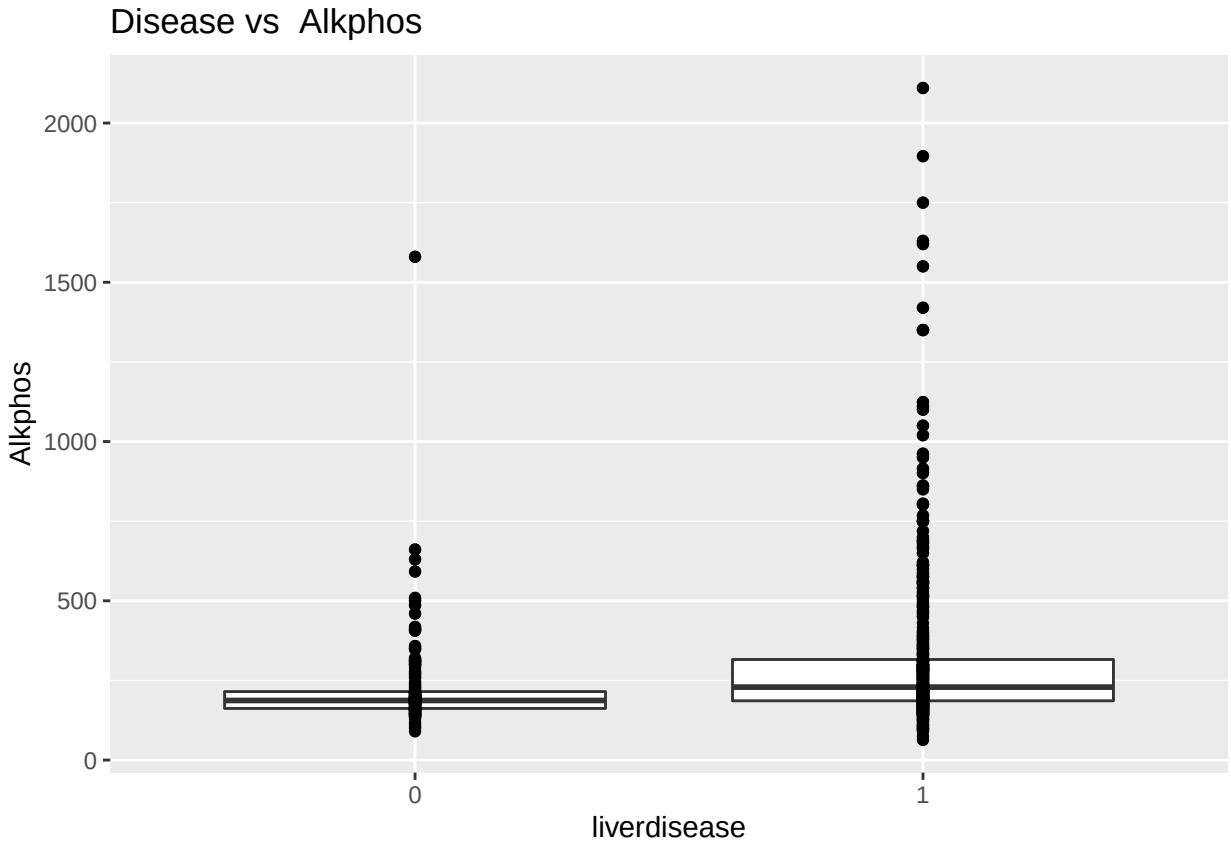
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Dir_Bil),sd=sd(Dir_Bil),se=sd(Dir_Bil)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Dir_Bil")+
  xlab("liverdisease") + ylab("Dir_Bil")
```



Dir_Bil is higher in liver disease group

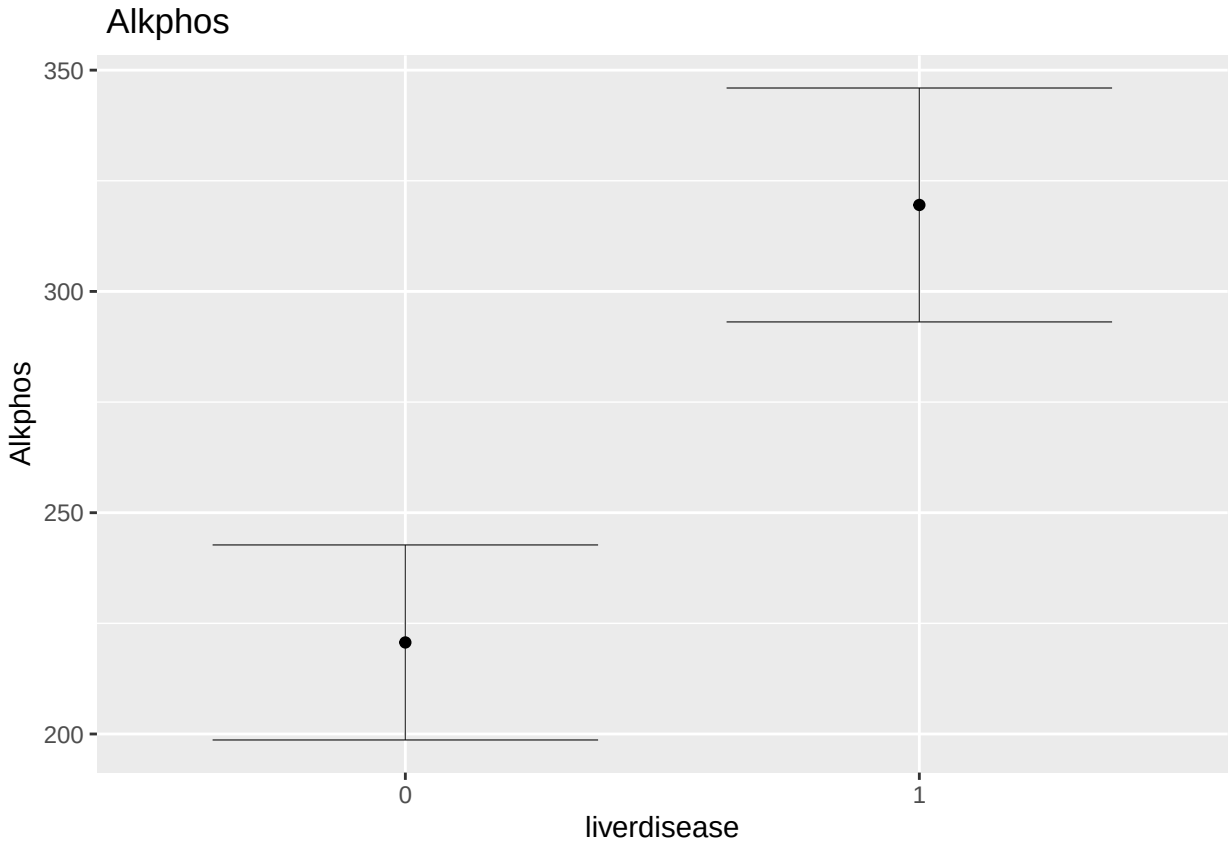
Alkphos expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease,  Alkphos))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs  Alkphos ") +
  xlab("liverdisease") + ylab(" Alkphos ")
```



Alkphos expression based on liver disease (mean+/-2se)

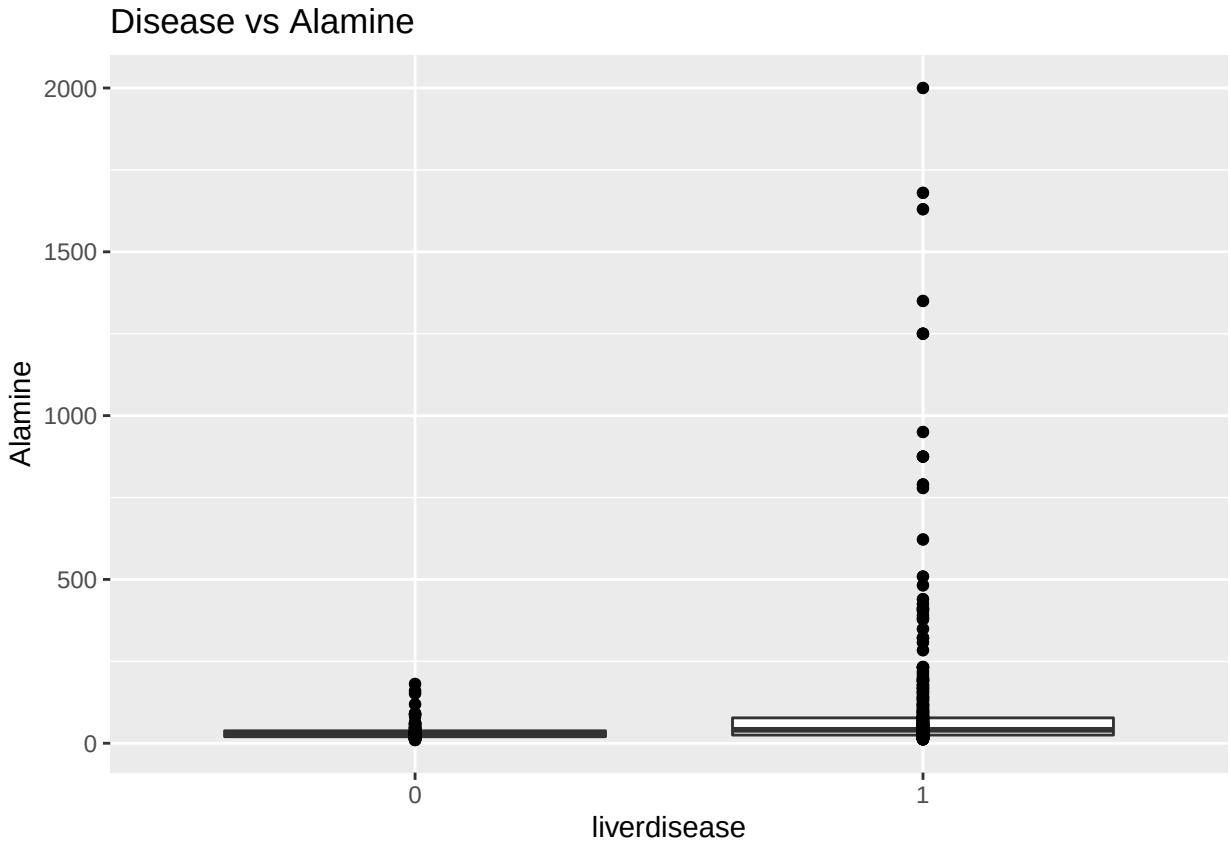
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean( Alkphos),sd=sd( Alkphos ),se=sd( Alkphos )/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle(" Alkphos ")+
  xlab("liverdisease") + ylab(" Alkphos ")
```



Alkphos is higher in liver disease group.

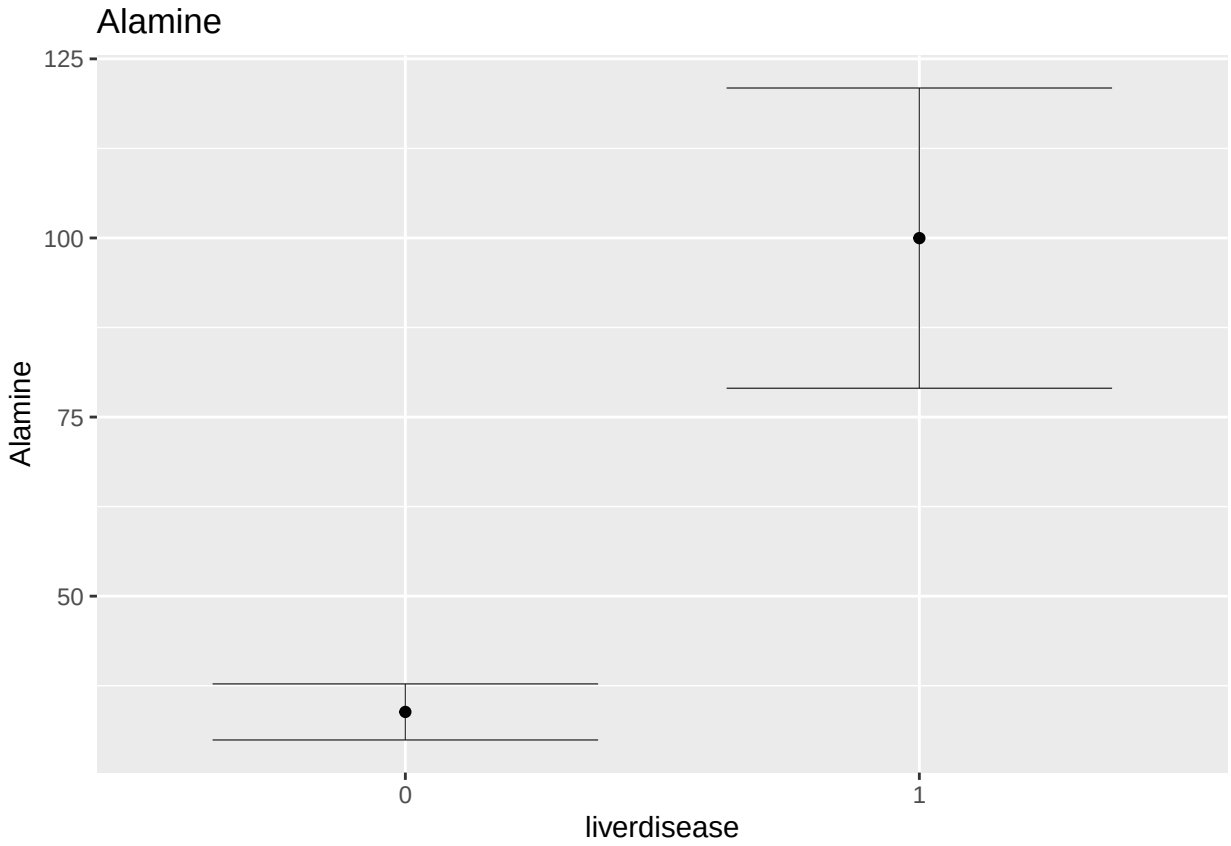
Alamine expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease, Alamine ))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Alamine ") +
  xlab("liverdisease") + ylab(" Alamine ")
```



Alamine expression based on liver disease (mean+/-2se)

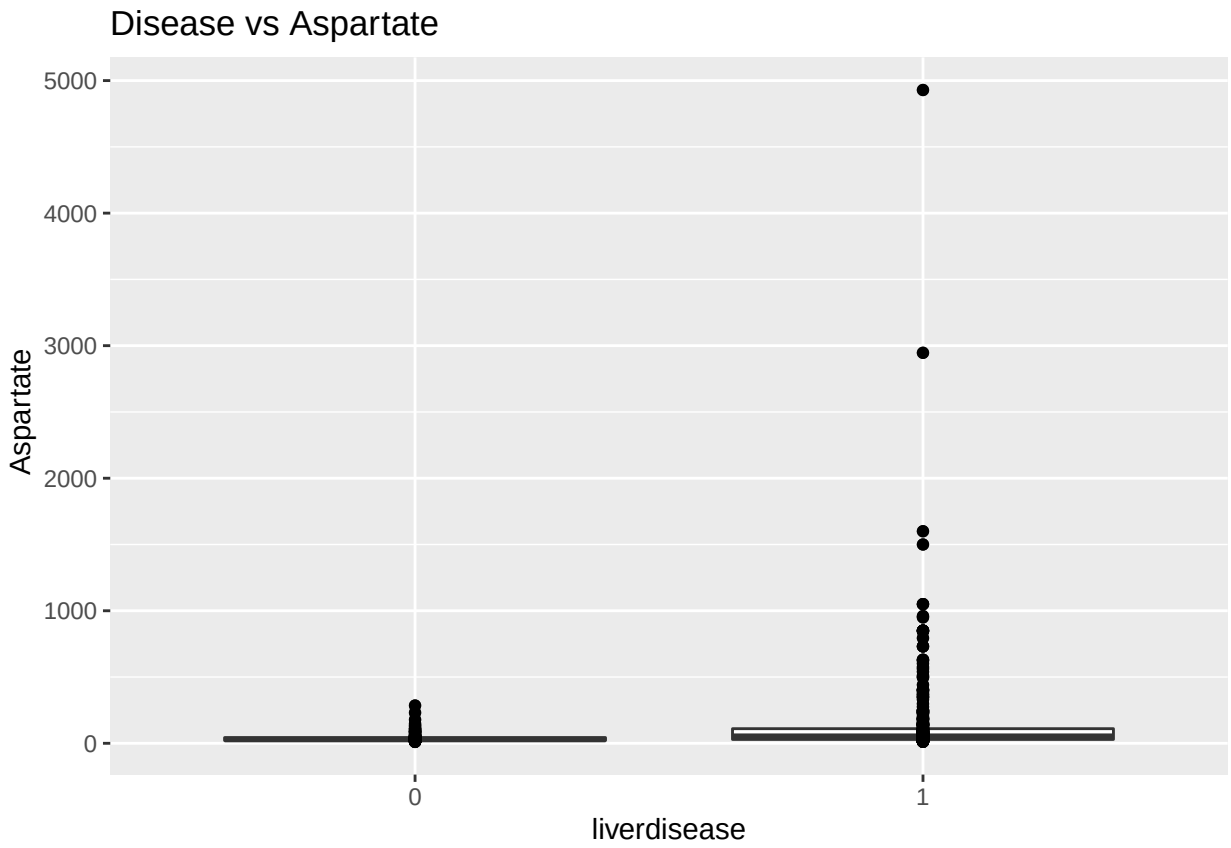
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Alamine ),sd=sd(Alamine),se=sd(Alamine)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Alamine")+
  xlab("liverdisease") + ylab(" Alamine ")
```



Alamine is higher in liver disease group.

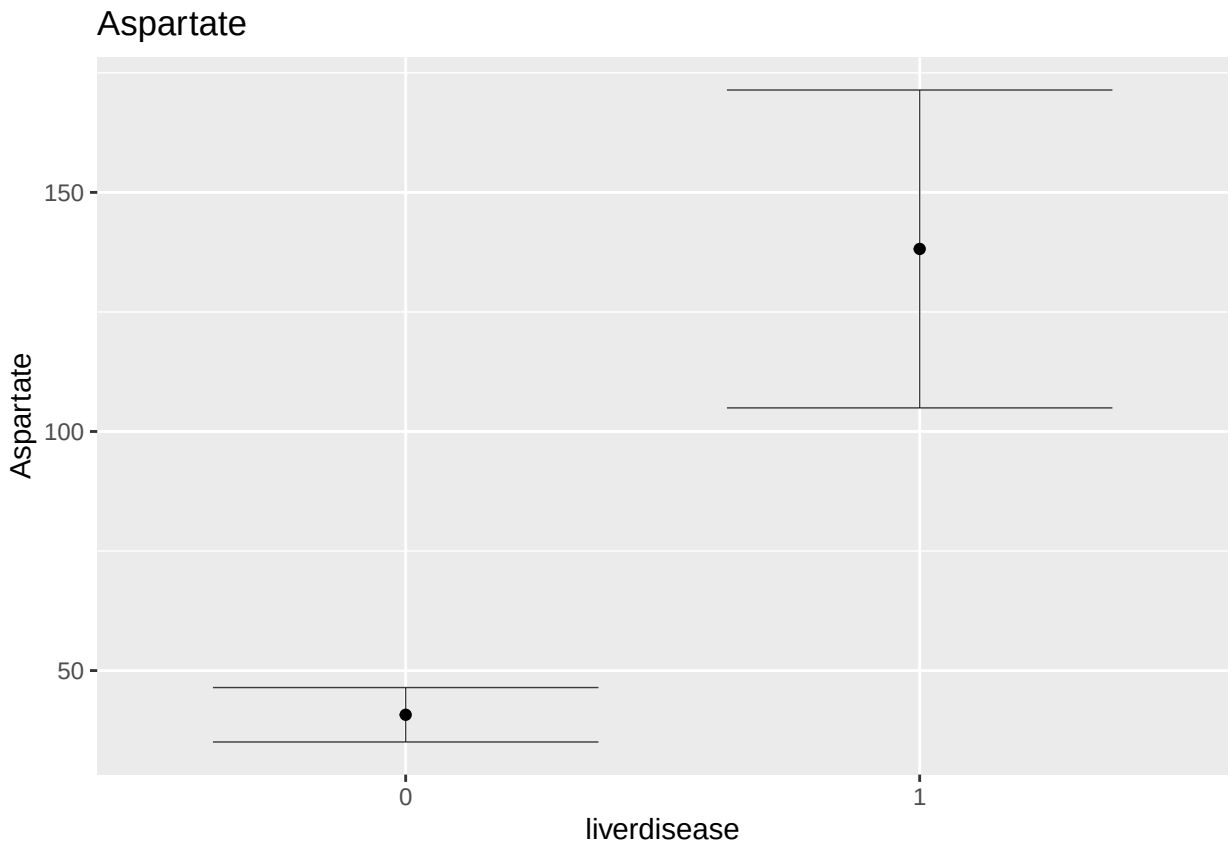
Aspartate expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease, Aspartate))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Aspartate") +
  xlab("liverdisease") + ylab(" Aspartate")
```



Aspartate expression based on liver disease (mean+/-2se)

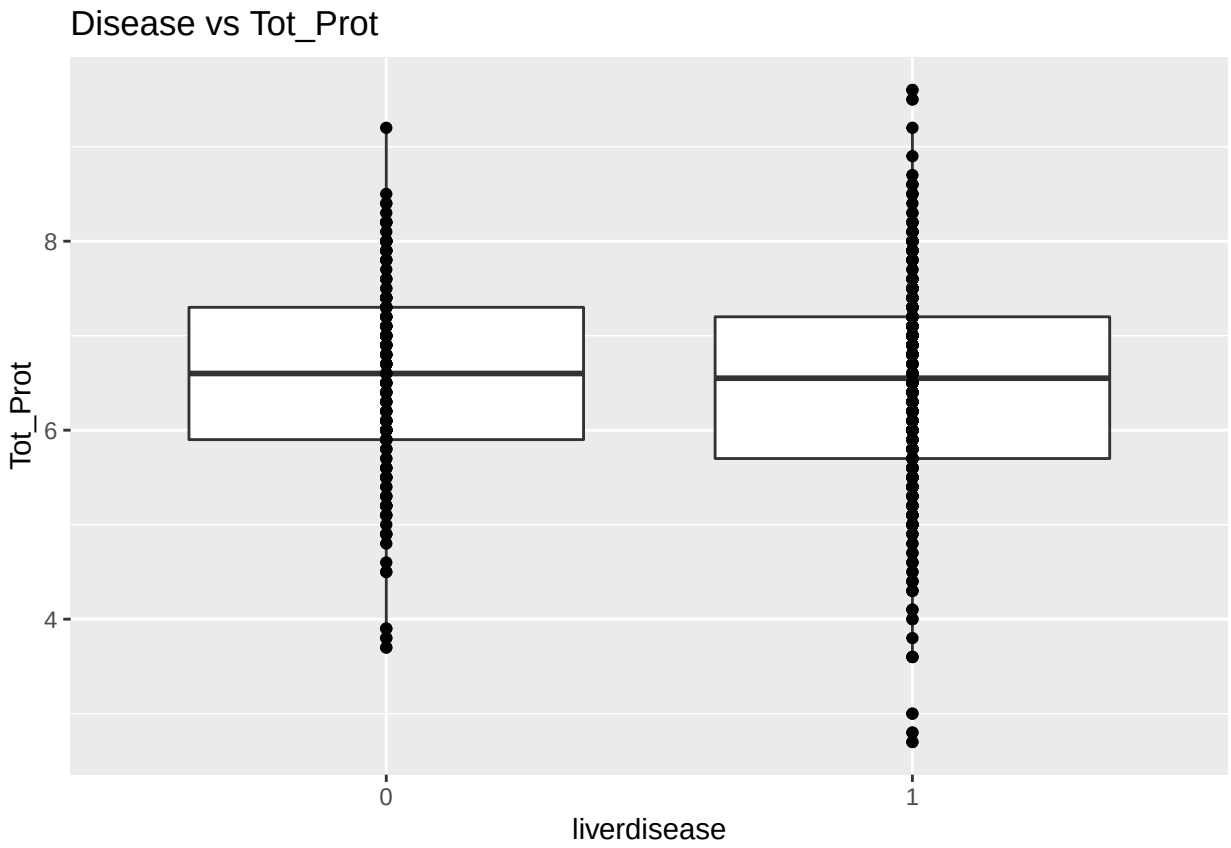
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Aspartate),sd=sd(Aspartate),se=sd(Aspartate)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Aspartate")+
  xlab("liverdisease") + ylab("Aspartate")
```



Aspartate is higher in liver disease group.

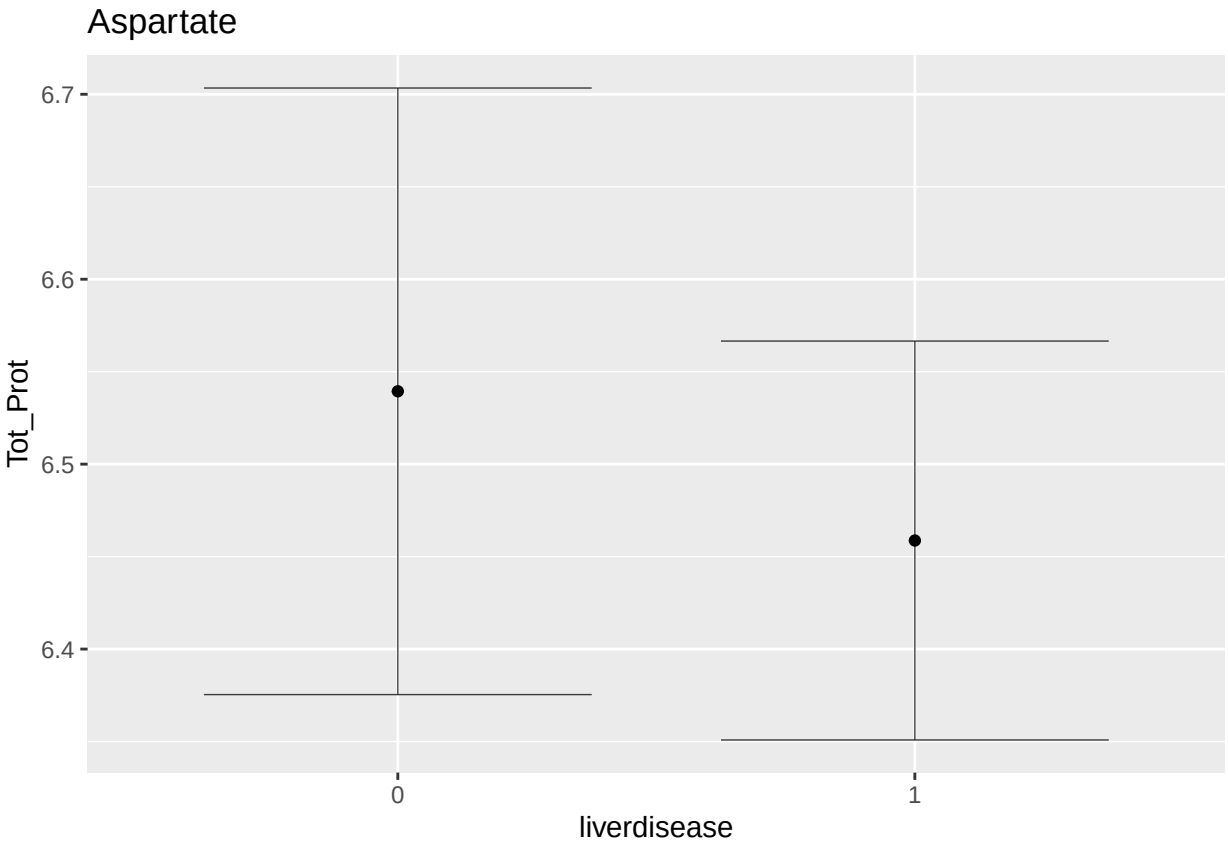
Tot_Prot expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease, Tot_Prot))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Tot_Prot") +
  xlab("liverdisease") + ylab("Tot_Prot")
```



Tot_Prot expression based on liver disease (mean+/-2se)

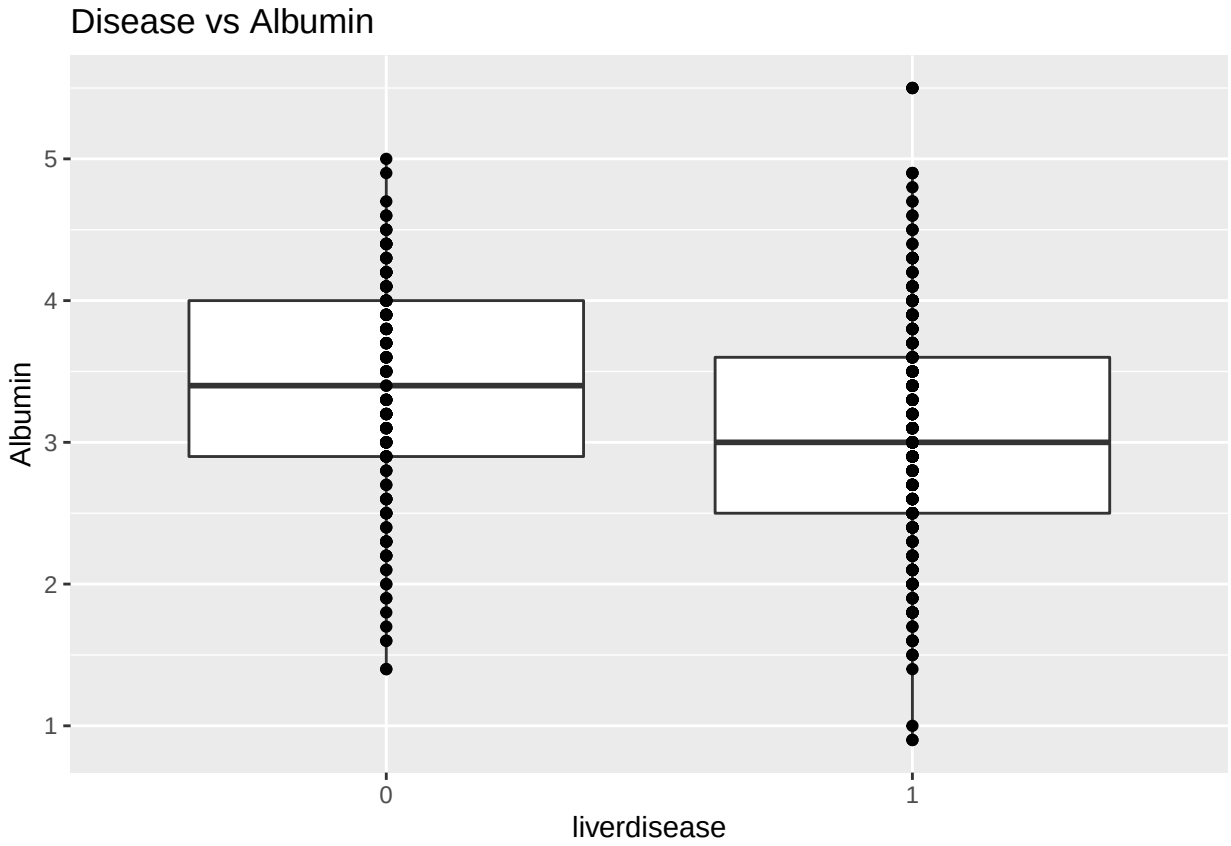
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Tot_Prot ),sd=sd(Tot_Prot),se=sd(Tot_Prot)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Aspartate")+
  xlab("liverdisease") + ylab("Tot_Prot")
```



Tot_Prot is lower in liver disease group.

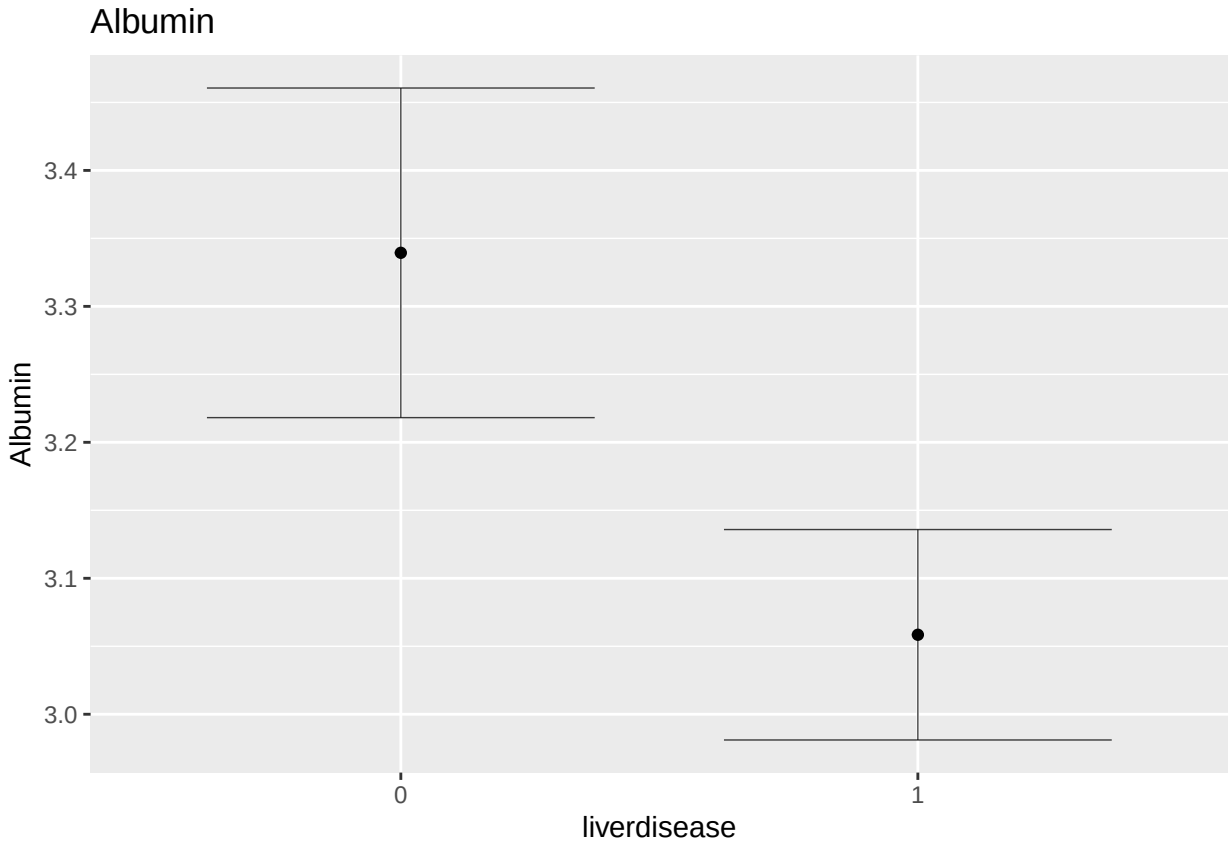
Albumin expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease, Albumin))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs Albumin") +
  xlab("liverdisease") + ylab("Albumin")
```



Albumin expression based on liver disease (mean+/-2se)

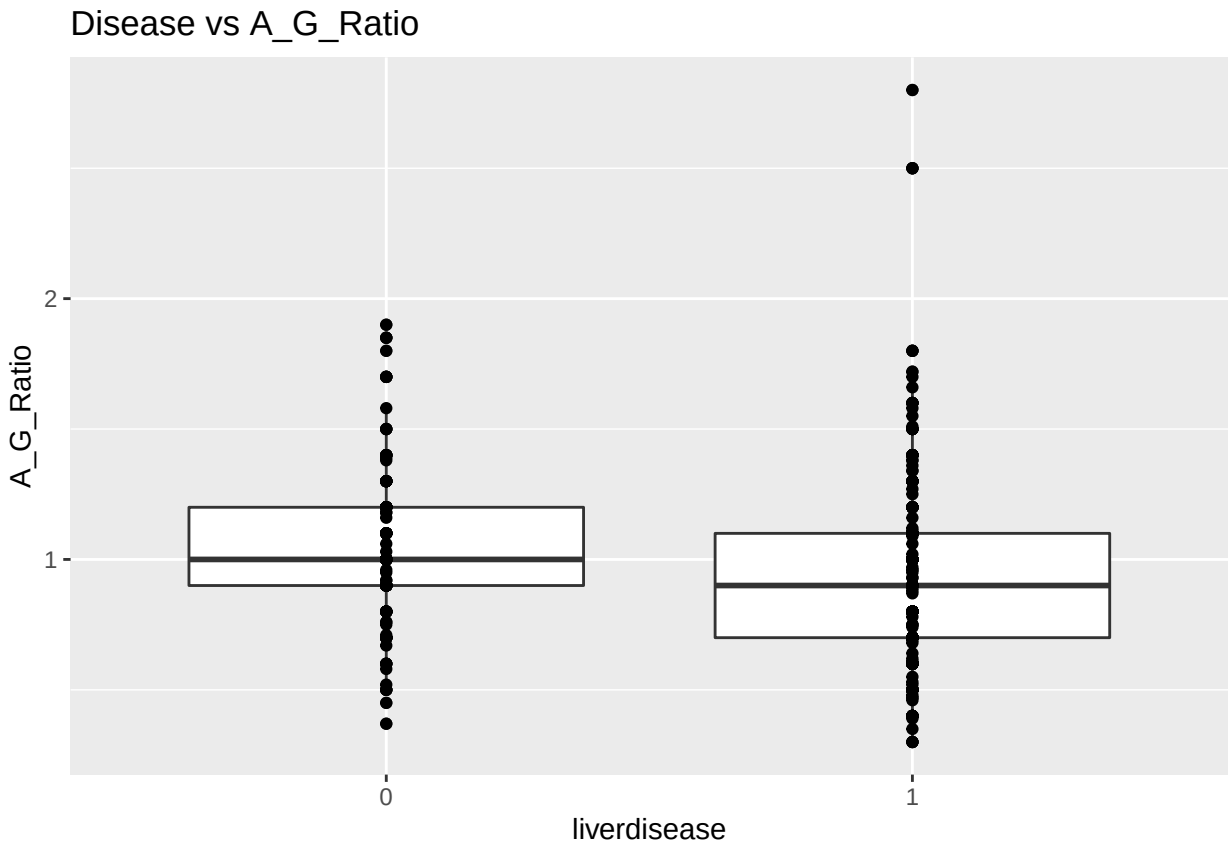
```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(Albumin),sd=sd(Albumin),se=sd(Albumin)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("Albumin")+
  xlab("liverdisease") + ylab("Albumin")
```



Albumin is lower in liver disease group.

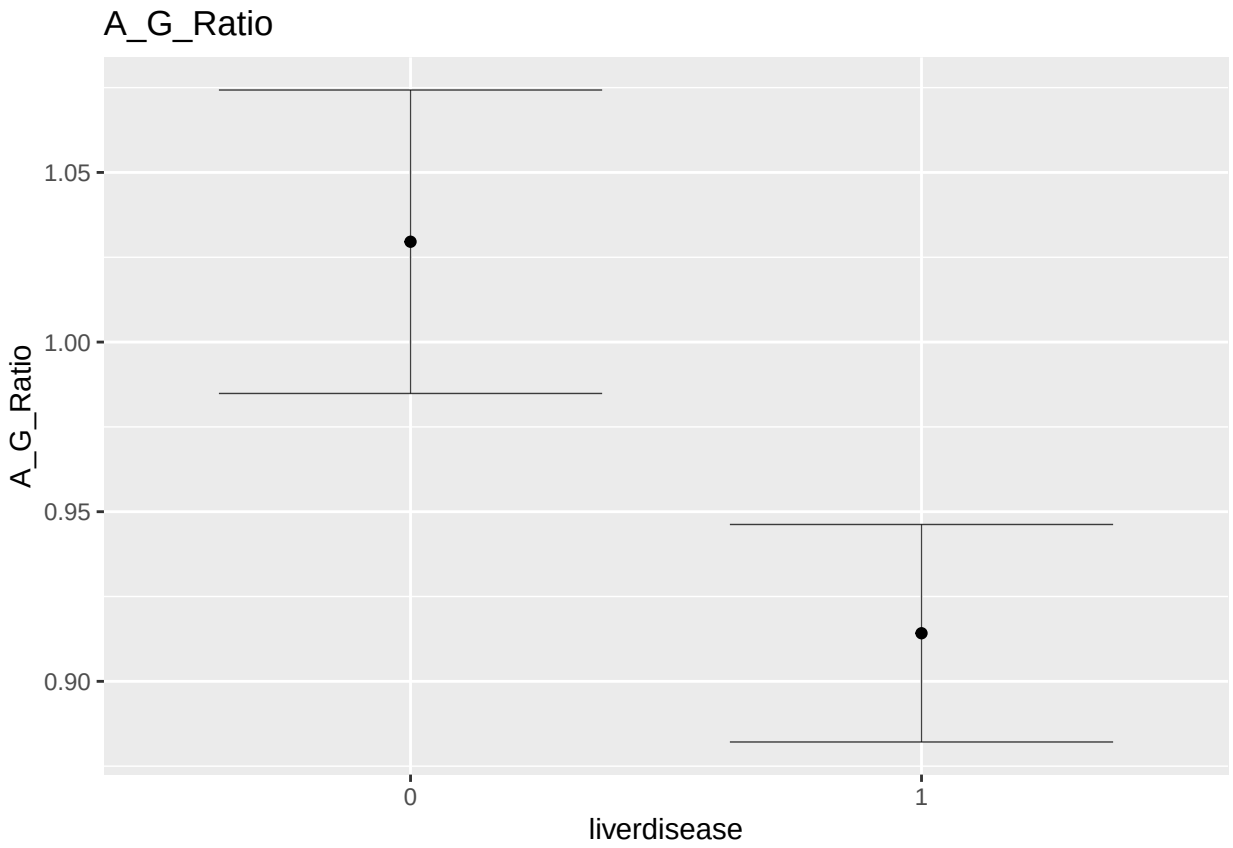
##A_G_Ratio expression based on liver disease

```
liver_data %>%
  ggplot(aes(liverdisease, A_G_Ratio))+
  geom_boxplot()+geom_point()+
  ggtitle("Disease vs A_G_Ratio") +
  xlab("liverdisease") + ylab("A_G_Ratio")
```



A_G_Ratio expression based on liver disease (mean+/-2se)

```
liver_data %>% group_by(liverdisease) %>%
  summarize(n=n(), avg=mean(A_G_Ratio),sd=sd(Albumin),se=sd(A_G_Ratio)/sqrt(n()))%>%
  ggplot(aes(x=liverdisease, y=avg, ymin=avg-2*se, ymax=avg+2*se))+
  geom_point()+
  geom_errorbar(width=0.75, colour="black", alpha=0.75, size=0.25)+
  ggtitle("A_G_Ratio")+
  xlab("liverdisease") + ylab("A_G_Ratio")
```



A_G_Ratio is lower in liver disease group.

Build the prediction system

separete data set to train_set(70%) and test_set(30%)

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

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
test_index1 <- createDataPartition(y = liver_data$Disease, times = 1, p = 0.3, list = FALSE)
train_set <- liver_data[-test_index1,]
test_set<- liver_data[test_index1,]
dim(train_set)
```

```
## [1] 405  12
```

```
dim(test_set)
```

```
## [1] 174  12
```

** build models**

Since “no disease” and have “liver disease” is a “0” and “1” binary, I think the Accuracy of each Model is more important to evaluate a model. But I will still calculate the rmse(residual mean square error).

model 1. liner Regression Model

create liner Regression Model

```
fit_lm <- lm(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                Tot_Prot + Albumin + A_G_Ratio, data = train_set)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
fit_lm$coef
```

```
##   (Intercept)           Age       SexMale       Tot_Bil       Dir_Bil
##  1.3751027736  0.0018702171  0.1013778847 -0.0111272186  0.0412676302
##       Alkphos       Alamine     Aspartate      Tot_Prot       Albumin
##  0.0002776724  0.0005353460 -0.0001087727  0.0636946488 -0.1340749744
##     A_G_Ratio
##  0.0487311160
```

get a prediction based on liner Regression Model

```
prediction_lm<- predict(fit_lm, test_set, type = "response")

Disease_lm <- ifelse(prediction_lm > 0.5, "1", "0") %>% factor
```

The Accuracy of liner Regression Model

```
confusionMatrix(Disease_lm, test_set$liverdisease)$table
```

```
## Warning in confusionMatrix.default(Disease_lm, test_set$liverdisease): Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
##           Reference
## Prediction   0   1
##          0   0   0
##          1  46 128
```

```
Accuracy_lm <- confusionMatrix(Disease_lm, test_set$liverdisease)$overall[["Accuracy"]]
```

```
## Warning in confusionMatrix.default(Disease_lm, test_set$liverdisease): Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
Accuracy_lm
```

```
## [1] 0.7356322
```

The accuracy of this liner Regression Model is 0.7356322.

rmse of liner Regression Model

```
model_lm_rmse <- RMSE(prediction_lm, test_set$Disease)
model_lm_rmse
```

```
## [1] 1.07452
```

model_lm_rmse is 1.0745201.

model 2. Logistic Regression Model

create Logistic Regression Model

```
fit_glm <- glm(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                 Tot_Prot + Albumin + A_G_Ratio, data = train_set, family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
fit_glm$coef
```

```
##   (Intercept)           Age       SexMale       Tot_Bil       Dir_Bil
## -2.2552284788  0.0115113314  0.2293940120 -0.3582400517  1.1293101633
##       Alkphos       Alamine     Aspartate      Tot_Prot       Albumin
##  0.0019650017  0.0189563803  0.0003299318  0.5191002170 -1.0487972826
##     A_G_Ratio
##  0.9683662611
```

get a prediction based on Logistic Regression Model

```
prediction_glm<- predict(fit_glm, test_set, type = "response")

Disease_glm <- ifelse(prediction_glm > 0.5, "1", "0") %>% factor
```

The Accuracy of Logistic Regression Model

```
confusionMatrix(Disease_glm, test_set$liverdisease)$table
```

```
##            Reference
## Prediction   0   1
##          0  14  18
##          1  32 110
```

```
Accuracy_glm <-confusionMatrix(Disease_glm, test_set$liverdisease)$overall[["Accuracy"]]
Accuracy_glm
```

```
## [1] 0.7126437
```

The accuracy of this Logistic Regression Model is 0.7126437.

rmse of Logistic Regression Model

```
model_glm_rmse <- RMSE(Accuracy_glm, test_set$Disease)
model_glm_rmse
```

```
## [1] 0.4415948
```

Model_glm_rmse is 0.4415948.

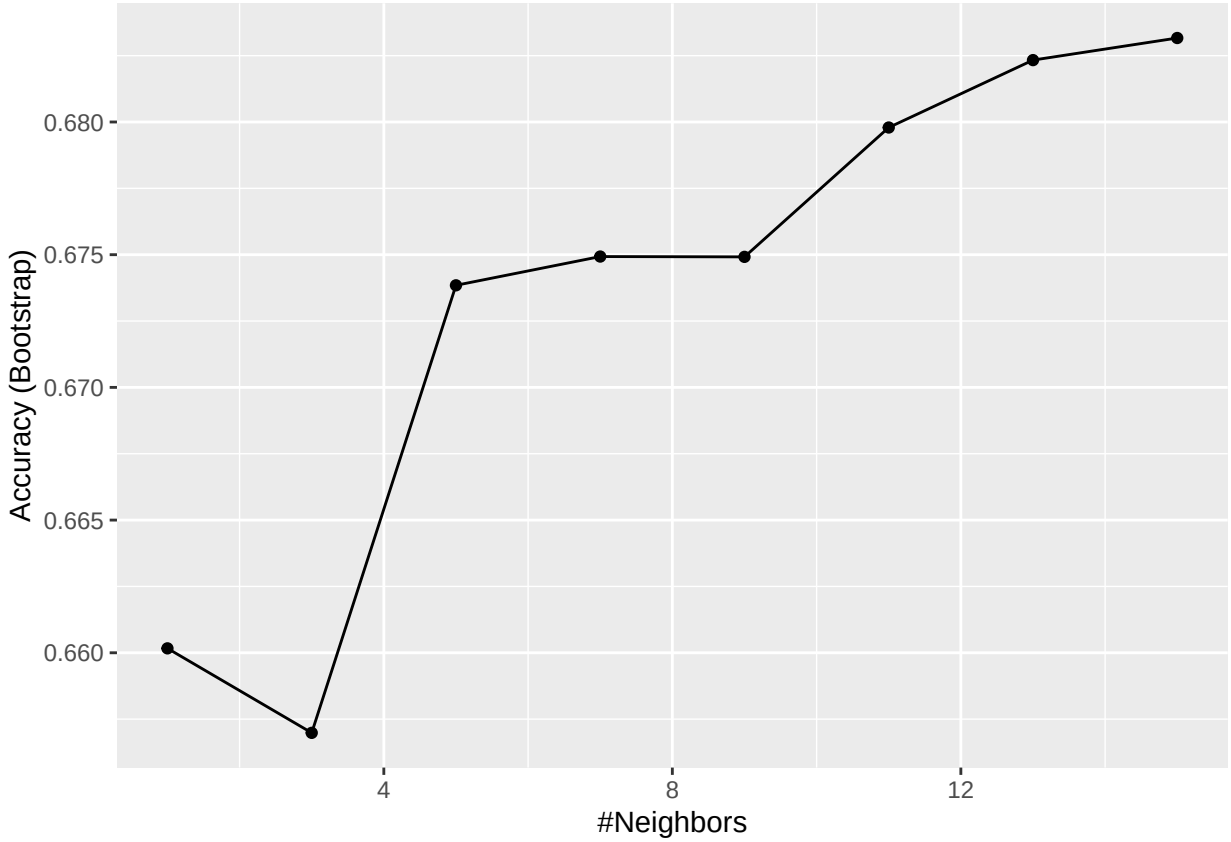
model 3. knn Model

find the best K for KNN model

```
fit_knnK <- train(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                Tot_Prot + Albumin + A_G_Ratio,   method = "knn",
              tuneGrid = data.frame(k = seq(1, 15, 2)),
              data = train_set)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
ggplot(fit_knnK)
```




```
fit_knnK
```

```
## k-Nearest Neighbors
##
## 405 samples
##  10 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 405, 405, 405, 405, 405, 405, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.6601660  0.1832394
##    3  0.6569830  0.1555606
##    5  0.6738465  0.1780718
##    7  0.6749303  0.1710184
##    9  0.6749182  0.1609944
##   11  0.6797932  0.1684023
##   13  0.6823327  0.1699910
##   15  0.6831650  0.1646835
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 15.
```

When k=13, knn model get the highest Accuracy.

create Logistic Regression Model use k=13

```
fit_knn <- knn3(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                  Tot_Prot + Albumin + A_G_Ratio, data = train_set, k=13)
```

get a prediction based on knn Model

```
prediction_knn <- predict(fit_knn, test_set, type = "class")
prediction_knn
```

```
##   [1] 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0
##  [75] 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1
## [112] 1 1 1 0 0 1 0 1 1 1 0 1 0 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1
## [149] 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0
## Levels: 0 1
```

The Accuracy of knn Model

```
confusionMatrix(data = prediction_knn, reference = test_set$liverdisease)$table
```

```
##            Reference
## Prediction   0   1
##          0  11  20
##          1  35 108
```

```
Accuracy_knn <- confusionMatrix(data = prediction_knn, reference = test_set$liverdisease)$overall["Accu

Accuracy_knn
```

```
## Accuracy
## 0.683908
```

The accuracy of this knn Model is “r Accuracy_knn”.

rmse of knn Model

```
model_knn_rmse <- RMSE(Accuracy_knn, test_set$Disease)
model_knn_rmse
```

```
## [1] 0.444019
```

Model_knn_rmse is 0.444019.

model 4. qda Model

create qda Model

```
fit_qda <- train(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                     Tot_Prot + Albumin + A_G_Ratio, data = train_set)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

get a prediction based on qda Model

```
prediction_qda <- predict(fit_qda, test_set)
prediction_qda
```

```
##   [1] 0 0 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
##  [38] 0 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
##  [75] 1 1 1 0 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 0 0 0 1 0 1 1 1 1 1 1 0 0 1
## [112] 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 0
## [149] 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## Levels: 0 1
```

compute accuracy

```
confusionMatrix(predict(fit_qda, test_set), test_set$liverdisease)$table
```

```
##            Reference
## Prediction   0   1
##          0  14  22
##          1  32 106
```

```
Accuracy_qda<-confusionMatrix(predict(fit_qda, test_set), test_set$liverdisease)$overall["Accuracy"]
Accuracy_qda
```

```
##  Accuracy
## 0.6896552
```

The accuracy of this qda Model is 0.6896552.

rmse of qda Model

```
model_qda_rmse <- RMSE(Accuracy_qda, test_set$Disease)
model_qda_rmse
```

```
## [1] 0.4433862
```

Model_qda_rmse is 0.4433862.

model 5. lda Model

create lda Model

```
fit_lda <- train(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                   Tot_Prot + Albumin + A_G_Ratio, data = train_set)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

get a prediction based on lda Model

```
prediction_lda <- predict(fit_lda, test_set)
prediction_lda
```

```
##   [1] 0 0 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1
##  [38] 0 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
##  [75] 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 0 1 0 1 1 1 1 1 1 0 0 1
## [112] 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1
## [149] 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## Levels: 0 1
```

compute accuracy

```
confusionMatrix(predict(fit_qda, test_set), test_set$liverdisease)$table
```

```
##            Reference
## Prediction   0   1
##          0  14  22
##          1  32 106
```

```
Accuracy_lda <- confusionMatrix(predict(fit_lda, test_set), test_set$liverdisease)$overall["Accuracy"]
Accuracy_lda
```

```
## Accuracy
## 0.683908
```

The accuracy of this lda Model is 0.683908.

rmse of lda Model

```
model_lda_rmse <- RMSE(Accuracy_lda, test_set$Disease)
model_lda_rmse
```

```
## [1] 0.444019
```

model_lda_rmse is 0.444019.

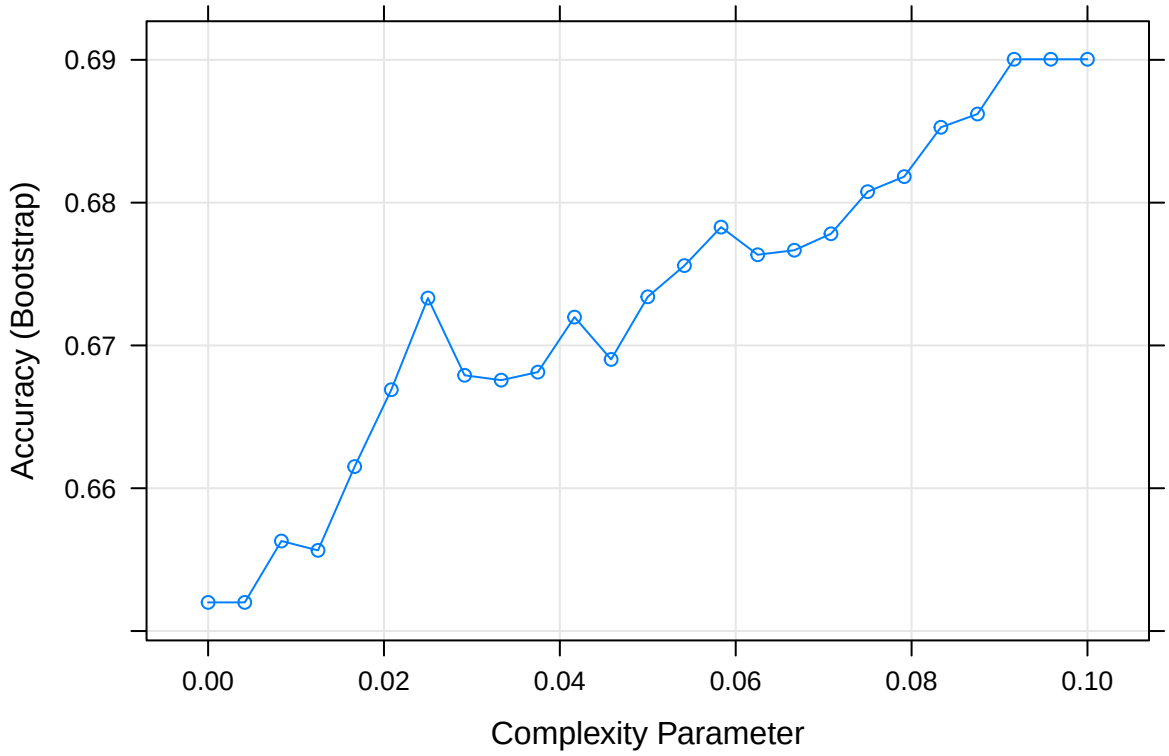
model 6. Classification (Decision) Trees

fit a classification tree and plot it

```
fit_tree <- train(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspartate +
                    Tot_Prot + Albumin + A_G_Ratio,
                   method = "rpart",
                   tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
                   data = train_set)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
plot(fit_tree)
```



get a prediction based on Decision Trees Model

```
prediction_Decision_Trees <- predict(fit_tree, test_set)
prediction_Decision_Trees
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [112] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [149] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## Levels: 0 1
```

It get too much "1" (liver disease)

compute accuracy

```
confusionMatrix(predict(fit_tree, test_set), test_set$liverdisease)$table
```

```
##           Reference
## Prediction   0   1
##          0   0   0
##          1  46 128
```

```
Accuracy_Decision_Trees <- confusionMatrix(predict(fit_tree, test_set), test_set$liverdisease)$overall[
Accuracy_Decision_Trees
```

```
##  Accuracy
## 0.7356322
```

Even the Accuracy_Decision_Trees is as highas 0.7356322, but the prediction is all "1"—-liver disease, I think we should not trust this model.

rmse of classification tree Model

```
model_Decision_Trees_rmse <- RMSE(Accuracy_Decision_Trees, test_set$Disease)
model_Decision_Trees_rmse
```

```
## [1] 0.440996
```

model 7. randomForest Trees

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

create a randomForest tree model

```
fit_randomForest <- randomForest(liverdisease ~ Age + Sex + Tot_Bil + Dir_Bil + Alkphos + Alamine + Aspa
                      Tot_Prot + Albumin + A_G_Ratio, data = train_set)
```

get a prediction based on randomForest Trees

```
prediction_randomForest <- predict(fit_randomForest, test_set)
prediction_randomForest
```

```
##   7   8  13  16  27  29  32  35  36  38  44  46  48  53  59  60  62  69  70  71
##   0   0   1   1   1   1   1   0   1   1   1   1   1   1   1   0   1   0   1   1
##  76  77  80  90  94 101 102 104 105 113 116 117 118 119 120 121 129 133 139 141
##   1   1   1   1   1   0   1   0   1   1   1   1   1   1   1   1   1   0   1   1
## 149 153 154 159 162 166 167 169 172 173 176 187 188 194 196 197 198 199 202 206
##   1   0   1   1   1   1   1   1   1   1   0   1   1   1   1   1   0   1   1   1
## 207 213 214 215 217 218 219 221 223 224 225 226 227 228 232 233 234 239 240 243
##   1   1   1   1   1   1   1   1   1   1   1   1   0   0   1   1   1   0   1   0
## 248 251 256 257 261 262 267 268 269 272 273 274 284 285 287 290 294 298 300 301
##   0   1   1   1   1   1   0   0   1   1   1   1   1   0   0   1   1   0   0   0
## 303 304 317 318 322 327 329 333 335 336 340 347 348 349 353 355 359 364 367 368
##   1   0   1   1   1   1   1   1   0   0   1   0   1   1   0   1   0   0   0   1
## 370 372 376 379 382 390 393 394 395 396 397 409 416 418 425 426 430 431 436 439
##   0   0   1   1   0   1   1   1   1   1   1   1   1   1   1   0   1   1   0   0
## 440 442 447 451 454 455 459 460 462 472 477 478 484 485 499 507 510 520 527 528
##   1   1   1   1   1   0   1   1   1   1   1   0   0   1   0   1   1   1   1   1
## 530 542 546 550 556 562 564 565 569 571 572 573 576 578
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## Levels: 0 1
```

compute accuracy

```
confusionMatrix(predict(fit_randomForest, test_set), test_set$liverdisease)$table
```

```
##            Reference
## Prediction   0   1
##          0  16  25
##          1  30 103
```

```
Accuracy_randomForest<- confusionMatrix(predict(fit_randomForest, test_set), test_set$liverdisease)$ove
Accuracy_randomForest
```

```
## Accuracy
## 0.683908
```

rmse of randomForest Model

```
model_randomForest_rmse <- RMSE(Accuracy_randomForest, test_set$Disease)
model_randomForest_rmse
```

```
## [1] 0.444019
```

model 8.Regularization model

create a test set that exclude "NA"

```
test_set1 <- test_set %>%
  semi_join(train_set, by = "Age") %>%
  semi_join(train_set, by = "Sex") %>%
  semi_join(train_set, by = "Tot_Bil")%>%
  semi_join(train_set, by = "Dir_Bil")%>%
  semi_join(train_set, by = "Alkphos")%>%
  semi_join(train_set, by = "Alamine")%>%
  semi_join(train_set, by = "Tot_Prot")%>%
  semi_join(train_set, by = "Aspartate")%>%
  semi_join(train_set, by = "Albumin")%>%
  semi_join(train_set, by = "A_G_Ratio")

dim(test_set1)
```

```
## [1] 77 12
```

compute the average based on each predictor

```
mu_Disease<- mean(train_set$Disease)
mu_Disease
```

```
## [1] 0.7061728
```

average by Age

```
Age_avgs <- train_set %>%
  group_by(Age) %>%
  summarize(b_Age = mean(Disease - mu_Disease))

Age_avgs
```

```
## # A tibble: 68 x 2
##      Age  b_Age
```

```
##    <int>  <dbl>
##  1     4 -0.706
##  2     7 -0.206
##  3    10  0.294
##  4    12 -0.206
##  5    13 -0.206
##  6    14 -0.206
##  7    15  0.294
##  8    16  0.294
##  9    17 -0.456
## 10    18  0.294
## # ... with 58 more rows
```

average by Sex

```
Sex_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  group_by(Sex) %>%
  summarize(b_Sex = mean(Disease - mu_Disease -b_Age ))

Sex_avgs
```

```
## # A tibble: 2 x 2
##   Sex      b_Sex
##   <chr>    <dbl>
## 1 Female -0.0581
## 2 Male    0.0183
```

average by Tot_Bil

```
Tot_Bil_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  group_by(Tot_Bil) %>%
  summarize(b_Tot_Bil = mean(Disease - mu_Disease -b_Age-b_Sex))

Tot_Bil_avgs
```

```
## # A tibble: 89 x 2
##    Tot_Bil b_Tot_Bil
##      <dbl>     <dbl>
##  1     0.5   -0.316
##  2     0.6   -0.0284
##  3     0.7   -0.184
##  4     0.8   -0.0579
##  5     0.9    0.0546
##  6     1     -0.0174
##  7     1.1   -0.0434
##  8     1.2    0.0791
```

```
##  9     1.3   -0.190
## 10     1.4   -0.117
## # ... with 79 more rows
```

average by Dir_Bil

```
Dir_Bil_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  group_by(Dir_Bil) %>%
  summarize(b_Dir_Bil = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil ))

Dir_Bil_avgs
```

```
## # A tibble: 66 x 2
##    Dir_Bil b_Dir_Bil
##      <dbl>     <dbl>
##  1     0.1    0.0605
##  2     0.2   -0.0137
##  3     0.3   -0.0824
##  4     0.4    0.129
##  5     0.5    0.117
##  6     0.6   -0.122
##  7     0.7   -0.0448
##  8     0.8   -0.0163
##  9     0.9    0.118
## 10     1      0.0864
## # ... with 56 more rows
```

average by Alkphos

```
Alkphos_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  group_by(Alkphos) %>%
  summarize(b_Alkphos = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil ))

Alkphos_avgs
```

```
## # A tibble: 210 x 2
##    Alkphos b_Alkphos
##      <int>     <dbl>
##  1      63    0.589
##  2      75    0.582
##  3      90   -0.342
##  4      92    0.0167
```

```
##  5      97    0.110
##  6      98    0.0406
##  7     100   -0.776
##  8     105   -0.519
##  9     110    0.493
## 10     114   -0.634
## # ... with 200 more rows
```

average by Alamine

```
Alamine_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  group_by(Alamine) %>%
  summarize(b_Alamine = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil - b_Alkphos))

Alamine_avgs
```

```
## # A tibble: 124 x 2
##    Alamine b_Alamine
##      <int>     <dbl>
##  1      10   -0.279
##  2      11   -0.124
##  3      12    0.0216
##  4      13    0.186
##  5      14   -0.0717
##  6      15   -0.0968
##  7      16    0.0117
##  8      17    0.0775
##  9      18    0.0197
## 10      19   -0.338
## # ... with 114 more rows
```

average by Aspartate

```
Aspartate_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  left_join(Alamine_avgs, by='Alamine') %>%
  group_by(Aspartate) %>%
  summarize(b_Aspartate = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil - b_Alkphos -

Aspartate_avgs
```

```
## # A tibble: 145 x 2
##    Aspartate b_Aspartate
##        <int>       <dbl>
##  1        10     -0.0471
##  2        12     -0.183
##  3        13      0.0459
##  4        14      0.128
##  5        15     -0.0725
##  6        16      0.0954
##  7        17     -0.0359
##  8        18      0.365
##  9        19     -0.0236
## 10        20      0.0101
## # ... with 135 more rows
```

average by Tot_Prot

```
Tot_Prot_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  left_join(Alamine_avgs, by='Alamine') %>%
  left_join(Aspartate_avgs, by='Aspartate') %>%
  group_by(Tot_Prot) %>%
  summarize(b_Tot_Prot = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil - b_Alkphos -

Tot_Prot_avgs
```

```
## # A tibble: 55 x 2
##    Tot_Prot b_Tot_Prot
##       <dbl>      <dbl>
##  1      2.7    -0.300
##  2      2.8    -0.171
##  3      3      -0.254
##  4      3.6     0
##  5      3.8    -0.0389
##  6      3.9    -0.520
##  7      4       0
##  8      4.1     0.283
##  9      4.3     0.0554
## 10      4.4     0.217
## # ... with 45 more rows
```

average by Albumin

```
Albumin_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
```

```
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  left_join(Alamine_avgs, by='Alamine') %>%
  left_join(Aspartate_avgs, by='Aspartate') %>%
  left_join(Tot_Prot_avgs, by='Tot_Prot') %>%
  group_by(Albumin) %>%
  summarize(b_Albumin = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil - b_Alkphos - b

Albumin_avgs
```

```
## # A tibble: 39 x 2
##    Albumin b_Albumin
##      <dbl>     <dbl>
##  1     0.9   0
##  2     1     0
##  3     1.4  -0.0138
##  4     1.5   0.0749
##  5     1.6  -0.0393
##  6     1.7  -0.0606
##  7     1.8   0.00900
##  8     1.9  -0.0880
##  9     2     0.0334
## 10     2.1   0.00326
## # ... with 29 more rows
```

average by A__G__Ratio

```
A_G_Ratio_avgs <- train_set %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  left_join(Alamine_avgs, by='Alamine') %>%
  left_join(Aspartate_avgs, by='Aspartate') %>%
  left_join(Tot_Prot_avgs, by='Tot_Prot') %>%
  left_join(Albumin_avgs, by='Albumin') %>%
  group_by(A_G_Ratio) %>%
  summarize(b_A_G_Ratio = mean(Disease - mu_Disease - b_Age - b_Sex -b_Tot_Bil - b_Dir_Bil - b_Alkphos -

A_G_Ratio_avgs
```

```
## # A tibble: 58 x 2
##    A_G_Ratio b_A_G_Ratio
##        <dbl>       <dbl>
##  1      0.3     -0.00129
##  2      0.37    -0.0761
##  3      0.39    -0.121
##  4      0.4      0.0608
##  5      0.45    -0.0138
```

```
##  6      0.46    -0.0523
##  7      0.47     0.122
##  8      0.48    -0.0325
##  9      0.5     -0.0433
## 10      0.52     0.0579
## # ... with 48 more rows
```

Data set for Regularization model

```
Regularization_predicted_Disease <- test_set1 %>%
  left_join(Age_avgs, by='Age') %>%
  left_join(Sex_avgs, by='Sex') %>%
  left_join(Tot_Bil_avgs, by='Tot_Bil') %>%
  left_join(Dir_Bil_avgs, by='Dir_Bil') %>%
  left_join(Alkphos_avgs, by='Alkphos') %>%
  left_join(Alamine_avgs, by='Alamine') %>%
  left_join(Aspartate_avgs, by='Aspartate') %>%
  left_join(Tot_Prot_avgs, by='Tot_Prot') %>%
  left_join(Albumin_avgs, by='Albumin') %>%
  left_join(A_G_Ratio_avgs, by='A_G_Ratio') %>%
  mutate(pred = mu_Disease + b_Age + b_Sex + b_Tot_Bil + b_Dir_Bil + b_Alkphos + b_Alamine + b_Aspartat
  .$pred

Regularization_predicted_Disease
```

```
##  [1] -0.06613039  0.31954702  0.59546995  1.05300786  0.56323404 -0.01810190
##  [7]  0.60352179  0.57276685  0.68175343  0.69259138  0.69222845  0.40959616
## [13]  1.10015642 -0.06361972 -0.02055745  0.99272929  0.92742826  1.37623921
## [19]  0.98555935  0.78380267  1.08063476  1.06648094  0.66535378  0.39838349
## [25]  0.87621985  0.25886318  0.54871080  0.82534023  0.55715606  0.56296536
## [31]  1.02652074  1.00403012  1.07353060  0.12211857  1.01968227  0.95134781
## [37] -0.26547608  1.04625945  0.59752734  0.32615065  0.18734181  0.43037080
## [43]  0.19707816  0.75871505  1.12486704  0.62032495  0.48957876  0.19065746
## [49]  0.30509702  1.20301098  0.69982336  0.44466491  0.17584823  0.47419257
## [55]  0.99428424  0.54377820  1.05731634  0.63589084  0.39383838  0.25155252
## [61]  0.22586565  0.53555230  0.73371268  0.95171517 -0.17879937  0.22079525
## [67]  1.09589198  0.65144046  1.03359540  0.24413782  0.27954451  0.83402700
## [73] -0.07822574  0.58312883  0.56098587  0.77493272  0.14218008
```

transfer numeric to “0”–no disease, “1”–liver disease

```
Regularization_predicted_Disease1 <- ifelse(Regularization_predicted_Disease> 0.5, "1", "0") %>% factor
Regularization_predicted_Disease1
```

```
##  [1] 0 0 1 1 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 1 0 1
## [39] 1 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 0 0 1 1 1 0 0 1 0 1 1 1
## [77] 0
## Levels: 0 1
```

compute accuracy

```
confusionMatrix(Regularization_predicted_Disease1, test_set1$liverdisease)$table
```

```
##           Reference
## Prediction  0  1
##          0 14 15
##          1 15 33
```

```
Accuracy_Regularization <-confusionMatrix(Regularization_predicted_Disease1, test_set1$liverdisease)$ov
Accuracy_Regularization
```

```
## [1] 0.6103896
```

The Accuracy of Regularization Model is not that high.

rmse of Regularization_model

```
Regularization_model_rmse <- RMSE(Regularization_predicted_Disease, test_set1$Disease)
Regularization_model_rmse
```

```
## [1] 0.5589416
```

The Accuracy of this model is not as high as the previous one and the the rmse of this model is not as small as the previous one, so this model is not that good.

model 9. Tuning regularization model

add Penalized least squares with different tuning

I add tunning to penalty large estimates that come from small sample size.

```
lambdas_dl <- seq(0, 40, 0.5)

all_reg_lambda_dl_rmses <- sapply(lambdas_dl, function(l){

  mu_Disease<- mean(train_set$Disease)

  b_iAge <- train_set %>%
    group_by(Age) %>%
    summarize(b_iAge = sum(Disease - mu_Disease)/(n()+l))

  b_iSex <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    group_by(Sex) %>%
    summarize(b_iSex = mean(Disease - mu_Disease - b_iAge)/(n()+l))
```

```r
b_iTot_Bil <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  group_by(Tot_Bil) %>%
  summarize(b_iTot_Bil = mean(Disease - mu_Disease - b_iAge - b_iSex)/(n()+l))

b_iDir_Bil <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  group_by(Dir_Bil) %>%
  summarize(b_iDir_Bil = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil)/(n()+l))

b_iAlkphos <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  group_by(Alkphos) %>%
  summarize(b_iAlkphos = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil)/(n()+l

b_iAlamine <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  group_by(Alamine) %>%
  summarize(b_iAlamine = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iAl

b_iAspartate <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  group_by(Aspartate) %>%
  summarize(b_iAspartate = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_i


b_iTot_Prot <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  left_join(b_iAspartate, by='Aspartate') %>%
  group_by(Tot_Prot) %>%
  summarize(b_iTot_Prot = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iA
```

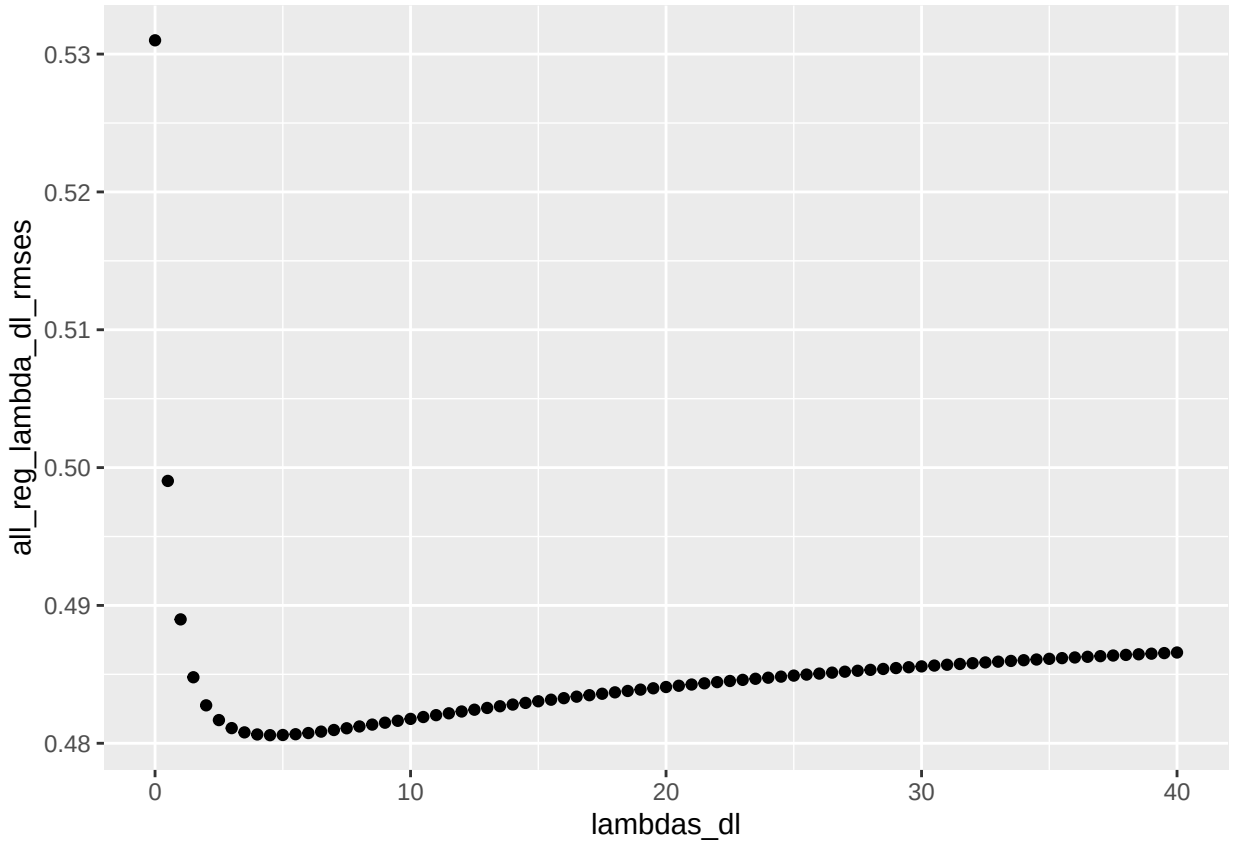
```
  b_iAlbumin <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    left_join(b_iAlkphos, by='Alkphos') %>%
    left_join(b_iAlamine, by='Alamine') %>%
    left_join(b_iAspartate, by='Aspartate') %>%
    left_join(b_iTot_Prot, by='Tot_Prot') %>%
    group_by(Albumin) %>%
    summarize(b_iAlbumin = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iAl


  b_iA_G_Ratio <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    left_join(b_iAlkphos, by='Alkphos') %>%
    left_join(b_iAlamine, by='Alamine') %>%
    left_join(b_iAspartate, by='Aspartate') %>%
    left_join(b_iTot_Prot, by='Tot_Prot') %>%
    left_join(b_iAlbumin, by='Albumin') %>%
    group_by(A_G_Ratio) %>%
    summarize(b_iA_G_Ratio = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iA

  Regularization_d_predicted_Disease <- test_set1 %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    left_join(b_iAlkphos, by='Alkphos') %>%
    left_join(b_iAlamine, by='Alamine') %>%
    left_join(b_iAspartate, by='Aspartate') %>%
    left_join(b_iTot_Prot, by='Tot_Prot') %>%
    left_join(b_iAlbumin, by='Albumin') %>%
    left_join(b_iA_G_Ratio, by='A_G_Ratio') %>%
    mutate(pred = mu_Disease + b_iAge + b_iSex + b_iTot_Bil+ b_iDir_Bil + b_iAlkphos + b_iAlamine + b_iA
    pull(pred)

  Regularization_d_predicted_Disease1 <- ifelse(Regularization_d_predicted_Disease> 0.5, "1", "0") %>% f


  return(RMSE(Regularization_d_predicted_Disease, test_set1$Disease))
})

all_reg_lambda_dl_rmses
```

```
##  [1] 0.5309996 0.4990337 0.4889876 0.4847885 0.4827521 0.4816815 0.4810999
##  [8] 0.4807890 0.4806384 0.4805873 0.4806005 0.4806563 0.4807408 0.4808448
## [15] 0.4809620 0.4810881 0.4812199 0.4813552 0.4814925 0.4816304 0.4817680
## [22] 0.4819048 0.4820402 0.4821738 0.4823054 0.4824348 0.4825618 0.4826863
## [29] 0.4828084 0.4829279 0.4830449 0.4831594 0.4832713 0.4833807 0.4834877
```

```
## [36] 0.4835923 0.4836946 0.4837945 0.4838922 0.4839876 0.4840810 0.4841722
## [43] 0.4842614 0.4843487 0.4844340 0.4845174 0.4845991 0.4846789 0.4847571
## [50] 0.4848335 0.4849084 0.4849816 0.4850534 0.4851236 0.4851925 0.4852599
## [57] 0.4853259 0.4853906 0.4854540 0.4855162 0.4855771 0.4856369 0.4856955
## [64] 0.4857530 0.4858094 0.4858647 0.4859190 0.4859723 0.4860246 0.4860760
## [71] 0.4861264 0.4861760 0.4862246 0.4862724 0.4863194 0.4863655 0.4864108
## [78] 0.4864554 0.4864992 0.4865423 0.4865847
```

summary with corresponding rmse and find the lambda that get the smallest rmse

```
qplot(lambdas_dl, all_reg_lambda_dl_rmses)
```




```
lambda_rmse <- lambdas_dl [which.min(all_reg_lambda_dl_rmses)]
lambda_rmse
```

```
## [1] 4.5
```

```
all_reg_lambda_dl_rmses[which.min(all_reg_lambda_dl_rmses)]
```

```
## [1] 0.4805873
```

When lambda=4.5, we got the smallest rmse 10. It seems add Penalized least squares tuning not improve rsme.

prediction Accuracy by add Penalized least squares with different tuning

```
lambdas_dl <- seq(0, 40, 0.5)

all_reg_lambda_dl_Accuracy <- sapply(lambdas_dl, function(k){

  mu_Disease<- mean(train_set$Disease)

  b_iAge <- train_set %>%
    group_by(Age) %>%
    summarize(b_iAge = sum(Disease - mu_Disease)/(n()+k))

  b_iSex <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    group_by(Sex) %>%
    summarize(b_iSex = mean(Disease - mu_Disease - b_iAge)/(n()+k))

  b_iTot_Bil <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    group_by(Tot_Bil) %>%
    summarize(b_iTot_Bil = mean(Disease - mu_Disease - b_iAge - b_iSex)/(n()+k))

  b_iDir_Bil <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    group_by(Dir_Bil) %>%
    summarize(b_iDir_Bil = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil)/(n()+k))

  b_iAlkphos <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    group_by(Alkphos) %>%
    summarize(b_iAlkphos = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil)/(n()+k

  b_iAlamine <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    left_join(b_iAlkphos, by='Alkphos') %>%
    group_by(Alamine) %>%
    summarize(b_iAlamine = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iAl

  b_iAspartate <- train_set %>%
    left_join(b_iAge, by='Age') %>%
    left_join(b_iSex, by='Sex') %>%
    left_join(b_iTot_Bil, by='Tot_Bil') %>%
    left_join(b_iDir_Bil, by='Dir_Bil') %>%
    left_join(b_iAlkphos, by='Alkphos') %>%
```

```
  left_join(b_iAlamine, by='Alamine') %>%
  group_by(Aspartate) %>%
  summarize(b_iAspartate = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_i


b_iTot_Prot <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  left_join(b_iAspartate, by='Aspartate') %>%
  group_by(Tot_Prot) %>%
  summarize(b_iTot_Prot = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iA


b_iAlbumin <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  left_join(b_iAspartate, by='Aspartate') %>%
  left_join(b_iTot_Prot, by='Tot_Prot') %>%
  group_by(Albumin) %>%
  summarize(b_iAlbumin = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_iAl


b_iA_G_Ratio <- train_set %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  left_join(b_iAspartate, by='Aspartate') %>%
  left_join(b_iTot_Prot, by='Tot_Prot') %>%
  left_join(b_iAlbumin, by='Albumin') %>%
  group_by(A_G_Ratio) %>%
  summarize(b_iA_G_Ratio = mean(Disease - mu_Disease - b_iAge - b_iSex - b_iTot_Bil- b_iDir_Bil - b_i

Regularization_d_predicted_Disease <- test_set1 %>%
  left_join(b_iAge, by='Age') %>%
  left_join(b_iSex, by='Sex') %>%
  left_join(b_iTot_Bil, by='Tot_Bil') %>%
  left_join(b_iDir_Bil, by='Dir_Bil') %>%
  left_join(b_iAlkphos, by='Alkphos') %>%
  left_join(b_iAlamine, by='Alamine') %>%
  left_join(b_iAspartate, by='Aspartate') %>%
  left_join(b_iTot_Prot, by='Tot_Prot') %>%
  left_join(b_iAlbumin, by='Albumin') %>%
```

```
    left_join(b_iA_G_Ratio, by='A_G_Ratio') %>%
    mutate(pred = mu_Disease + b_iAge + b_iSex + b_iTot_Bil+ b_iDir_Bil + b_iAlkphos + b_iAlamine + b_i
    pull(pred)

  Regularization_d_predicted_Disease1 <- ifelse(Regularization_d_predicted_Disease> 0.5, "1", "0") %>% 


  return(confusionMatrix(Regularization_d_predicted_Disease1, test_set1$liverdisease)$overall[["Accurac

})
```

```
## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.

## Warning in confusionMatrix.default(Regularization_d_predicted_Disease1, : Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
all_reg_lambda_dl_Accuracy
```

```
##  [1] 0.6103896 0.6103896 0.5974026 0.5974026 0.6103896 0.6103896 0.5844156
##  [8] 0.5844156 0.5974026 0.6233766 0.6233766 0.6233766 0.6233766 0.6103896
## [15] 0.6103896 0.6103896 0.6233766 0.6233766 0.6103896 0.6233766 0.6233766
## [22] 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766
## [29] 0.6233766 0.6233766 0.6233766 0.6233766 0.6103896 0.6103896 0.6103896
## [36] 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896
## [43] 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896 0.6103896
## [50] 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766
## [57] 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766
## [64] 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766
## [71] 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766 0.6233766
## [78] 0.6233766 0.6233766 0.6233766 0.6233766
```

the Accuracy of Tuning_regularization_model

```
lambdas_dl [which.min(all_reg_lambda_dl_rmses)]
```

```
## [1] 4.5
```

```
all_reg_lambda_dl_Accuracy[which.min(all_reg_lambda_dl_rmses)]
```

```
## [1] 0.6233766
```

the rmse of Tuning_regularization_model

```
lambdas_dl [which.max(all_reg_lambda_dl_Accuracy)]
```

```
## [1] 4.5
```

```
all_reg_lambda_dl_rmses[which.max(all_reg_lambda_dl_Accuracy)]
```

```
## [1] 0.4805873
```

the highest accuracy of Tuning_regularization_model

```
the_highest_acccuracy_of_Tuning_regularization_model<-  all_reg_lambda_dl_Accuracy [which.max(all_reg_la
the_highest_acccuracy_of_Tuning_regularization_model
```

```
## [1] 0.6233766
```

the smallest rmse of Tuning_regularization_model

```
the_smallest_rmse_of_all_Tuning_regularization_model <- all_reg_lambda_dl_rmses[which.min(all_reg_lambda
the_smallest_rmse_of_all_Tuning_regularization_model
```

```
## [1] 0.4805873
```

Based on the highest accuracy and the smallest rsme, this model does not looks like a good one.

III.Result

I try 9 models, summary as the following: ## ** summary of accuracy and rmse **

```
asscracy_rmse_summary<-data_frame ( Method = c("liner Regression Model", "Logistic Regression Model","kn
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## Please use `tibble()` instead.
```

```
asscracy_rmse_summary
```

```
## # A tibble: 9 x 3
##   Method                       Accuracy  rmse
##   <chr>                           <dbl> <dbl>
## 1 liner Regression Model          0.736 1.07
## 2 Logistic Regression Model       0.713 0.442
## 3 knn Model                       0.684 0.444
## 4 qda Model                       0.690 0.443
## 5 lda Model                       0.684 0.444
## 6 Decision Trees                  0.736 0.441
## 7 random Forest Trees             0.684 0.444
## 8 Regularization model            0.610 0.559
## 9 Tuning regularization model     0.623 0.481
```

The 2 highest accuracy models —liner Regression Model and Decision Trees give all prediction "1"–liver disease, this is not true. So I will exclude these 2 models. When running the Regularization model and Tuning regularization model, I remove "NA"from the data set, this make the data size too small, which is only 77, 12, maybe this is the reason the accuracy of Regularization model is not that high, which is 0.6103896, and rmse 0.5589416 is larger compare to other model. In Tuning regularization model model, the highest accuracy and the smallest rmse are not with the same lambda. Tuning regularization model get accuracy of 0.6233766 which is not the highest of all the 9 model and the corresponding rmse is 0.4805873. Thinking of both accuracy and rmsem, qda Model get the highest accuracy 0.6896552 and the smallest rmse 0.4433862, the prediction table looks reasonable too. So I think the qda model is the best prediction model for this liver disease data set.

IV. Conclusion

This project is to build a model that can predict a liver disease status based on features of "Age", "Sex", "Tot_Bil", "Dir_Bil", "Alkphos", "Alamine", "Aspartate", "Tot_Prot", "Albumin", "A_G_Ratio". After review the data set, I find all these features are related to disease status. So we use all these features in building models. 9 models are created—-liner Regression Model, Logistic Regression Model, knn Model, qda

Model, lda Model, Decision Trees, random Forest Trees, Regularization model, Tuning regularization model, the qda Model is the best one with a accuracy of 0.6896552 and a rmse of 0.4433862. Building of a disease prediction model will create a new method for disease diagnosis in public health and clinic.

Limitation: These data set size is not large enough to get good prdiction, espically when I exculded missing and "NA" value.

In the future, as more samples are added to this data set, we can try again. or maybe try some more model, for example, PCA.