Documentation for the project on Predicting Financial Strength of Companies using Machine Learning

Software used- R Studio

Codes used to deploy machine learning algorithms on the model

# Code to perform multinomial logistic regression and creating the f score column

1. ***# Connecting to the data set***
2. data <- read.csv("Data Set.csv",TRUE,",")
3. class(data)
4. ***# Removing Unwanted columns***
5. data<-data[ ,-17:-19]
6. ***# Checking that the unwanted columns are removed***
7. names(data)
8. ***# Creating a new column to store the F score of a company and assigning it the value NA***
9. data$F.score <- NA
10. ***# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0***
11. a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

1. ***# Converting the a0 value to numeric from character type and assigning the value to variable - a***
2. a<-as.numeric(a0)
3. ***# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0***
4. b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

1. ***# Converting the b0 value to numeric from character type and assigning the value to variable - b***
2. b<-as.numeric(b0)
3. ***# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0***
4. c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

1. ***# Converting the c0 value to numeric from character type and assigning the value to variable - c***
2. c<-as.numeric(c0)
3. ***# Replacing NA values with 1***
4. c[which(is.na(c))]<-1
5. ***# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0***
6. d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

1. ***#*** ***Converting the d0 value to numeric from character type and assigning the value to variable - d***
2. d<-as.numeric(d0)
3. ***# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0***
4. e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

1. ***# Converting the e0 value to numeric from character type and assigning the value to variable - e***
2. e<-as.numeric(e0)
3. ***# Replacing NA values with 1***
4. e[which(is.na(e))]<-1
5. ***# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0***
6. f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

1. ***# Converting the f0 value to numeric from character type and assigning the value to variable - f***
2. f<-as.numeric(f0)
3. ***# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0***
4. g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1, ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))
5. ***# Converting the g0 value to numeric from character type and assigning the value to variable - g***

g<-as.numeric(g0)

1. ***# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0***
2. h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

1. ***# Converting the h0 value to numeric from character type and assigning the value to variable - h***
2. h<-as.numeric(h0)
3. ***# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0***
4. i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

1. ***# Converting the i0 value to numeric from character type and assigning the value to variable - i***
2. i<-as.numeric(i0)
3. ***# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score***
4. data$F.score <- a+b+c+d+e+f+g+h+i
5. data$F.score[which(is.na(data$F.score))]<-0
6. **#Creating new Column that determines the status of every F score**
7. for(i in 1:nrow(data)){
8. if(data$F.score[i]>=7){
9. data$status[i]<-"Financially Strong"
10. }else if (data$F.score[i]<=3){
11. data$status[i]<-"Financially weak"
12. }else {
13. data$status[i]<-"Financially Moderate"
14. }
15. }
16. **#Removing Rows containing NA values**
17. data1<-na.omit(data)
18. **#Extracting Independent and dependent Variable**
19. x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]
20. y<-data1[ ,c("status")]
21. dataset<-data.frame(x,y)
22. dataset$Net.Income<-as.numeric(dataset$Net.Income)
23. dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)
24. dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)
25. dataset$y=factor(dataset$y)
26. sapply(dataset,class)
27. print(dataset)
28. dataset$y<-relevel(dataset$y,ref ='Financially Strong')
29. **#splitting the data**
30. split<-sample.split(dataset,SplitRatio=0.8)
31. split
32. train<-subset(dataset,split=="TRUE")
33. test<-subset(dataset,split=="FALSE")
34. mymodel<-multinom(y ~ Net.Income+ROA+operating.cash.flow.2020+Long.term.debt.2019+Long.term.debt.2020+Current.Ratio.2019+Current.Ratio.2020+Outstanding.Shares.in.2019+Outstanding.Shares.in.2020+Gross.margin.2019+Gross.margin.2020+Asset.Turnover.Ratio.2019+Asset.Turnover.Ratio.2020,data = train)
35. summary(mymodel)
36. res<-predict(mymodel,train)
37. res
38. confmatrix<- table(Actual\_value=train$y, Predicted\_value=res)
39. confmatrix
40. Acurracy<-sum(diag(confmatrix))/sum(confmatrix)
41. Acurracy
42. **#plotting the correlation values**
43. d<-train[1:13]
44. cr<-cor(d)
45. corrplot(cr,type = "upper")
46. corrplot(cr,type = "lower")
47. **#Creating the ROC curve**
48. pred<-predict(mymodel,test,type = "prob")
49. r<- multiclass.roc(test$y,pred,percent=T)
50. rs<-r[["rocs"]]
51. r1<-rs[[1]][[1]]
52. r2<-rs[[2]][[1]]
53. r3<-rs[[3]][[2]]
54. plot.roc(r1,col='blue')
55. plot.roc(r2,add=T,col='green')
56. plot.roc(r3,add=T,col='red')

Code to perform Random forest algorithm

library(caTools)

install.packages("randomForest")

library(randomForest)

library(caret)

**# Connecting to the data set**

data <- read.csv("Data Set.csv",TRUE,",")

class(data)

**# Removing Unwanted columns**

data<-data[ ,-17:-19]

**# Checking that the unwanted columns are removed**

names(data)

**# Creating a new column to store the F score of a company and assigning it the value NA**

data$F.score <- NA

**# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0**

a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

**# Converting the a0 value to numeric from character type and assigning the value to variable - a**

a<-as.numeric(a0)

**# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0**

b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

**# Converting the b0 value to numeric from character type and assigning the value to variable - b**

b<-as.numeric(b0)

**# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0**

c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

**# Converting the c0 value to numeric from character type and assigning the value to variable - c**

c<-as.numeric(c0)

**# Replacing NA values with 1**

c[which(is.na(c))]<-1

**# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0**

d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

**# Converting the d0 value to numeric from character type and assigning the value to variable - d**

d<-as.numeric(d0)

**# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0**

e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

**# Converting the e0 value to numeric from character type and assigning the value to variable - e**

e<-as.numeric(e0)

**# Replacing NA values with 1**

e[which(is.na(e))]<-1

**# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

**# Converting the f0 value to numeric from character type and assigning the value to variable - f**

f<-as.numeric(f0)

**# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0**

g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1,

ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))

**# Converting the g0 value to numeric from character type and assigning the value to variable - g**

g<-as.numeric(g0)

**# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

**# Converting the h0 value to numeric from character type and assigning the value to variable - h**

h<-as.numeric(h0)

**# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

**# Converting the i0 value to numeric from character type and assigning the value to variable - i**

i<-as.numeric(i0)

**# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score**

data$F.score <- a+b+c+d+e+f+g+h+i

data$F.score[which(is.na(data$F.score))]<-0

#**Creating new Column that determines the status of every F score**

for(i in 1:nrow(data)){

if(data$F.score[i]>=7){

data$status[i]<-"Financially Strong"

}else if (data$F.score[i]<=3){

data$status[i]<-"Financially weak"

}else {

data$status[i]<-"Financially Moderate"

}

}

#Removing Rows containing NA values

data1<-na.omit(data)

**#Extracting Independent and dependent Variable**

x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]

y<-data1[ ,c("status")]

dataset<-data.frame(x,y)

dataset$Net.Income<-as.numeric(dataset$Net.Income)

dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)

dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)

dataset$y=factor(dataset$y)

sapply(dataset,class)

table(dataset$y)

set.seed(123)

split<-sample.split(dataset,SplitRatio=0.7)

split

train<-subset(dataset,split=="TRUE")

test<-subset(dataset,split=="FALSE")

set.seed(222)

rf<-randomForest(y~.,data = train,

ntree=200,

mtry=1,

importance=TRUE,

proximity=TRUE)

**#prediction using train**

p1<-predict(rf,train)

confusionMatrix(p1,train$y)

**#prediction using test**

p2<-predict(rf,test)

confusionMatrix(p2,test$y)

**#error Rate of random forest**

plot(rf)

**#tune mtry**

t<-tuneRF(train[,-14],train[,14],

stepFactor = 0.7,

plot= TRUE,

ntreeTry=200,

trace = TRUE,

improve = 0.05)

varImpPlot(rf,

sort=TRUE,

n.var = 10,

main = "Top 10 - Variable Importance")

MDSplot(rf,train$y)

***Code to perform k means algorithm***

library(factoextra)

**# Connecting to the data set**

data <- read.csv("Data Set.csv",TRUE,",")

class(data)

**# Removing Unwanted columns**

data<-data[ ,-17:-19]

**# Checking that the unwanted columns are removed**

names(data)

**# Creating a new column to store the F score of a company and assigning it the value NA**

data$F.score <- NA

**# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0**

a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

**# Converting the a0 value to numeric from character type and assigning the value to variable - a**

a<-as.numeric(a0)

**# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0**

b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

**# Converting the b0 value to numeric from character type and assigning the value to variable - b**

b<-as.numeric(b0)

**# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0**

c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

**# Converting the c0 value to numeric from character type and assigning the value to variable - c**

c<-as.numeric(c0)

**# Replacing NA values with 1**

c[which(is.na(c))]<-1

**# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0**

d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

**# Converting the d0 value to numeric from character type and assigning the value to variable - d**

d<-as.numeric(d0)

**# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0**

e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

**# Converting the e0 value to numeric from character type and assigning the value to variable - e**

e<-as.numeric(e0)

**# Replacing NA values with 1**

e[which(is.na(e))]<-1

**# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

**# Converting the f0 value to numeric from character type and assigning the value to variable - f**

f<-as.numeric(f0)

**# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0**

g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1,

ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))

**# Converting the g0 value to numeric from character type and assigning the value to variable - g**

g<-as.numeric(g0)

**# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

**# Converting the h0 value to numeric from character type and assigning the value to variable - h**

h<-as.numeric(h0)

**# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

**# Converting the i0 value to numeric from character type and assigning the value to variable - i**

i<-as.numeric(i0)

**# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score**

data$F.score <- a+b+c+d+e+f+g+h+i

data$F.score[which(is.na(data$F.score))]<-0

**#Creating new Column that determines the status of every F score**

for(i in 1:nrow(data)){

if(data$F.score[i]>=7){

data$status[i]<-"Financially Strong"

}else if (data$F.score[i]<=3){

data$status[i]<-"Financially weak"

}else {

data$status[i]<-"Financially Moderate"

}

}

**#Removing Rows containing NA values**

data1<-na.omit(data)

**#Extracting Independent and dependent Variable**

x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]

y<-data1[ ,c("status")]

dataset<-data.frame(x,y)

dataset$Net.Income<-as.numeric(dataset$Net.Income)

dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)

dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)

dataset$y=factor(dataset$y)

sapply(dataset,class)

print(dataset)

dataset=data1

data1=data1[,-1:-3]

data1=data1[,-15]

data\_scale=scale(data1)

**#distance**

data1<-dist(data\_scale)

**#calculating how many clusters we need**

fviz\_nbclust(data\_scale,kmeans,method = "wss")+

labs(subtitle = "Elbow method")

km.out<-kmeans(data\_scale,centers = 3,nstart = 169)

print(km.out)

km.cluster<-km.out$cluster

rownames(data\_scale)<-paste(dataset$Company.Name,1:dim(dataset)[1],sep = "\_")

ggp<-fviz\_cluster(list(data=data\_scale,cluster=km.cluster))

ggp+ # Modify labels and colors

scale\_color\_manual(labels = c("Financially Strong", "Financially Weak", "Financially Moderate"),

values = c("red", "green", "blue"))

Code to perform SVM algorithm

**# Connecting to the data set**

data <- read.csv("Data Set.csv",TRUE,",")

class(data)

**# Removing Unwanted columns**

data<-data[ ,-17:-19]

**# Checking that the unwanted columns are removed**

names(data)

**# Creating a new column to store the F score of a company and assigning it the value NA**

data$F.score <- NA

**# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0**

a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

**# Converting the a0 value to numeric from character type and assigning the value to variable - a**

a<-as.numeric(a0)

**# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0**

b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

**# Converting the b0 value to numeric from character type and assigning the value to variable - b**

b<-as.numeric(b0)

**# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0**

c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

**# Converting the c0 value to numeric from character type and assigning the value to variable - c**

c<-as.numeric(c0)

**# Replacing NA values with 1**

c[which(is.na(c))]<-1

**# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0**

d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

**# Converting the d0 value to numeric from character type and assigning the value to variable - d**

d<-as.numeric(d0)

**# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0**

e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

**# Converting the e0 value to numeric from character type and assigning the value to variable - e**

e<-as.numeric(e0)

**# Replacing NA values with 1**

e[which(is.na(e))]<-1

**# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

**# Converting the f0 value to numeric from character type and assigning the value to variable - f**

f<-as.numeric(f0)

**# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0**

g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1,

ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))

**# Converting the g0 value to numeric from character type and assigning the value to variable - g**

g<-as.numeric(g0)

**# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

**# Converting the h0 value to numeric from character type and assigning the value to variable - h**

h<-as.numeric(h0)

**# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

**# Converting the i0 value to numeric from character type and assigning the value to variable - i**

i<-as.numeric(i0)

**# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score**

data$F.score <- a+b+c+d+e+f+g+h+i

data$F.score[which(is.na(data$F.score))]<-0

**#Creating new Column that determines the status of every F score**

for(i in 1:nrow(data)){

if(data$F.score[i]>=7){

data$status[i]<-"Financially Strong"

}else if (data$F.score[i]<=3){

data$status[i]<-"Financially weak"

}else {

data$status[i]<-"Financially Moderate"

}

}

**#Removing Rows containing NA values**

data1<-na.omit(data)

**#Extracting Independent and dependent Variable**

x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]

y<-data1[ ,c("status")]

dataset<-data.frame(x,y)

dataset$Net.Income<-as.numeric(dataset$Net.Income)

dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)

dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)

dataset$y=factor(dataset$y)

sapply(dataset,class)

print(dataset)

library(caTools)

**#splitting the data**

split<-sample.split(dataset,SplitRatio=0.8)

split

train<-subset(dataset,split=="TRUE")

test<-subset(dataset,split=="FALSE")

library(e1071)

plot(dataset)

mymodel<-svm(y~.,data=test)

summary(mymodel)

pred<-predict(mymodel,test)

tab<-table(predicted=pred, actual=test$y)

tab

Acurracy<-sum(diag(tab))/sum(tab)

Acurracy

1-Acurracy

plot(mymodel,data=test,

Net.Income~Gross.margin.2020)

Deploying the model on a API using plumber application

***Code used to create the RDS file which is used for building the API***

**library(caTools)**

install.packages("randomForest")

library(randomForest)

library(caret)

**# Connecting to the data set**

data <- read.csv("Data Set.csv",TRUE,",")

class(data)

**# Removing Unwanted columns**

data<-data[ ,-17:-19]

**# Checking that the unwanted columns are removed**

names(data)

**# Creating a new column to store the F score of a company and assigning it the value NA**

data$F.score <- NA

**# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0**

a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

**# Converting the a0 value to numeric from character type and assigning the value to variable - a**

a<-as.numeric(a0)

**# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0**

b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

**# Converting the b0 value to numeric from character type and assigning the value to variable - b**

b<-as.numeric(b0)

**# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0**

c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

**# Converting the c0 value to numeric from character type and assigning the value to variable - c**

c<-as.numeric(c0)

**# Replacing NA values with 1**

c[which(is.na(c))]<-1

**# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0**

d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

**# Converting the d0 value to numeric from character type and assigning the value to variable - d**

d<-as.numeric(d0)

**# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0**

e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

**# Converting the e0 value to numeric from character type and assigning the value to variable - e**

e<-as.numeric(e0)

**# Replacing NA values with 1**

e[which(is.na(e))]<-1

**# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

**# Converting the f0 value to numeric from character type and assigning the value to variable - f**

f<-as.numeric(f0)

**# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0**

g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1,

ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))

**# Converting the g0 value to numeric from character type and assigning the value to variable - g**

g<-as.numeric(g0)

**# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

**# Converting the h0 value to numeric from character type and assigning the value to variable - h**

h<-as.numeric(h0)

**# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0**

i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

**# Converting the i0 value to numeric from character type and assigning the value to variable - i**

i<-as.numeric(i0)

**# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score**

data$F.score <- a+b+c+d+e+f+g+h+i

data$F.score[which(is.na(data$F.score))]<-0

**#Creating new Column that determines the status of every F score**

for(i in 1:nrow(data)){

if(data$F.score[i]>=7){

data$status[i]<-"Financially Strong"

}else if (data$F.score[i]<=3){

data$status[i]<-"Financially weak"

}else {

data$status[i]<-"Financially Moderate"

}

}

**#Removing Rows containing NA values**

data1<-na.omit(data)

**#Extracting Independent and dependent Variable**

x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]

y<-data1[ ,c("status")]

dataset<-data.frame(x,y)

dataset$Net.Income<-as.numeric(dataset$Net.Income)

dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)

dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)

dataset$y=factor(dataset$y)

sapply(dataset,class)

table(dataset$y)

set.seed(123)

split<-sample.split(dataset,SplitRatio=0.7)

split

train<-subset(dataset,split=="TRUE")

test<-subset(dataset,split=="FALSE")

set.seed(222)

rf<-randomForest(y~.,data = train,

ntree=200,

mtry=1,

importance=TRUE,

proximity=TRUE)

**#prediction using train**

p1<-predict(rf,train)

confusionMatrix(p1,train$y)

**#prediction using test**

p2<-predict(rf,test)

confusionMatrix(p2,test$y)

**#error Rate of random forest**

plot(rf)

**#tune mtry**

t<-tuneRF(train[,-14],train[,14],

stepFactor = 0.7,

plot= TRUE,

ntreeTry=200,

trace = TRUE,

improve = 0.05)

varImpPlot(rf,

sort=TRUE,

n.var = 10,

main = "Top 10 - Variable Importance")

MDSplot(rf,train$y)

library(dplyr)

**## Make a prediction function**

NewPredictions <- function(rf, train){

new.predictions <- predict(object = rf, newdata = train)

return(new.predictions)

}

modellist <- vector(mode = 'list')

**# Save fitted model here.**

modellist$modelobject <- rf

modellist$NewPredictions <- NewPredictions

saveRDS(object = modellist , file = 'HH.rds')

names(dataset)

***Code used to create the API***

library(plumber)

library(randomForest)

#\* return the input

#\*

#\* @get /patrol

function(messg = ""){

list(messg = paste0("Hi I am listening '", messg, "'"))

}

## Load the model

modellist = readRDS("HH.rds")

## Lets make the predictions

#\* @param Net.Income

#\* @param ROA

#\* @param operating.cash.flow.2020

#\* @param Long.term.debt.2019

#\* @param Long.term.debt.2020

#\* @param Current.Ratio.2019

#\* @param Current.Ratio.2020

#\* @param Outstanding.Shares.in.2019

#\* @param Outstanding.Shares.in.2020

#\* @param Gross.margin.2019

#\* @param Gross.margin.2020

#\* @param Asset.Turnover.Ratio.2019

#\* @param Asset.Turnover.Ratio.2020

#\* @get /predict

predictions <- function(Net.Income, ROA, operating.cash.flow.2020, Long.term.debt.2019,Long.term.debt.2020,Current.Ratio.2019,Current.Ratio.2020,Outstanding.Shares.in.2019,Outstanding.Shares.in.2020,Gross.margin.2019,Gross.margin.2020,Asset.Turnover.Ratio.2019,Asset.Turnover.Ratio.2020){

Net.Income <- as.numeric(Net.Income )

ROA <- as.numeric(ROA)

operating.cash.flow.2020 <- as.numeric(operating.cash.flow.2020)

Long.term.debt.2019 <- as.numeric(Long.term.debt.2019)

Long.term.debt.2020 <- as.numeric(Long.term.debt.2020)

Current.Ratio.2019 <- as.numeric(Current.Ratio.2019)

Current.Ratio.2020 <- as.numeric(Current.Ratio.2020)

Outstanding.Shares.in.2019 <- as.numeric(Outstanding.Shares.in.2019)

Outstanding.Shares.in.2020 <- as.numeric(Outstanding.Shares.in.2020)

Gross.margin.2019 <- as.numeric(Gross.margin.2019)

Gross.margin.2020 <- as.numeric(Gross.margin.2020)

Asset.Turnover.Ratio.2019 <- as.numeric(Asset.Turnover.Ratio.2019)

Asset.Turnover.Ratio.2020 <- as.numeric(Asset.Turnover.Ratio.2020)

X.new <- data.frame(Net.Income= Net.Income,

ROA = ROA ,

operating.cash.flow.2020=operating.cash.flow.2020,

Long.term.debt.2019 = Long.term.debt.2019,

Long.term.debt.2020=Long.term.debt.2020,

Current.Ratio.2019=Current.Ratio.2019,

Current.Ratio.2020=Current.Ratio.2020,

Outstanding.Shares.in.2019=Outstanding.Shares.in.2019,

Outstanding.Shares.in.2020=Outstanding.Shares.in.2020,

Gross.margin.2019=Gross.margin.2019,

Gross.margin.2020=Gross.margin.2020,

Asset.Turnover.Ratio.2019=Asset.Turnover.Ratio.2019,

Asset.Turnover.Ratio.2020=Asset.Turnover.Ratio.2020)

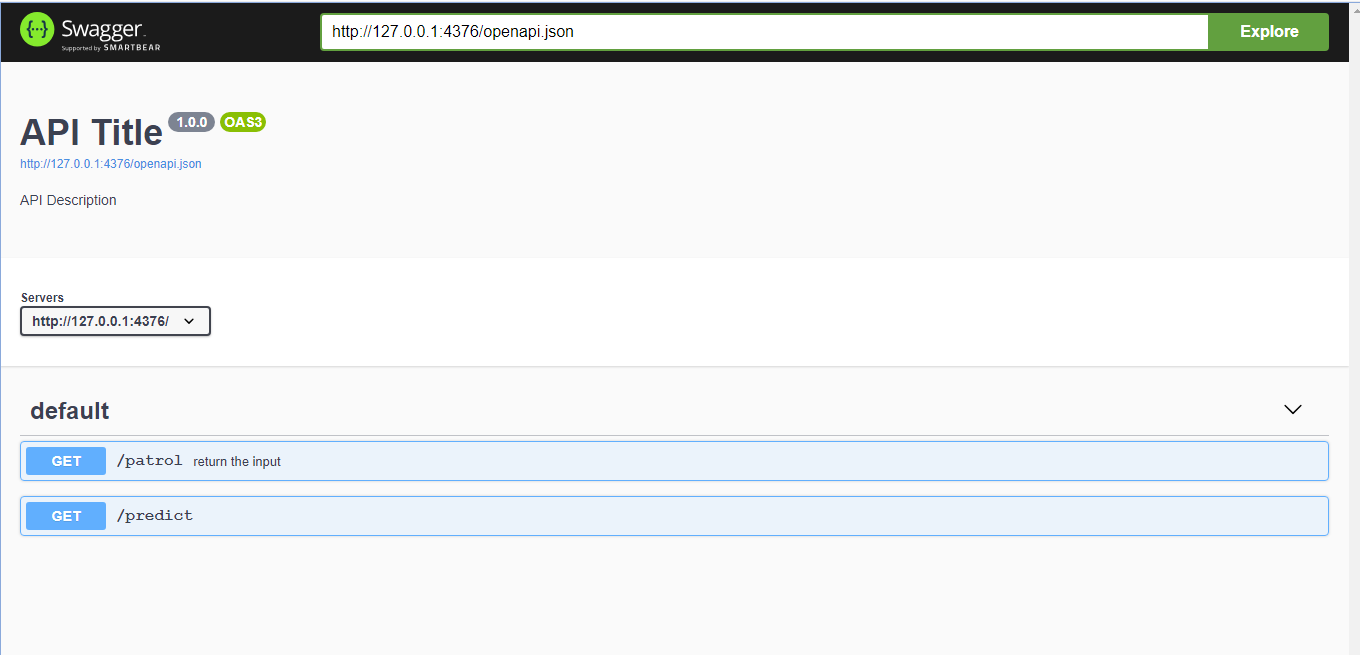
y.pred <- modellist$NewPredictions(modellist$modelobject,X.new)

return(y.pred)

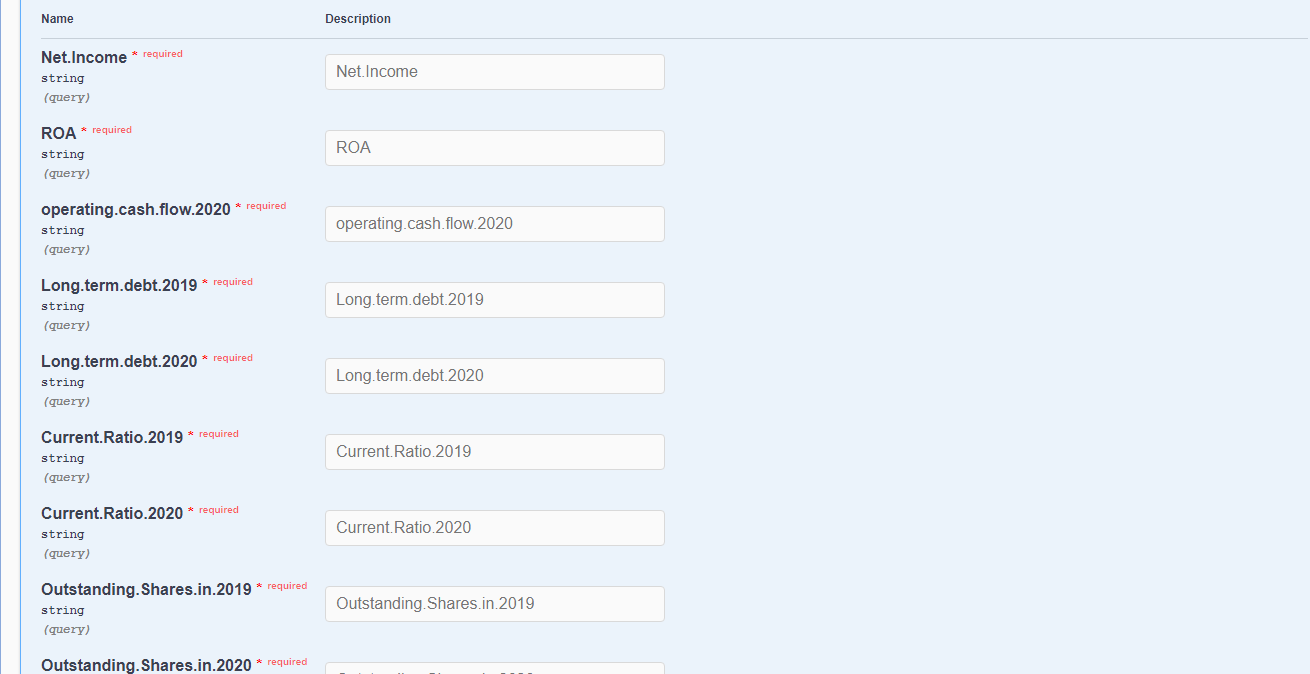
}

API Interface

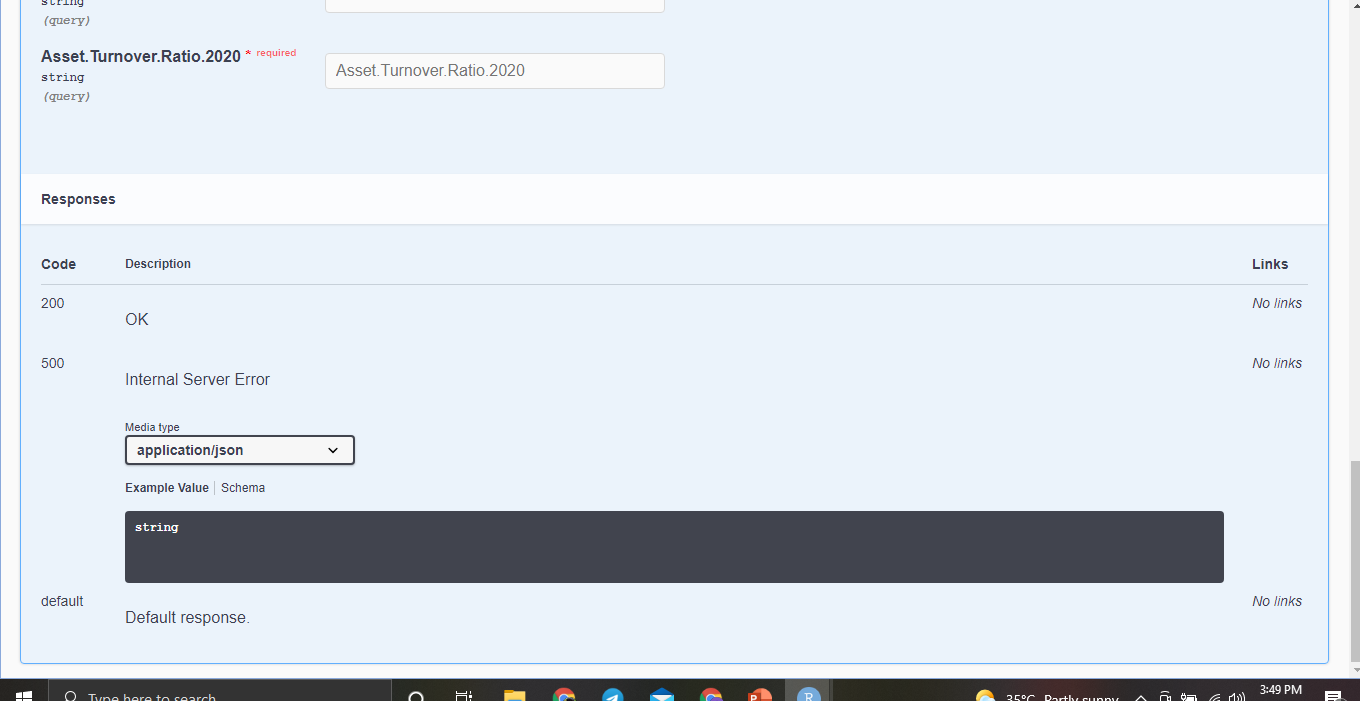
***Below image represents the API interface***

******

***Below picture shows input parameters section to be imputed by a user***

******

***Below section shows the output section of the API***

******

Deploying the model on a Web Application using R Shiny application

***Code used to create the RDS file which is used for building the web application***

library(caTools)

install.packages("randomForest")

library(randomForest)

library(caret)

# Connecting to the data set

data <- read.csv("Data Set.csv",TRUE,",")

class(data)

# Removing Unwanted columns

data<-data[ ,-17:-19]

# Checking that the unwanted columns are removed

names(data)

# Creating a new column to store the F score of a company and assigning it the value NA

data$F.score <- NA

# Assigning score to a0 by Analyzing the Net income column if the net income of a company is +VE then the value is 1 else its assigned as 0

a0 <- ifelse(data$Net.Income>0,a0<-1,

ifelse(data$Net.Income<0,a0<-0,NA\_character\_))

# Converting the a0 value to numeric from character type and assigning the value to variable - a

a<-as.numeric(a0)

# Assigning score to b0 by Analyzing the ROA column if the ROA of a company is +VE then the value is 1 else its assigned as 0

b0<- ifelse(data$ROA>0,b0<-1,

ifelse(data$ROA<0,b0<-0,NA\_character\_))

# Converting the b0 value to numeric from character type and assigning the value to variable - b

b<-as.numeric(b0)

# Assigning score to c0 by Analyzing the operating cash flow column if the operating cash flow of a company is +VE then the value is 1 else its assigned as 0

c0<-ifelse(data$operating.cash.flow.2020>0,c0<-1,

ifelse(data$operating.cash.flow.2020<0,c0<-0,NA\_character\_))

# Converting the c0 value to numeric from character type and assigning the value to variable - c

c<-as.numeric(c0)

# Replacing NA values with 1

c[which(is.na(c))]<-1

# Assigning score to d0 by comparing the operating cash flow column with net income if the operating cash flow of a company is greater then the value is 1 else its assigned as 0

d0<-ifelse(data$operating.cash.flow.2020>data$Net.Income,d0<-1,

ifelse(data$operating.cash.flow.2020<data$Net.Income,d0<-0,NA\_character\_))

# Converting the d0 value to numeric from character type and assigning the value to variable - d

d<-as.numeric(d0)

# Assigning score to e0 by comparing the Long term debt of 2020 column with Long term debt 2019 if the Long term debt 2020 of a company is less than its previous year than the value is 1 else its assigned as 0

e0<-ifelse(data$Long.term.debt.2020<data$Long.term.debt.2019,e0<-1,

ifelse(data$Long.term.debt.2020>data$Long.term.debt.2019,e0<-0,NA\_character\_))

# Converting the e0 value to numeric from character type and assigning the value to variable - e

e<-as.numeric(e0)

# Replacing NA values with 1

e[which(is.na(e))]<-1

# Assigning score to f0 by comparing the current ratio of 2020 column with current ratio of 2019 if the current ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0

f0<-ifelse(data$Current.Ratio.2020>data$Current.Ratio.2019,f0<-1,

ifelse(data$Current.Ratio.2020<data$Current.Ratio.2019,f0<-0,NA\_character\_))

# Converting the f0 value to numeric from character type and assigning the value to variable - f

f<-as.numeric(f0)

# Assigning score to g0 by comparing the Outstanding Shares of 2019 column with Outstanding Shares of 2020 if the Outstanding Shares of 2020 of a company is equal to its previous year than the value is 1 else its assigned as 0

g0<-ifelse(data$Outstanding.Shares.in.2019==data$Outstanding.Shares.in.2020,g0<-1,

ifelse(data$Outstanding.Shares.in.2019!=data$Outstanding.Shares.in.2020,g0<-0,NA\_character\_))

# Converting the g0 value to numeric from character type and assigning the value to variable - g

g<-as.numeric(g0)

# Assigning score to h0 by comparing the gross margin of 2020 column with gross margin of 2019 if the gross margin of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0

h0<-ifelse(data$Gross.margin.2020>data$Gross.margin.2019,h0<-1,

ifelse(data$Gross.margin.2020<data$Gross.margin.2019,h0<-0,NA\_character\_))

# Converting the h0 value to numeric from character type and assigning the value to variable - h

h<-as.numeric(h0)

# Assigning score to i0 by comparing the Asset Turnover Ratio of 2020 column with Asset Turnover Ratio of 2019 if the Asset Turnover Ratio of 2020 of a company is greater than its previous year than the value is 1 else its assigned as 0

i0<-ifelse(data$Asset.Turnover.Ratio.2020>data$Asset.Turnover.Ratio.2019,i0<-1,

ifelse(data$Asset.Turnover.Ratio.2020<data$Asset.Turnover.Ratio.2019,i0<-0,NA\_character\_))

# Converting the i0 value to numeric from character type and assigning the value to variable - i

i<-as.numeric(i0)

# Summing the values of a,b,c,d,e,f,g,h,i and populating the column F score

data$F.score <- a+b+c+d+e+f+g+h+i

data$F.score[which(is.na(data$F.score))]<-0

#Creating new Column that determines the status of every F score

for(i in 1:nrow(data)){

if(data$F.score[i]>=7){

data$status[i]<-"Financially Strong"

}else if (data$F.score[i]<=3){

data$status[i]<-"Financially weak"

}else {

data$status[i]<-"Financially Moderate"

}

}

#Removing Rows containing NA values

data1<-na.omit(data)

#Extracting Independent and dependent Variable

x<-data1[ ,c("Net.Income","ROA","operating.cash.flow.2020","Long.term.debt.2019","Long.term.debt.2020","Current.Ratio.2019","Current.Ratio.2020","Outstanding.Shares.in.2019","Outstanding.Shares.in.2020","Gross.margin.2019","Gross.margin.2020","Asset.Turnover.Ratio.2019","Asset.Turnover.Ratio.2020")]

y<-data1[ ,c("status")]

dataset<-data.frame(x,y)

dataset$Net.Income<-as.numeric(dataset$Net.Income)

dataset$Outstanding.Shares.in.2019<-as.numeric(dataset$Outstanding.Shares.in.2019)

dataset$Outstanding.Shares.in.2020<-as.numeric(dataset$Outstanding.Shares.in.2020)

dataset$y=factor(dataset$y)

sapply(dataset,class)

table(dataset$y)

set.seed(123)

split<-sample.split(dataset,SplitRatio=0.7)

split

train<-subset(dataset,split=="TRUE")

test<-subset(dataset,split=="FALSE")

set.seed(222)

rf<-randomForest(y~.,data = train,

ntree=200,

mtry=1,

importance=TRUE,

proximity=TRUE)

# Save model to RDS file

saveRDS(rf, "Webapp.rds")

***Code used to create the Web application***

# Import libraries

library(shiny)

library(data.table)

library(randomForest)

# Read in the RF model

model <- readRDS("Webapp.rds")

####################################

# User interface #

####################################

ui <- pageWithSidebar(

# Page header

headerPanel('Predicting Financial Strength of Companies using Machine Learning'),

# Input values

sidebarPanel(

#HTML("<h3>Input parameters</h3>"),

tags$label(

h3('Input parameters'),

div("All input parameters are mandatory\*", style = "color:red")),

numericInput("Net.Income",

label = "Net Income",

value = ""),

numericInput("ROA",

label = "Return on Assets",

value = ""),

numericInput("operating.cash.flow.2020",

label = "Operating cash flow 2020",

value = ""),

numericInput("Long.term.debt.2019",

label = "Long term debt 2019",

value = ""),

numericInput("Long.term.debt.2020",

label = "Long term debt 2020",

value = ""),

numericInput("Current.Ratio.2019",

label = "Current Ratio 2019",

value = ""),

numericInput("Current.Ratio.2020",

label = "Current Ratio 2020",

value = ""),

numericInput("Outstanding.Shares.in.2019",

label = "Outstanding Shares in 2019",

value = ""),

numericInput("Outstanding.Shares.in.2020",

label = "Outstanding Shares in 2020",

value = ""),

numericInput("Gross.margin.2019",

label = "Gross margin 2019",

value = ""),

numericInput("Gross.margin.2020",

label = "Gross margin 2020",

value = ""),

numericInput("Asset.Turnover.Ratio.2019",

label = "Asset Turnover Ratio 2019",

value = ""),

numericInput("Asset.Turnover.Ratio.2020",

label = "Asset Turnover Ratio 2020",

value = ""),

actionButton("submitbutton", "Predict",

class = "btn btn-primary")

),

mainPanel(

tags$label(h3('Status/Output')), # Status/Output Text Box

verbatimTextOutput('contents'),

h3(tableOutput('tabledata')), # Prediction results table

br(),

br(),

br(),

br(),

h3("Description:"),

h4("This Web Application predicts Financial strength of companies using machine learning. The financial status of the company can be acquired by inputting the data in input parameters panel and click on predict to obtain the results. "),

h4("By,"),

h4('Siddeshwar'),

h4('MBA Student | Dayananda Sagar University | Bengaluru'),

h4('Data Analyst Intern '),

h4("",

a("AmberTAG Analytics",

href = "https://www.ambertag.com/"))

)

)

####################################

# Server #

####################################

server<- function(input, output, session) {

# Input Data

datasetInput <- reactive({

df <- data.frame(

Name = c("Net Income",

"ROA",

"operating cash.flow 2020",

"Long term debt 2019",

"Long term debt 2020",

"Current Ratio 2019",

"Current Ratio 2020",

"Outstanding Shares in 2019",

"Outstanding Shares in 2020",

"Gross margin 2019",

"Gross margin 2020",

"Asset Turnover Ratio 2019",

"Asset Turnover Ratio 2020"),

Value = as.character(c(input$Net.Income,

input$ROA,

input$operating.cash.flow.2020,

input$Long.term.debt.2019,

input$Long.term.debt.2020,

input$Current.Ratio.2019,

input$Current.Ratio.2020,

input$Outstanding.Shares.in.2019,

input$Outstanding.Shares.in.2020,

input$Gross.margin.2019,

input$Gross.margin.2020,

input$Asset.Turnover.Ratio.2019,

input$Asset.Turnover.Ratio.2020)),

stringsAsFactors = FALSE)

y <- 0

df <- rbind(df, y)

input <- transpose(df)

write.table(input,"input.csv", sep=",", quote = FALSE, row.names = FALSE, col.names = FALSE)

test <- read.csv(paste("input", ".csv", sep=""), header = TRUE)

Output <- data.frame(Prediction=predict(model,test))

print(Output)

})

# Status/Output Text Box

output$contents <- renderPrint({

if (input$submitbutton>0) {

isolate("Prediction complete.")

} else {

return("Server is ready for Prediction.")

}

})

# Prediction results table

output$tabledata <- renderTable({

if (input$submitbutton>0) {

isolate(datasetInput())

}

})

}

####################################

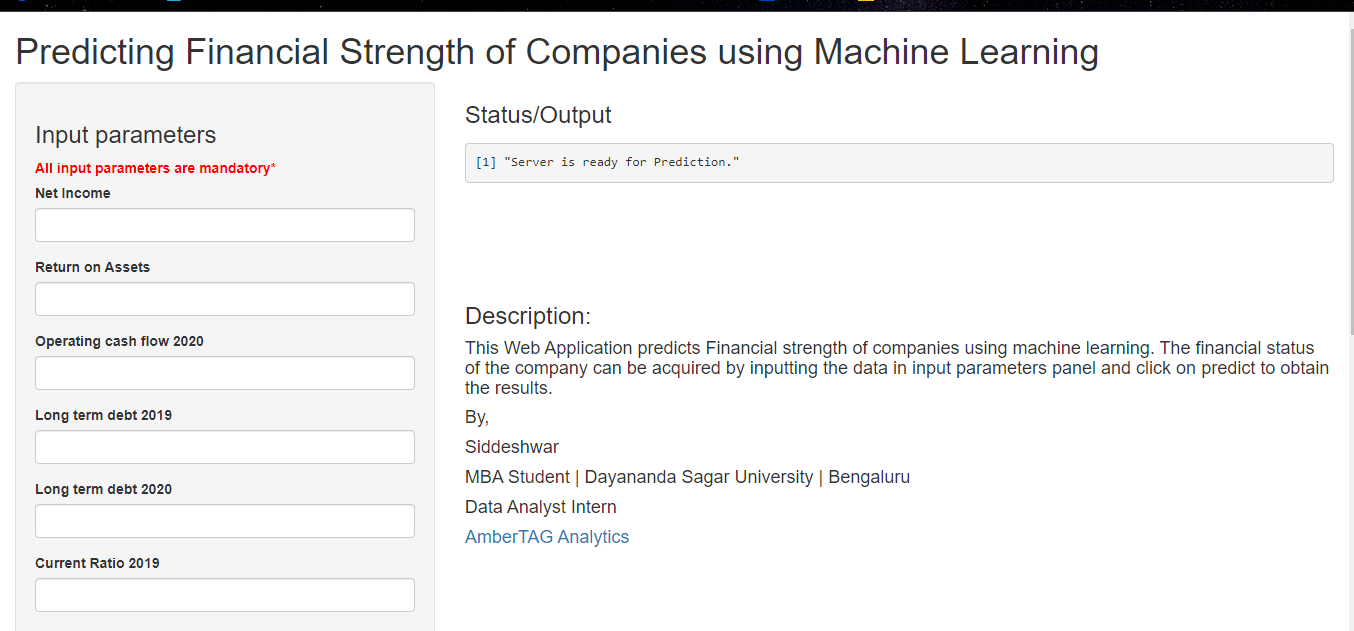
# Create the shiny app #

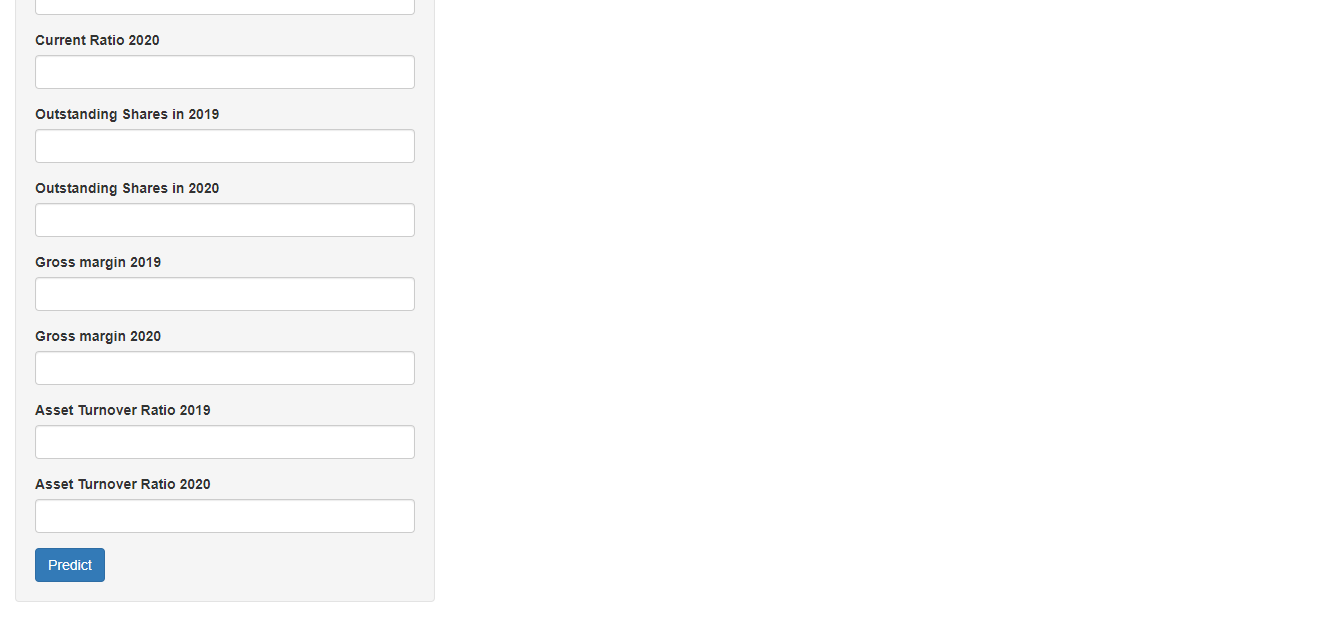
####################################

shinyApp(ui = ui, server = server)

Web Application Interface

***Below is the overall interface of the web application created***





***Web application link***

<https://project-at.shinyapps.io/Financial_strength/?_ga=2.60671661.1224010819.1647936926-1823690415.1647936926>