Practice and Assignment 7

Revision of Data frames, Generalized function for entropy
Decision Trees, CART

1. **DATA FRAMES REVISION**

 # DATA FRAMES REVISION

```
d = read.csv("ClassificationSimpleLab.csv")
d
d[2]
d[,2]
typeof(d[2])
typeof(d[,2])
d$age
d[2,]
d[2,2]
d[ncol(d)]
d[,ncol(d)]
names(d)
colnames(d)[1]
typeof(d[1])
class(d[1])
typeof(d[,1])
class(d[,1])

is.list(d)
is.vector(d)
is.data.frame(d)
is.list(d)
is.table(d)
is.matrix(d)
is.array(d)

is.vector(d[,1])
is.vector(d[1,])
is.vector(d$RID)
is.vector(d[1])
is.list(d[1])
l=list(c(1,2,3))
l
as.data.frame(l)
l= list(1,2,3)
l
as.data.frame(l)
```

2. GENERALIZED FUNCTION FOR COMPUTING INFORMATION GAIN

 # GENERALIZED FUNCTION FOR COMPUTING INFORMATION GAIN
 # ARNAB HAS GIVEN A SIMILAR CODE

 # INFORMATION GAIN THROUGH ENTROPY USED IN DECISION TREES

```
entropy <- function (x,y) {
 t<-table(x, y)
 H<-0
 for (i in 1:nrow(t))
 { prop<-t[i,]/sum(t[i,])
   Htmp <- -(prop[1]*log2(prop[1]))-(prop[2]*log2(prop[2]))
   HH<-ifelse(is.na(Htmp),0,Htmp)
   H<-H+(table(x)[i]/length(x))*HH
 }
 return(H)
}

gain  <- function(x,y) {
 e <- entropy(x,y)
 f <- table(y)
 nr<-f[1]+f[2]
 r <- -(f[1]/nr)*log2(f[1]/nr)-(f[2]/nr)*log2(f[2]/nr)
 eage<- r -e
 v = c(e,eage)
 return(v)
}

#Main Code below

{
d <- read.csv("ClassificationSimpleLab.csv")
dd = d
m=-Inf
ii=-1
for(i in 2:(ncol(d)-1))
{
dc = d[,i]
dclass = d[,ncol(d)]
print(colnames(d[i]))
v <- gain(dc, dclass)
if(v[2] > m)
 { m=v[2]
   ii=i
 }
print(paste("Entropy = ", unname(v[1]), "Gain =", unname(v[2])))
```

```
}
print(paste("Maximum gain = ", m , " for attribute = ", colnames(d[ii])))
}
```

3. **DECISION TREES C5.0**

#DECISION TREES C5.0

#GO STEP BY STEP AND INTERPRET WHAT IS HAPPENING

```
Install.packages(“caret”)
library(caret)

install.packages("C50")
library(C50)

install.packages("modeldata")
library(modeldata)

data(credit_data)

str(credit_data)

vars <- c("Home", "Seniority")
cc=credit_data[, c(vars, "Status")]
str(credit_data[, c(vars, "Status")])
str(cc)

sz=nrow(cc)*0.7
in_train <- sample(1:nrow(cc), size = sz)
train_data <- cc[ in_train,]
test_data  <- cc[-in_train,]

tree_mod <- C5.0(x = train_data[,vars], y = train_data$Status)

tree_mod
plot(tree_mod)
summary(tree_mod)

#What happens if rules=TRUE is used

tree_mod <- C5.0(x = train_data[,vars], y = train_data$Status, rules=TRUE)

tree_mod
plot(tree_mod)
summary(tree_mod)

predict.train <- predict(tree_mod, newdata=train_data, type="class")
```

```
strain = table(train_data$Status, predict.train, dnn = c("Actual", "Predicted"))
strain

predict.test <- predict(tree_mod, newdata=test_data, type="class")
s = table(test_data$Status, predict.test, dnn = c("Actual", "Predicted"))
s

accuracytrain <- sum(diag(strain))/sum(strain)
accuracytest <- sum(diag(s))/sum(s)
accuracytrain
accuracytest
```

4. **OUTPUT Sensitivity and Specificity of the classifier**

 # OUTPUT Sensitivity and Specificity of the classifier – code not given

5. **Do 3 & 4 for the data file "ClassificationSimplecases.csv"**

6. **Create a table to compare Accuracy, Sensitivity and Specificity for the Training set and Test set using various training partitions of 40%, 50%, 60%, 70% and 80% for the file "ClassificationSimplecases.csv"**