**[For Classification Dataset]**

**-------------------------------------------------------------------------------------------**

**Q2.1 Compare the performance of 10 machine learning models for given classification data set for the data partition of 70-30%.**

A2.1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Method | Package | Sensitivity | Specificity | Precision | Recall | Accuracy |
| M1 | adaboost | fastAdaboost | 0.642 | 0.398 | 0.539 | 0.642 | 52.57 |
| M2 | glmnet | glmnet | 0.702 | 0.345 | 0.533 | 0.702 | 52.93 |
| M3 | knn | class | 0.620 | 0.395 | 0.521 | 0.620 | 51.17 |
| M4 | obliqueRF | obliqueRF | 0.649 | 0.360 | 0.528 | 0.649 | 51.14 |
| M5 | ksvm | kernlab | 0.699 | 0.358 | 0.536 | 0.699 | 53.33 |
| M6 | multinom | car, nnet | 0.688 | 0.357 | 0.523 | 0.688 | 52.43 |
| M7 | pls | caret | 0.705 | 0.366 | 0.544 | 0.705 | 54.17 |
| M8 | rpart | rpart | 0.715 | 0.324 | 0.537 | 0.715 | 52.83 |
| M9 | nnet | nnet | 0.572 | 0.474 | 0.539 | 0.572 | 52.47 |
| M10 | randomForest | randomForest | 0.584 | 0.466 | 0.549 | 0.584 | 52.8 |

**CODES**

**adaboost**

library(ada)

library(hmeasure)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "ada"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ada(formula, trainDataset, control=rpart::rpart.control(maxdepth=30, cp=0.010000, minsplit=20,xval=10), iter=50)

Predicted <- (predict(model, testDataset))

PredictedProb <- predict(model, testDataset, type= "prob")[,2]

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

accuracy

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**glmnet**

library(hmeasure)

library(kernlab)

library(caret)

library(gmodels)

library(glmnet)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "glmnet"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

train\_labels = (train\_labels)

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- train(formula, trainDataset, method="glmnet")

Predicted <- round(as.numeric(predict(model, testDataset)))

PredictedProb <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**knn**

library(kernlab)

library(caret)

library(class)

library(gmodels)

iterations = 10

arr\_r=list()

arr\_mae=list()

arr\_R=list()

arr\_acc=list()

modelName <- "knn"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

dataset <- dataset[sample(nrow(dataset)),]

totalDataset <- nrow(dataset)

for(i in 0:iterations){

target <- names(dataset)[1]

inputs <- setdiff(names(dataset),target)

n=21

inputs <-sample(inputs,n)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

test\_labels <- dataset[(totalDataset \* training/100):totalDataset,c(target)]

model <- knn(train = trainDataset, test = testDataset, cl= train\_labels, k=50 )

CrossTable(x= test\_labels, y = model, prop.chisq = FALSE)

}

data <- cbind(arr\_r, arr\_R, arr\_acc, arr\_mae)

write.csv(data, file=paste("E:/101510028/",modelName,"-regression.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**obliqueRF**

library(obliqueRF)

library(hmeasure)

library(kernlab)

library(caret)

library(class)

library(gmodels)

iterations=3

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "orf"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

test\_labels <- dataset[(totalDataset \* training/100):totalDataset,c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- obliqueRF(data.matrix(trainDataset), data.matrix(train\_labels), mtry = 2, ntree = 100)

Predicted <- round(as.numeric(predict(model, testDataset)))

CrossTable(x= test\_labels, y = Predicted, prop.chisq = FALSE)

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**ksvm**

library(kernlab)

library(hmeasure)

iterations=15

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

arr\_nu = list()

arr\_eps = list()

modelName <- "ksvm"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

nu= runif(1)

epsilon= runif(1)

model <- ksvm(formula, trainDataset, nu= nu, epsilon= epsilon, kernel="rbfdot", prob.model=TRUE)

Predicted <- round(as.numeric(predict(model, testDataset)))

PredictedProb <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

arr\_eps[i] <- epsilon

arr\_nu[i] <- nu

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc, arr\_eps, arr\_nu)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**multinom**

library(nnet)

library(car)

library(hmeasure)

iterations=1

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "multinom"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- multinom(formula, trainDataset,trace=FALSE, maxit=1000)

Predicted <- round(as.numeric(predict(model, testDataset))-1)

PredictedProb <- predict(model, testDataset, type="probs")

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**pls**

library(hmeasure)

library(kernlab)

library(caret)

library(gmodels)

iterations=11

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "pls"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

train\_labels = as.factor(train\_labels)

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model<-train(formula,trainDataset,method="pls")

Predicted <- round(as.numeric(predict(model, testDataset)))

PredictedProb <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**rpart**

library(rpart)

library(hmeasure)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "decisiontree"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- rpart(formula, trainDataset, method="class", parms=list(split="information"), control=rpart.control(usesurrogate=0, maxsurrogate=0))

Predicted <- predict(model, testDataset, type="class")

PredictedProb <- predict(model, testDataset, type= "prob")[,1]

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

accuracy

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**nnet**

library(nnet)

library(hmeasure)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "nnet"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <-nnet(formula, trainDataset, size=10, linout=TRUE, skip=TRUE, MaxNWts=10000, trace=FALSE, maxit=100)

Predicted <- round(as.numeric(predict(model, testDataset)))

PredictedProb <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**randomForest**

library(randomForest)

library(hmeasure)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "rf"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- randomForest(trainDataset, train\_labels, ntree=500, mtry=2)

Predicted <- round(as.numeric(predict(model, testDataset)))

PredictedProb <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Q2.2 Ensemble the models from Table 2.1 for given classification data set on data partition of 70-30%.**

A2.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Combination of | Sensitivity | Specificity | Precision | Recall | Accuracy |
| E1 | M1, M5, M6, M7, M10 | 0.682 | 0.358 | 0.526 | 0.682 | 52.07 |
| E2 | M1, M2, M4 | 0.813 | 0.235 | 0.521 | 0.813 | 52.8 |
| E3 | M2, M4, M6, M8, M10 | 0.653 | 0.385 | 0.538 | 0.653 | 52.53 |
| E4 | M5, M7, M8 | 0.689 | 0.36 | 0.53 | 0.689 | 52.67 |
| E5 | M1, M2, M6, M8, M10 | 0.672 | 0.355 | 0.515 | 0.672 | 51.4 |

**CODES**

**M1, M5 ,M6, M7, M10**

library(randomForest)

library(hmeasure)

library(car)

library(kernlab)

library(caret)

library(ada)

iterations=10

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "combo1"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- randomForest(trainDataset, train\_labels, ntree=500, mtry=2)

Predicted1 <- round(as.numeric(predict(model, testDataset)))

PredictedProb1 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- multinom(formula, trainDataset,trace=FALSE, maxit=1000)

Predicted2 <- round(as.numeric(predict(model, testDataset))-1)

PredictedProb2 <- predict(model, testDataset, type="probs")

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ksvm(formula, trainDataset, nu= 0.414848076, epsilon= 0.263102617, kernel="rbfdot", prob.model=TRUE)

Predicted3 <- round(as.numeric(predict(model, testDataset)))

PredictedProb3 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

train\_labels = as.factor(train\_labels)

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model<-train(formula,trainDataset,method="pls")

Predicted4 <- round(as.numeric(predict(model, testDataset)))

PredictedProb4 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ada(formula, trainDataset, control=rpart::rpart.control(maxdepth=30, cp=0.010000, minsplit=20,xval=10), iter=50)

Predicted5 <- (predict(model, testDataset))

PredictedProb5 <- predict(model, testDataset, type= "prob")[,2]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

for(j in 1:length(Predicted1)){

a=0 #0's

b=0 #1's

if (Predicted1[j]==0){

a=a+1

}

if(Predicted1[j]==1){

b=b+1

}

if (Predicted2[j]==0){

a=a+1

}

if(Predicted2[j]==1){

b=b+1

}

if (Predicted3[j]==0){

a=a+1

}

if(Predicted3[j]==1){

b=b+1

}

if (Predicted4[j]==0){

a=a+1

}

if(Predicted4[j]==1){

b=b+1

}

if (Predicted5[j]==0){

a=a+1

}

if(Predicted5[j]==1){

b=b+1

}

if(a>b){

Predicted[j]=0

}

if(b>a){

Predicted[j]=1

}

}

PredictedProb = (PredictedProb1 + PredictedProb2 + PredictedProb3 + PredictedProb4 + PredictedProb5)/5

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**M1, M2, M4**

library(randomForest)

library(hmeasure)

library(car)

library(kernlab)

library(caret)

library(glmnet)

library(rpart)

library(obliqueRF)

iterations=1

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "combo2"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

test\_labels <- dataset[(totalDataset \* training/100):totalDataset,c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- obliqueRF(data.matrix(trainDataset), data.matrix(train\_labels), mtry = 2, ntree = 100)

Predicted1 <- round(as.numeric(predict(model, testDataset)))

PredictedProb1 <- predict(model, testDataset)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- train(formula, trainDataset, method="glmnet")

Predicted2 <- round(as.numeric(predict(model, testDataset)))

PredictedProb2 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ada(formula, trainDataset, control=rpart::rpart.control(maxdepth=30, cp=0.010000, minsplit=20,xval=10), iter=50)

Predicted3 <- (predict(model, testDataset))

PredictedProb3 <- predict(model, testDataset, type= "prob")[,2]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

for(j in 1:length(Predicted1)){

a=0 #0's

b=0 #1's

if (Predicted1[j]==0){

a=a+1

}

if(Predicted1[j]==1){

b=b+1

}

if (Predicted2[j]==0){

a=a+1

}

if(Predicted2[j]==1){

b=b+1

}

if (Predicted3[j]==0){

a=a+1

}

if(Predicted3[j]==1){

b=b+1

}

if (Predicted4[j]==0){

a=a+1

}

if(Predicted4[j]==1){

b=b+1

}

if (Predicted5[j]==0){

a=a+1

}

if(Predicted5[j]==1){

b=b+1

}

if(a>b){

Predicted[j]=0

}

if(b>a){

Predicted[j]=1

}

}

#PredictedProb = (PredictedProb1 + PredictedProb2 + PredictedProb3)/3

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

#Manually calculating spec, sens etc... from confusion matrix

tp= ConfusionMatrix[1,1]

tn= ConfusionMatrix[2,2]

fp= ConfusionMatrix[2,1]

fn= ConfusionMatrix[1,2]

sens[1]= tp/(tp+fn)

spec[1]= tn/(tn+fp)

prec[1]= tp/(tp+fp)

#------------------------------------------------------------------------------------------------------#

#EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

#sens[i] <- EvaluationsParameters$Sens

#spec[i] <- EvaluationsParameters$Spec

#prec[i] <- EvaluationsParameters$Precision

#reca[i] <- EvaluationsParameters$Recall

acc[1] <- accuracy

reca[1]= sens[1]

#EvaluationsParameters$Accuracy <- accuracy

#rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**M2, M4, M6, M8, M10**

library(randomForest)

library(hmeasure)

library(car)

library(kernlab)

library(caret)

library(glmnet)

library(rpart)

library(obliqueRF)

iterations=1

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "combo3"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- randomForest(trainDataset, train\_labels, ntree=500, mtry=2)

Predicted1 <- round(as.numeric(predict(model, testDataset)))

PredictedProb1 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- multinom(formula, trainDataset,trace=FALSE, maxit=1000)

Predicted2 <- round(as.numeric(predict(model, testDataset))-1)

PredictedProb2 <- predict(model, testDataset, type="probs")

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- train(formula, trainDataset, method="glmnet")

Predicted3 <- round(as.numeric(predict(model, testDataset)))

PredictedProb3 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- rpart(formula, trainDataset, method="class", parms=list(split="information"), control=rpart.control(usesurrogate=0, maxsurrogate=0))

Predicted4 <- predict(model, testDataset, type="class")

PredictedProb4 <- predict(model, testDataset, type= "prob")[,1]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

test\_labels <- dataset[(totalDataset \* training/100):totalDataset,c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- obliqueRF(data.matrix(trainDataset), data.matrix(train\_labels), mtry = 2, ntree = 100)

Predicted5 <- round(as.numeric(predict(model, testDataset)))

PredictedProb5 <- predict(model, testDataset)

#-----------------------------------------------------------------------------------------------------#

for(j in 1:length(Predicted1)){

a=0 #0's

b=0 #1's

if (Predicted1[j]==0){

a=a+1

}

if(Predicted1[j]==1){

b=b+1

}

if (Predicted2[j]==0){

a=a+1

}

if(Predicted2[j]==1){

b=b+1

}

if (Predicted3[j]==0){

a=a+1

}

if(Predicted3[j]==1){

b=b+1

}

if (Predicted4[j]==0){

a=a+1

}

if(Predicted4[j]==1){

b=b+1

}

if (Predicted5[j]==0){

a=a+1

}

if(Predicted5[j]==1){

b=b+1

}

if(a>b){

Predicted[j]=0

}

if(b>a){

Predicted[j]=1

}

}

#PredictedProb = (PredictedProb1 + PredictedProb2 + PredictedProb3 + PredictedProb4 + PredictedProb5)/5

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

#Manually calculating spec, sens etc... from confusion matrix

tp= ConfusionMatrix[1,1]

tn= ConfusionMatrix[2,2]

fp= ConfusionMatrix[2,1]

fn= ConfusionMatrix[1,2]

sens[1]= tp/(tp+fn)

spec[1]= tn/(tn+fp)

prec[1]= tp/(tp+fp)

#------------------------------------------------------------------------------------------------------#

#EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

#sens[i] <- EvaluationsParameters$Sens

#spec[i] <- EvaluationsParameters$Spec

#prec[i] <- EvaluationsParameters$Precision

#reca[i] <- EvaluationsParameters$Recall

acc[1] <- accuracy

reca[1]= sens[1]

#EvaluationsParameters$Accuracy <- accuracy

#rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**M5, M7, M8**

library(randomForest)

library(hmeasure)

library(car)

library(kernlab)

library(caret)

library(glmnet)

library(rpart)

library(obliqueRF)

iterations=1

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "combo4"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

train\_labels = as.factor(train\_labels)

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model<-train(formula,trainDataset,method="pls")

Predicted1 <- round(as.numeric(predict(model, testDataset)))

PredictedProb1 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- rpart(formula, trainDataset, method="class", parms=list(split="information"), control=rpart.control(usesurrogate=0, maxsurrogate=0))

Predicted2 <- predict(model, testDataset, type="class")

PredictedProb2 <- predict(model, testDataset, type= "prob")[,1]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ksvm(formula, trainDataset, nu= 0.414848076, epsilon= 0.263102617, kernel="rbfdot", prob.model=TRUE)

Predicted3 <- round(as.numeric(predict(model, testDataset)))

PredictedProb3 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

for(j in 1:length(Predicted1)){

a=0 #0's

b=0 #1's

if (Predicted1[j]==0){

a=a+1

}

if(Predicted1[j]==1){

b=b+1

}

if (Predicted2[j]==0){

a=a+1

}

if(Predicted2[j]==1){

b=b+1

}

if (Predicted3[j]==0){

a=a+1

}

if(Predicted3[j]==1){

b=b+1

}

if (Predicted4[j]==0){

a=a+1

}

if(Predicted4[j]==1){

b=b+1

}

if (Predicted5[j]==0){

a=a+1

}

if(Predicted5[j]==1){

b=b+1

}

if(a>b){

Predicted[j]=0

}

if(b>a){

Predicted[j]=1

}

}

PredictedProb = (PredictedProb1 + PredictedProb2 + PredictedProb3)/3

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**M1,M2, M6, M8, M10**

library(randomForest)

library(hmeasure)

library(car)

library(kernlab)

library(caret)

library(glmnet)

library(rpart)

iterations=1

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

modelName <- "combo5"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- setdiff(names(dataset),target)

n <- length(inputs)

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels <- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- randomForest(trainDataset, train\_labels, ntree=500, mtry=2)

Predicted1 <- round(as.numeric(predict(model, testDataset)))

PredictedProb1 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- multinom(formula, trainDataset,trace=FALSE, maxit=1000)

Predicted2 <- round(as.numeric(predict(model, testDataset))-1)

PredictedProb2 <- predict(model, testDataset, type="probs")

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

train\_labels<- dataset[1:(totalDataset \* training/100),c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- train(formula, trainDataset, method="glmnet")

Predicted3 <- round(as.numeric(predict(model, testDataset)))

PredictedProb3 <- predict(model, testDataset)

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- rpart(formula, trainDataset, method="class", parms=list(split="information"), control=rpart.control(usesurrogate=0, maxsurrogate=0))

Predicted4 <- predict(model, testDataset, type="class")

PredictedProb4 <- predict(model, testDataset, type= "prob")[,1]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- ada(formula, trainDataset, control=rpart::rpart.control(maxdepth=30, cp=0.010000, minsplit=20,xval=10), iter=50)

Predicted5 <- (predict(model, testDataset))

PredictedProb5 <- predict(model, testDataset, type= "prob")[,2]

Actual <- as.double(unlist(testDataset[target]))

#-----------------------------------------------------------------------------------------------------#

for(j in 1:length(Predicted1)){

a=0 #0's

b=0 #1's

if (Predicted1[j]==0){

a=a+1

}

if(Predicted1[j]==1){

b=b+1

}

if (Predicted2[j]==0){

a=a+1

}

if(Predicted2[j]==1){

b=b+1

}

if (Predicted3[j]==0){

a=a+1

}

if(Predicted3[j]==1){

b=b+1

}

if (Predicted4[j]==0){

a=a+1

}

if(Predicted4[j]==1){

b=b+1

}

if (Predicted5[j]==0){

a=a+1

}

if(Predicted5[j]==1){

b=b+1

}

if(a>b){

Predicted[j]=0

}

if(b>a){

Predicted[j]=1

}

}

PredictedProb = (PredictedProb1 + PredictedProb2 + PredictedProb3 + PredictedProb4 + PredictedProb5)/5

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Q2.3 Study 5 feature selection techniques for given classification data set and report Top five features.**

**[Hint: Use FSelecter, Boruta, etc]**

A2.3

|  |  |
| --- | --- |
| Feature Selection Technique | Top 5 Features |
| Random Approach | F18, F10, F6, F15, F1 |
| Boruta | F17, F20, F14, F4, F10 |
| SES | F17, F20, F18, F19, F17 |
| MMPC | F17, F20, F18, F19, F17 |
| randomVarImpsRF | F4, F1, F9, F18, F10 |

**CODES**

**Random Approach (using decision tree)**

library(rpart)

library(hmeasure)

iterations=100

sens = list()

spec = list()

reca = list()

prec = list()

acc = list()

para= list()

modelName <- "decisiontree"

InputDataFileName="E:/101510028/classificationDataSet.csv"

training = 70

dataset <- read.csv(InputDataFileName)

totalDataset <- nrow(dataset)

target <- names(dataset)[1]

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

for (i in 1:iterations){

dataset <- dataset[sample(nrow(dataset)),]

inputs <- sample(names(dataset[,2:21]), 5)

inputs

n <- length(inputs)

trainDataset <- dataset[1:(totalDataset \* training/100),c(inputs, target)]

testDataset <- dataset[(totalDataset \* training/100):totalDataset,c(inputs, target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

model <- rpart(formula, trainDataset, method="class", parms=list(split="information"), control=rpart.control(usesurrogate=0, maxsurrogate=0))

Predicted <- predict(model, testDataset, type="class")

PredictedProb <- predict(model, testDataset, type= "prob")[,1]

Actual <- as.double(unlist(testDataset[target]))

ConfusionMatrix <- misclassCounts(Predicted,Actual)$conf.matrix

EvaluationsParameters <- round(HMeasure(Actual,PredictedProb)$metrics,3)

accuracy <- round(mean(Actual==Predicted) \*100,2)

accuracy

sens[i] <- EvaluationsParameters$Sens

spec[i] <- EvaluationsParameters$Spec

prec[i] <- EvaluationsParameters$Precision

reca[i] <- EvaluationsParameters$Recall

acc[i] <- accuracy

para[i] <- list(inputs)

EvaluationsParameters$Accuracy <- accuracy

rownames(EvaluationsParameters[i])=modelName

}

data <- cbind(sens, spec, prec, reca, acc, para)

write.csv(data, file=paste("E:/101510028/",modelName,"-classification.csv",sep=''), row.names=TRUE)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Boruta + SES + MMPC + randomVarImpsRF**

library(randomForest)

library(varSelRF)

library(Boruta)

library(MXM)

InputDataFileName="E:/101510028/classificationDataSet.csv"

data <- read.csv(InputDataFileName)

data <- data[sample(nrow(data)),]

target <- names(dataset)[1]

inputs <- setdiff(names(data),target)

trainDataset <- data[,c(inputs)]

train\_labels <- data[,c(target)]

formula <- as.formula(paste(target, "~", paste(c(inputs), collapse = "+")))

forest <- randomForest(trainDataset, train\_labels, ntree = 200, importance = TRUE)

model = randomVarImpsRF(trainDataset, train\_labels,forest, numrandom = 1,usingCluster = FALSE)

write.csv(model, file=paste("E:/101510028/","randomVArImp.csv",sep=''), row.names=TRUE)

Boruta.Short <- Boruta(train\_labels~ ., data = trainDataset , maxRuns = 12)

x=attStats(Boruta.Short)

write.csv(x, file=paste("E:/101510028/","boruta.csv",sep=''), row.names=TRUE)

x=SES(train\_labels, trainDataset, max\_k = 3, threshold = 0.05, test = NULL, ini = NULL,wei = NULL, user\_test = NULL, hash = FALSE, hashObject = NULL, robust = FALSE,ncores = 2, logged = FALSE)

summary(x)

x=MMPC(train\_labels, trainDataset, max\_k = 3, threshold = 0.05, test = NULL, ini = NULL,wei = NULL, user\_test = NULL, hash = FALSE, hashObject = NULL, robust = FALSE,ncores = 1, backward = FALSE, logged = FALSE)

summary(x)

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------