Machhine Learning | HW3 | Rohan Thorat | (201)-744-4816

Comparison between LDA algorithm, KNN algorithm and Logistic Regression over Breast Cancer Wisconsin Dataset to classify Malignant vs. Benign breast tumors.

Inference for the Project
Model Accuracy for Test data
Ranking based on Accuracy for LDA, KNN and Logistic Regression

1) **LDA - 97.18%**
2) **KNN - 97.05%**
3) **Logistic Regression - 91.37%**

Naming the variables
Id Number – V1
Diagnosis – V2

Diving Deep into the analysis for LDA

R Code for LDA

```
#Inputting raw data
Wdbc3<-read.csv("/Users/rohan/Desktop/DS630_MachineLearning/HW3/wdbc.data.txt")
#View(wdbc3)

#Naming raw data
names(wdbc3)<-c("V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11","V12","V13","V14",
        "V15","V16","V17","V18","V19","V20","V21","V22","V23","V24","V25","V26","V27",
        "V28","V29","V30","V31","V32")
#View(wdbc3)

#removing id number
wdbc3 <- wdbc3[,c(-1)]
summary(wdbc3)

#splitting data
train <- wdbc3[1:426, ]
test <- wdbc3[427:568, ]

#using MASS library for LDA
library(MASS)
wdbclda <- lda(V2~., data = train)
summary(wdbclda)

#running prediction on training data
pred_train <- predict(wdbclda, train, type= "response")
ls(pred_train)
pred_train$class
table(train$V2,pred_train$class)
```

#creating confusionMatrix for training data
library(caret)
confusionMatrix(table(train$V2,pred_train$class), positive = "M")

	B	M
B	249	1
M	12	164

#Accuracy for training data
Accuracy : 0.9695

#running prediction on testing data
pred_test <- predict(wdbclda, test, type = "response")
table(test$V2,pred_test$class)

#creating confusionMatrix() for test data
confusionMatrix(table(test$V2,pred_test$class), positive = "M")

	B	M
B	106	1
M	3	32

#Accuracy for testing data
Accuracy : 0.9718

Diving Deep into the analysis for KNN

#Inputting raw data

```
wdbc2<-read.csv("/Users/rohan/Desktop/DS630_MachineLearning/HW3/wdbc.data.txt")
#View(wdbc2)
```

#Naming raw data

```
names(wdbc2)<-c("V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11","V12","V13","V14",
            "V15","V16","V17","V18","V19","V20","V21","V22","V23","V24","V25","V26","V27",
            "V28","V29","V30","V31","V32")
#View(wdbc2)
```

#removing id number

```
wdbc2 <- wdbc2[,c(-1)]
summary(wdbc2)
```

#Scaling Data

```
wdbc2Normalized <- as.data.frame(scale(wdbc2[-1]))
```

#splitting data

```
train <- wdbc2Normalized[1:426, ]
test <- wdbc2Normalized[427:568, ]
trainLabels <- wdbc2[1:426, 1]
testLabels <- wdbc2[427:568, 1]
```

#running CLASS package for KNN

```
library(class)
```

#taking k approximately equal to square root of the number of rows in the training data

```
k <- 20
wdbcknn <- knn(train = train,test = test,cl = trainLabels,k)

ActualVsPredicted <- cbind(testLabels, wdbcknn)
colnames(ActualVsPredicted) <- c('actual', 'predicted')
percentage <- sum(apply(ActualVsPredicted, 1,
                  function(row) { ifelse(row[1] == row[2], 1, 0) }
)) / dim(ActualVsPredicted)[1]
```

#running gmodels library for crosstable function

library (gmodels)

CrossTable(x = testLabels, y = wdbcknn, prop.chisq = F, dnn = c('actual', 'predicted'))

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  142


             | predicted
      actual |         B |         M | Row Total |
-------------|-----------|-----------|-----------|
           B |       107 |         0 |       107 |
             |     1.000 |     0.000 |     0.754 |
             |     0.982 |     0.000 |           |
             |     0.754 |     0.000 |           |
-------------|-----------|-----------|-----------|
           M |         2 |        33 |        35 |
             |     0.057 |     0.943 |     0.246 |
             |     0.018 |     1.000 |           |
             |     0.014 |     0.232 |           |
-------------|-----------|-----------|-----------|
Column Total |       109 |        33 |       142 |
             |     0.768 |     0.232 |           |
-------------|-----------|-----------|-----------|
```

> #correctly detect as being malignant

> recall=33/(33+2)

> recall

[1] 0.9428571

> #predict benign tumors although they are malignant

> score=(2((1*0.9428)/(1+0.9428)))*

> score

[1] 0.970558

Diving Deep into the analysis for Logistic Regression

#Inputting raw data

```
wdbc1<-read.csv("/Users/rohan/Desktop/DS630_MachineLearning/HW3/wdbc.data.txt")
#View(wdbc1)
```

#Naming raw data

```
names(wdbc1)<-c("V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11","V12","V13","V14",
        "V15","V16","V17","V18","V19","V20","V21","V22","V23","V24","V25","V26","V27",
        "V28","V29","V30","V31","V32")
#View(wdbc)
```

#Removing id number

```
wdbc1 <- wdbc1[,c(-1)]
summary(wdbc1)
```

#Removing outliers after comparing the histograms for all the variables

```
wdbc1<-subset(wdbc1,V4<35 & V7<0.15 & V13<1.5 & V12<0.09 & V14<3 & V17<0.20 & V20<0.045)
summary(wdbc1)
```

#Checking for NA values

```
complete.cases(wdbc1)
```

#Random Sampling of data as 75% Train and 25% Test

```
indexes = sample(1:nrow(wdbc1), size=0.75*nrow(wdbc1))
```

#Splitting data

```
train = wdbc1[indexes,]
dim(train)
test = wdbc1[-indexes,]
dim(test)
```

#Running the GLM model for logistic regression with the significant values alone

```
model11<-glm(V2~V4+V7+V12+V13+V14+V17+V20,data=train,family = binomial)
summary(model11)
```

#Running prediction on training data

```
pred_train<-predict(model11,train, type = "response")
train$result<-ifelse(pred_train >0.5, "M","B")
cv<-table(train$result,train$V2)
cv
```

	B	M
B	265	17
M	10	125

#Accuracy for training data

Accuracy : 0.9353

#Creating confusionMatrix() for test data

```
confusionMatrix(cv,positive = "M")
```

#Running prediction on testing data

```
pred_test <- predict(model11,test,type="response")
test$result<-ifelse(pred_test > 0.5 ,"M","B")
cv_1<-table(test$result,test$V2)
cv_1
```

	B	M
B	72	16
M	4	47

#Accuracy for test data

Accuracy : 0.8561

#Creating confusionMatrix() for test data

```
confusionMatrix(cv_1,positive = "M")
```

#ROCR curve

```
#install.packages("ROCR")
library(ROCR)
ROCRpred<-prediction(pred_test, test$V2)
ROCRperf<-performance(ROCRpred,'tpr','fpr')
plot(ROCRperf,colorize = TRUE, text.adj = c(-0.2,1.7))
```