FinalProject

Stephen Yu

2023-12-06

Reading Data and Cleaning

```
predict <- read.csv("X_pca.csv", header = FALSE)
response <- read.csv("y_sentimentonly.csv")[, -1]
factoredresponse <- as.integer(factor(response)) - 1
moviedata <- cbind(predict, factoredresponse)
```

Modelling

```
library(mclust)
```

```
## Package 'mclust' version 6.0.0
## Type 'citation("mclust")' for citing this R package in publications.
```

```
library(class)
library(MASS)
set.seed(100)

# K-Fold CV
# k <- floor(sqrt(nrow(moviedata)))
k <- 5
size <- nrow(moviedata) / k

# Initialize Results
resultLR <- matrix(0, nrow = 3, ncol = 3)
rownames(resultLR) <-  c("False", "True", "Accuracy")
colnames(resultLR) <- c("Precision", "Recall", "F1-Score")

resultKNN <- matrix(0, nrow = 3, ncol = 3)
rownames(resultKNN) <-  c("False", "True", "Accuracy")
colnames(resultKNN) <- c("Precision", "Recall", "F1-Score")

resultLDA <- matrix(0, nrow = 3, ncol = 3)
rownames(resultLDA) <-  c("False", "True", "Accuracy")
colnames(resultLDA) <- c("Precision", "Recall", "F1-Score")

resultQDA <- matrix(0, nrow = 3, ncol = 3)
rownames(resultQDA) <-  c("False", "True", "Accuracy")
colnames(resultQDA) <- c("Precision", "Recall", "F1-Score")

my.thres <- .5
```

```r
for(i in 0:(k - 1)){

  # Train-Validation Split
  i.test <- seq((round(i * size) + 1), round((i + 1) * size))
  sa.train <- moviedata[-i.test, ]
  sa.test <- moviedata[i.test, ]

  # Logistic Regression
  ml <- glm(factoredresponse ~ ., data = sa.train, family = "binomial")
  pred.prob.test <- predict(ml, sa.test[, -ncol(sa.test)], type = "response")
  predicted <- pred.prob.test > my.thres
  tableLog <- table("Reference" = sa.test[ ,ncol(sa.test)], "Predicted" = predicted)
  resultLR[1, 1] <- resultLR[1, 1] + ((tableLog[1, 1] / (tableLog[1, 1] + tableLog[2, 1])) / k) # False
  resultLR[2, 1] <- resultLR[2, 1] + ((tableLog[2, 2] / (tableLog[2, 2] + tableLog[1, 2])) / k) # True
  resultLR[1, 2] <- resultLR[1, 2] + ((tableLog[1, 1] / (tableLog[1, 1] + tableLog[1, 2])) / k) # False
  resultLR[2, 2] <- resultLR[2, 2] + ((tableLog[2, 2] / (tableLog[2, 2] + tableLog[2, 1])) / k) # True
  resultLR[3, 3] <- resultLR[3, 3] + ((tableLog[1, 1] + tableLog[2, 2]) / nrow(sa.test) / k) # Accuracy

  # KNN
  m.knn <- knn(sa.train[, -ncol(sa.train)], sa.test[, -ncol(sa.test)], sa.train[, ncol(sa.train)], k =
  table.knn <- table("Reference" = sa.test[, ncol(sa.test)], "Predicted" = m.knn)
  resultKNN[1, 1] <- resultKNN[1, 1] + ((table.knn[1, 1] / (table.knn[1, 1] + table.knn[2, 1])) / k) #
  resultKNN[2, 1] <- resultKNN[2, 1] + ((table.knn[2, 2] / (table.knn[2, 2] + table.knn[1, 2])) / k) #
  resultKNN[1, 2] <- resultKNN[1, 2] + ((table.knn[1, 1] / (table.knn[1, 1] + table.knn[1, 2])) / k) #
  resultKNN[2, 2] <- resultKNN[2, 2] + ((table.knn[2, 2] / (table.knn[2, 2] + table.knn[2, 1])) / k) #
  resultKNN[3, 3] <- resultKNN[3, 3] + ((table.knn[1, 1] + table.knn[2, 2]) / nrow(sa.test) / k) # Accu

  # LDA
  lda.mod <- lda(factoredresponse ~ ., data = sa.train)
  pred.lda.test <- predict(lda.mod, sa.test[, -ncol(sa.test)])
  tableLDA <- table("Reference" = sa.test[, ncol(sa.test)], "Predicted" = pred.lda.test$class)
  resultLDA[1, 1] <- resultLDA[1, 1] + ((tableLDA[1, 1] / (tableLDA[1, 1] + tableLDA[2, 1])) / k) # Fal
  resultLDA[2, 1] <- resultLDA[2, 1] + ((tableLDA[2, 2] / (tableLDA[2, 2] + tableLDA[1, 2])) / k) # Tru
  resultLDA[1, 2] <- resultLDA[1, 2] + ((tableLDA[1, 1] / (tableLDA[1, 1] + tableLDA[1, 2])) / k) # Fal
  resultLDA[2, 2] <- resultLDA[2, 2] + ((tableLDA[2, 2] / (tableLDA[2, 2] + tableLDA[2, 1])) / k) # Tru
  resultLDA[3, 3] <- resultLDA[3, 3] + ((tableLDA[1, 1] + tableLDA[2, 2]) / nrow(sa.test) / k) # Accura

  # QDA
  qda.mod <- qda(factoredresponse ~ ., data = sa.train)
  pred.qda.test <- predict(qda.mod, sa.test[, -ncol(sa.test)])
  tableQDA <- table("Reference" = sa.test[, ncol(sa.test)], "Predicted" = pred.qda.test$class)
  resultQDA[1, 1] <- resultQDA[1, 1] + ((tableQDA[1, 1] / (tableQDA[1, 1] + tableQDA[2, 1])) / k) # Fal
  resultQDA[2, 1] <- resultQDA[2, 1] + ((tableQDA[2, 2] / (tableQDA[2, 2] + tableQDA[1, 2])) / k) # Tru
  resultQDA[1, 2] <- resultQDA[1, 2] + ((tableQDA[1, 1] / (tableQDA[1, 1] + tableQDA[1, 2])) / k) # Fal
  resultQDA[2, 2] <- resultQDA[2, 2] + ((tableQDA[2, 2] / (tableQDA[2, 2] + tableQDA[2, 1])) / k) # Tru
  resultQDA[3, 3] <- resultQDA[3, 3] + ((tableQDA[1, 1] + tableQDA[2, 2]) / nrow(sa.test) / k) # Accura

}

# F-1 Score LR
resultLR[1, 3] <- (2 * resultLR[1, 1] * resultLR[1, 2]) / (resultLR[1, 1] + resultLR[1, 2])
resultLR[2, 3] <- (2 * resultLR[2, 1] * resultLR[2, 2]) / (resultLR[2, 1] + resultLR[2, 2])
resultLR[3, c(1, 2)] <- NA
```

```r
resultLR
```

```
##           Precision    Recall  F1-Score
## False     0.8563877 0.8269763 0.8414251
## True      0.8326881 0.8613968 0.8467992
## Accuracy         NA        NA 0.8441600
```

```r
# F-1 Score KNN
resultKNN[1, 3] <- (2 * resultKNN[1, 1] * resultKNN[1, 2]) / (resultKNN[1, 1] + resultKNN[1, 2])
resultKNN[2, 3] <- (2 * resultKNN[2, 1] * resultKNN[2, 2]) / (resultKNN[2, 1] + resultKNN[2, 2])
resultKNN[3, c(1, 2)] <- NA
resultKNN
```

```
##           Precision    Recall  F1-Score
## False     0.7847922 0.7615025 0.7729720
## True      0.7683595 0.7912213 0.7796229
## Accuracy         NA        NA 0.7763200
```

```r
# F-1 Score LDA
resultLDA[1, 3] <- (2 * resultLDA[1, 1] * resultLDA[1, 2]) / (resultLDA[1, 1] + resultLDA[1, 2])
resultLDA[2, 3] <- (2 * resultLDA[2, 1] * resultLDA[2, 2]) / (resultLDA[2, 1] + resultLDA[2, 2])
resultLDA[3, c(1, 2)] <- NA
resultLDA
```

```
##           Precision   Recall  F1-Score
## False     0.8615304 0.805047 0.8323316
## True      0.8169835 0.870676 0.8429757
## Accuracy         NA       NA 0.8378200
```

```r
# F-1 Score QDA
resultQDA[1, 3] <- (2 * resultQDA[1, 1] * resultQDA[1, 2]) / (resultQDA[1, 1] + resultQDA[1, 2])
resultQDA[2, 3] <- (2 * resultQDA[2, 1] * resultQDA[2, 2]) / (resultQDA[2, 1] + resultQDA[2, 2])
resultQDA[3, c(1, 2)] <- NA
resultQDA
```

```
##           Precision    Recall  F1-Score
## False     0.7976051 0.7555377 0.7760017
## True      0.7678302 0.8083555 0.7875719
## Accuracy         NA        NA 0.7819600
```

Model Comparison

```r
# Accuracy
accuracy <- matrix(c(resultLR[3, 3], resultKNN[3, 3], resultLDA[3, 3], resultQDA[3, 3]), ncol = 1)
rownames(accuracy) <- c("LR", "KNN", "LDA", "QDA")
colnames(accuracy) <- "Accuracy"
accuracy
```

```
##     Accuracy
## LR   0.84416
## KNN  0.77632
## LDA  0.83782
## QDA  0.78196
```

```r
# True Precision
trueprecision <- matrix(c(resultLR[2, 1], resultKNN[2, 1], resultLDA[2, 1], resultQDA[2, 1],
                          resultLR[2, 2], resultKNN[2, 2], resultLDA[2, 2], resultQDA[2, 2],
```

```
                          resultLR[2, 3], resultKNN[2, 3], resultLDA[2, 3], resultQDA[2, 3]), ncol = 3)
rownames(trueprecision) <- c("LR", "KNN", "LDA", "QDA")
colnames(trueprecision) <- c("T Precision", "T Recall", "T F1-Score")
trueprecision
```

```
##     T Precision  T Recall T F1-Score
## LR    0.8326881 0.8613968  0.8467992
## KNN   0.7683595 0.7912213  0.7796229
## LDA   0.8169835 0.8706760  0.8429757
## QDA   0.7678302 0.8083555  0.7875719
```

```
# False Precision
falseprecision <- matrix(c(resultLR[1, 1], resultKNN[1, 1], resultLDA[1, 1], resultQDA[1, 1],
                          resultLR[1, 2], resultKNN[1, 2], resultLDA[1, 2], resultQDA[1, 2],
                          resultLR[1, 3], resultKNN[1, 3], resultLDA[1, 3], resultQDA[1, 3]), ncol = 3)
rownames(falseprecision) <- c("LR", "KNN", "LDA", "QDA")
colnames(falseprecision) <- c("F Precision", "F Recall", "F F1-Score")
falseprecision
```

```
##     F Precision  F Recall F F1-Score
## LR    0.8563877 0.8269763  0.8414251
## KNN   0.7847922 0.7615025  0.7729720
## LDA   0.8615304 0.8050470  0.8323316
## QDA   0.7976051 0.7555377  0.7760017
```

ROC Curves

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

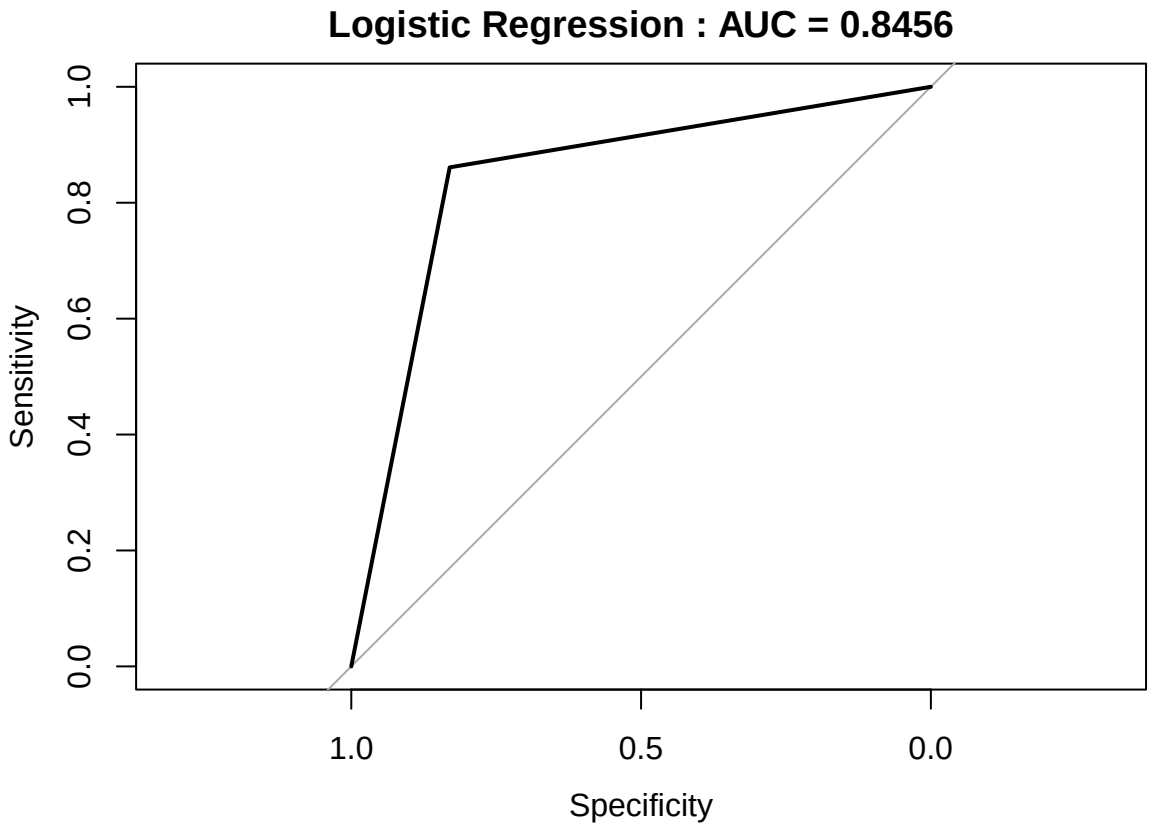
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
# Logistic Regression
roc_score <- roc(sa.test[, ncol(sa.test)], as.numeric(predicted))
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

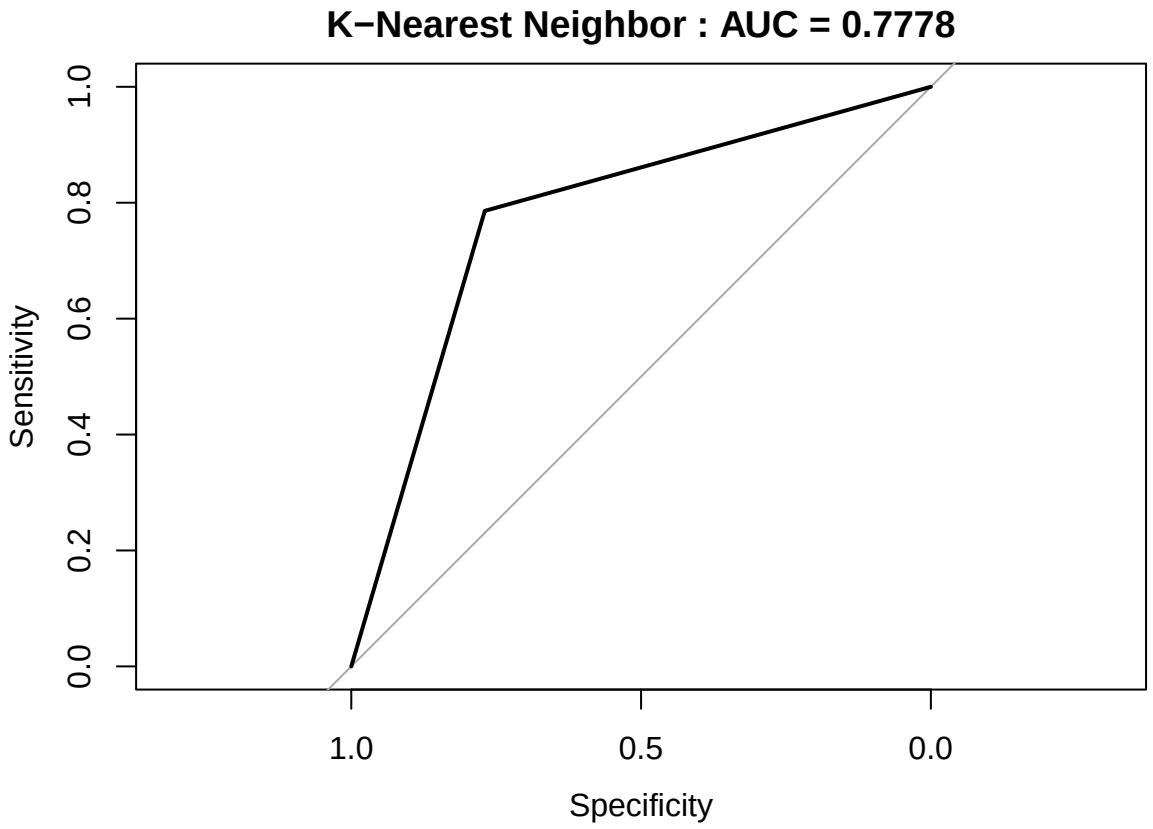
```
plot(roc_score, main = paste0("Logistic Regression : AUC = ", round(roc_score$auc, 4)))
```




```
# KNN
roc_score <- roc(sa.test[, ncol(sa.test)], as.numeric(m.knn))
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

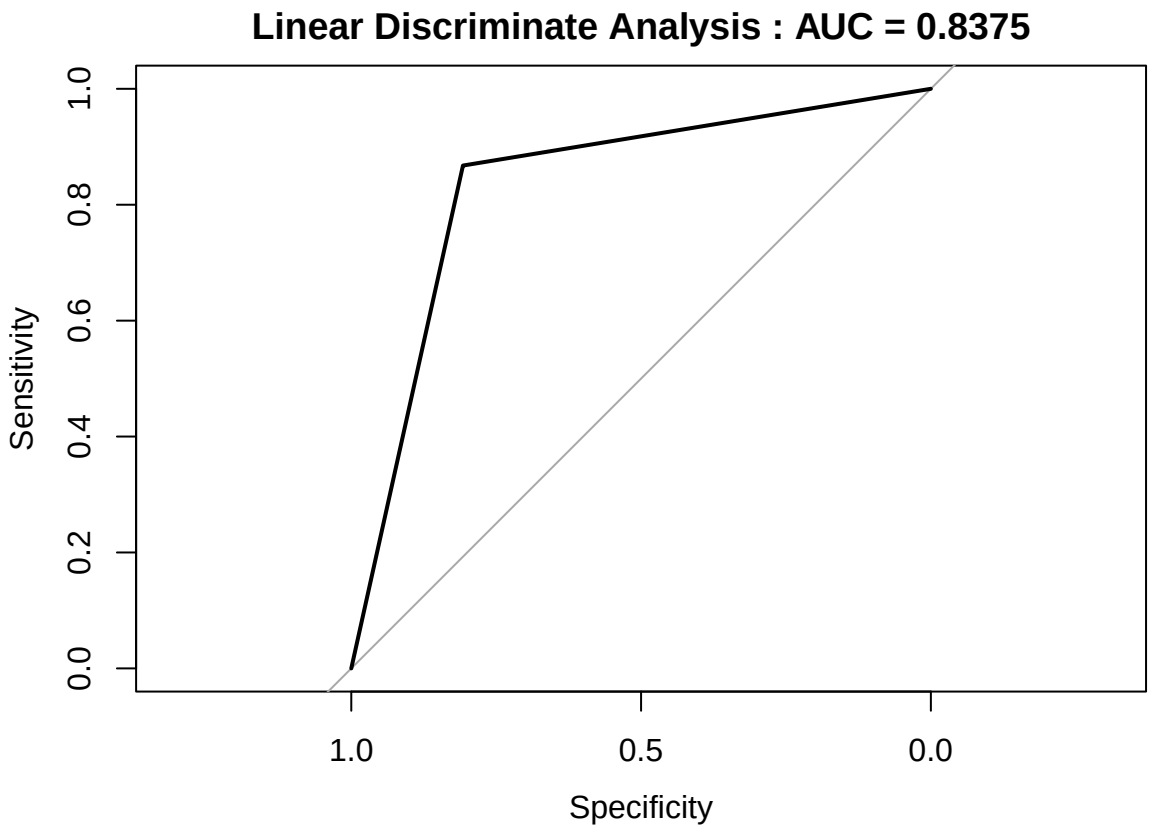
```
plot(roc_score, main = paste0("K-Nearest Neighbor : AUC = ", round(roc_score$auc, 4)))
```




```
# LDA
roc_score <- roc(sa.test[, ncol(sa.test)], as.numeric(pred.lda.test$class))
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

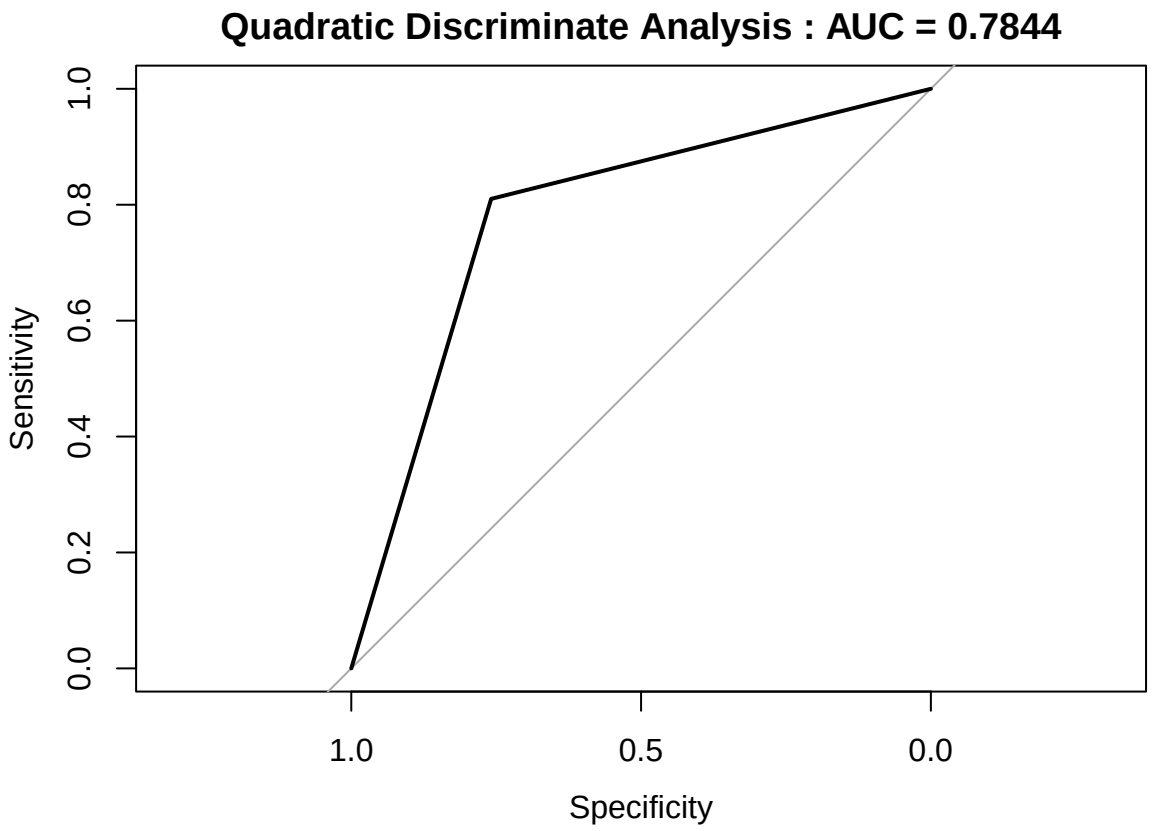
```
plot(roc_score, main = paste0("Linear Discriminate Analysis : AUC = ", round(roc_score$auc, 4)))
```




```
# QDA
roc_score <- roc(sa.test[, ncol(sa.test)], as.numeric(pred.qda.test$class))
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot(roc_score, main = paste0("Quadratic Discriminate Analysis : AUC = ", round(roc_score$auc, 4)))
```


Quadratic Discriminate Analysis : AUC = 0.7844
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1.0
0.5
0.0
Specificity