Statistical Learning Final Project

Yaniv Bronshtein, Benjamin Barnett, Krishna Piratla

11/19/2021

Import the necessary libraries

```
library(MASS)
library(ROCR)
library(tree)
library(class)
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-2
```

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(rpart)
library(gam)
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.20
```

Set the 4X2 grid for the ROC-curves

```
# par(mfrow=c(4,2))
```

Import the dataset

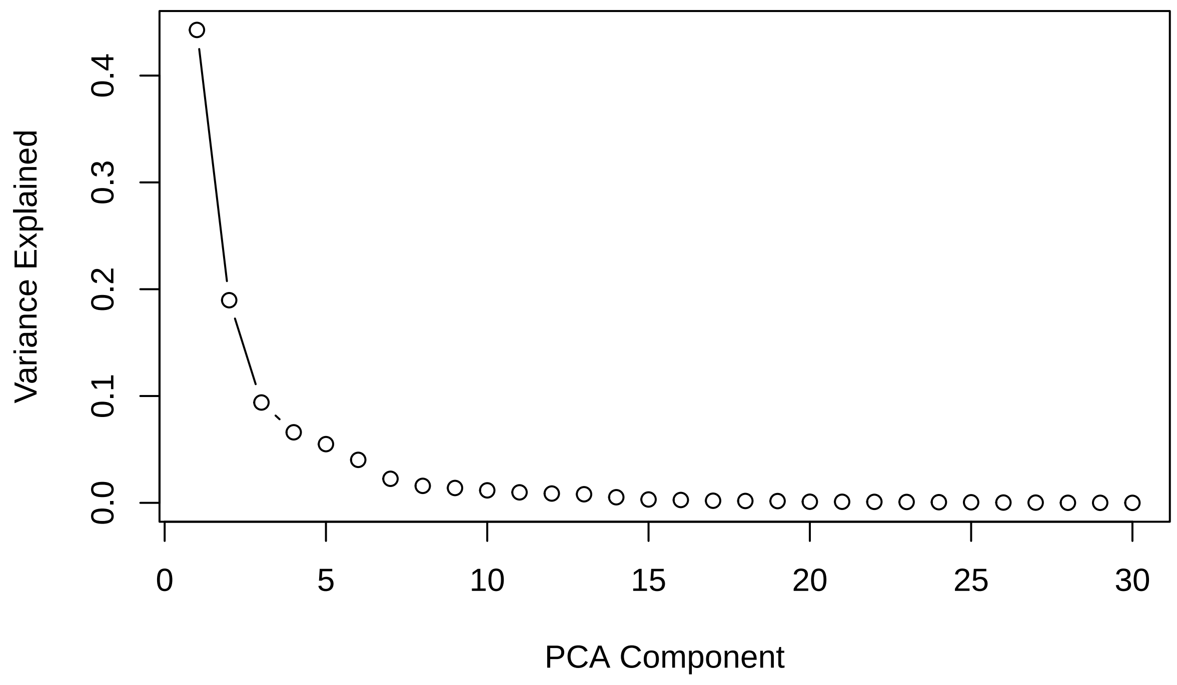
```
cancer <- read.csv('breast-cancer.csv')
```

Perform principle component analysis to minimize the number of features to be used for classification

```
result <- prcomp(cancer[,-c(1,2,33)], scale=TRUE)
```

Determine the variance explained by PCA

```
var_explained <- result$sdev^2 / sum(result$sdev^2)
plot(c(1:30),
     var_explained,
     type='b',
     xlab='PCA Component',
     ylab='Variance Explained')
```



Extract the first 7 principle components

```
pca <- result$x; pca <- pca[,1:7]
# cancer <- cancer[,c(2,23,24,27,28,31)]
```

Create train and test sets for PCA and diagnosis

```
set.seed(1)
train <- sample(1:nrow(cancer), 0.5*nrow(cancer))

pca_train <- data.frame(pca[train,])
pca_test <- data.frame(pca[-train,])

diagnosis_train <- cancer$diagnosis[train]
diagnosis_test <- cancer$diagnosis[-train]
```

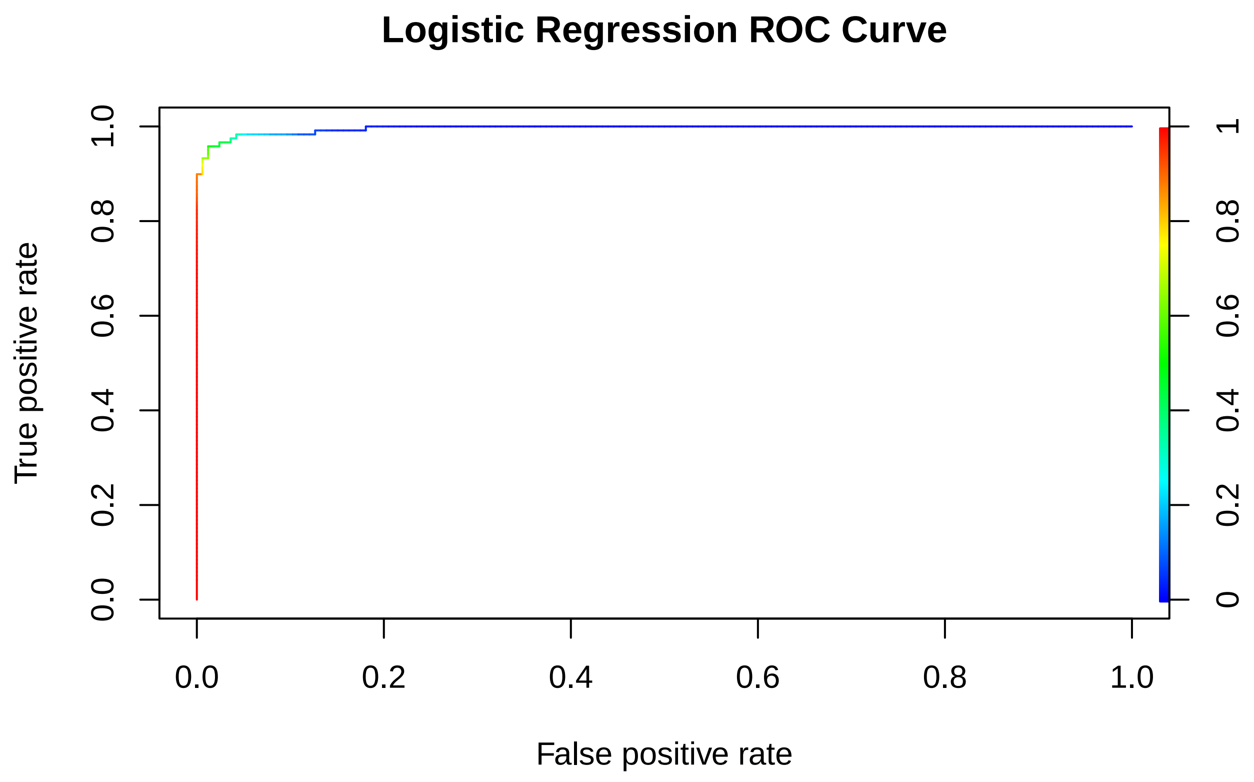
Fit a logistic regression model and generate the roc_curve for it

```
glm_fit <- suppressWarnings(glm(as.factor(diagnosis_train) ~., data=pca_train, family=binomial))
glm_prb <- predict(glm_fit, pca_test, type="response")

roc_prd1 <- prediction(glm_prb, diagnosis_test)
roc_prf1 <- performance(roc_prd1,"tpr","fpr")
```

Plot Logistic Regression roc curve

```r
plot(roc_prf1, colorize=TRUE, main="Logistic Regression ROC Curve")
```



Get the AUC value for our ROC curve

```r
auc1 <- as.numeric((performance(roc_prd1,"auc"))@y.values)
auc1
```

```
## [1] 0.9960514
```

Generate the confusion matrix and compute the accuracy for logistic regression for a 0.2 threshold

```r
glm_prd <- rep("B", 285); glm_prd[glm_prb > .2] <- "M"
c1_20 <- table(glm_prd, diagnosis_test)
a1_20 <- (table(glm_prd, diagnosis_test)[1] + table(glm_prd, diagnosis_test)[4])/285
c1_20
```

```
##        diagnosis_test
## glm_prd   B   M
##       B 155   2
##       M  11 117
```

```r
a1_20
```

```
## [1] 0.954386
```

Generate the confusion matrix and compute the accuracy for logistic regression for a 0.5 threshold

```r
glm_prd <- rep("B", 285); glm_prd[glm_prb > .5] <- "M"
c1_50 <- table(glm_prd, diagnosis_test)
a1_50 <- (table(glm_prd, diagnosis_test)[1] + table(glm_prd, diagnosis_test)[4])/285
c1_50
```

```
##         diagnosis_test
## glm_prd   B   M
##       B 164   6
##       M   2 113
```

```
a1_50
```

```
## [1] 0.9719298
```

Generate the confusion matrix and compute the accuracy for logistic regression for a 0.8 threshold

```
glm_prd <- rep("B", 285); glm_prd[glm_prb > .8] <- "M"
c1_80 <- table(glm_prd, diagnosis_test)
a1_80 <- (table(glm_prd, diagnosis_test)[1] + table(glm_prd, diagnosis_test)[4])/285
c1_80
```

```
##         diagnosis_test
## glm_prd   B   M
##       B 166  12
##       M   0 107
```

```
a1_80
```

```
## [1] 0.9578947
```

Perform Linear Discriminant analysis

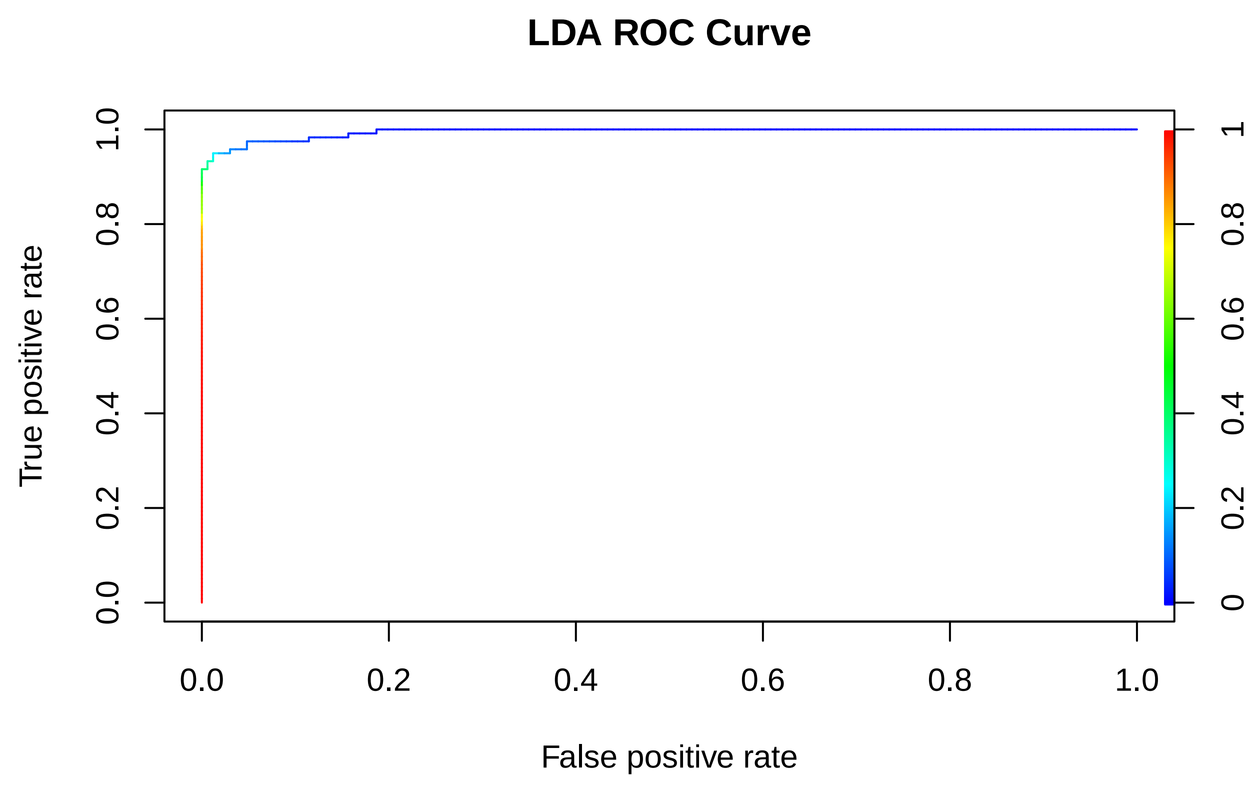
```
lda_fit <- lda(as.factor(diagnosis_train) ~., data=pca_train)
lda_prb <- predict(lda_fit, pca_test, type="response")
```

Generate the roc_curve and predictions for the lda model

```
roc_prd2 <- prediction(lda_prb$posterior[,2], diagnosis_test)
roc_prf2 <- performance(roc_prd2,"tpr","fpr")
```

Plot the roc_curve for lda

```
plot(roc_prf2, colorize=TRUE, main="LDA ROC Curve")
```



Get the Area under curve value from the roc_curve for lda

```
auc2 <- as.numeric((performance(roc_prd2,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for LDA for a 0.2 threshold

```
lda_prd <- rep("B", 285); lda_prd[lda_prb$posterior[,2] > .2] <- "M"
c2_20 <- table(lda_prd, diagnosis_test)
a2_20 <- (table(lda_prd, diagnosis_test)[1] + table(lda_prd, diagnosis_test)[4])/285
c2_20
```

```
##        diagnosis_test
## lda_prd   B   M
##       B 164   6
##       M   2 113
```

```
a2_20
```

```
## [1] 0.9719298
```

Generate the confusion matrix and compute the accuracy for LDA for a 0.5 threshold

```
lda_prd <- rep("B", 285); lda_prd[lda_prb$posterior[,2] > .5] <- "M"
c2_50 <- table(lda_prd, diagnosis_test)
a2_50 <- (table(lda_prd, diagnosis_test)[1] + table(lda_prd, diagnosis_test)[4])/285
c2_50
```

```
##        diagnosis_test
## lda_prd   B   M
##       B 166  14
##       M   0 105
```

```
a2_50
```

```
## [1] 0.9508772
```

Generate the confusion matrix and compute the accuracy for LDA for a 0.8 threshold

```
lda_prd <- rep("B", 285); lda_prd[lda_prb$posterior[,2] > .8] <- "M"
c2_80 <- table(lda_prd, diagnosis_test)
a2_80 <- (table(lda_prd, diagnosis_test)[1] + table(lda_prd, diagnosis_test)[4])/285
c2_80
```

```
##        diagnosis_test
## lda_prd   B   M
##       B 166  25
##       M   0  94
```

```
a2_80
```

```
## [1] 0.9122807
```

Perform Quadratic Discriminant analysis

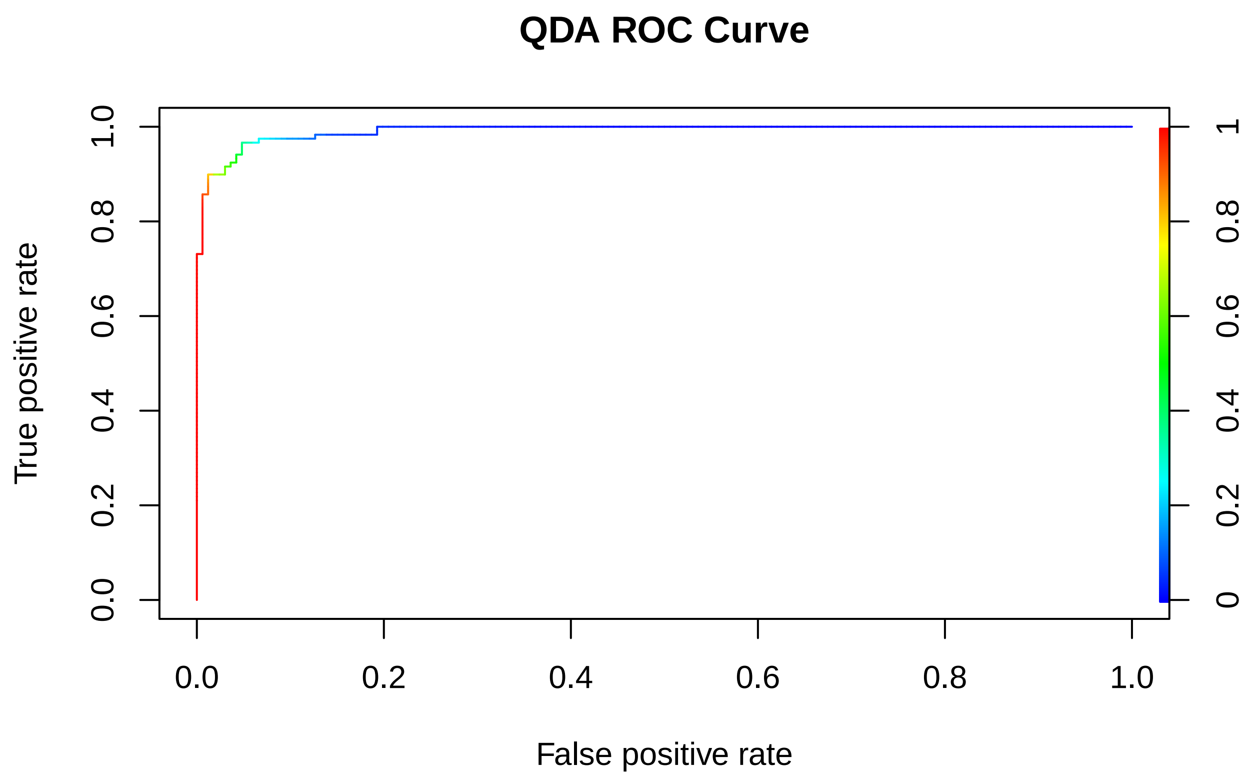
```
qda_fit <- qda(as.factor(diagnosis_train) ~., data=pca_train)
qda_prb <- predict(qda_fit, pca_test, type="response")
```

Generate the roc_curve and predictions for the qda model

```
roc_prd3 <- prediction(qda_prb$posterior[,2], diagnosis_test)
roc_prf3 <- performance(roc_prd3,"tpr","fpr")
```

Plot the roc_curve for qda

```
plot(roc_prf3, colorize=TRUE, main="QDA ROC Curve")
```



Get the AUC value for the roc_curve for qda

```
auc3 <- as.numeric((performance(roc_prd3,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for QDA for a 0.2 threshold

```
qda_prd <- rep("B", 285); qda_prd[qda_prb$posterior[,2] > .2] <- "M"
c3_20 <- table(qda_prd, diagnosis_test)
a3_20 <- (table(qda_prd, diagnosis_test)[1] + table(qda_prd, diagnosis_test)[4])/285
c3_20
```

```
##          diagnosis_test
## qda_prd   B   M
##         B 152   3
##         M  14 116
```

```
a3_20
```

```
## [1] 0.9403509
```

Generate the confusion matrix and compute the accuracy for QDA for a 0.5 threshold

```
qda_prd <- rep("B", 285); qda_prd[qda_prb$posterior[,2] > .5] <- "M"
c3_50 <- table(qda_prd, diagnosis_test)
a3_50 <- (table(qda_prd, diagnosis_test)[1] + table(qda_prd, diagnosis_test)[4])/285
a3_50
```

```
## [1] 0.9438596
```

```
a3_50
```

```
## [1] 0.9438596
```

Generate the confusion matrix and compute the accuracy for QDA for a 0.8 threshold

```
qda_prd <- rep("B", 285); qda_prd[qda_prb$posterior[,2] > .8] <- "M"
c3_80 <- table(qda_prd, diagnosis_test)
a3_80 <- (table(qda_prd, diagnosis_test)[1] + table(qda_prd, diagnosis_test)[4])/285
c3_80
```

```
##          diagnosis_test
## qda_prd   B   M
##         B 164  13
##         M   2 106
```

```
a3_80
```

```
## [1] 0.9473684
```

Train a general additive model and generate predictions

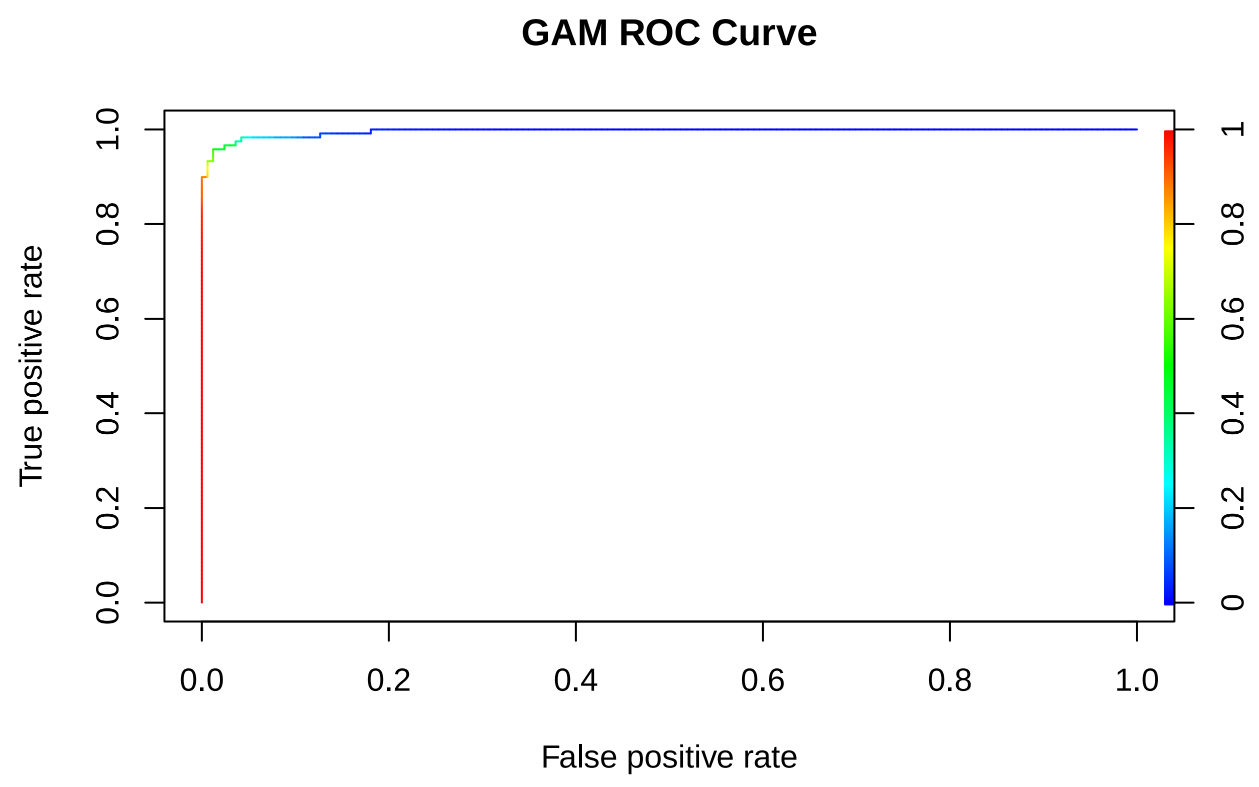
```
gam_fit <- gam(as.factor(diagnosis_train) ~., data=pca_train, family=binomial)
gam_prb <- predict(gam_fit, pca_test, type="response")
```

Generate the roc_curve and predictions for the GAM

```
roc_prd4 <- prediction(gam_prb, diagnosis_test)
roc_prf4 <- performance(roc_prd4,"tpr","fpr")
```

Plot the ROC curve for the GAM

```
plot(roc_prf4, colorize=TRUE, main="GAM ROC Curve")
```



Get the AUC value for the GAM

```
auc4 <- as.numeric((performance(roc_prd4,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for GAM for a 0.2 threshold

```
gam_prd <- rep("B", 285); gam_prd[gam_prb > .2] <- "M"
c4_20 <- table(gam_prd, diagnosis_test)
a4_20 <- (table(gam_prd, diagnosis_test)[1] + table(gam_prd, diagnosis_test)[4])/285
c4_20
```

```
##         diagnosis_test
## gam_prd   B   M
##       B 155   2
##       M  11 117
```

```
a4_20
```

```
## [1] 0.954386
```

Generate the confusion matrix and compute the accuracy for GAM for a 0.5 threshold

```
gam_prd <- rep("B", 285); gam_prd[gam_prb > .5] <- "M"
c4_50 <- table(gam_prd, diagnosis_test)
a4_50 <- (table(gam_prd, diagnosis_test)[1] + table(gam_prd, diagnosis_test)[4])/285
c4_50
```

```
##         diagnosis_test
## gam_prd   B   M
##       B 164   6
##       M   2 113
```

```
a4_50
```

```
## [1] 0.9719298
```

Generate the confusion matrix and compute the accuracy for GAM for a 0.8 threshold

```
gam_prd <- rep("B", 285); gam_prd[gam_prb > .8] <- "M"
c4_80 <- table(gam_prd, diagnosis_test)
a4_80 <- (table(gam_prd, diagnosis_test)[1] + table(gam_prd, diagnosis_test)[4])/285
c4_80
```

```
##         diagnosis_test
## gam_prd   B   M
##         B 166  12
##         M   0 107
```

```
a4_80
```

```
## [1] 0.9578947
```

Train a Random Forest Model and generate predictions

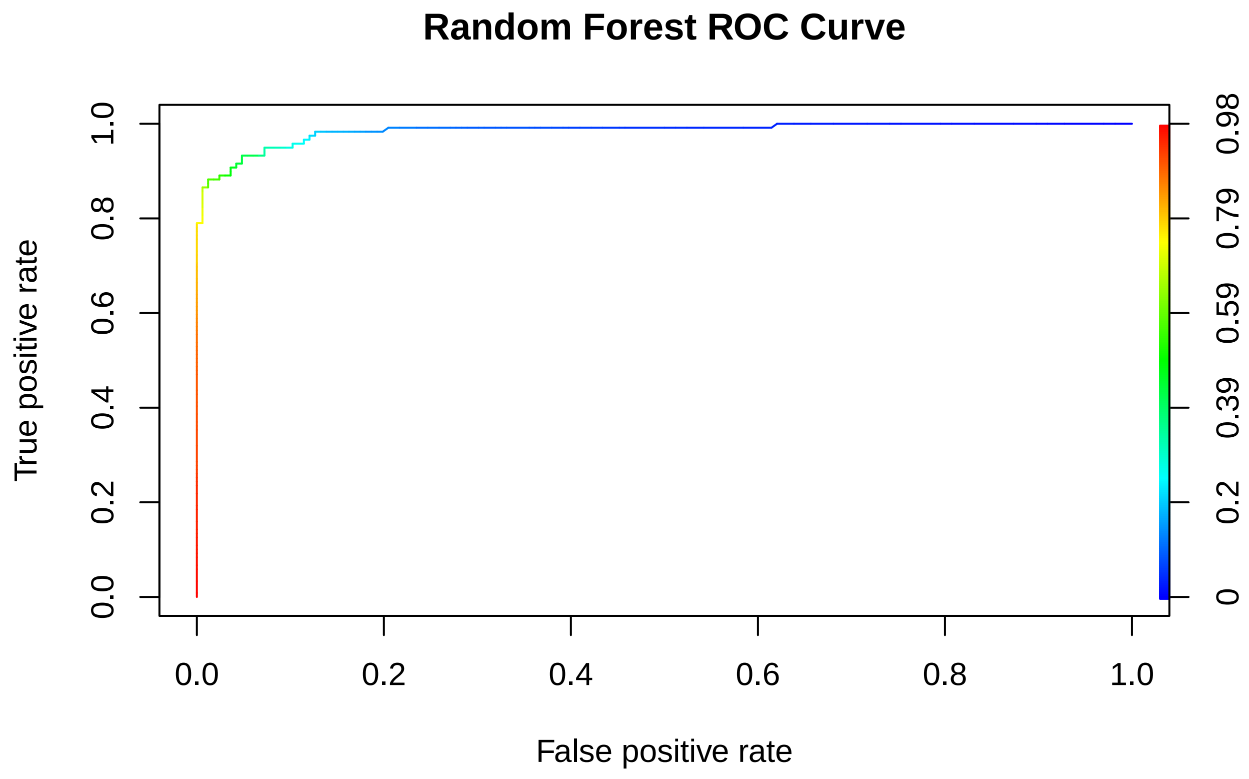
```
rnf_fit <- randomForest(as.factor(diagnosis_train) ~., data=pca_train)
rnf_prb <- predict(rnf_fit, pca_test, type="prob")
```

Generate the ROC curve for Random Forest

```
roc_prd5 <- prediction(rnf_prb[,2], diagnosis_test)
roc_prf5 <- performance(roc_prd5,"tpr","fpr")
```

Plot the ROC Curve for Random Forest

```
plot(roc_prf5, colorize=TRUE, main="Random Forest ROC Curve")
```



Get the AUC value for Random Forest

```
auc5 <- as.numeric((performance(roc_prd5,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for Random Forest for a 0.2 threshold

```
rnf_prd <- rep("B", 285); rnf_prd[rnf_prb[,2] > .2] <- "M"
c5_20 <- table(rnf_prd, diagnosis_test)
a5_20 <- (table(rnf_prd, diagnosis_test)[1] + table(rnf_prd, diagnosis_test)[4])/285
c5_20
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 145   3
##       M  21 116
```

```
a5_20
```

```
## [1] 0.9157895
```

Generate the confusion matrix and compute the accuracy for Random Forest for a 0.5 threshold

```
rnf_prd <- rep("B", 285); rnf_prd[rnf_prb[,2] > .5] <- "M"
c5_50 <- table(rnf_prd, diagnosis_test)
a5_50 <- (table(rnf_prd, diagnosis_test)[1] + table(rnf_prd, diagnosis_test)[4])/285
c5_50
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 161  13
##       M   5 106
```

```
a5_50
```

```
## [1] 0.9368421
```

Generate the confusion matrix and compute the accuracy for Random Forest for a 0.8 threshold

```
rnf_prd <- rep("B", 285); rnf_prd[rnf_prb[,2] > .8] <- "M"
c5_80 <- table(rnf_prd, diagnosis_test)
a5_80 <- (table(rnf_prd, diagnosis_test)[1] + table(rnf_prd, diagnosis_test)[4])/285
c5_80
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 166  40
##       M   0  79
```

```
a5_80
```

```
## [1] 0.8596491
```

Train a Generalized Boosted Regression Model and generate predictions

```
bst_fit <- gbm(ifelse(diagnosis_train == 'B', 0, 1) ~., data=pca_train, distribution="bernoulli", n.tre
bst_prb <- predict(bst_fit, pca_test, type="response")
```

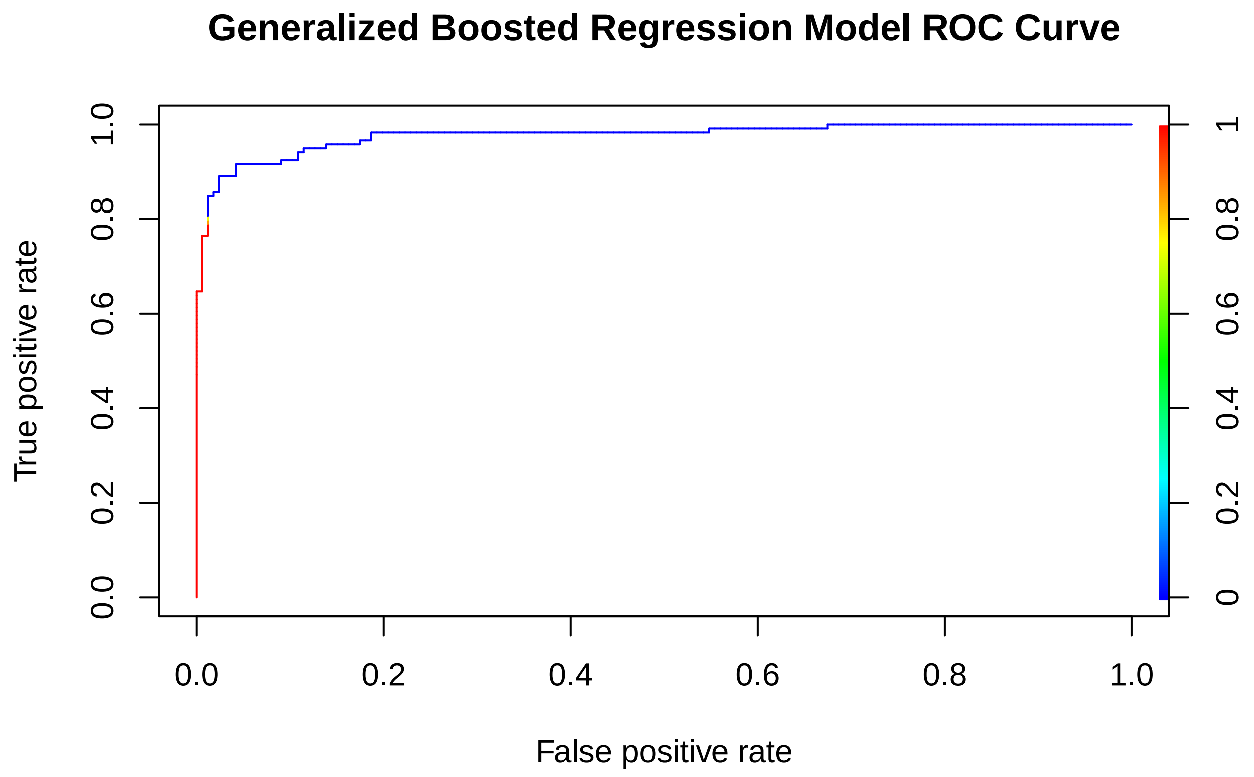
```
## Using 5000 trees...
```

Generate the ROC curve for Generalized Boosted Regression Model Model

```
roc_prd6 <- prediction(bst_prb, diagnosis_test)
roc_prf6 <- performance(roc_prd6,"tpr","fpr")
```

Plot the ROC curve for Generalized Boosted Regression Model Model

```
plot(roc_prf6, colorize=TRUE, main="Generalized Boosted Regression Model ROC Curve")
```



Extract the AUC value for the Generalized Boosted Regression Model Model

```
auc6 <- as.numeric((performance(roc_prd6,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for Generalized Boosted Regression Model for a 0.2 threshold

```
bst_prd <- rep("B", 285); bst_prd[bst_prb > .2] <- "M"
c6_20 <- table(bst_prd, diagnosis_test)
a6_20 <- (table(bst_prd, diagnosis_test)[1] + table(bst_prd, diagnosis_test)[4])/285
c6_20
```

```
##        diagnosis_test
## bst_prd   B   M
##       B 164  24
##       M   2  95
```

```
a6_20
```

```
## [1] 0.9087719
```

Generate the confusion matrix and compute the accuracy for Generalized Boosted Regression Model for a 0.5 threshold

```
bst_prd <- rep("B", 285); bst_prd[bst_prb > .5] <- "M"
c6_50 <- table(bst_prd, diagnosis_test)
a6_50 <- (table(bst_prd, diagnosis_test)[1] + table(bst_prd, diagnosis_test)[4])/285
c6_50
```

```
##        diagnosis_test
## bst_prd   B   M
##       B 164  24
##       M   2  95
```

```
a6_50
```

```
## [1] 0.9087719
```

Generate the confusion matrix and compute the accuracy for Generalized Boosted Regression Model for a 0.8 threshold

```
bst_prd <- rep("B", 285); bst_prd[bst_prb > .8] <- "M"
c6_80 <- table(bst_prd, diagnosis_test)
a6_80 <- (table(bst_prd, diagnosis_test)[1] + table(bst_prd, diagnosis_test)[4])/285
c6_80
```

```
##        diagnosis_test
## bst_prd   B   M
##       B 164  25
##       M   2  94
```

```
a6_80
```

```
## [1] 0.9052632
```

Perform tuning before running the Linear Support Vector Machine algorithm

```
get_tuning <- tune(svm, diagnosis ~ ., data=data.frame(cbind(diagnosis = as.factor(diagnosis_train), pca
```

Train Linear Support Vector Machine Model

```
svm_linear <- svm(diagnosis ~ ., data=data.frame(cbind(diagnosis = as.factor(diagnosis_train), pca_trai
kernel='linear', cost=get_tuning$best.performance, probability=TRUE)
```

Generate the Linear SVM predictions

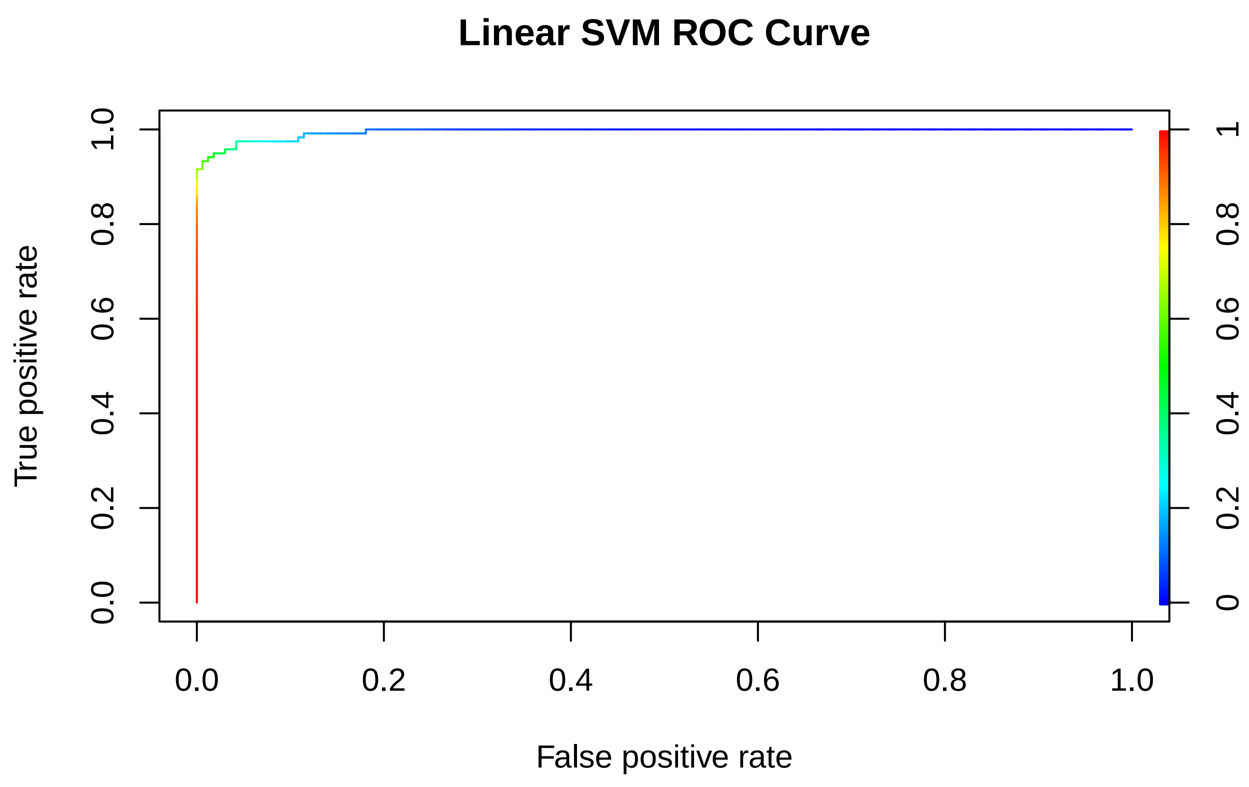
```
svm_prb <- predict(svm_linear, pca_test, probability=TRUE)
svm_prb <- attr(svm_prb, 'probabilities')[,2]
```

Generate the roc curve for Linear SVM

```
roc_prd7 <- prediction(svm_prb, diagnosis_test)
roc_prf7 <- performance(roc_prd7,"tpr","fpr")
```

Plot the ROC Curve for Linear SVM

```
plot(roc_prf7, colorize=TRUE, main="Linear SVM ROC Curve")
```



Extract the AUC value for the Linear SVM model

```
auc7 <- as.numeric((performance(roc_prd7,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for Linear SVM for a 0.2 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .2] <- "M"
c7_20 <- table(svm_prd, diagnosis_test)
a7_20 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c7_20
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 148   3
##       M  18 116
```

```
a7_20
```

```
## [1] 0.9263158
```

Generate the confusion matrix and compute the accuracy for Linear SVM for a 0.5 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .5] <- "M"
c7_50 <- table(svm_prd, diagnosis_test)
a7_50 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c7_50
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 164   8
##       M   2 111
```

```
a7_50
```

```
## [1] 0.9649123
```

Generate the confusion matrix and compute the accuracy for Linear SVM for a 0.8 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .8] <- "M"
c7_80 <- table(svm_prd, diagnosis_test)
a7_80 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c7_80
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 166  18
##       M   0 101
```

```
a7_80
```

```
## [1] 0.9368421
```

Perform tuning before running the Radial Support Vector Machine algorithm

```
get_tuning <- tune(svm, diagnosis ~ ., data=data.frame(cbind(diagnosis = as.factor(diagnosis_train), pca
```

Train Radial Support Vector Machine Model

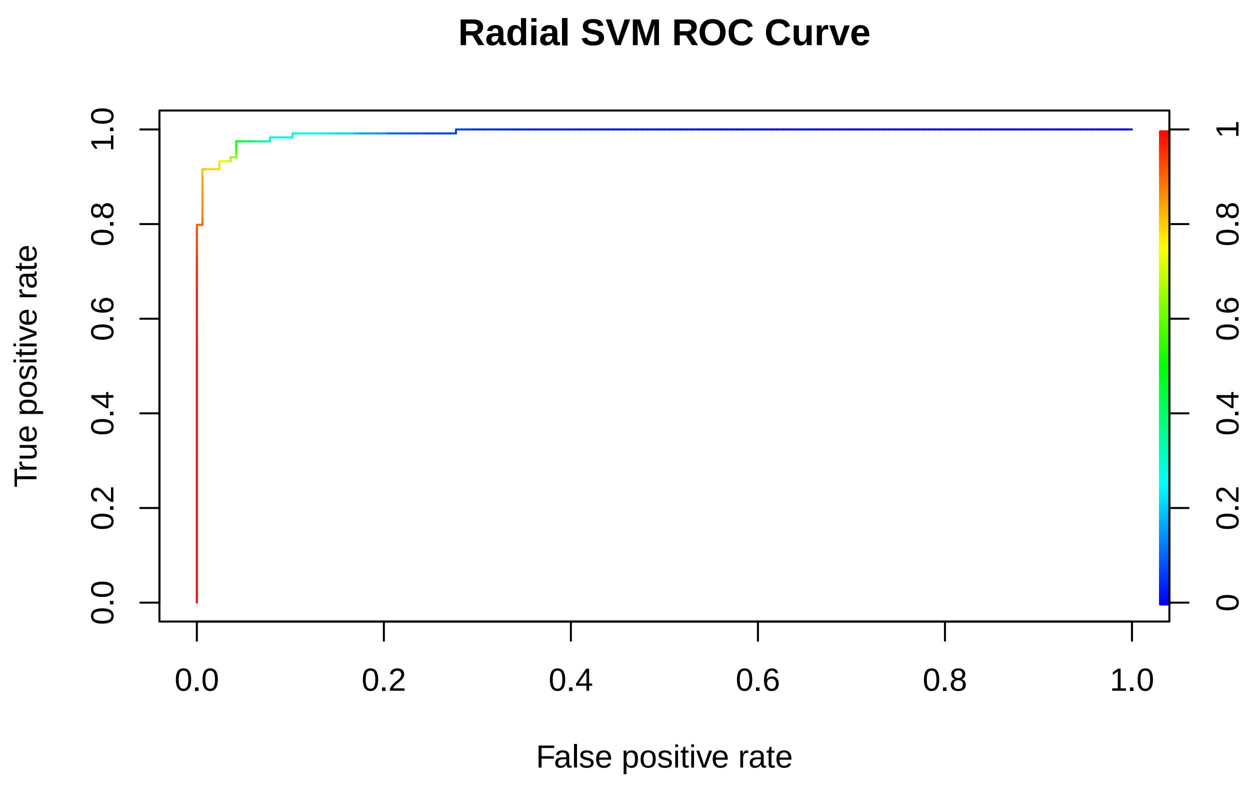
```
svm_radial <- svm(diagnosis ~ ., data=data.frame(cbind(diagnosis = as.factor(diagnosis_train), pca_trai
kernel='radial', cost=get_tuning$best.performance, probability=TRUE)
```

Generate the Radial SVM predictions

```
svm_prb <- predict(svm_radial, pca_test, probability=TRUE)
svm_prb <- attr(svm_prb, 'probabilities')[,2]
```

Generate the roc curve for Radial SVM

```
roc_prd8 <- prediction(svm_prb, diagnosis_test)
roc_prf8 <- performance(roc_prd8,"tpr","fpr")
plot(roc_prf8, colorize=TRUE, main="Radial SVM ROC Curve")
```



Extract the AUC value for the Radial SVM model

```
auc8 <- as.numeric((performance(roc_prd8,"auc"))@y.values)
```

Generate the confusion matrix and compute the accuracy for Radial SVM for a 0.2 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .2] <- "M"
c8_20 <- table(svm_prd, diagnosis_test)
a8_20 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c8_20
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 139   1
##       M  27 118
```

```
a8_20
```

```
## [1] 0.9017544
```

Generate the confusion matrix and compute the accuracy for Radial SVM for a 0.5 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .5] <- "M"
c8_50 <- table(svm_prd, diagnosis_test)
a8_50 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c8_50
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 159   4
##       M   7 115
```

```
a8_50
```

```
## [1] 0.9614035
```

Generate the confusion matrix and compute the accuracy for Radial SVM for a 0.8 threshold

```
svm_prd <- rep("B", 285); svm_prd[svm_prb > .8] <- "M"
c8_80 <- table(svm_prd, diagnosis_test)
a8_80 <- (table(svm_prd, diagnosis_test)[1] + table(svm_prd, diagnosis_test)[4])/285
c8_80
```

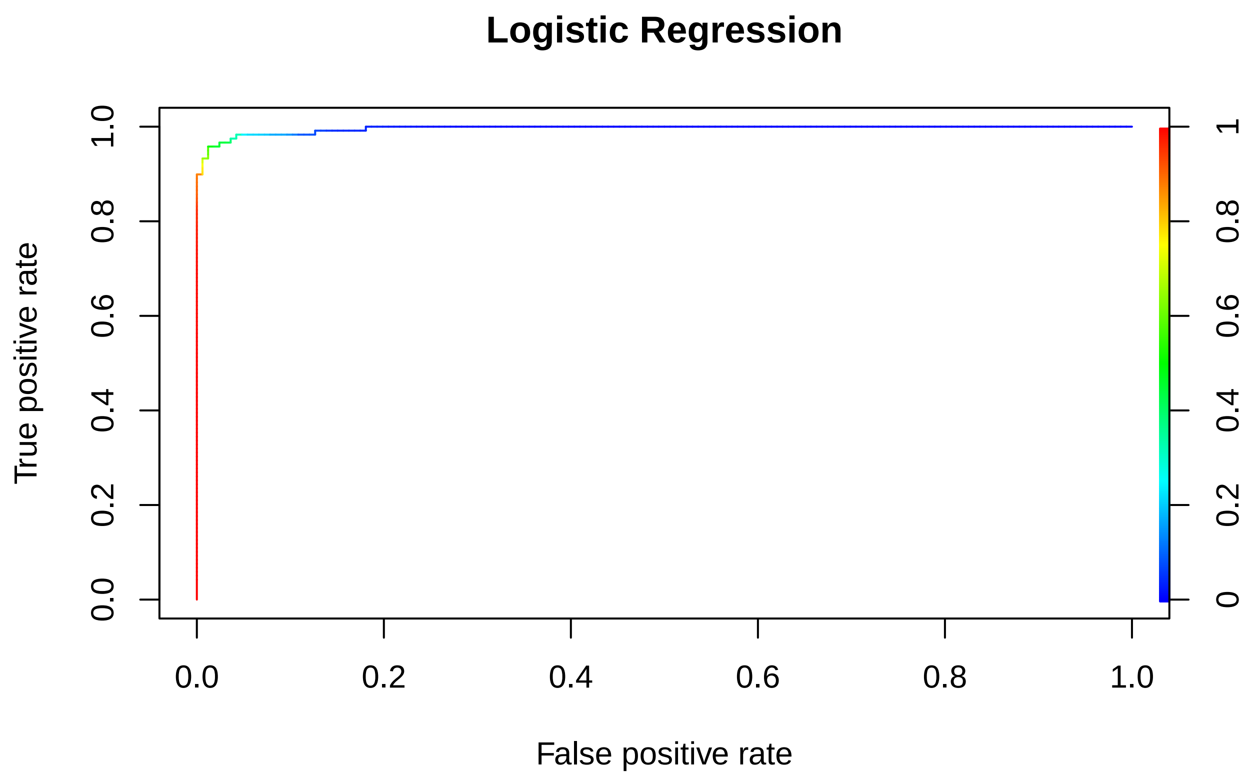
```
##          diagnosis_test
## svm_prd   B   M
##        B 165  10
##        M   1 109
```

```
a8_80
```

```
## [1] 0.9614035
```

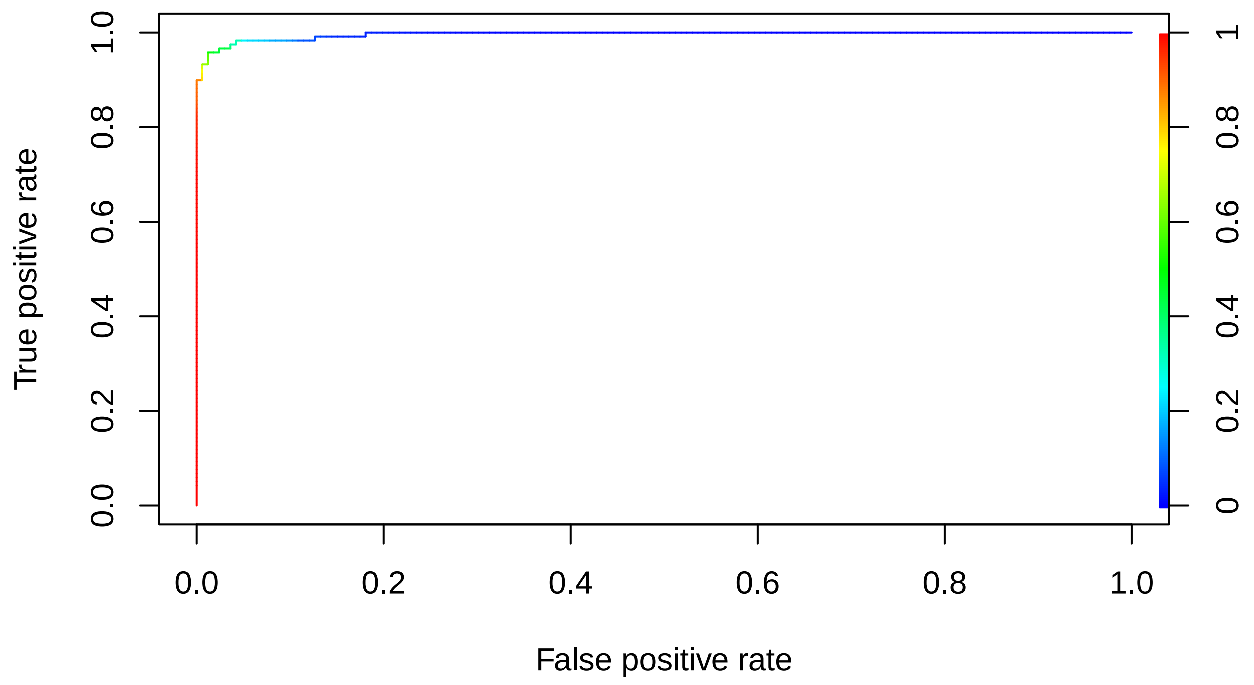
Plot of all ROC curves

```
# par(mfrow=c(1,2))
plot(roc_prf1, colorize=TRUE, main="Logistic Regression")
```



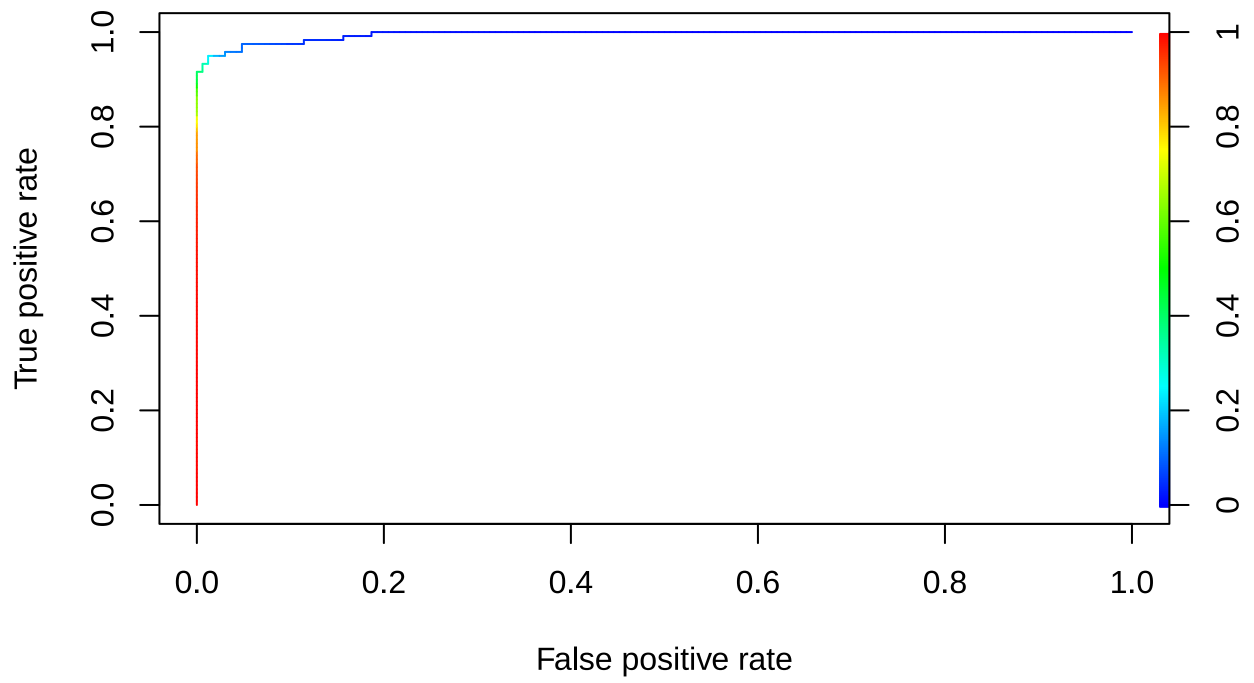
```
plot(roc_prf4, colorize=TRUE, main="GAM")
```

GAM

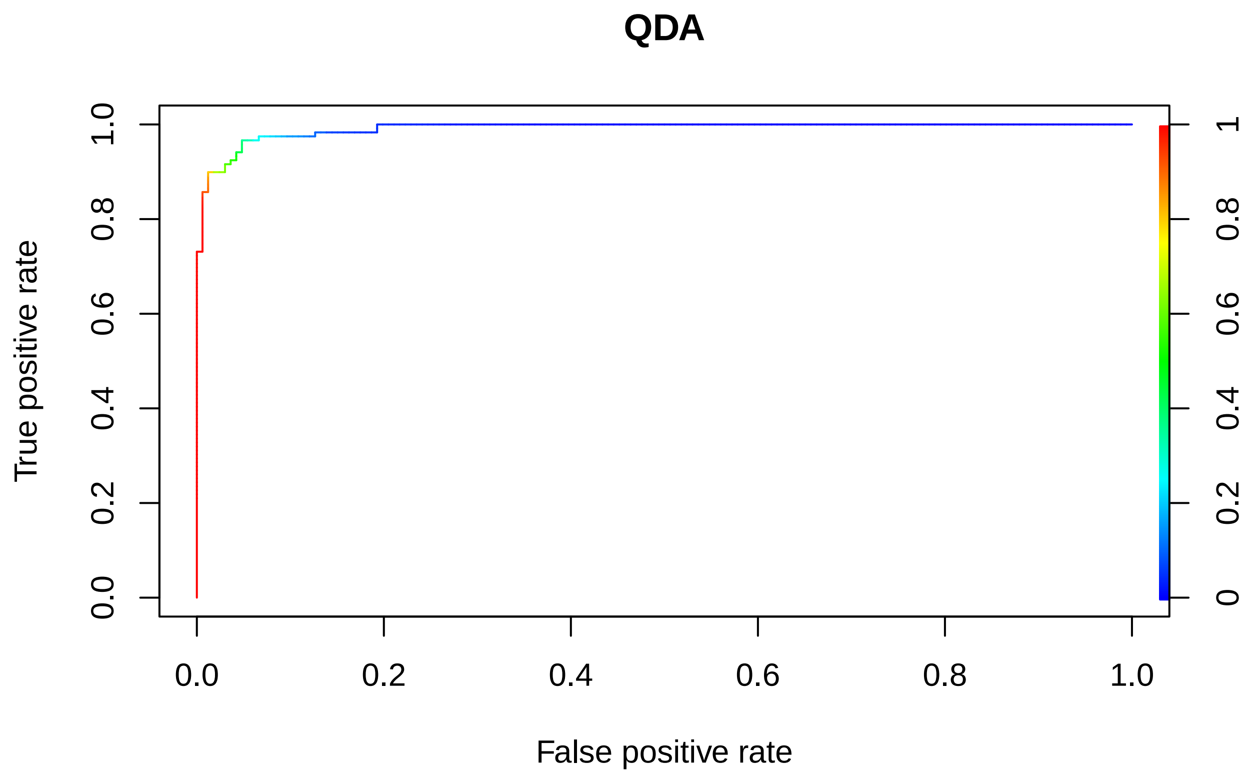


```
# par(mfrow=c(1,2))
plot(roc_prf2, colorize=TRUE, main="LDA")
```

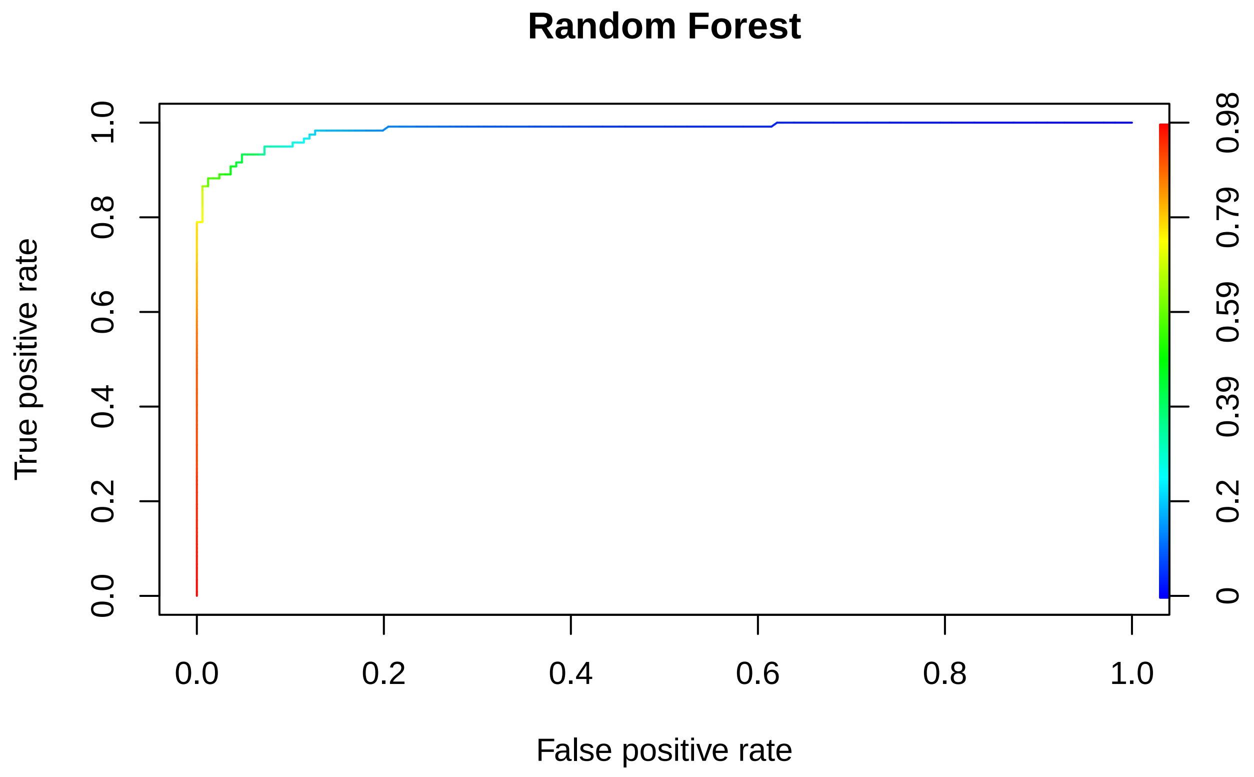
LDA



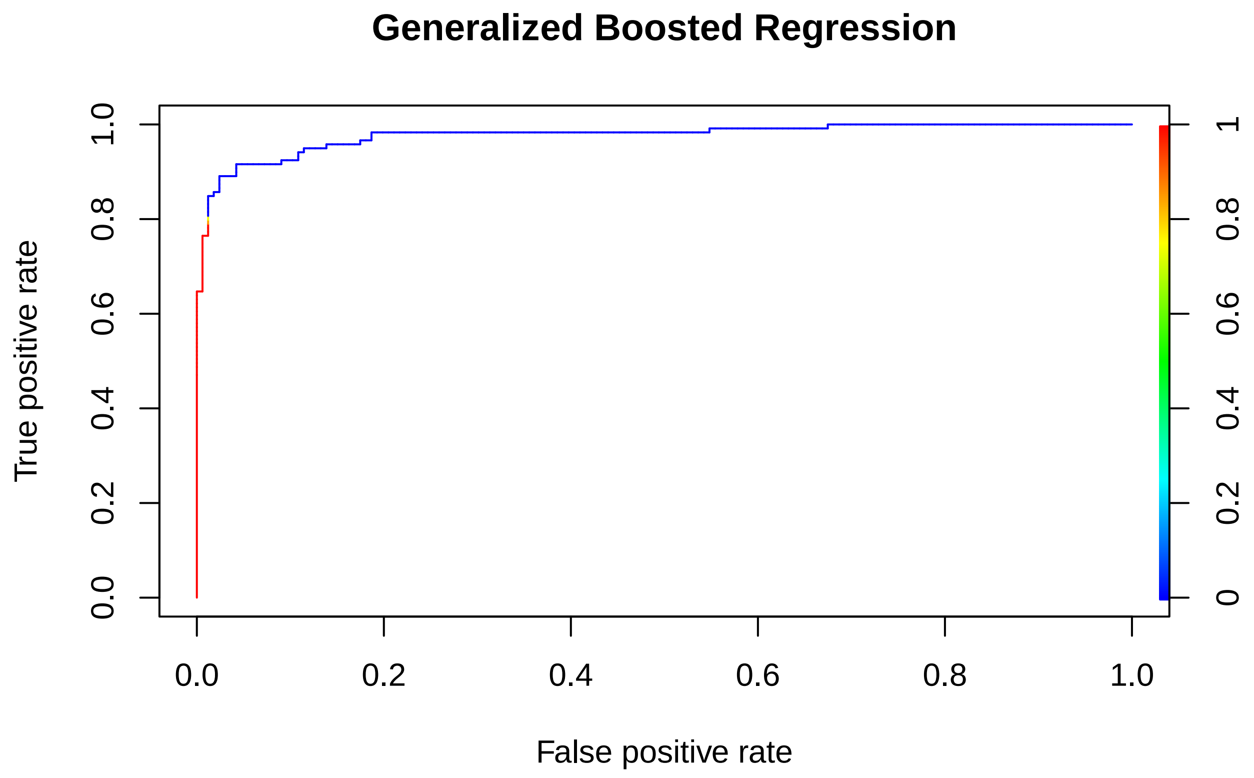
```r
plot(roc_prf3, colorize=TRUE, main="QDA")
```



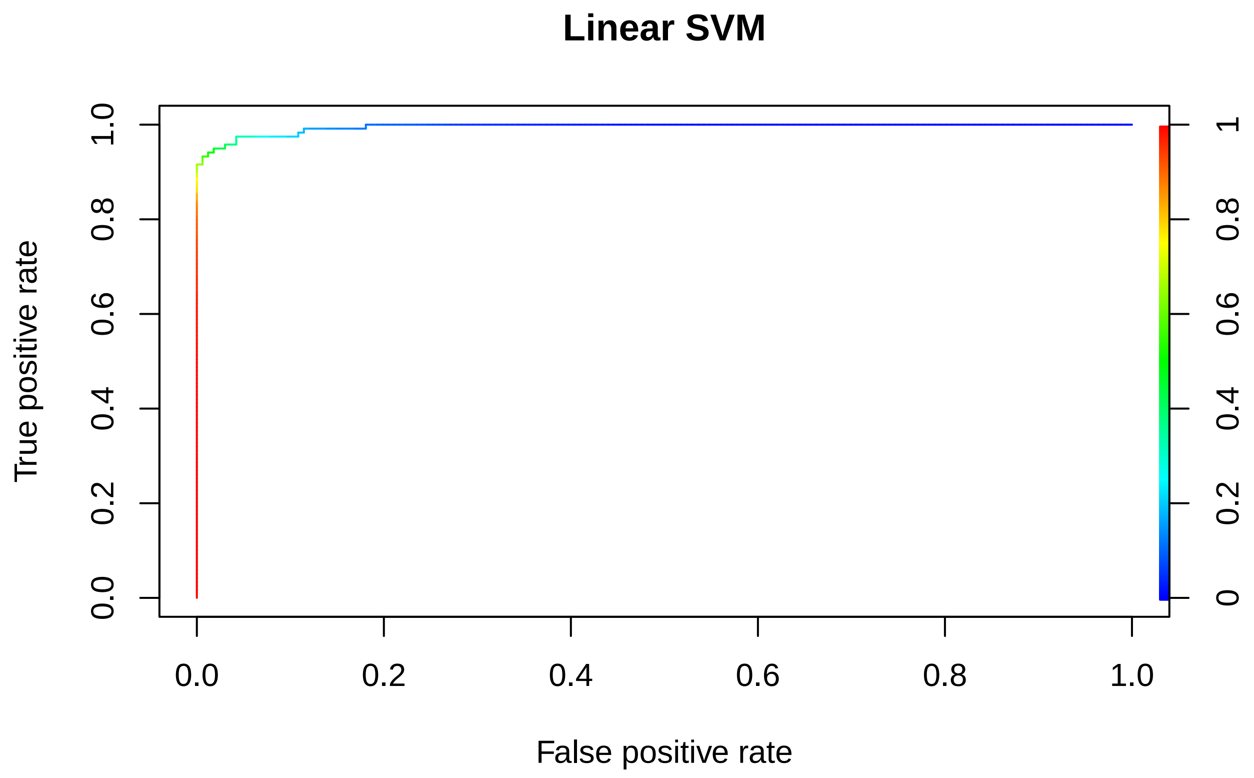
```r
# par(mfrow=c(1,2))
plot(roc_prf5, colorize=TRUE, main="Random Forest")
```



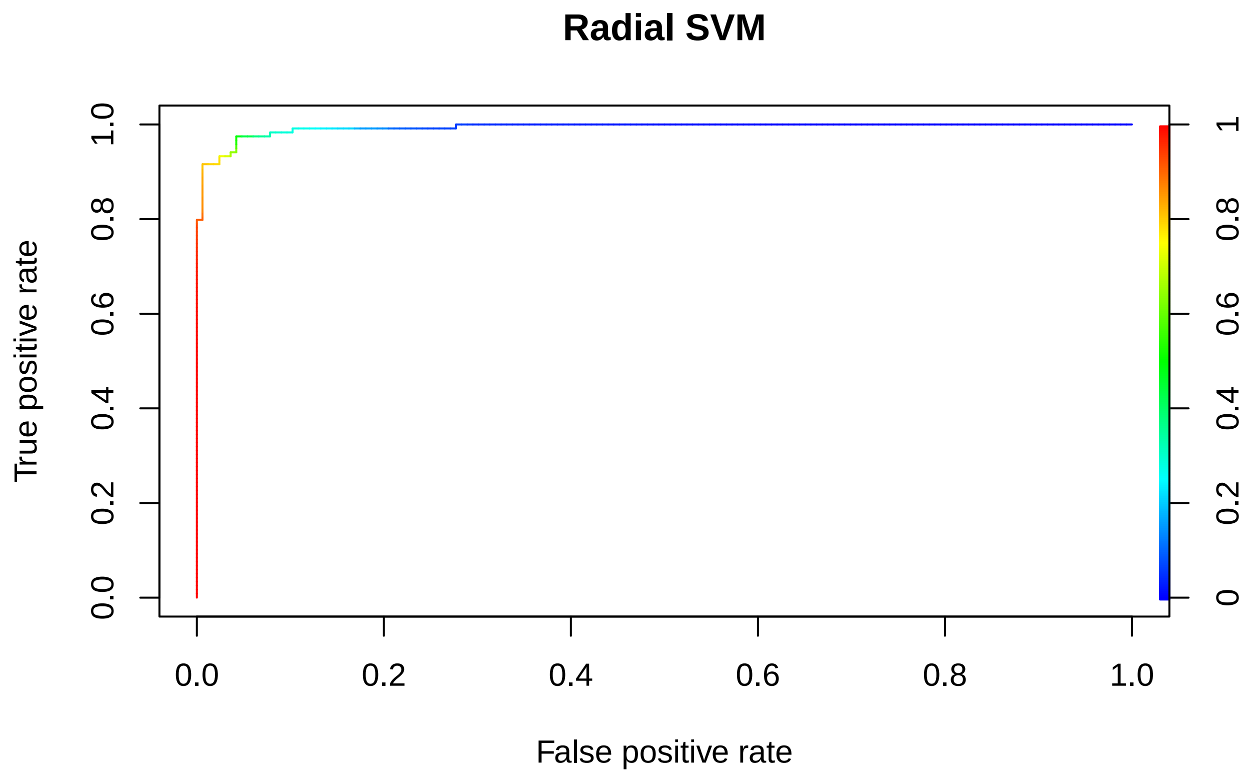
```r
plot(roc_prf6, colorize=TRUE, main="Generalized Boosted Regression")
```



```r
# par(mfrow=c(1,2))
plot(roc_prf7, colorize=TRUE, main="Linear SVM")
```



```r
plot(roc_prf8, colorize=TRUE, main="Radial SVM")
```



Function floor plotting confusion matrices

```r
plot_confusion_accuracy <- function(a,c){
  cat("Threshold: 0.2 |",'Accuracy:',a[1],'\n\n')
  c[1]
  cat("\n************************************\n")

  cat("Threshold: 0.5 |",'Accuracy:',a[2],'\n\n')
  c[2]
  cat("\n************************************\n")
  cat("Threshold: 0.8 |",'Accuracy:',a[3],'\n\n')
  c[3]

}
```

Plotting all confusion matrices

```
## Threshold: 0.2 | Accuracy: 0.954386

##        diagnosis_test
## glm_prd   B   M
##       B 155   2
##       M  11 117

##
## ************************************

## Threshold: 0.5 | Accuracy: 0.9719298

##        diagnosis_test
## glm_prd   B   M
##       B 164   6
##       M   2 113
##
## **********************************
## Threshold: 0.8 | Accuracy: 0.9578947
##        diagnosis_test
## glm_prd   B   M
##       B 166  12
##       M   0 107
## Threshold: 0.2 | Accuracy: 0.9719298
##        diagnosis_test
## lda_prd   B   M
##       B 164   6
##       M   2 113
##
## **********************************
## Threshold: 0.5 | Accuracy: 0.9508772
##        diagnosis_test
## lda_prd   B   M
##       B 166  14
##       M   0 105
##
## **********************************
## Threshold: 0.8 | Accuracy: 0.9122807
##        diagnosis_test
## lda_prd   B   M
##       B 166  25
##       M   0  94
## Threshold: 0.2 | Accuracy: 0.9403509
##        diagnosis_test
## qda_prd   B   M
##       B 152   3
##       M  14 116
##
## **********************************
## Threshold: 0.5 | Accuracy: 0.9438596
##        diagnosis_test
## qda_prd   B   M
##       B 159   9
##       M   7 110
##
## **********************************
## Threshold: 0.8 | Accuracy: 0.9473684
##        diagnosis_test
## qda_prd   B   M
##       B 164  13
##       M   2 106
```

```
## Threshold: 0.2 | Accuracy: 0.954386
```

```
##        diagnosis_test
## gam_prd   B   M
##       B 155   2
##       M  11 117
```

```
##
## **********************************
```

```
## Threshold: 0.5 | Accuracy: 0.9719298
```

```
##        diagnosis_test
## gam_prd   B   M
##       B 164   6
##       M   2 113
```

```
##
## **********************************
```

```
## Threshold: 0.8 | Accuracy: 0.9578947
```

```
##        diagnosis_test
## gam_prd   B   M
##       B 166  12
##       M   0 107
```

```
## Threshold: 0.2 | Accuracy: 0.9157895
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 145   3
##       M  21 116
```

```
##
## **********************************
```

```
## Threshold: 0.5 | Accuracy: 0.9368421
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 161  13
##       M   5 106
```

```
##
## **********************************
```

```
## Threshold: 0.8 | Accuracy: 0.8596491
```

```
##        diagnosis_test
## rnf_prd   B   M
##       B 166  40
##       M   0  79
```

```
## Threshold: 0.2 | Accuracy: 0.9087719
```

```
##        diagnosis_test
## bst_prd   B   M
##       B 164  24
##       M   2  95
##
## **********************************
## Threshold: 0.5 | Accuracy: 0.9087719
##        diagnosis_test
## bst_prd   B   M
##       B 164  24
##       M   2  95
##
## **********************************
## Threshold: 0.8 | Accuracy: 0.9052632
##        diagnosis_test
## bst_prd   B   M
##       B 164  25
##       M   2  94
## Threshold: 0.2 | Accuracy: 0.9263158
##        diagnosis_test
## svm_prd   B   M
##       B 148   3
##       M  18 116
##
## **********************************
## Threshold: 0.5 | Accuracy: 0.9649123
##        diagnosis_test
## svm_prd   B   M
##       B 164   8
##       M   2 111
##
## **********************************
## Threshold: 0.8 | Accuracy: 0.9368421
##        diagnosis_test
## svm_prd   B   M
##       B 166  18
##       M   0 101
## Threshold: 0.2 | Accuracy: 0.9017544
##        diagnosis_test
## svm_prd   B   M
##       B 139   1
##       M  27 118
##
## **********************************
## Threshold: 0.5 | Accuracy: 0.9614035
##        diagnosis_test
## svm_prd   B   M
##       B 159   4
##       M   7 115
```

```
##
## ***********************************
```

```
## Threshold: 0.8 | Accuracy: 0.9614035
```

```
##        diagnosis_test
## svm_prd   B   M
##       B 165  10
##       M   1 109
```

AUC

```
auc1
```

```
## [1] 0.9960514
```

```
auc4
```

```
## [1] 0.9960514
```

```
auc2
```

```
## [1] 0.9947859
```

```
auc3
```

```
## [1] 0.991141
```

```
auc5
```

```
## [1] 0.9853701
```

```
auc6
```

```
## [1] 0.976663
```

```
auc7
```

```
## [1] 0.9952921
```

```
auc8
```

```
## [1] 0.9933178
```