BUDA535Module3

## Problem 1 :

Using the Carseats data in the ISLR package create a categorical binary response such that we’re seeing if location sales are above or below the median value. That is CarseatsSales). Fit a logistic regression model using all predictors available. Explore the classifier that could be created from this model using an ROC curve. Discuss any trade-offs you see in the classification performance in the model.

library(ISLR)  
library(ROCR)

## Warning: package 'ROCR' was built under R version 3.6.3

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.6.3

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

library(caret)

## Warning: package 'caret' was built under R version 3.6.2

## Loading required package: lattice

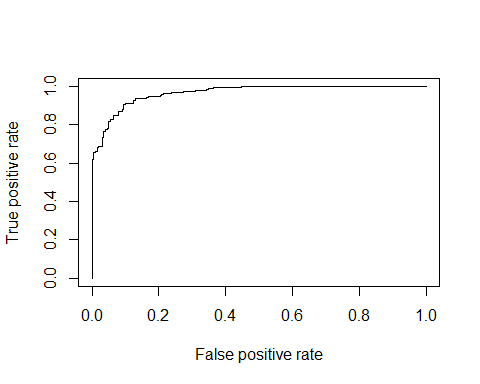
## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.2

library(e1071)

## Warning: package 'e1071' was built under R version 3.6.3

data(Carseats)  
#head(Carseats)  
  
Carseats$Sales\_factor<- as.factor(Carseats$Sales > median(Carseats$Sales))  
  
logistic.model <- glm(Sales\_factor ~ .-Sales, data=Carseats, family="binomial")  
#summary(logistic.model)  
  
pred <- predict(logistic.model,type="response")  
sales\_pred = prediction(pred,Carseats$Sales\_factor)  
p1 = performance(sales\_pred,"tpr","fpr")  
plot(p1)



confusionMatrix(as.factor(pred > .7),Carseats$Sales\_factor, positive = "TRUE")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 191 38  
## TRUE 10 161  
##   
## Accuracy : 0.88   
## 95% CI : (0.8441, 0.9102)  
## No Information Rate : 0.5025   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7598   
##   
## Mcnemar's Test P-Value : 9.735e-05   
##   
## Sensitivity : 0.8090   
## Specificity : 0.9502   
## Pos Pred Value : 0.9415   
## Neg Pred Value : 0.8341   
## Prevalence : 0.4975   
## Detection Rate : 0.4025   
## Detection Prevalence : 0.4275   
## Balanced Accuracy : 0.8796   
##   
## 'Positive' Class : TRUE   
##

confusionMatrix(as.factor(pred > .9),Carseats$Sales\_factor, positive = "TRUE")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 199 67  
## TRUE 2 132  
##   
## Accuracy : 0.8275   
## 95% CI : (0.7868, 0.8632)  
## No Information Rate : 0.5025   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6544   
##   
## Mcnemar's Test P-Value : 1.312e-14   
##   
## Sensitivity : 0.6633   
## Specificity : 0.9900   
## Pos Pred Value : 0.9851   
## Neg Pred Value : 0.7481   
## Prevalence : 0.4975   
## Detection Rate : 0.3300   
## Detection Prevalence : 0.3350   
## Balanced Accuracy : 0.8267   
##   
## 'Positive' Class : TRUE   
##

confusionMatrix(as.factor(pred > .61),Carseats$Sales\_factor, positive = "TRUE")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 185 28  
## TRUE 16 171  
##   
## Accuracy : 0.89   
## 95% CI : (0.8552, 0.9189)  
## No Information Rate : 0.5025   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.7799   
##   
## Mcnemar's Test P-Value : 0.09725   
##   
## Sensitivity : 0.8593   
## Specificity : 0.9204   
## Pos Pred Value : 0.9144   
## Neg Pred Value : 0.8685   
## Prevalence : 0.4975   
## Detection Rate : 0.4275   
## Detection Prevalence : 0.4675   
## Balanced Accuracy : 0.8898   
##   
## 'Positive' Class : TRUE   
##

#library(dplyr)  
#nrow(filter(Carseats,Carseats$Sales\_factor=="TRUE"))  
#nrow(filter(Carseats,Carseats$Sales\_factor=="FALSE"))

### Interpretation :

**Data:** Created a new column with factor of Sales > median. Used the Sales factor as response to see the sales high and low from median.

Predicted model on carseats data set. Constructed a ROC curve by plotting the true positive rate (TPR) against the false positive rate (FPR) with performance classifier method. ROC shows trade-offs between TPR and FPR and possible cutoff values, which are the probabilities assigned to each observation. Classifiers that give curves close to top-left corner indicates better performance.

Used Positive = “TRUE” in confusion matrix to identify true positive rate (positive results) for sales that are > median.

As seen in the ROC curve, if we want the prediction probability of true positive values with 70%, we will get 10% false positive values. Considering the probability threshold > 0.70 in confusion matrix for Sales to check if above or below the median, 161 observations are true positives and 191 observations are false positives. That means, 81% (sensitivity = 0.809) of Sales are predicted to be true positive (above median) in this model.

Now setting probability threashold > .90, False positive and true positives observations are increased and other 2 are decreases. In addition, model accuracy decreases. Hence, if we go toward X-axis or 90 degree angle, False positive rate will increase. That means, 66% of results are true positive and 99% of results are true negative which sounds a little unrealistic.

So lowering the probability threshold might give better results and can achieve the highest possible sensitivity while maintaining a low false positive rate.

If we predict the sales at 0.61, where the cut off of the curve starts, 185 observations are false positive(92% Speicificity) and 171 are true positive(86% Sensitivity). Accuracy of the model is 89% and error rate is 11%.

Overall, based on ROC curve, the performance of classifier in terms of misclassification counts and Error rate varies with probability cut-offs(threshold) and to evaluate/compare (two or more) such classifiers, we need to consider a suitable range of its values or fix one. For a good model, the ROC curve should rise steeply which indicates that the true positive rate (y-axis) increases faster than the false positive rate (x-axis) as the probability threshold decreases.

For this binomial model to see if location sales are above or below the median value, ROC curve rises steeply till 0.61 of y-axis and than it incline towards the x-axis until the point 1.0 of (y-axis). So the best classificaiton rule or best probability cut-off can be considered at 0.62 of y-axis. It is best to choose the cutoff value after taking into consideration the trade offs between sensitivity and specificity.

## Problem 2

Using the Default data in the ISLR package use Naive Bayes, LDA, and QDA to create classification models to predict credit defaults and compare the results. To make this comparison you should create a testing sample of 10% of each default class, and compare the results on that hold out sample only. Give any insights you may have about these models.

library(ISLR)  
library(e1071)  
library(MASS)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':  
##   
## select

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

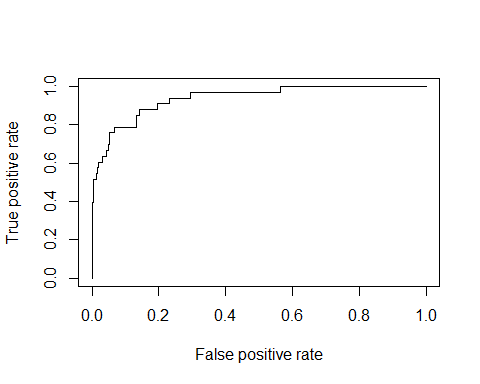
data("Default")  
  
#hold out  
library(caret)  
set.seed(2367)  
  
train\_in <- createDataPartition(Default$default,p=.90, list = FALSE, times=1)  
train\_data <- Default[train\_in,]  
test\_data <- Default[-train\_in,]  
  
#nrow(Default)  
#nrow(train\_data)  
#nrow(test\_data)  
#nrow(filter(test\_data,test\_data$default == "Yes"))  
#nrow(filter(test\_data,test\_data$default == "No"))  
  
  
## Naive Bayes  
model\_naive <- naiveBayes(default~.,data=train\_data)  
model\_naive

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## No Yes   
## 0.96667037 0.03332963   
##   
## Conditional probabilities:  
## student  
## Y No Yes  
## No 0.7085392 0.2914608  
## Yes 0.6166667 0.3833333  
##   
## balance  
## Y [,1] [,2]  
## No 803.1785 457.6149  
## Yes 1748.9940 336.6471  
##   
## income  
## Y [,1] [,2]  
## No 33510.44 13292.81  
## Yes 32018.30 13890.24

#pred.naive <- predict(model\_naive, newdata = test\_data, type="class")  
pred.naive <- predict(model\_naive, newdata = test\_data)  
nb.cm1 <- table(true = test\_data$default, predicted = pred.naive)  
nb.cm2 = confusionMatrix(nb.cm1, positive = "Yes")  
  
## LDA  
model\_LDA <- lda(default~.,data=train\_data)  
model\_LDA

## Call:  
## lda(default ~ ., data = train\_data)  
##   
## Prior probabilities of groups:  
## No Yes   
## 0.96667037 0.03332963   
##   
## Group means:  
## studentYes balance income  
## No 0.2914608 803.1785 33510.44  
## Yes 0.3833333 1748.9940 32018.30  
##   
## Coefficients of linear discriminants:  
## LD1  
## studentYes -1.624046e-01  
## balance 2.234982e-03  
## income 3.012718e-06

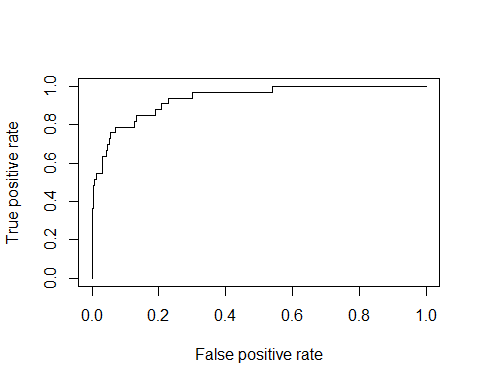
pred.lda <- predict(model\_LDA,test\_data)  
#sum(pred.lda$posterior[, 1] >= .25)  
#sum(pred.lda$posterior[, 2] > .25)  
pred.prob.lda = prediction(pred.lda$posterior[,2], test\_data$default)  
p1.lda = performance(pred.prob.lda,"tpr","fpr")  
plot(p1.lda)



lda.cm1 <- table(test\_data$default, pred.lda$class)  
lda.cm2 = confusionMatrix(lda.cm1, positive = "Yes")  
  
## QDA  
model\_QDA <- qda(default~.,data=train\_data)  
model\_QDA

## Call:  
## qda(default ~ ., data = train\_data)  
##   
## Prior probabilities of groups:  
## No Yes   
## 0.96667037 0.03332963   
##   
## Group means:  
## studentYes balance income  
## No 0.2914608 803.1785 33510.44  
## Yes 0.3833333 1748.9940 32018.30

pred.qda <- predict(model\_QDA,test\_data)  
pred.prob.qda = prediction(pred.qda$posterior[,2], test\_data$default)  
p1.qda = performance(pred.prob.qda,"tpr","fpr")  
plot(p1.qda)



qda.cm1 <- table(test\_data$default, pred.qda$class)  
qda.cm2 = confusionMatrix(qda.cm1, positive = "Yes")  
  
# Confusion Matrix  
list(title = "Confusion Matrix Table",  
 LDA\_Model = lda.cm2$table,  
 QDA\_Model = qda.cm2$table,  
 naiveBayes\_Model = nb.cm2$table)

## $title  
## [1] "Confusion Matrix Table"  
##   
## $LDA\_Model  
##   
## No Yes  
## No 965 1  
## Yes 23 10  
##   
## $QDA\_Model  
##   
## No Yes  
## No 964 2  
## Yes 22 11  
##   
## $naiveBayes\_Model  
## predicted  
## true No Yes  
## No 959 7  
## Yes 23 10

# Prediction Eval  
list(title = "Prediction Eval",  
 LDA\_model = lda.cm2$table %>% prop.table() %>% round(3),  
 QDA\_model = qda.cm2$table %>% prop.table() %>% round(3),  
 NaiveBayes\_Model = nb.cm2$table %>% prop.table() %>% round(3))

## $title  
## [1] "Prediction Eval"  
##   
## $LDA\_model  
##   
## No Yes  
## No 0.966 0.001  
## Yes 0.023 0.010  
##   
## $QDA\_model  
##   
## No Yes  
## No 0.965 0.002  
## Yes 0.022 0.011  
##   
## $NaiveBayes\_Model  
## predicted  
## true No Yes  
## No 0.960 0.007  
## Yes 0.023 0.010

# Error Rate  
test\_data %>%  
 mutate(lda.pred = (pred.lda$class),  
 qda.pred = (pred.qda$class),  
 naive.pred = (pred.naive)) %>%   
 summarize(lda.error = mean(default != lda.pred),  
 qda.error = mean(default != qda.pred),  
 naive.error = mean(default != naive.pred))

## lda.error qda.error naive.error  
## 1 0.02402402 0.02402402 0.03003003

# Precision and other values  
  
list(title = "Precision, Sensitivity, Specificity",  
 LDA\_model = lda.cm2$byClass,  
 QDA\_model = qda.cm2$byClass,  
 naiveBayes\_model = nb.cm2$byClass)

## $title  
## [1] "Precision, Sensitivity, Specificity"  
##   
## $LDA\_model  
## Sensitivity Specificity Pos Pred Value   
## 0.90909091 0.97672065 0.30303030   
## Neg Pred Value Precision Recall   
## 0.99896480 0.30303030 0.90909091   
## F1 Prevalence Detection Rate   
## 0.45454545 0.01101101 0.01001001   
## Detection Prevalence Balanced Accuracy   
## 0.03303303 0.94290578   
##   
## $QDA\_model  
## Sensitivity Specificity Pos Pred Value   
## 0.84615385 0.97768763 0.33333333   
## Neg Pred Value Precision Recall   
## 0.99792961 0.33333333 0.84615385   
## F1 Prevalence Detection Rate   
## 0.47826087 0.01301301 0.01101101   
## Detection Prevalence Balanced Accuracy   
## 0.03303303 0.91192074   
##   
## $naiveBayes\_model  
## Sensitivity Specificity Pos Pred Value   
## 0.58823529 0.97657841 0.30303030   
## Neg Pred Value Precision Recall   
## 0.99275362 0.30303030 0.58823529   
## F1 Prevalence Detection Rate   
## 0.40000000 0.01701702 0.01001001   
## Detection Prevalence Balanced Accuracy   
## 0.03303303 0.78240685

### Interpretation :

**Naive Bayes method :** Based on the summary of Naive Bayes model, prior probabilities are No = 0.966 and Yes = 0.033. Output also provides Conditional probabilties for each predictor in each class with mean and standard deviation with column 1 and 2 respectively of the outcome variable marked as Y.

**LDA Model :** Based on the summary of LDA model, prior probabilities are No = 0.966 and yes = 0.033 which means 96.6% of the observations on training set are customers who are not defaulters and 3% are defaulted. Output also provides the group means which predict within each class.

**QDA Model :** Summary of QDA is same as LDA on the training set. Summary contains the prior probabilities and the group means. But it does not contain the coefficients of the linear discriminants which was in LDA because this is Quatratic model for prediction and not linear.

DId prediction for all three models on test data and constructed the ROC curve for LDA and QDA.

**Prediction Comparisons/ Confusion Matrix :** Displayed prediction/confusion matrix in percentage form for all three models. Results shows that all models perform in a very similar manner. 96% of the observations are predicted as true negatives and about 1% are true positives. All three Models predicted 3% customers will not default but they actually did. Models predicted Less than 1% of customers will default but they never did.

**Error Rates and other Comparisons :** Estimated error rates for all three models. It is seen in the table that error rate for naive bayes model is slightly less than LDA and QDA models. If we look at the precision for each model, LDA model has a slightly low precision than the QDA and naive Bayes model. We can increase the precision of the model by adjusting the posterior probability threshold with considering the sensitivity and specificity too. ROC curves looks almost similar for LDA and QDA models. LDA model have probability cut off just below 0.25 on y-axis and QDA model have probability cut off at around 0.25 on y-axis. Based on the statistics, naive bayes perform good with error rates. But if we consider the precision, LDA is better. LDA and QDA can be used with continous variables while naive bayes can only be used with categorial variables. LDA and QDA are giving good assumptions based on confusion matrix that predicted true negatives and true positives. We can decrease the prior probability cutoff from .5 (this the default considered) to 0.25 and can get better prediction for true positives and true negatives as per ROC curves of LDA and QDA.