ActSc 632 Assignment 2 Solutions

Spring 2023

Department of Statistics and Actuarial Science, University of Waterloo

Question 1

Determine the number of predictors in this data set, and whether they are quantitative of qualita- tive. How many observations are there? How many observations are classified as a “bad credit”? “good credit”?

```r
library(tree)
library(MASS)
library(CASdatasets)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sp
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)
```

```r
library(pROC)
```

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

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

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

```r
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

```r
library(class)


# Load the data
data(credit)

within(credit,{
installment_rate <-factor(installment_rate)
residence_since <- factor(residence_since)
existing_credits <- factor(existing_credits)
num_dependents <- factor(num_dependents)
})
```

```r
summary(credit)
```

```
##  checking_status    duration    credit_history    purpose    credit_amount
##  A11:274         Min.   : 4.0   A30: 40        A43    :280   Min.   :  250
##  A12:269         1st Qu.:12.0   A31: 49        A40    :234   1st Qu.: 1366
##  A13: 63         Median :18.0   A32:530        A42    :181   Median : 2320
##  A14:394         Mean   :20.9   A33: 88        A41    :103   Mean   : 3271
##                  3rd Qu.:24.0   A34:293        A49    : 97   3rd Qu.: 3972
##                  Max.   :72.0                  A46    : 50   Max.   :18424
##                                                (Other): 55
##  savings   employment installment_rate personal_status other_parties
##  A61:603   A71: 62    Min.   :1.000    A91: 50         A101:907
##  A62:103   A72:172    1st Qu.:2.000    A92:310         A102: 41
##  A63: 63   A73:339    Median :3.000    A93:548         A103: 52
##  A64: 48   A74:174    Mean   :2.973    A94: 92
##  A65:183   A75:253    3rd Qu.:4.000
##                       Max.   :4.000
##
##  residence_since property_magnitude      age        other_payment_plans
##  Min.   :1.000   A121:282           Min.   :19.00   A141:139
##  1st Qu.:2.000   A122:232           1st Qu.:27.00   A142: 47
##  Median :3.000   A123:332           Median :33.00   A143:814
##  Mean   :2.845   A124:154           Mean   :35.55
##  3rd Qu.:4.000                      3rd Qu.:42.00
##  Max.   :4.000                      Max.   :75.00
##
##  housing    existing_credits   job      num_dependents  telephone
##  A151:179   Min.   :1.000    A171: 22   Min.   :1.000   A191:596
##  A152:713   1st Qu.:1.000    A172:200   1st Qu.:1.000   A192:404
##  A153:108   Median :1.000    A173:630   Median :1.000
##             Mean   :1.407    A174:148   Mean   :1.155
##             3rd Qu.:2.000               3rd Qu.:1.000
##             Max.   :4.000               Max.   :2.000
##
##  foreign_worker     class
##  A201:963       Min.   :0.0
##  A202: 37       1st Qu.:0.0
##                 Median :0.0
##                 Mean   :0.3
##                 3rd Qu.:1.0
##                 Max.   :1.0
##
```

```r
print(c(sum(credit$class == 1),
        sum(credit$class == 0)))
```

```
## [1] 300 700
```

There are 20 predictors. Duration, credit amount, age are quantitative, and the other are qualitative. Depending on the encoding used for the data set (which differs from source to source) the predictors installment rate, residence since, existing credits and num dependents may be encoded as integers and thus be a priori be considered quantitative. But when looking at the meaning of the different values these predictors can take, it seems best to treat them as categorical. It won't matter in our analysis though, as these 4 predictors do not end up being considered in our models. and the others are categorical. There are 300 observations out of 1000 classified as bad credit.

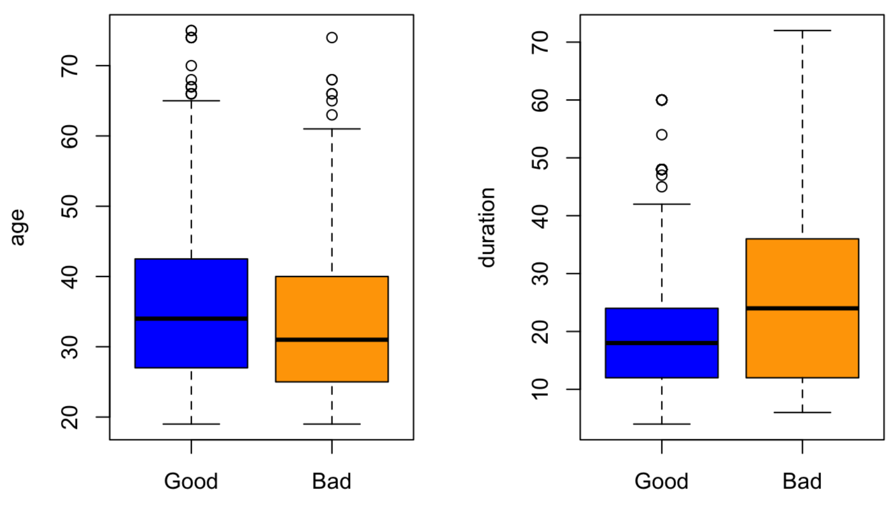
Question 2

To explore the data further, produce the following plots:

- for each of the predictors age and duration, make a box plot showing the distribution of the observations, separately for the “good” and “bad” observations;
- for the pairs duration & savings and duration & credit history, plot the observations (using duration on the x axis) and use different symbols for the “good” and “bad” observations.

Comments on the plots you obtained.

```
par(mfrow=c(1,2))

# Create a box plot of 'age' for the two classes, displayed inline
boxplot(credit$age[credit$class==0], credit$age[credit$class==1],
        names=c("Good","Bad"), ylab = "age", col=c("blue","orange"))

# Create a box plot of 'duration' for the two classes, displayed inline
boxplot(credit$duration[credit$class==0], credit$duration[credit$class==1],
        names=c("Good","Bad"), ylab = "duration", col=c("blue","orange"))
```



Clearly the bad credit observations seems to have a longer duration, and the distribution of the duration has a larger variance; they also seem to be slightly younger compared to the good credit.

Question 3

Randomly split your data in 80% of the observations for training and 20% for testing.

```
set.seed(1)
train <- sample(1:nrow(credit), 0.8 * nrow(credit))

credit$class <- as.factor(credit$class)

credit.train<-credit[train, ]
credit.test<-credit[-train, ]
```

Question 4

For this sub-question, you should work with the following predictors: age, duration, purpose, credit history, and savings. For each of

- logistic regression,
- linear discriminant analysis,
- quadratic discriminant analysis,

do the following:

a. Use the training data to build each classifier and then use the test data set to determine the confusion matrix, the overall error rate, Type-I error, Type-II error.
b. Plot the ROC curve for the three methods considered and determine the AUC (area under the curve).

```
myerr<-function(tab){
E=c(0,0,0)
E[1]<-(tab[1,2]+tab[2,1])/sum(tab[,])
E[2]<-tab[2,1]/sum(tab[,1])
E[3]<-tab[1,2]/sum(tab[,2])
return(E)
}

#logistic
log.5<-glm(class~age+duration+purpose+credit_history+savings,data=credit.train,family="binomial")

probLogReg5<-predict(log.5,newdata=credit.test,type="response")
confusion.log5<-table(probLogReg5>0.5, credit.test$class)

myerr(confusion.log5)
```

```
## [1] 0.24500000 0.05263158 0.62686567
```

```
##lda
lda.5<-lda(class~age+duration+purpose+credit_history+savings,data=credit.train)
conf5.lda<-table(predict(lda.5, newdata=credit.test)$class, credit.test$class)
myerr(conf5.lda)
```

```
## [1] 0.24000000 0.03759398 0.64179104
```

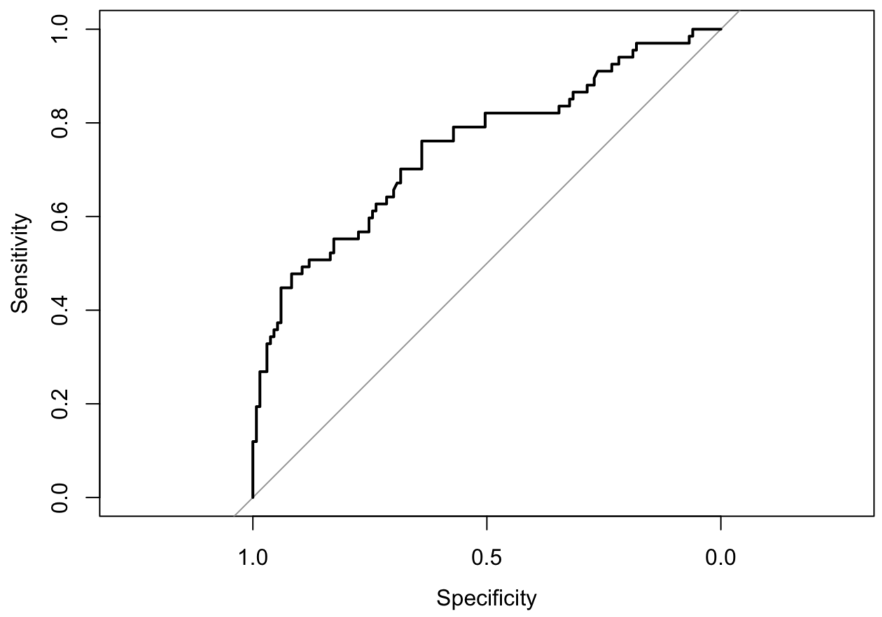
```
#qda
qda.5<-qda(class~age+duration+purpose+credit_history+savings,data=credit.train)
conf5.qda<-table(predict(qda.5, newdata=credit.test)$class, credit.test$class)
myerr(conf5.qda)
```

```
## [1] 0.3150000 0.2556391 0.4328358
```

```
#ROC curves

roc.log5<-roc(credit.test$class,probLogReg5,plot=TRUE)
```

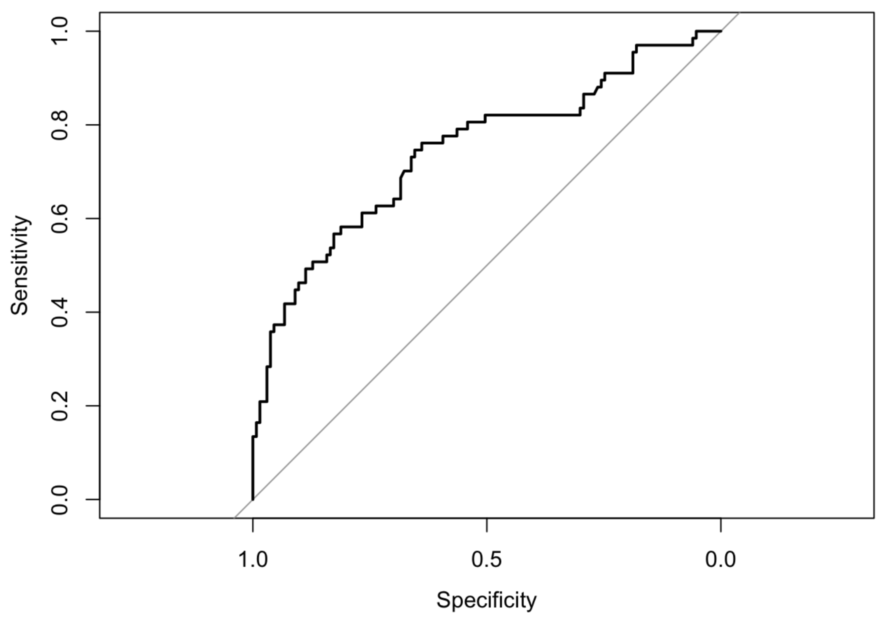
```
## Setting levels: control = 0, case = 1
```

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



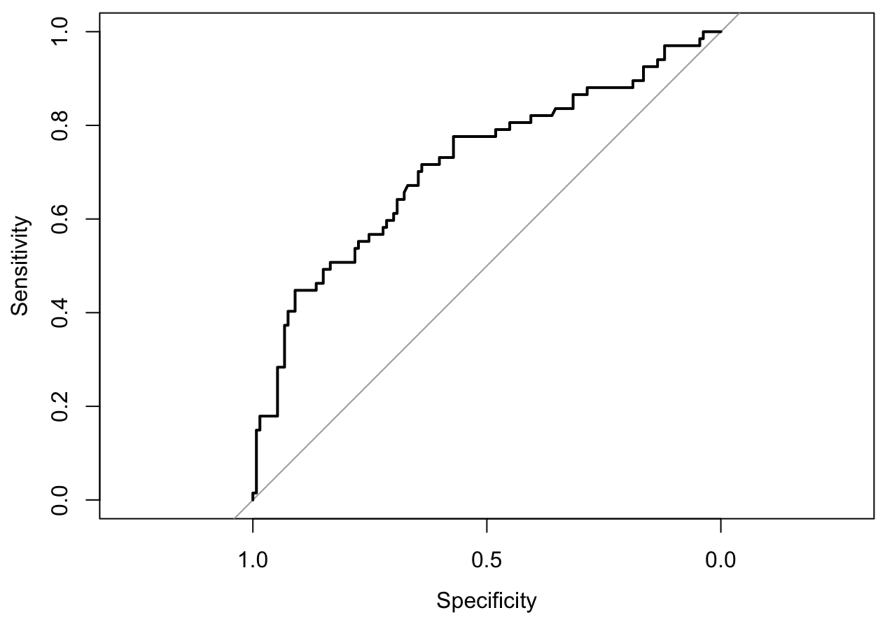
```
roc.lda5<-roc(credit.test$class,predict(lda.5, newdata=credit.test)$posterior[,2],plot=TRUE)
```

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



```
roc.qda5<-roc(credit.test$class,predict(qda.5, newdata=credit.test)$posterior[,2],plot=TRUE)
```

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



Question 5

Next you wish to use the K nearest neighbours (KNN) as a classifier for this problem, using the three predictors age, duration and credit amount. Apply the KNN approach for each of K = 1, 3, 5, and then for each value of K, use the test set to produce the confusion matrix, determine the overall error rate, Type-I error and Type-II error. Which choice of K seems the best?

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```
## Loading required package: lattice
```

```
y<-c(2,5,13)
credit.train.knn<-credit.train[,y]
credit.test.knn <- credit.test[,y]
for(i in 1:length(y)){
credit.train.knn[,i]<-as.numeric(credit.train[,y[i]])
credit.test.knn[,i]<-as.numeric(credit.test[,y[i]])
}

train_data_means <- apply(credit.train.knn, 2, mean)
train_data_sds <- apply(credit.train.knn, 2, sd)

train.X <- scale(credit.train.knn, center = train_data_means, scale = train_data_sds)
test.X  <- scale(credit.test.knn, center = train_data_means, scale = train_data_sds)

train.Y<-credit.train$class
test.Y<-credit.test$class

knn.k1<-knn(train.X,test.X,train.Y,k=1)
knn.tab1<-table(knn.k1,test.Y)
myerr(knn.tab1)
```

```
## [1] 0.4150000 0.3308271 0.5820896
```

```
knn.k3<-knn(train.X,test.X,train.Y,k=3)
knn.tab3<-table(knn.k3,test.Y)
myerr(knn.tab3)
```

```
## [1] 0.4050000 0.2255639 0.7611940
```

```
knn.k5<-knn(train.X,test.X,train.Y,k=5)
knn.tab5<-table(knn.k5,test.Y)
myerr(knn.tab5)
```

```
## [1] 0.3550000 0.1578947 0.7462687
```

The kNN with K=5 provides the best overall classification accuracy. However, one should consider smaller values of K when minimizing Type-II error is the top priority.

Question 6

Simply using recursive binary partitioning, obtain a tree for this classifcation problem.

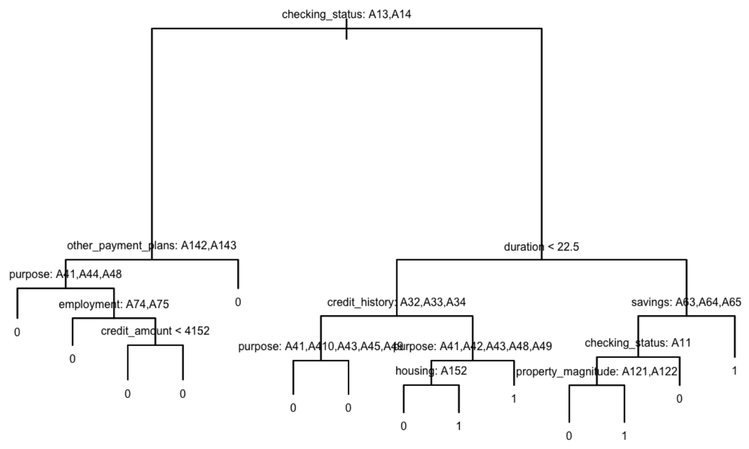
- How many leaves does your tree have?
- How many factors were used to build this tree?
- What is the deviance for this tree? (If you used something else than the default definition of deviance in R, please specify how is deviance determined).
- Plot the tree you obtained using R. There should be enough information that given an observation, one could determine in which leaf it ends up.
- Use your tree to make predictions for the test data set. Produce the confusion matrix corresponding to your tree and plot the ROC curve.

```
tree.credit<-tree(class~.,data=credit.train)
summary(tree.credit)
```

```
##
## Classification tree:
## tree(formula = class ~ ., data = credit.train)
## Variables actually used in tree construction:
##  [1] "checking_status"      "other_payment_plans" "purpose"
##  [4] "employment"           "credit_amount"       "duration"
##  [7] "credit_history"       "housing"             "savings"
## [10] "property_magnitude"
## Number of terminal nodes:  14
## Residual mean deviance:  0.9012 = 708.3 / 786
## Misclassification error rate: 0.2125 = 170 / 800
```

So there are 11 leaves and 7 factors were used. The deviance is 0.9321. Note that since the tree will depend on the training set, which is randomly chosen, you may have obtained a tree that has a quite different structure, with a different subset of factors used, and a different number of terminal nodes. As mentioned in class, this method has a high variance, which is why the results can be so different.

```
plot(tree.credit)
text(tree.credit,pretty=0,cex=0.5)
```



We recall that the categories listed on a node are those used to determine which observations go in the left child.

```
pred.tree.cred<-predict(tree.credit,newdata=credit.test,type="class")

tree.tab<-table(pred.tree.cred,credit.test$class)

myerr(tree.tab)
```

```
## [1] 0.2800000 0.1353383 0.5671642
```

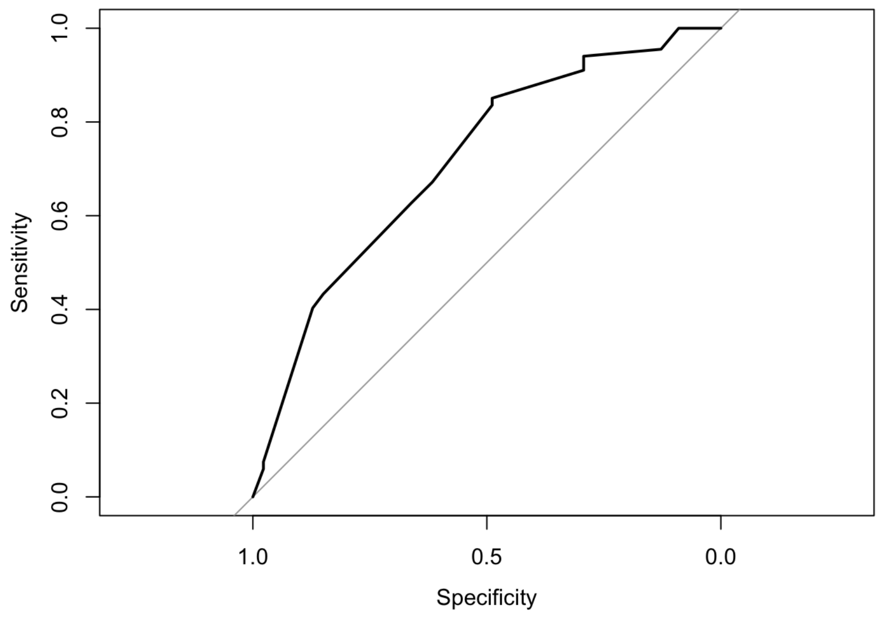
```
tree.dev.pred <- predict(tree.credit,newdata=credit.test,type="tree")

print(c("Deviance:", deviance(tree.dev.pred)))
```

```
## [1] "Deviance:"        "266.232606602815"
```

```
roc(credit.test$class,predict(tree.credit,newdata=credit.test,type="vector")[,2],plot=TRUE)
```

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

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



```
##
## Call:
## roc.default(response = credit.test$class, predictor = predict(tree.credit,     newdata = credit.test, type =
"vector")[, 2], plot = TRUE)
##
## Data: predict(tree.credit, newdata = credit.test, type = "vector")[, 2] in 133 controls (credit.test$class 0)
< 67 cases (credit.test$class 1).
## Area under the curve: 0.7164
```

The ROC curve has AUC of 0.7124.

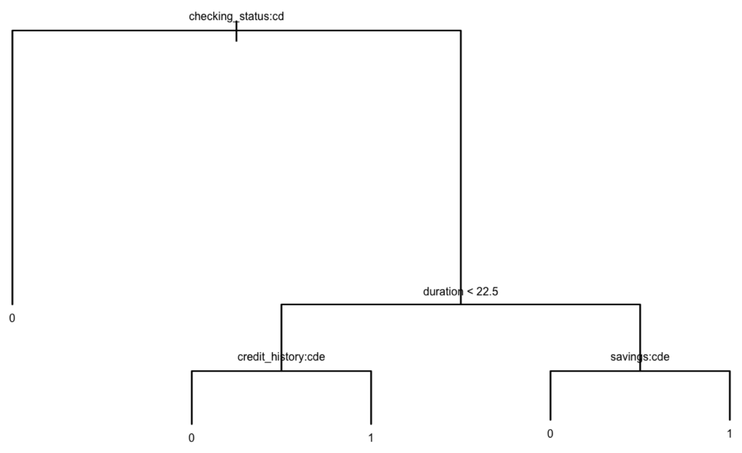
Question 7

Now try to use pruning to see if you can improve your results.

```
cv.credit<-cv.tree(tree.credit,FUN=prune.misclass)
print(cv.credit)
```

```
## $size
## [1] 14  9  7  5  4  1
##
## $dev
## [1] 205 205 203 207 228 244
##
## $k
## [1]     -Inf  0.00000  2.50000  3.50000 11.00000 13.33333
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
prune.credit<-prune.misclass(tree.credit,best=5)
plot(prune.credit)
text(prune.credit,cex=0.5)
```



```
summary(prune.credit)
```

```
##
## Classification tree:
## snip.tree(tree = tree.credit, nodes = c(2L, 12L, 13L, 14L))
## Variables actually used in tree construction:
## [1] "checking_status" "duration"        "credit_history"  "savings"
## Number of terminal nodes:  5
## Residual mean deviance:  1.027 = 816.2 / 795
## Misclassification error rate: 0.2275 = 182 / 800
```

```
pred.prune.cred<-predict(prune.credit,credit.test,type="class")
prune.tab<-table(pred.prune.cred,credit.test$class)

myerr(prune.tab)
```

```
## [1] 0.2800000 0.1127820 0.6119403
```

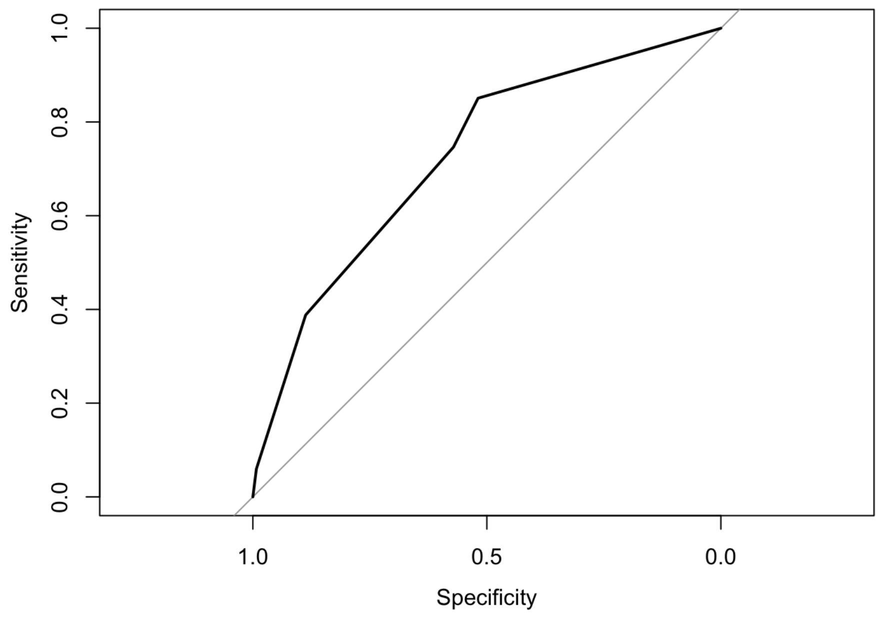
```
prtree.dev.pred<- predict(prune.credit,newdata=credit.test,type="tree")
print(c("Deviance:", deviance(prtree.dev.pred)))
```

```
## [1] "Deviance:"        "224.792077637098"
```

```
pred.prune.cred.prob<-predict(prune.credit,credit.test)
roc.prune.cred<-roc(credit.test$class,pred.prune.cred.prob[,2],plot=TRUE)
```

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

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



```
print(c("AUC", roc.prune.cred$auc))
```

```
## [1] "AUC"               "0.725002805521266"
```

The overall error has slightly improved. The deviance has improved to 323.3462, and the ROC curve now has AUC of 0.7383.

Question 8

Now try bagging and random forests to see if you can improve your results.

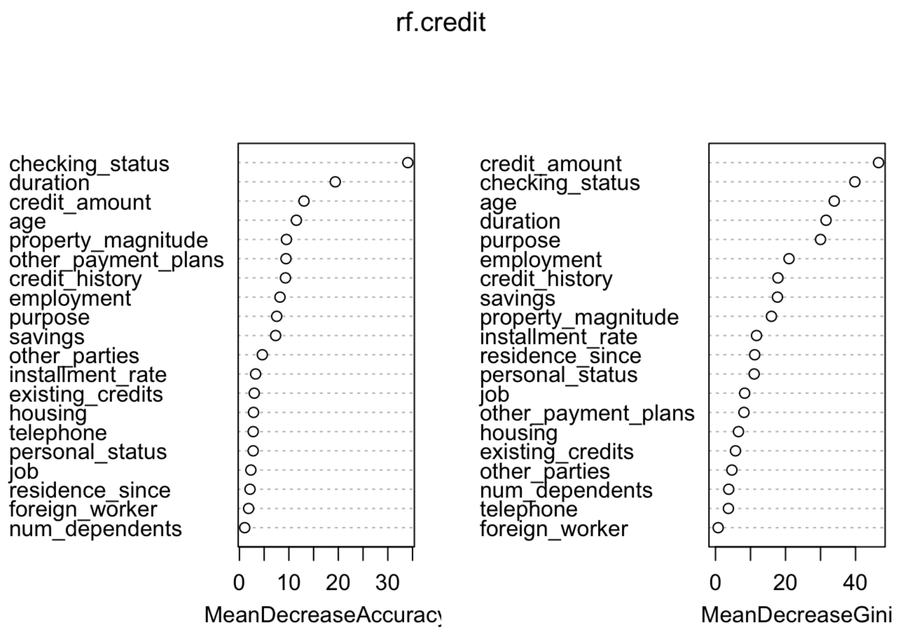
```
set.seed(5)
fullbag.credit<-randomForest(class~.,data=credit.train,mtry=20,importance=TRUE)
yhat.bag<-predict(fullbag.credit,newdata=credit.test)
bag.tab<-table(yhat.bag,credit.test$class)

bag.tab
```

```
##
## yhat.bag   0   1
##        0 116  34
##        1  17  33
```

```
rf.credit<-randomForest(class~.,data=credit.train,mtry=10,importance=TRUE)

yhat.rf<-predict(rf.credit,newdata=credit.test)
rf.tab<-table(yhat.rf,credit.test$class)

rf.tab
```

```
##
## yhat.rf   0   1
##       0 122  35
##       1  11  32
```

```
varImpPlot(rf.credit)
```



We see that whether we use the accuracy (prediction error) or Gini index, the predictor checking status seems very important, as well as duration. Other important predictors are credit history and savings, purpose, age and credit amount. These are pretty consistent with the (limited set of) predictors that were used to construct the pruned tree.

Question 9

Conclude by making a suggestion as to which of the above methods is the best for this problem.

```
res.table <- data.frame(matrix(ncol = 4, nrow = 4))

colnames(res.table) <- c("Method", "Overall error", "Type I error", "Type II error")

res.table[1,1] <- "Single Tree"
res.table[1, 2:4] <- myerr(tree.tab)

res.table[2,1] <- "Pruned Tree"
res.table[2, 2:4] <- myerr(prune.tab)

res.table[3,1] <- "Bagging"
res.table[3, 2:4] <- myerr(bag.tab)

res.table[4,1] <- "Random Forest"
res.table[4, 2:4] <- myerr(rf.tab)

res.table
```

```
##          Method Overall error Type I error Type II error
## 1   Single Tree         0.280   0.13533835     0.5671642
## 2   Pruned Tree         0.280   0.11278195     0.6119403
## 3       Bagging         0.255   0.12781955     0.5074627
## 4 Random Forest         0.230   0.08270677     0.5223881
```

Based on our results, random forest has the best overall prediction error and Type-I error, and bagging has the best Type-II error. Given that bagging and random forest have much less variance, our assessment of their performance is much more reliable than for the single tree and pruned tree and as such, we would recommend using one of those two methods.