ISYE-6501
Fall 2024

Homework 7

Question 10.1

Using the same crime data set uscrime.txt as in Question 8.2 and 9.1, find the best model you can using

a) a regression tree model, and

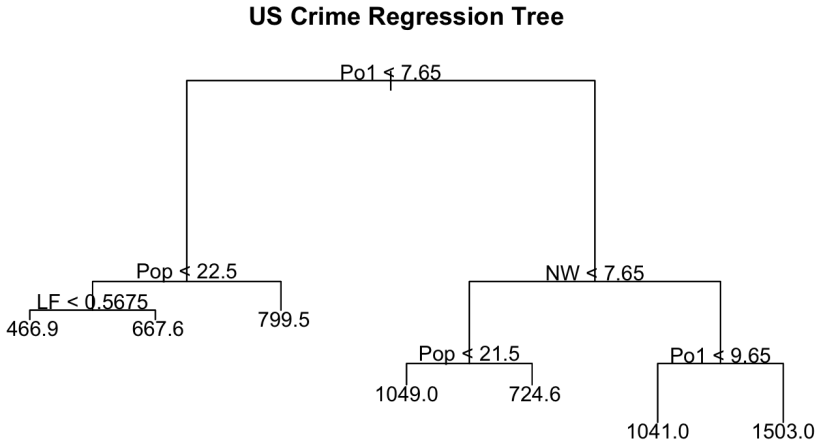
TAKEAWAYS:

After running the tree function on the unpruned tree with 7 leafs and the pruned tree with 5 leafs, we can see that the quality of fit of the unpruned tree is better. The reason is because when looking at the summary of the two trees, the residual mean deviance of the unpruned tree is lower (47390) compared to that of the pruned tree (54210). Also after calculating the R-squared of the two trees we can see that the unpruned tree has a higher R-squared of 0.724 compared to the pruned tree with an R-squared of 0.669.

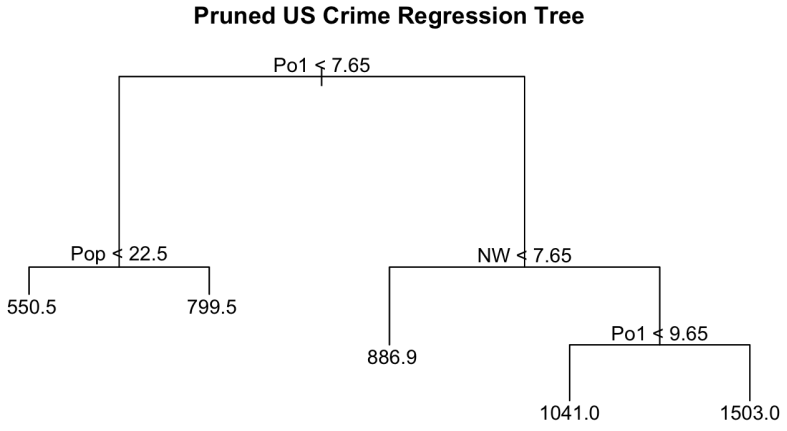
Below is the code/ R markdown, find the input in black, the comments in green, and the output in blue.

CODE:

```
> #load the data uscrime data into a table named data.
> data <- read.table("uscrime.txt", header = TRUE)
> #load the correct packages used
> library(randomForest)
> library(tree)
> library(caret)
> #apply the tree function to create a regression tree using the uscrime dataset and check its contents.
> crimetree <- tree(Crime~.,data=data)
> summary(crimetree)
Regression tree:
tree(formula = Crime ~ ., data = data)
Variables actually used in tree construction:
[1] "Po1" "Pop" "LF"  "NW"
Number of terminal nodes:  7
Residual mean deviance:  47390 = 1896000 / 40
Distribution of residuals:
   Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-573.900  -98.300   -1.545    0.000  110.600  490.100
> #then we plot the tree
> plot(crimetree)
> text(crimetree)
> title("US Crime Regression Tree")
```




```
> #prune the regression tree, analyze and plot
> prunedcrimetree <- prune.tree(crimetree, best = 5)
> summary(prunedcrimetree)
Regression tree:
snip.tree(tree = crimetree, nodes = c(4L, 6L))
Variables actually used in tree construction:
[1] "Po1" "Pop" "NW"
Number of terminal nodes:  5
Residual mean deviance:  54210 = 2277000 / 42
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -573.9  -107.5    15.5     0.0   122.8   490.1
> plot(prunedcrimetree)
> text(prunedcrimetree)
> title("Pruned US Crime Regression Tree")
```




```
> #Let's calculate the quality of fit of the unpruned tree.
> crimetreepred <- predict(crimetree, data=data[,1:15])
> RSS <- sum((crimetreepred-data[,16])^2)
> TSS <- sum((data[,16]-mean(data[,16]))^2)
```

```
> unprunedR2 <- 1-RSS/TSS
> unprunedR2
[1] 0.7244962
> #Now calculate the quality of the pruned tree.
> prunedcrimetreepred <- predict(prunedcrimetree, data=data[,1:15])
> RSS <- sum((prunedcrimetreepred-data[,16])^2)
> TSS <- sum((data[,16]-mean(data[,16]))^2)
> prunedR2 <- 1-RSS/TSS
> prunedR2
[1] 0.6691333
```

b) a random forest model.

TAKEAWAYS:

After running the randomForest function on the data set, and testing the function with multiple nodesizes, I found that the highest R-squared value was 0.425 with a node size of 5. We now can see that for this data set, using a forest model is not as accurate as using the regression tree. This is possibly due to the forest model overfitting because of the relatively smaller size of the data set.

CODE:

```
> #Now we move onto the forest model.
> crimeforest <- randomForest(Crime~., data=data, importance=TRUE, nodesize = 5)
> crimeforestpred <- predict(crimeforest, data=data[,16])
> RSS <- sum((crimeforestpred-data[,16])^2)
> crimeforestR2 <- 1-RSS/TSS
> crimeforestR2
[1] 0.4248584
```

Question 10.2

Describe a situation or problem from your job, everyday life, current events, etc., for which a logistic regression model would be appropriate. List some (up to 5) predictors that you might use.

As a data analyst for a hospital, a logistic regression model would be appropriate for determining whether or not a patient will be readmitted within 30 days of discharge. We try to minimize readmissions within 30 days because that could be a sign that there was insufficient care done in the initial visit or that a diagnosis was missed. Please find below 5 predictors that I would use.

- Age (The older the patient the higher the chance of readmission)
- Length of stay (higher length of stay could be an indicator of severe conditions)
- Comorbidities Index (A score that indicates the severity of comorbid conditions)
- Discharge Disposition (A patient discharge to a skilled nursing facility would have less likelihood of being readmitted compared to someone who is discharged to home)
- Location of admission (Patient in the intensive care unit tend to have higher readmission rates)

Question 10.3

1. Using theGermanCredit data set germancredit.txt, use logistic regression to find a good predictive model for whether credit applicants are good credit risks or not. Show your model (factors used and their coefficients), the software output, and the quality of fit. You can use the glm function in R. To get a logistic regression (logit) model on data where the response is either zero or one.

2. Because the model gives a result between 0 and 1, it requires setting a threshold probability to separate between "good" and "bad" answers. In this data set, they estimate that incorrectly identifying a bad customer as good, is 5 times worse than incorrectly classifying a good customer as bad. Determine a good threshold probability based on your model.

TAKEAWAYS:

After creating a confusion matrix with the train data from my datatrainmodel, I found that there was a good amount of misclassification of bad borrowers and as said in the problem, misclassifying bad borrowers is 5 times worse that misclassifying a good borrower. I decided to make a new model and messed around with the threshold value and seeing the accuracy output. I found that using a threshold of 0.7 was gave the best accuracy at about 0.76. Which in turn gave us a very high specificity of 0.956 or 96.5%. This means that the model accurately predicts the negative cases 96.5% of the time. I was satisfied with the model.

CODE:

```
> #load the data uscrime data into a table named data.
> data <- read.table("germancredit.txt", header = FALSE)
> #Start off by replacing the 1,2 with 0, and 1.
> data$V21[data$V21==1] <- 0
> data$V21[data$V21==2] <- 1
> #Next we split our data into training and validation sets
> datapart <- createDataPartition(data$V21, times = 1, p = 0.7, list=FALSE)
> datatrain <- data[datapart,]
> datavalid <- data[-datapart,]
> table(datatrain$V21)

  0   1
473 227
> table(datavalid$V21)

  0   1
227  73
> #Next we create a model with the Training data
> datatrainmodel <- glm(V21~., data=datatrain, family=binomial(link="logit"))
> summary(datatrainmodel)

Call:
```

```
glm(formula = V21 ~ ., family = binomial(link = "logit"), data = datatrain)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.227e+00  1.334e+00   0.920 0.357713
V1A12       -2.092e-01  2.722e-01  -0.769 0.442125
V1A13       -6.868e-01  4.411e-01  -1.557 0.119463
V1A14       -1.837e+00  2.902e-01  -6.329 2.46e-10 ***
V2           2.857e-02  1.183e-02   2.415 0.015733 *
V3A31       -9.769e-01  7.426e-01  -1.316 0.188328
V3A32       -1.623e+00  6.253e-01  -2.595 0.009452 **
V3A33       -2.165e+00  6.652e-01  -3.254 0.001137 **
V3A34       -2.942e+00  6.393e-01  -4.601 4.20e-06 ***
V4A41       -1.685e+00  4.908e-01  -3.432 0.000599 ***
V4A410      -1.006e+00  8.678e-01  -1.159 0.246328
V4A42       -8.905e-01  3.226e-01  -2.760 0.005771 **
V4A43       -1.002e+00  2.993e-01  -3.348 0.000814 ***
V4A44        4.824e-01  8.913e-01   0.541 0.588324
V4A45       -1.729e-01  6.253e-01  -0.277 0.782143
V4A46        6.035e-02  4.655e-01   0.130 0.896856
V4A48       -1.649e+00  1.398e+00  -1.180 0.237986
V4A49       -7.363e-01  4.163e-01  -1.769 0.076944 .
V5           1.423e-04  5.675e-05   2.508 0.012154 *
V6A62       -5.312e-01  3.837e-01  -1.385 0.166154
V6A63       -7.659e-02  4.308e-01  -0.178 0.858881
V6A64       -1.189e+00  5.828e-01  -2.040 0.041326 *
V6A65       -9.217e-01  3.159e-01  -2.918 0.003527 **
V7A72        1.961e-01  5.166e-01   0.380 0.704158
V7A73        1.376e-01  5.060e-01   0.272 0.785654
V7A74       -4.348e-01  5.428e-01  -0.801 0.423071
V7A75       -3.564e-02  5.143e-01  -0.069 0.944755
V8           3.294e-01  1.058e-01   3.113 0.001854 **
V9A92       -5.835e-01  4.899e-01  -1.191 0.233634
V9A93       -1.299e+00  4.826e-01  -2.691 0.007114 **
V9A94       -7.393e-01  5.710e-01  -1.295 0.195399
V10A102      4.701e-01  4.664e-01   1.008 0.313487
V10A103     -1.242e+00  6.031e-01  -2.060 0.039389 *
V11          1.508e-02  1.068e-01   0.141 0.887721
V12A122      3.992e-02  3.088e-01   0.129 0.897145
V12A123      1.106e-01  2.866e-01   0.386 0.699517
V12A124      8.058e-01  5.178e-01   1.556 0.119669
V13         -5.874e-03  1.086e-02  -0.541 0.588613
```

```
V14A142     2.321e-01  4.902e-01   0.473 0.635875
V14A143    -6.442e-01  2.908e-01  -2.215 0.026740 *
V15A152     3.297e-02  2.872e-01   0.115 0.908623
V15A153    -7.374e-01  5.901e-01  -1.250 0.211424
V16         5.312e-01  2.312e-01   2.298 0.021572 *
V17A172     2.738e-01  7.559e-01   0.362 0.717160
V17A173     3.799e-01  7.318e-01   0.519 0.603686
V17A174     3.671e-01  7.336e-01   0.500 0.616812
V18         3.721e-02  3.017e-01   0.123 0.901866
V19A192    -4.353e-01  2.461e-01  -1.769 0.076896 .
V20A202    -1.650e+00  8.087e-01  -2.041 0.041298 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 882.08  on 699  degrees of freedom
Residual deviance: 614.40  on 651  degrees of freedom
AIC: 712.4

Number of Fisher Scoring iterations: 5

> #Now we predict using the validation data
> datapredict <- predict(datatrainmodel, newdata=datavalid[,-21], type = "response")
> table(datavalid$V21, round(datapredict))

    0   1
 0 192  35
 1  42  31
> #Now we must remove the unsignficant variables from the training and validation data due to
misclassification of bad borrowers.
> datatrain$V1A14[datatrain$V1 == "A14"] <- 1
> datatrain$V1A14[datatrain$V1 != "A14"] <- 0
> datatrain$V3A34[datatrain$V3 == "A34"] <- 1
> datatrain$V3A34[datatrain$V3 != "A34"] <- 0
> datatrain$V4A41[datatrain$V4 == "A41"] <- 1
> datatrain$V4A41[datatrain$V4 != "A41"] <- 0
> datatrain$V4A43[datatrain$V4 == "A43"] <- 1
> datatrain$V4A43[datatrain$V4 != "A43"] <- 0
> datavalid$V1A14[datavalid$V1 == "A14"] <- 1
> datavalid$V1A14[datavalid$V1 != "A14"] <- 0
> datavalid$V3A34[datavalid$V3 == "A34"] <- 1
```

```
> datavalid$V3A34[datavalid$V3 != "A34"] <- 0
> datavalid$V4A41[datavalid$V4 == "A41"] <- 1
> datavalid$V4A41[datavalid$V4 != "A41"] <- 0
> datavalid$V4A43[datavalid$V4 == "A43"] <- 1
> datavalid$V4A43[datavalid$V4 != "A43"] <- 0
> #create new model
> datatrainmodel2 <- glm(V21 ~ V1A14+V2+V3A34+V4A41+V4A43, data = datatrain,
family=binomial(link="logit"))
> summary(datatrainmodel2)

Call:
glm(formula = V21 ~ V1A14 + V2 + V3A34 + V4A41 + V4A43, family = binomial(link = "logit"),
    data = datatrain)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.600818   0.203534  -2.952  0.00316 **
V1A14       -1.632865   0.216501  -7.542 4.63e-14 ***
V2           0.040983   0.007849   5.222 1.77e-07 ***
V3A34       -0.954892   0.227669  -4.194 2.74e-05 ***
V4A41       -1.185566   0.380482  -3.116  0.00183 **
V4A43       -0.670907   0.216216  -3.103  0.00192 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 882.08  on 699  degrees of freedom
Residual deviance: 721.84  on 694  degrees of freedom
AIC: 733.84

Number of Fisher Scoring iterations: 5

> #now create a new confusion matrix on the modified data sets
> datapredict2 <- predict(datatrainmodel2, newdata=datavalid[,-21], type="response")
> confmatrix <- as.matrix(table(round(datapredict2), datavalid$V21))
> names(dimnames(confmatrix)) <- c("Model", "True")
> confmatrix
     True
Model   0   1
    0 189  39
    1  38  34
```

```
> #Now going to mess with the threshold to find the best value.
> confmatrix2 <- as.matrix(table(round(datapredict2 > 0.7), datavalid$V21))
> names(dimnames(confmatrix2)) <- c("Model", "True")
> confmatrix2
    True
Model   0   1
   0 217  63
   1  10  10
> #Now calculate the specificity and accuracy
> specificity <- (confmatrix2[1,1])/(confmatrix2[1,1]+confmatrix2[2,1])
> specificity
[1] 0.9559471
> accuracy <-
(confmatrix2[1,1]+confmatrix2[2,2])/(confmatrix2[1,1]+confmatrix2[1,2]+confmatrix2[2,1]+confmatrix2[
2,2])
> accuracy
[1] 0.7566667
```