Project 7

Marcus McKenzie

7.1

Using the crime data set uscrime.txt, find the best model you can using (a) a regression tree model, and (b) a random forest model. In R, you can use the tree package or the r part package, and the randomForest package. For each model, describe one or two qualitative takeaways you get from analyzing the results.

Load Libraries:

Load data:

```
#data <- read.table("C:/Users/katie/Documents/Analytics/uscrime.txt", header = TRUE, stringsAsFactors =
data <- read.table("uscrime.txt", header = TRUE, stringsAsFactors = FALSE)

data
```

```
##       M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 1  15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2  14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3  14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4  13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5  14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6  12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
## 7  12.7  1 11.1  8.2  7.9 0.519  98.2   4 13.9 0.097 3.8   6200 16.8 0.042100
## 8  13.1  1 10.9 11.5 10.9 0.542  96.9  50 17.9 0.079 3.5   4720 20.6 0.040099
## 9  15.7  1  9.0  6.5  6.2 0.553  95.5  39 28.6 0.081 2.8   4210 23.9 0.071697
## 10 14.0  0 11.8  7.1  6.8 0.632 102.9   7  1.5 0.100 2.4   5260 17.4 0.044498
## 11 12.4  0 10.5 12.1 11.6 0.580  96.6 101 10.6 0.077 3.5   6570 17.0 0.016201
## 12 13.4  0 10.8  7.5  7.1 0.595  97.2  47  5.9 0.083 3.1   5800 17.2 0.031201
## 13 12.8  0 11.3  6.7  6.0 0.624  97.2  28  1.0 0.077 2.5   5070 20.6 0.045302
## 14 13.5  0 11.7  6.2  6.1 0.595  98.6  22  4.6 0.077 2.7   5290 19.0 0.053200
## 15 15.2  1  8.7  5.7  5.3 0.530  98.6  30  7.2 0.092 4.3   4050 26.4 0.069100
## 16 14.2  1  8.8  8.1  7.7 0.497  95.6  33 32.1 0.116 4.7   4270 24.7 0.052099
## 17 14.3  0 11.0  6.6  6.3 0.537  97.7  10  0.6 0.114 3.5   4870 16.6 0.076299
## 18 13.5  1 10.4 12.3 11.5 0.537  97.8  31 17.0 0.089 3.4   6310 16.5 0.119804
## 19 13.0  0 11.6 12.8 12.8 0.536  93.4  51  2.4 0.078 3.4   6270 13.5 0.019099
## 20 12.5  0 10.8 11.3 10.5 0.567  98.5  78  9.4 0.130 5.8   6260 16.6 0.034801
## 21 12.6  0 10.8  7.4  6.7 0.602  98.4  34  1.2 0.102 3.3   5570 19.5 0.022800
## 22 15.7  1  8.9  4.7  4.4 0.512  96.2  22 42.3 0.097 3.4   2880 27.6 0.089502
## 23 13.2  0  9.6  8.7  8.3 0.564  95.3  43  9.2 0.083 3.2   5130 22.7 0.030700
## 24 13.1  0 11.6  7.8  7.3 0.574 103.8   7  3.6 0.142 4.2   5400 17.6 0.041598
## 25 13.0  0 11.6  6.3  5.7 0.641  98.4  14  2.6 0.070 2.1   4860 19.6 0.069197
## 26 13.1  0 12.1 16.0 14.3 0.631 107.1   3  7.7 0.102 4.1   6740 15.2 0.041698
## 27 13.5  0 10.9  6.9  7.1 0.540  96.5   6  0.4 0.080 2.2   5640 13.9 0.036099
## 28 15.2  0 11.2  8.2  7.6 0.571 101.8  10  7.9 0.103 2.8   5370 21.5 0.038201
```

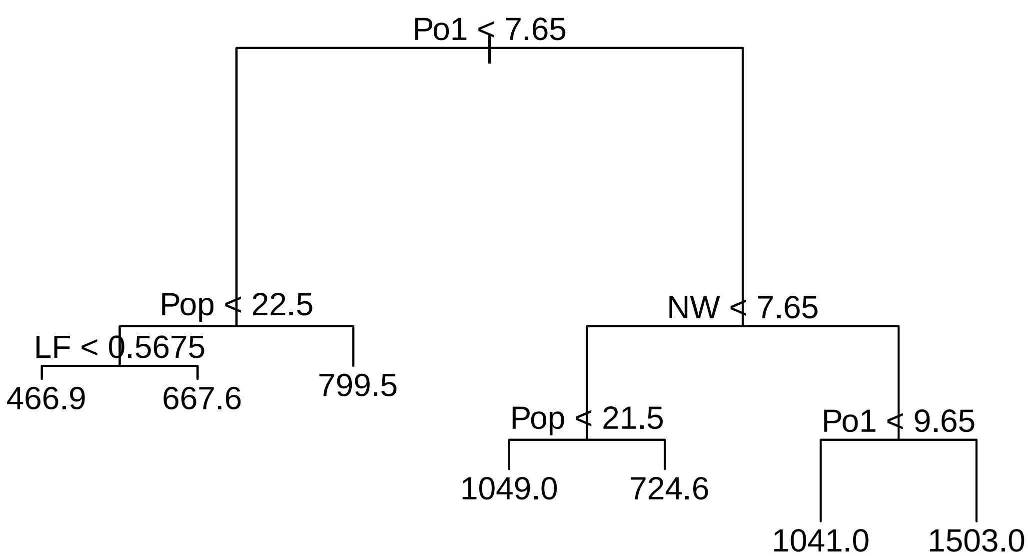
```
## 29 11.9  0 10.7 16.6 15.7 0.521  93.8 168  8.9 0.092 3.6   6370 15.4 0.023400
## 30 16.6  1  8.9  5.8  5.4 0.521  97.3  46 25.4 0.072 2.6   3960 23.7 0.075298
## 31 14.0  0  9.3  5.5  5.4 0.535 104.5   6  2.0 0.135 4.0   4530 20.0 0.041999
## 32 12.5  0 10.9  9.0  8.1 0.586  96.4  97  8.2 0.105 4.3   6170 16.3 0.042698
## 33 14.7  1 10.4  6.3  6.4 0.560  97.2  23  9.5 0.076 2.4   4620 23.3 0.049499
## 34 12.6  0 11.8  9.7  9.7 0.542  99.0  18  2.1 0.102 3.5   5890 16.6 0.040799
## 35 12.3  0 10.2  9.7  8.7 0.526  94.8 113  7.6 0.124 5.0   5720 15.8 0.020700
## 36 15.0  0 10.0 10.9  9.8 0.531  96.4   9  2.4 0.087 3.8   5590 15.3 0.006900
## 37 17.7  1  8.7  5.8  5.6 0.638  97.4  24 34.9 0.076 2.8   3820 25.4 0.045198
## 38 13.3  0 10.4  5.1  4.7 0.599 102.4   7  4.0 0.099 2.7   4250 22.5 0.053998
## 39 14.9  1  8.8  6.1  5.4 0.515  95.3  36 16.5 0.086 3.5   3950 25.1 0.047099
## 40 14.5  1 10.4  8.2  7.4 0.560  98.1  96 12.6 0.088 3.1   4880 22.8 0.038801
## 41 14.8  0 12.2  7.2  6.6 0.601  99.8   9  1.9 0.084 2.0   5900 14.4 0.025100
## 42 14.1  0 10.9  5.6  5.4 0.523  96.8   4  0.2 0.107 3.7   4890 17.0 0.088904
## 43 16.2  1  9.9  7.5  7.0 0.522  99.6  40 20.8 0.073 2.7   4960 22.4 0.054902
## 44 13.6  0 12.1  9.5  9.6 0.574 101.2  29  3.6 0.111 3.7   6220 16.2 0.028100
## 45 13.9  1  8.8  4.6  4.1 0.480  96.8  19  4.9 0.135 5.3   4570 24.9 0.056202
## 46 12.6  0 10.4 10.6  9.7 0.599  98.9  40  2.4 0.078 2.5   5930 17.1 0.046598
## 47 13.0  0 12.1  9.0  9.1 0.623 104.9   3  2.2 0.113 4.0   5880 16.0 0.052802
##       Time Crime
## 1  26.2011   791
## 2  25.2999  1635
## 3  24.3006   578
## 4  29.9012  1969
## 5  21.2998  1234
## 6  20.9995   682
## 7  20.6993   963
## 8  24.5988  1555
## 9  29.4001   856
## 10 19.5994   705
## 11 41.6000  1674
## 12 34.2984   849
## 13 36.2993   511
## 14 21.5010   664
## 15 22.7008   798
## 16 26.0991   946
## 17 19.1002   539
## 18 18.1996   929
## 19 24.9008   750
## 20 26.4010  1225
## 21 37.5998   742
## 22 37.0994   439
## 23 25.1989  1216
## 24 17.6000   968
## 25 21.9003   523
## 26 22.1005  1993
## 27 28.4999   342
## 28 25.8006  1216
## 29 36.7009  1043
## 30 28.3011   696
## 31 21.7998   373
## 32 30.9014   754
## 33 25.5005  1072
## 34 21.6997   923
```

```
## 35 37.4011   653
## 36 44.0004  1272
## 37 31.6995   831
## 38 16.6999   566
## 39 27.3004   826
## 40 29.3004  1151
## 41 30.0001   880
## 42 12.1996   542
## 43 31.9989   823
## 44 30.0001  1030
## 45 32.5996   455
## 46 16.6999   508
## 47 16.0997   849
```

```r
crime_tree <- tree(Crime ~ ., data = data)
```

Plot regression Tree model:

```r
plot(crime_tree)
text(crime_tree)
```



Calculate accuracy of tree model:

```r
tree_predict <- predict(crime_tree, data = data[,1:15])
rs <- sum((tree_predict - data[,16])^2)
ts <- sum((data[,16] - mean(data[,16]))^2)
```

```r
r <- 1 - rs/ts
r
```

```
## [1] 0.7244962
```

The accuracy of the tree model is around 72%, which I believe is fairly accurate for a prediction model.

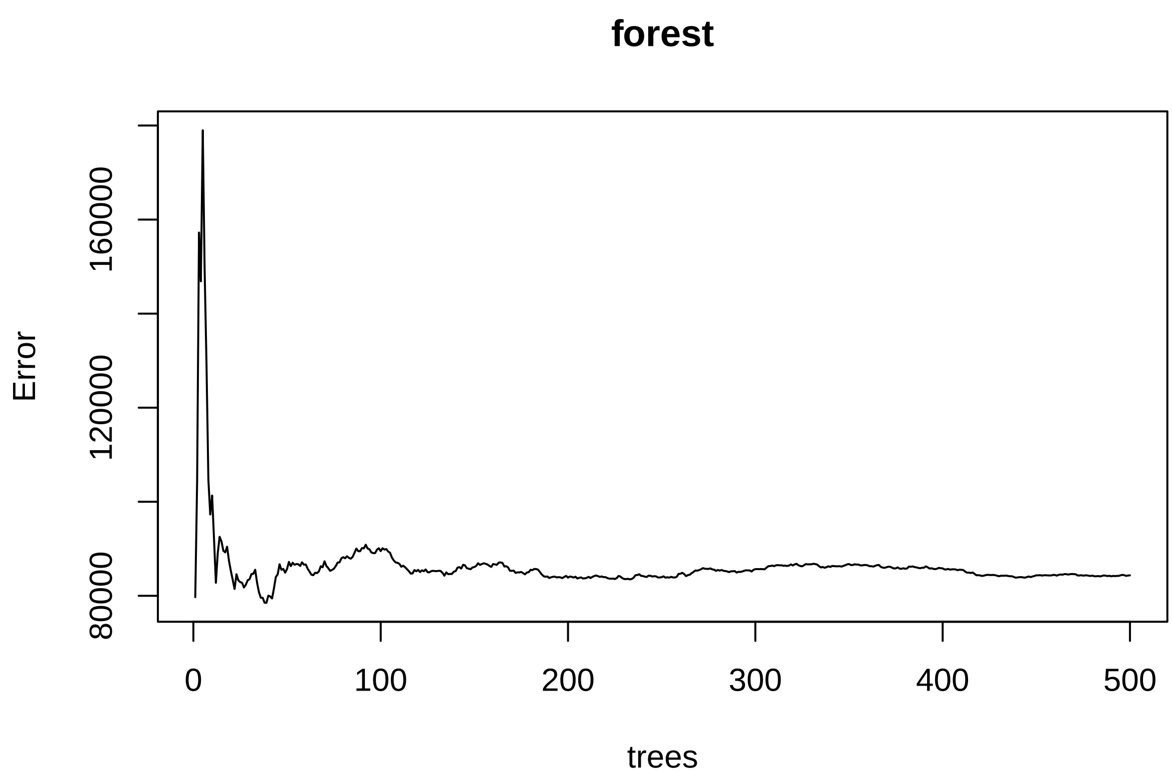
Now we are going to try to make similiar predictions with a random forest model and compare to the accuracy of the tree model.

Random forests:

```r
forest <- randomForest(Crime ~ ., data = data, mtry = 2)
forest
```

```
##
## Call:
##  randomForest(formula = Crime ~ ., data = data, mtry = 2)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##           Mean of squared residuals: 84344.94
##                     % Var explained: 42.39
```

```r
plot(forest)
```



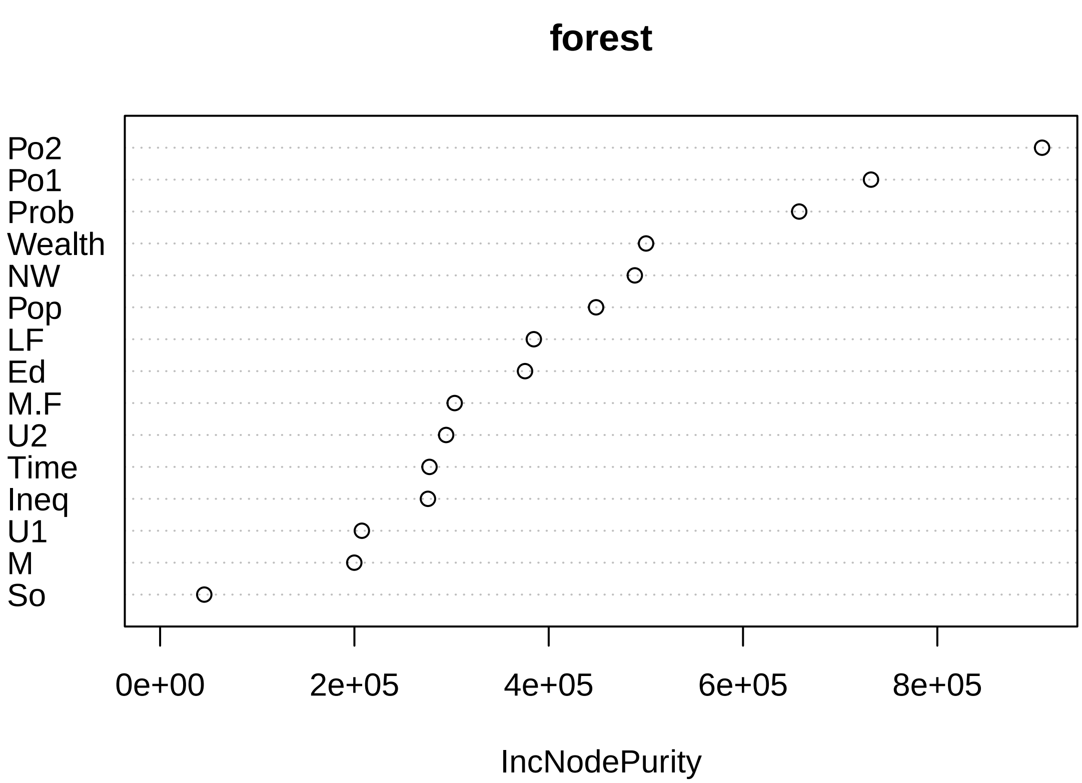
From this plot we can see that the around 100 there is a sharp increase in the error prior to a minimum error value.

Determine which factors are most important:

```
importance(forest, type = 2)
```

```
##          IncNodePurity
## M            199869.37
## So            45460.19
## Ed           375604.44
## Po1          731729.15
## Po2          907815.82
## LF           384668.48
## M.F          303186.49
## Pop          448762.77
## NW           488613.74
## U1           207676.87
## U2           294414.25
## Wealth       500209.58
## Ineq         275680.73
## Prob         657815.03
## Time         277288.20
```

```
varImpPlot(forest)
```



From the plot above we can see that Po1, Prob, Po2, Wealth, NW are the most important factors in the model.

```
pred_forest <- predict(forest)
ss_forest <- sum((pred_forest-data$Crime)^2)

ss_total <- sum((data$Crime - mean(data$Crime))^2)
r_forest <-  1 - (ss_forest/ss_total)

r_forest
```

```
## [1] 0.4238841
```

This accuracy value of 43% is slightly lower than that of the tree model, but is fairly consistent with the variance value shown previously in the model summary.

7.2

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.

Given the current state of the coronavirus it could be useful to predict whether or not someone is suffering from the coronavirus or simply has a flu. Some predictors that could be used for this analysis are: change or loss in taste, age(coronavirus spreads easier in older folks than influenza), blood clots in veins, length of symptoms and your location(whether you had been previously near coronavirus clusters)

7.3

1. Using the GermanCredit 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, use family=binomial(link="logit") in your glm function call.

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.

Load German data:

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

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

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

```
##    V1 V2  V3  V4   V5  V6  V7 V8  V9  V10 V11  V12 V13  V14  V15 V16  V17 V18
## 1 A11  6 A34 A43 1169 A65 A75  4 A93 A101   4 A121  67 A143 A152   2 A173   1
## 2 A12 48 A32 A43 5951 A61 A73  2 A92 A101   2 A121  22 A143 A152   1 A173   1
## 3 A14 12 A34 A46 2096 A61 A74  2 A93 A101   3 A121  49 A143 A152   1 A172   2
## 4 A11 42 A32 A42 7882 A61 A74  2 A93 A103   4 A122  45 A143 A153   1 A173   2
## 5 A11 24 A33 A40 4870 A61 A73  3 A93 A101   4 A124  53 A143 A153   2 A173   2
## 6 A14 36 A32 A46 9055 A65 A73  2 A93 A101   4 A124  35 A143 A153   1 A172   2
##    V19  V20 V21
## 1 A192 A201   1
## 2 A191 A201   2
## 3 A191 A201   1
## 4 A191 A201   1
## 5 A191 A201   2
## 6 A192 A201   1
```

Create training model/data:

```
data$V21[data$V21==1] <- 0
data$V21[data$V21==2] <- 1

train <-sample(nrow(data),0.7* nrow(data), replace = FALSE)

train_data <-data[train, ]
test_data <-data[-train, ]

model <-glm(V21~.,family =binomial(link ="logit"),data =train_data)
```

Determine roc:

```
pred <-predict(model, test_data, type = "response")

roc <-roc(test_data$V21,round(pred))
```

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

## Setting direction: controls < cases
```

```
roc
```

```
##
## Call:
## roc.default(response = test_data$V21, predictor = round(pred))
##
## Data: round(pred) in 208 controls (test_data$V21 0) < 92 cases (test_data$V21 1).
## Area under the curve: 0.679
```

Threshold .4

```
threshold<-as.integer(pred>0.4)

matrix<-as.matrix(table(threshold, test_data$V21))

matrix
```

```
##
## threshold   0   1
##         0 167  38
##         1  41  54
```

```r
acc <- (matrix[1,1]+matrix[2,2])/(matrix[1,1]+matrix[1,2]+matrix[2,1]+matrix[2,2])

acc
```

```
## [1] 0.7366667
```

```r
spec <- matrix[1,1]/matrix[1,1]+matrix[2,1]

spec
```

```
## [1] 42
```

Threshold .5

```r
threshold2<-as.integer(pred>0.5)

matrix2<-as.matrix(table(threshold2, test_data$V21))

matrix2
```

```
##
## threshold2   0   1
##          0 183  48
##          1  25  44
```

Accuracy for .5 threshold

```r
acc2 <- (matrix2[1,1]+matrix2[2,2])/(matrix2[1,1]+matrix2[1,2]+matrix2[2,1]+matrix2[2,2])

acc2
```

```
## [1] 0.7566667
```

```r
spec2 <- matrix2[1,1]/(matrix2[1,1]+matrix2[2,1])

spec2
```

```
## [1] 0.8798077
```

Threshold for .6

```r
threshold3<-as.integer(pred>0.6)

matrix3<-as.matrix(table(threshold3, test_data$V21))

matrix3
```

```
##
## threshold3   0   1
##          0 190  60
##          1  18  32
```

Accuracy for threshold .6

```r
acc3 <- (matrix3[1,1]+matrix3[2,2])/(matrix3[1,1]+matrix3[1,2]+matrix3[2,1]+matrix3[2,2])

acc3
```

```
## [1] 0.74
```

```r
spec3 <- matrix3[1,1]/(matrix3[1,1]+matrix3[2,1])

spec3
```

```
## [1] 0.9134615
```

threshold .7

```r
threshold4<-as.integer(pred>0.7)

matrix4<-as.matrix(table(threshold4, test_data$V21))

matrix4
```

```
##
## threshold4   0   1
##          0 197  68
##          1  11  24
```

Accuracy for threshold .7

```r
acc4 <- (matrix4[1,1]+matrix4[2,2])/(matrix4[1,1]+matrix4[1,2]+matrix4[2,1]+matrix4[2,2])

acc4
```

```
## [1] 0.7366667
```

```r
spec4 <- matrix4[1,1]/(matrix4[1,1]+matrix4[2,1])

spec4
```

```
## [1] 0.9471154
```

As we can see from some of the various thresholds that were tested, a threshold of .5 appears to provide the best predictions, while also having a fairly high specificity value.