ISYE6501x Homework 7

Done By: Joel Quek

Question 10.1

Using the same crime data set uscrime.txt as in Questions 8.2 and 9.1, 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 rpart package, and the randomForest package. For each model,describe one or two qualitative takeaways you get from analyzing the results (i.e., don't just stop when you have a good model, but interpret it too).

Dataset and Libraries

In [1]:

```
library(rpart)
library(rpart.plot)
library(tree)
library(RColorBrewer)
library(rattle)
library(caret)
library(randomForest)
library(data.table)
library(mltools)
library(ggplot2)
library(cowplot)
```

...

In [2]:

```
crime <- read.table("uscrime.txt", header=TRUE)
```

In [3]:

```
head(crime)
```

A data.frame: 6 × 16

	M	So	Ed	Po1	Po2	LF	M.F	Pop	NW	U1	U2	Wealth	Ineq	
	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	
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.
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.
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.
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.
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.
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.

In [4]:

```
summary(crime)
```

```
       M               So               Ed              Po1       
 Min.   :11.90   Min.   :0.0000   Min.   : 8.70   Min.   : 4.50  
 1st Qu.:13.00   1st Qu.:0.0000   1st Qu.: 9.75   1st Qu.: 6.25  
 Median :13.60   Median :0.0000   Median :10.80   Median : 7.80  
 Mean   :13.86   Mean   :0.3404   Mean   :10.56   Mean   : 8.50  
 3rd Qu.:14.60   3rd Qu.:1.0000   3rd Qu.:11.45   3rd Qu.:10.45  
 Max.   :17.70   Max.   :1.0000   Max.   :12.20   Max.   :16.60  
      Po2               LF              M.F              Pop        
 Min.   : 4.100   Min.   :0.4800   Min.   : 93.40   Min.   :  3.00  
 1st Qu.: 5.850   1st Qu.:0.5305   1st Qu.: 96.45   1st Qu.: 10.00  
 Median : 7.300   Median :0.5600   Median : 97.70   Median : 25.00  
 Mean   : 8.023   Mean   :0.5612   Mean   : 98.30   Mean   : 36.62  
 3rd Qu.: 9.700   3rd Qu.:0.5930   3rd Qu.: 99.20   3rd Qu.: 41.50  
 Max.   :15.700   Max.   :0.6410   Max.   :107.10   Max.   :168.00  
       NW              U1                U2            Wealth    
 Min.   : 0.20   Min.   :0.07000   Min.   :2.000   Min.   :2880  
 1st Qu.: 2.40   1st Qu.:0.08050   1st Qu.:2.750   1st Qu.:4595  
 Median : 7.60   Median :0.09200   Median :3.400   Median :5370  
 Mean   :10.11   Mean   :0.09547   Mean   :3.398   Mean   :5254  
 3rd Qu.:13.25   3rd Qu.:0.10400   3rd Qu.:3.850   3rd Qu.:5915  
 Max.   :42.30   Max.   :0.14200   Max.   :5.800   Max.   :6890  
      Ineq            Prob              Time           Crime       
 Min.   :12.60   Min.   :0.00690   Min.   :12.20   Min.   : 342.0  
 1st Qu.:16.55   1st Qu.:0.03270   1st Qu.:21.60   1st Qu.: 658.5  
 Median :17.60   Median :0.04210   Median :25.80   Median : 831.0  
 Mean   :19.40   Mean   :0.04709   Mean   :26.60   Mean   : 905.1  
 3rd Qu.:22.75   3rd Qu.:0.05445   3rd Qu.:30.45   3rd Qu.:1057.5  
 Max.   :27.60   Max.   :0.11980   Max.   :44.00   Max.   :1993.0  
```

Regression Tree

Source: [https://www.pluralsight.com/guides/explore-r-libraries:-rpart (https://www.pluralsight.com/guides/explore-r-libraries:-rpart)](https://www.pluralsight.com/guides/explore-r-libraries:-rpart)

In [5]:

```
set.seed(1)

tree_model<- rpart(Crime~ ., data = crime,method="anova" )

summary(tree_model)
```

```
Call:
rpart(formula = Crime ~ ., data = crime, method = "anova")
  n= 47 

          CP nsplit rel error    xerror      xstd
1 0.36296293      0 1.0000000 1.0303899 0.2549076
2 0.14814320      1 0.6370371 0.8900680 0.2149365
3 0.05173165      2 0.4888939 0.9096979 0.2393384
4 0.01000000      3 0.4371622 0.8893049 0.2346618

Variable importance
   Po1    Po2 Wealth   Ineq   Prob      M     NW    Pop   Time     Ed     LF 
    17     17     11     11     10     10      9      5      4      4      1 
    So 
     1 

Node number 1: 47 observations,    complexity param=0.3629629
  mean=905.0851, MSE=146402.7 
  left son=2 (23 obs) right son=3 (24 obs)
  Primary splits:
      Po1    < 7.65      to the left,  improve=0.3629629, (0 missing)
      Po2    < 7.2       to the left,  improve=0.3629629, (0 missing)
      Prob   < 0.0418485 to the right, improve=0.3217700, (0 missing)
      NW     < 7.65      to the left,  improve=0.2356621, (0 missing)
      Wealth < 6240      to the left,  improve=0.2002403, (0 missing)
  Surrogate splits:
      Po2    < 7.2       to the left,  agree=1.000, adj=1.000, (0 split)
      Wealth < 5330      to the left,  agree=0.830, adj=0.652, (0 split)
      Prob   < 0.043598  to the right, agree=0.809, adj=0.609, (0 split)
      M      < 13.25     to the right, agree=0.745, adj=0.478, (0 split)
      Ineq   < 17.15     to the right, agree=0.745, adj=0.478, (0 split)

Node number 2: 23 observations,    complexity param=0.05173165
  mean=669.6087, MSE=33880.15 
  left son=4 (12 obs) right son=5 (11 obs)
  Primary splits:
      Pop < 22.5      to the left,  improve=0.4568043, (0 missing)
      M   < 14.5      to the left,  improve=0.3931567, (0 missing)
      NW  < 5.4       to the left,  improve=0.3184074, (0 missing)
      Po1 < 5.75      to the left,  improve=0.2310098, (0 missing)
      U1  < 0.093     to the right, improve=0.2119062, (0 missing)
  Surrogate splits:
      NW   < 5.4       to the left,  agree=0.826, adj=0.636, (0 split)
      M    < 14.5      to the left,  agree=0.783, adj=0.545, (0 split)
      Time < 22.30055  to the left,  agree=0.783, adj=0.545, (0 split)
      So   < 0.5       to the left,  agree=0.739, adj=0.455, (0 split)
      Ed   < 10.85     to the right, agree=0.739, adj=0.455, (0 split)

Node number 3: 24 observations,    complexity param=0.1481432
  mean=1130.75, MSE=150173.4 
  left son=6 (10 obs) right son=7 (14 obs)
  Primary splits:
      NW   < 7.65      to the left,  improve=0.2828293, (0 missing)
      M    < 13.05     to the left,  improve=0.2714159, (0 missing)
      Time < 21.9001   to the left,  improve=0.2060170, (0 missing)
      M.F  < 99.2      to the left,  improve=0.1703438, (0 missing)
      Po1  < 10.75     to the left,  improve=0.1659433, (0 missing)
  Surrogate splits:
      Ed   < 11.45     to the right, agree=0.750, adj=0.4, (0 split)
      Ineq < 16.25     to the left,  agree=0.750, adj=0.4, (0 split)
      Time < 21.9001   to the left,  agree=0.750, adj=0.4, (0 split)
      Pop  < 30        to the left,  agree=0.708, adj=0.3, (0 split)
```

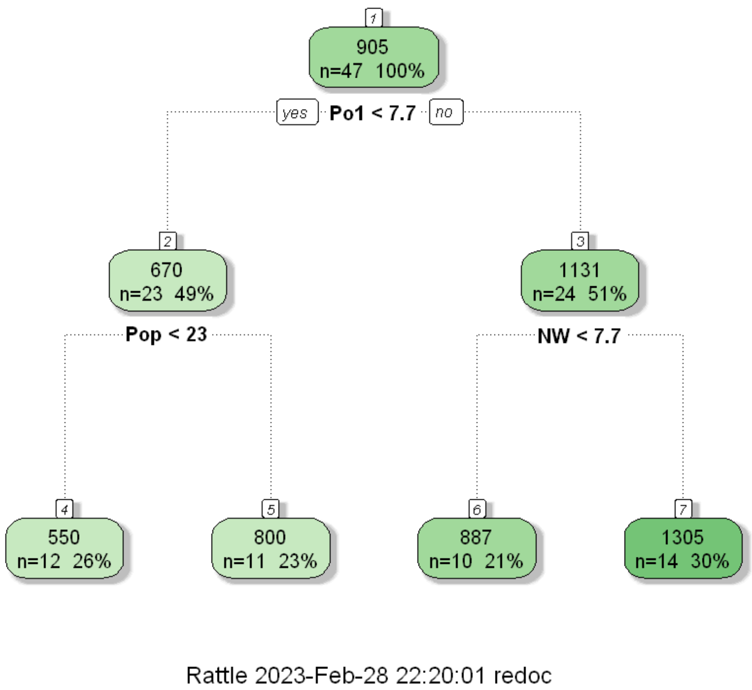
```
      LF    < 0.5885    to the right, agree=0.667, adj=0.2, (0 split)

Node number 4: 12 observations
  mean=550.5, MSE=20317.58

Node number 5: 11 observations
  mean=799.5455, MSE=16315.52

Node number 6: 10 observations
  mean=886.9, MSE=55757.49

Node number 7: 14 observations
  mean=1304.929, MSE=144801.8
```

Regression Tree Plot

Source: [https://discuss.analyticsvidhya.com/t/what-are-the-packages-required-to-plot-a-fancy-rpart-plot-in-r/6776/2 (https://discuss.analyticsvidhya.com/t/what-are-the-packages-required-to-plot-a-fancy-rpart-plot-in-r/6776/2)](https://discuss.analyticsvidhya.com/t/what-are-the-packages-required-to-plot-a-fancy-rpart-plot-in-r/6776/2)

In [6]:

```
fancyRpartPlot(tree_model)
# rpart.plot(tree_model) also works
```



From the Regression Tree only three variables are used

1. Po1
2. Pop
3. NW

Variable Importance

In [7]:

```
tree_model$variable.importance
```

Po1: 2497521.6813136 **Po2:** 2497521.6813136 **Wealth:** 1628818.48781322 **Ineq:** 1602211.95963445 **Prob:** 1520230.58862567 **M:** 1388627.84614747 **NW:** 1245883.78569375 **Pop:** 661770.552416714 **Time:** 601906.02365587 **Ed:** 569545.86447513 **LF:** 203872.534285714 **So:** 161800.795903701

Variable importance is determined by calculating the relative influence of each variable: whether that variable was selected to split on during the tree building process, and how much the squared error (over all trees) improved (decreased) as a result.

Source: https://h2o-release.s3.amazonaws.com/h2o/rel-yau/3/docs-website/h2o-docs/variable-importance.html#:~:text=Variable%20importance%20is%20determined%20by,(decreased)%20as%20a%20result (https://h2o-release.s3.amazonaws.com/h2o/rel-yau/3/docs-website/h2o-docs/variable-importance.html#:~:text=Variable%20importance%20is%20determined%20by,(decreased)%20as%20a%20result).

Model Evaluation

Source (Regression): https://medium.com/nerd-for-tech/implementing-decision-trees-in-r-regression-problem-using-rpart-c74cbd9e0b7b (https://medium.com/nerd-for-tech/implementing-decision-trees-in-r-regression-problem-using-rpart-c74cbd9e0b7b)

Source (Classification): https://www.pluralsight.com/guides/explore-r-libraries:-rpart (https://www.pluralsight.com/guides/explore-r-libraries:-rpart)

In [8]:

```
set.seed(100)
trainRowNumbers <- createDataPartition(crime$Crime, p=0.7, list=FALSE)
train <- crime[trainRowNumbers,]
test <- crime[-trainRowNumbers,]
dim(train); dim(test)
```

35 · 16

12 · 16

In [9]:

```
PredictCART_train = predict(tree_model, data = test, type="vector")
```

In [10]:

```
MAE <- function(actual,pred) {mean(abs(actual-pred))}

MAE(test$Crime,PredictCART_train)
```

```
Warning message in actual - pred:
"longer object length is not a multiple of shorter object length"
```

356.986791931473

In [11]:

```
y1 <- predict(tree_model, test)
MSE1 <- mean((y1-test$Crime)^2)
MSE1
```

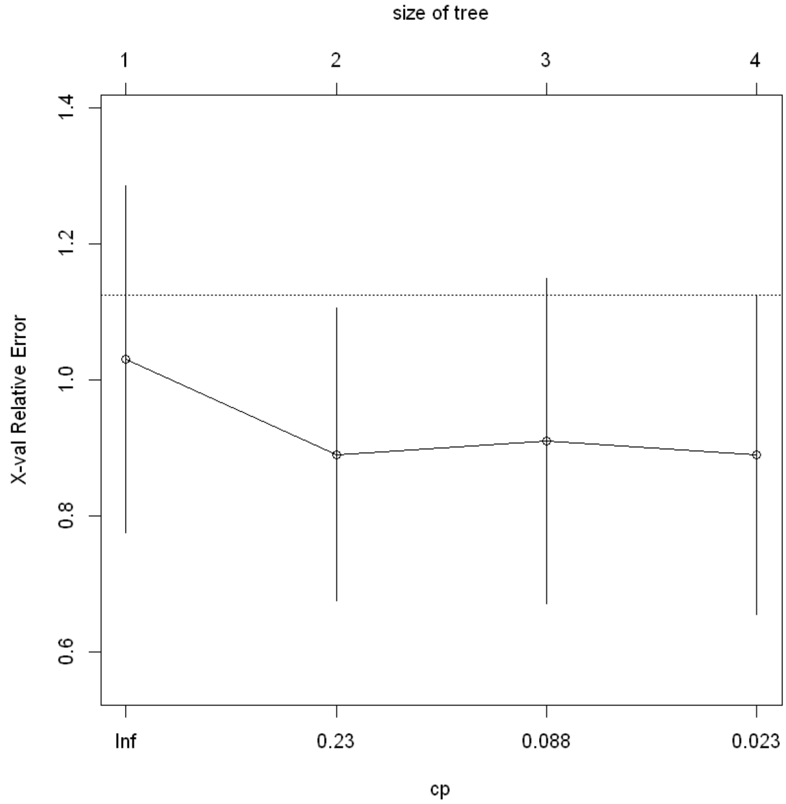
65045.3652938944

Regression Tree Complexity Parameter

In [12]:

```
printcp(tree_model)

plotcp(tree_model)
```

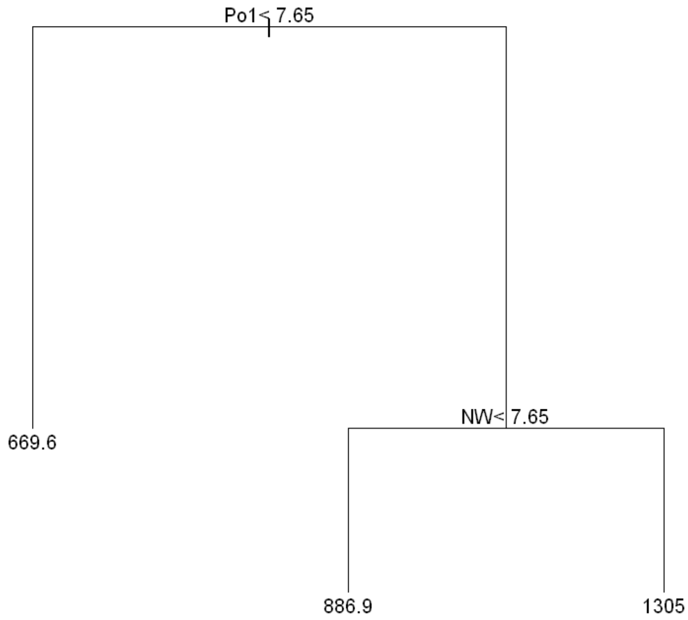
```
Regression tree:
rpart(formula = Crime ~ ., data = crime, method = "anova")

Variables actually used in tree construction:
[1] NW  Po1 Pop

Root node error: 6880928/47 = 146403

n= 47

        CP nsplit rel error  xerror    xstd
1 0.362963      0   1.00000 1.03039 0.25491
2 0.148143      1   0.63704 0.89007 0.21494
3 0.051732      2   0.48889 0.90970 0.23934
4 0.010000      3   0.43716 0.88930 0.23466
```



Pruning the Tree

In [13]:

```
pruned_model <- prune.rpart(tree_model,cp=0.10) # adjust the cp parameter to get different mo

plot(pruned_model)

text(pruned_model)

#fancyRpartPlot(tree_model)
```



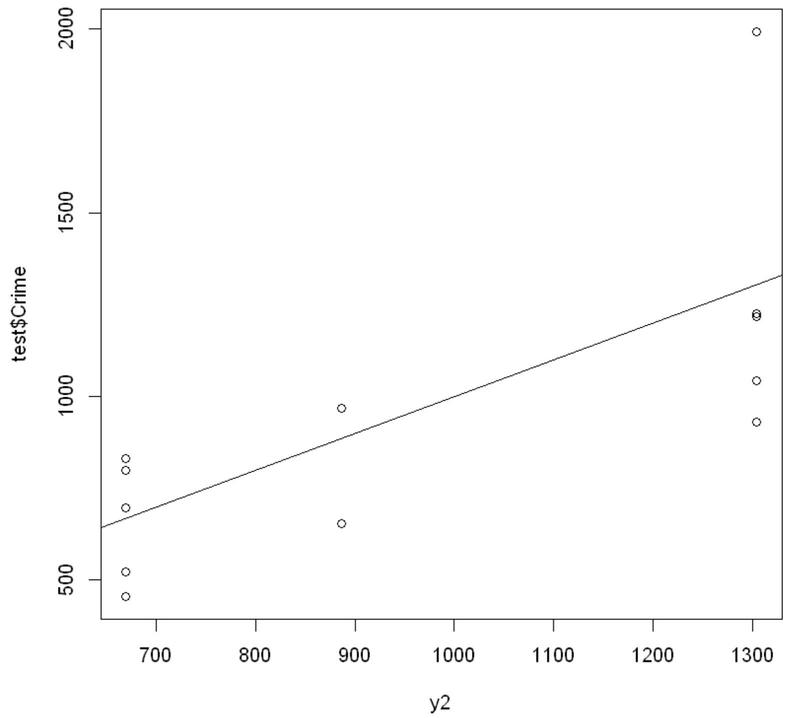
Test the Pruned Model

In [14]:

```
y2 <- predict(pruned_model, test)
```

In [15]:

```
plot(y2, test$Crime)

abline(0,1)
```



In [16]:

```
MSE2 <- mean((y2-test$Crime)^2)
MSE2
```

72477.785708081

The MSE is worse for the pruned tree

Regression Equations at the Leaves

In [17]:

```
pruned_model$where
```

2 · 5 · 2 · 5 · 4 · 4 · 5 · 5 · 2 · 2 · 5 · 2 · 2 · 2 · 2 · 5 · 2 · 5 · 4 · 5 ·
2 · 2 · 5 · 4 · 2 · 5 · 2 · 5 · 5 · 2 · 2 · 5 · 2 · 4 · 4 · 4 · 2 · 2 · 2 · 5 ·
2 · 2 · 2 · 4 · 2 · 4 · 4

In [18]:

```
leaf4 <-crime[which(pruned_model$where==4),]

leaf5 <-crime[which(pruned_model$where==5),]

leaf2 <-crime[which(pruned_model$where==2),]

#leaf6 <-crime[which(pruned_model$where==6),]
```

In [19]:

```
model_leaf4<-lm(Crime~., data=leaf4)
model_leaf4
#summary(leaf4)
```

```
Call:
lm(formula = Crime ~ ., data = leaf4)

Coefficients:
(Intercept)            M           So           Ed          Po1          Po2
   32527.85       258.27           NA       -46.38     -1168.92       612.42
         LF          M.F          Pop           NW           U1           U2
   16612.42      -384.45       -18.22       124.13      2064.68           NA
     Wealth         Ineq         Prob         Time
         NA           NA           NA           NA
```

In [20]:

```
model_leaf5<-lm(Crime~., data=leaf5)
model_leaf5
#summary(leaf7)
```

```
Call:
lm(formula = Crime ~ ., data = leaf5)

Coefficients:
(Intercept)            M           So           Ed          Po1          Po2
 -1.381e+04    8.012e+01   -2.827e+02    2.663e+02   -2.943e+02    3.571e+02
         LF          M.F          Pop           NW           U1           U2
 -1.648e+03    8.738e+01    1.154e+00    8.841e+00   -3.265e+04    5.783e+02
     Wealth         Ineq         Prob         Time
  2.416e-01    1.367e+02           NA           NA
```

In [21]:

```
model_leaf2<-lm(Crime~., data=leaf2)
model_leaf2
#summary(leaf6)
```

```
Call:
lm(formula = Crime ~ ., data = leaf2)

Coefficients:
(Intercept)            M           So           Ed          Po1          Po2
   -48.5477      45.8622     380.4815     187.9074      -3.5138      44.6382
         LF          M.F          Pop           NW           U1           U2
  1059.3652     -22.5521      10.6413       0.1010    4878.2802      -5.5126
     Wealth         Ineq         Prob         Time
    -0.1022       4.7779   -7317.4407     -20.0603
```

Observations

It appears that the R library produced the model with the lowest MSE.

Random Forest Model

Source (Stat Quest): https://www.youtube.com/watch?v=6EXPYzbfLCE (https://www.youtube.com/watch?v=6EXPYzbfLCE)

In [22]:

```
set.seed(1)
forest_model <- randomForest(Crime ~. , data=crime,keep.forest=T, importance=TRUE,class=)
print(forest_model )
```

```
Call:
 randomForest(formula = Crime ~ ., data = crime, keep.forest = T,      importa
nce = TRUE, class = )
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5

          Mean of squared residuals: 87407.26
                    % Var explained: 40.3
```

Variable Importance

In [23]:

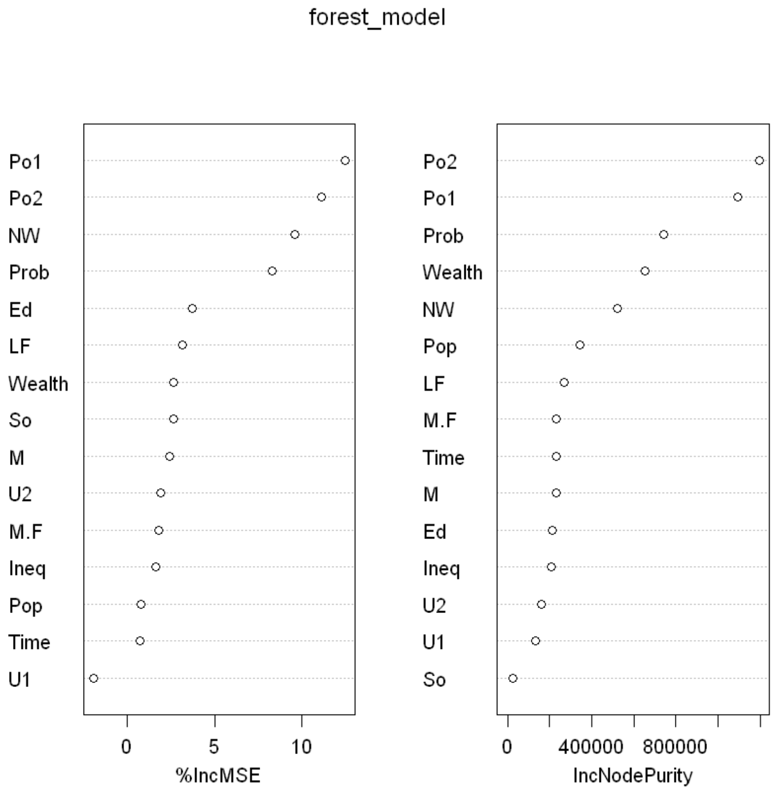
```
importance(forest_model )
```

A matrix: 15 × 2 of type dbl

	%IncMSE	IncNodePurity
M	2.3965203	228719.50
So	2.6589225	23456.11
Ed	3.7141435	213283.48
Po1	12.4862109	1095484.85
Po2	11.0894307	1197840.76
LF	3.1325318	270248.43
M.F	1.7951798	232785.09
Pop	0.7784942	343523.34
NW	9.5782376	520590.05
U1	-1.9438082	133339.11
U2	1.9167880	162803.16
Wealth	2.6613199	654055.84
Ineq	1.6040415	208959.89
Prob	8.3155853	743181.12
Time	0.7126839	231615.22

In [24]:

```
varImpPlot(forest_model )
```



Performance of Random Forest Model

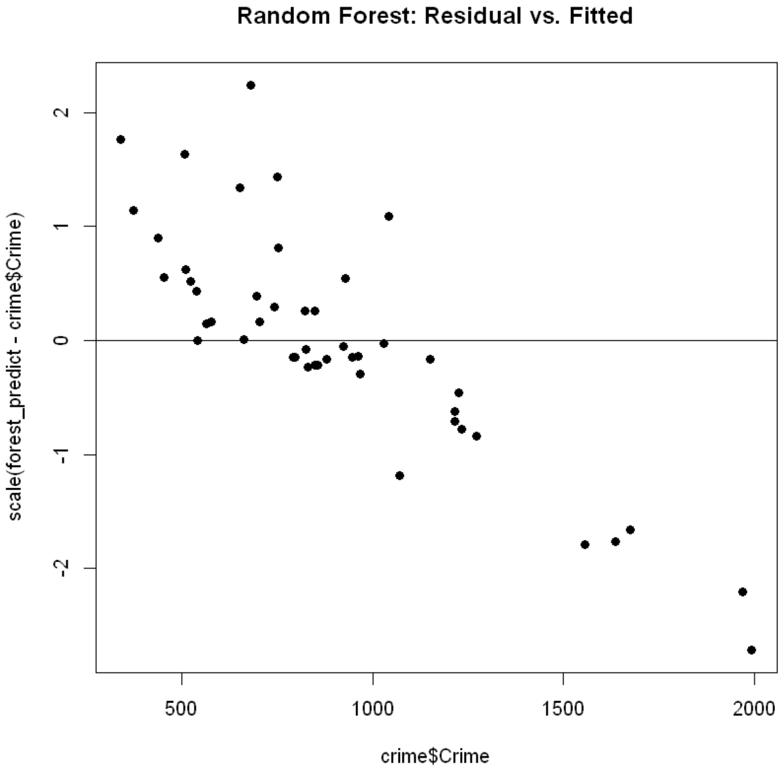
In [25]:

```
forest_predict <- predict(forest_model)
RSS <- sum((forest_predict-crime$Crime)^2)
# the residual sum of squares
TSS <- sum((mean(crime$Crime)-crime$Crime)^2)
#the total sum of squares
R_squared_forest<-1-(RSS/TSS)
R_squared_forest
```

0.402966921582338

In [26]:

```
#residual analysis
plot(crime$Crime, scale(forest_predict-crime$Crime), pch =19,main="Random Forest: Residual vs
abline(0,0)
```



Observations

R^2 is 0.402966921582338 which is a good score

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 an educator, there are times when I need to decide if a student is suitable for an Arts or Science program. Although there are tests with numeric scores used to mesaure, there are other intangible factors that should also be taken into account for instance:

1. Socio-economic factors
2. Gender
3. Race
4. Next-of-kin aptitude in the subjects

Such factors are categorical and would require logistic regression to model the response.

Question 10.3

1. Using the GermanCredit data set germancredit.txt from http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german (http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german) / (description at http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29 (http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29)), 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.

Source (Stat Quest): https://www.youtube.com/watch?v=C4N3_XJJ-jU (https://www.youtube.com/watch?v=C4N3_XJJ-jU)

In [27]:

```
credit <- read.table("germancredit.txt",sep = " ")
```

In [28]:

```
head(credit,5)
```

A data.frame: 5 × 21

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	⋯	V12	V13	V14
	<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	⋯	<chr>	<int>	<chr>
1	A11	6	A34	A43	1169	A65	A75	4	A93	A101	⋯	A121	67	A143
2	A12	48	A32	A43	5951	A61	A73	2	A92	A101	⋯	A121	22	A143
3	A14	12	A34	A46	2096	A61	A74	2	A93	A101	⋯	A121	49	A143
4	A11	42	A32	A42	7882	A61	A74	2	A93	A103	⋯	A122	45	A143
5	A11	24	A33	A40	4870	A61	A73	3	A93	A101	⋯	A124	53	A143

One-Hot Encode the Response Variable

In [33]:

```
set.seed(1)
credit_onehot <- one_hot(as.data.table(credit))
#one hot encoding the categorical variables
```

In [34]:

```
head(credit_onehot)
```

A data.table: 6 × 21

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	⋯	V12	V13	V14	V
<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	⋯	<chr>	<int>	<chr>	<c
A11	6	A34	A43	1169	A65	A75	4	A93	A101	⋯	A121	67	A143	A
A12	48	A32	A43	5951	A61	A73	2	A92	A101	⋯	A121	22	A143	A
A14	12	A34	A46	2096	A61	A74	2	A93	A101	⋯	A121	49	A143	A
A11	42	A32	A42	7882	A61	A74	2	A93	A103	⋯	A122	45	A143	A
A11	24	A33	A40	4870	A61	A73	3	A93	A101	⋯	A124	53	A143	A
A14	36	A32	A46	9055	A65	A73	2	A93	A101	⋯	A124	35	A143	A

In [35]:

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

In the data dictionary [http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29](http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29%5D)

1 = Good, 2 = Bad

Our one-hot encoding makes

0 = Good and 1 = Bad

In [36]:

```
head(credit_onehot)
```

A data.table: 6 × 21

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	⋯	V12	V13	V14	V
<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	<int>	<chr>	<chr>	⋯	<chr>	<int>	<chr>	<c
A11	6	A34	A43	1169	A65	A75	4	A93	A101	⋯	A121	67	A143	A
A12	48	A32	A43	5951	A61	A73	2	A92	A101	⋯	A121	22	A143	A
A14	12	A34	A46	2096	A61	A74	2	A93	A101	⋯	A121	49	A143	A
A11	42	A32	A42	7882	A61	A74	2	A93	A103	⋯	A122	45	A143	A
A11	24	A33	A40	4870	A61	A73	3	A93	A101	⋯	A124	53	A143	A
A14	36	A32	A46	9055	A65	A73	2	A93	A101	⋯	A124	35	A143	A

In [37]:

```
logistic <- glm(V21~.,data=credit_onehot, family="binomial")
```

In [39]:

```
summary(logistic) # pay attention to the coefficients
```

```
Call:
glm(formula = V21 ~ ., family = "binomial", data = credit_onehot)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.3410  -0.6994  -0.3752   0.7095   2.6116  

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept)  4.005e-01  1.084e+00   0.369 0.711869    
V1A12       -3.749e-01  2.179e-01  -1.720 0.085400 .  
V1A13       -9.657e-01  3.692e-01  -2.616 0.008905 ** 
V1A14       -1.712e+00  2.322e-01  -7.373 1.66e-13 ***
V2           2.786e-02  9.296e-03   2.997 0.002724 ** 
V3A31        1.434e-01  5.489e-01   0.261 0.793921    
V3A32       -5.861e-01  4.305e-01  -1.362 0.173348    
V3A33       -8.532e-01  4.717e-01  -1.809 0.070470 .  
V3A34       -1.436e+00  4.399e-01  -3.264 0.001099 ** 
V4A41       -1.666e+00  3.743e-01  -4.452 8.51e-06 ***
V4A410      -1.489e+00  7.764e-01  -1.918 0.055163 .  
V4A42       -7.916e-01  2.610e-01  -3.033 0.002421 ** 
V4A43       -8.916e-01  2.471e-01  -3.609 0.000308 ***
V4A44       -5.228e-01  7.623e-01  -0.686 0.492831    
V4A45       -2.164e-01  5.500e-01  -0.393 0.694000    
V4A46        3.628e-02  3.965e-01   0.092 0.927082    
V4A48       -2.059e+00  1.212e+00  -1.699 0.089297 .  
V4A49       -7.401e-01  3.339e-01  -2.216 0.026668 *  
V5           1.283e-04  4.444e-05   2.887 0.003894 ** 
V6A62       -3.577e-01  2.861e-01  -1.250 0.211130    
V6A63       -3.761e-01  4.011e-01  -0.938 0.348476    
V6A64       -1.339e+00  5.249e-01  -2.551 0.010729 *  
V6A65       -9.467e-01  2.625e-01  -3.607 0.000310 ***
V7A72       -6.691e-02  4.270e-01  -0.157 0.875475    
V7A73       -1.828e-01  4.105e-01  -0.445 0.656049    
V7A74       -8.310e-01  4.455e-01  -1.866 0.062110 .  
V7A75       -2.766e-01  4.134e-01  -0.669 0.503410    
V8           3.301e-01  8.828e-02   3.739 0.000185 ***
V9A92       -2.755e-01  3.865e-01  -0.713 0.476040    
V9A93       -8.161e-01  3.799e-01  -2.148 0.031718 *  
V9A94       -3.671e-01  4.537e-01  -0.809 0.418448    
V10A102      4.360e-01  4.101e-01   1.063 0.287700    
V10A103     -9.786e-01  4.243e-01  -2.307 0.021072 *  
V11          4.776e-03  8.641e-02   0.055 0.955920    
V12A122      2.814e-01  2.534e-01   1.111 0.266630    
V12A123      1.945e-01  2.360e-01   0.824 0.409743    
V12A124      7.304e-01  4.245e-01   1.721 0.085308 .  
V13         -1.454e-02  9.222e-03  -1.576 0.114982    
V14A142     -1.232e-01  4.119e-01  -0.299 0.764878    
V14A143     -6.463e-01  2.391e-01  -2.703 0.006871 ** 
V15A152     -4.436e-01  2.347e-01  -1.890 0.058715 .  
V15A153     -6.839e-01  4.770e-01  -1.434 0.151657    
V16          2.721e-01  1.895e-01   1.436 0.151109    
V17A172      5.361e-01  6.796e-01   0.789 0.430160    
V17A173      5.547e-01  6.549e-01   0.847 0.397015    
V17A174      4.795e-01  6.623e-01   0.724 0.469086    
V18          2.647e-01  2.492e-01   1.062 0.288249    
V19A192     -3.000e-01  2.013e-01  -1.491 0.136060    
V20A202     -1.392e+00  6.258e-01  -2.225 0.026095 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1221.73  on 999  degrees of freedom
Residual deviance:  895.82  on 951  degrees of freedom
AIC: 993.82

Number of Fisher Scoring iterations: 5
```

For the factors whose p-values are below 0.05, the log(odds) and log(odds ratios) are both statistically significant.

AIC (Akaike Information Criterion) is 993.82 which in this context is just the Residual Deviance adjusted for the number of parameters in the model.

AIC can be used to compare one model to another.

Model Evaluation - Pseudo R^2

Source (Stat Quest): [https://www.youtube.com/watch?v=C4N3_XJJ-jU (https://www.youtube.com/watch?v=C4N3_XJJ-jU)](https://www.youtube.com/watch?v=C4N3_XJJ-jU)

In [40]:

```
ll.null<-logistic$null.deviance/-2
ll.proposed<-logistic$deviance/-2
(ll.null-ll.proposed)/ll.null
```

0.266762043171136

The Pseudo R^2 can be interpreted as the overall effect size

In [43]:

```
1-pchisq(2*(ll.proposed-ll.null),df=(length(logistic$coefficients)-1))
```

0

In this case the p-value is tiny, so the R^2 calue isn't due to randomness.

Logistic Curve

In [45]:

```
predicted.data<-data.frame(probability.of.credit=logistic$fitted.values,credit=credit_onehot$
```

In [46]:

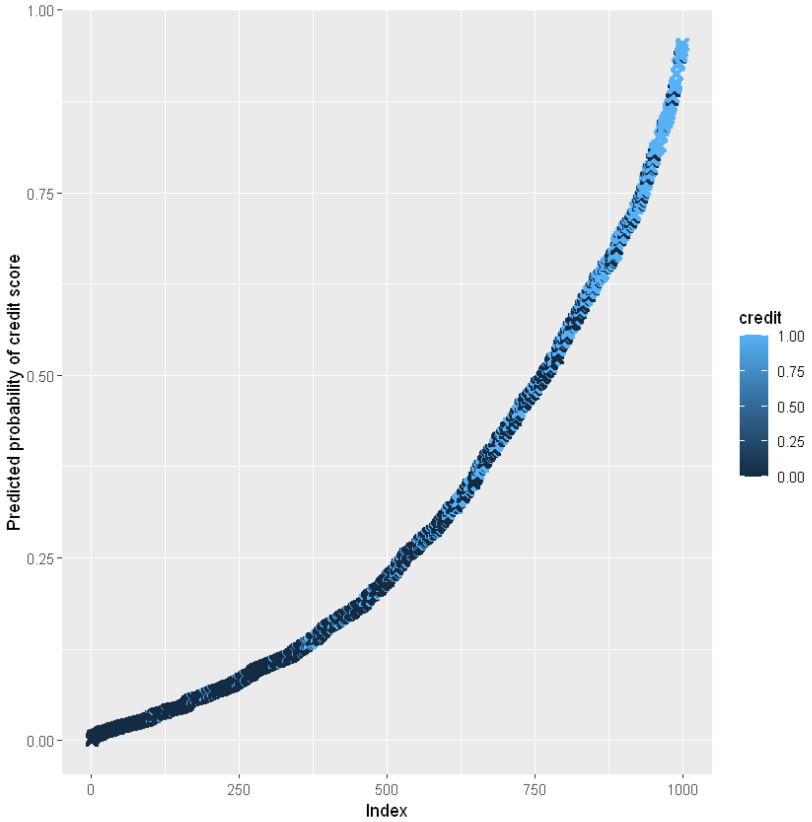
```
predicted.data<-predicted.data[order(predicted.data$probability.of.credit,decreasing=FALSE),
```

In [47]:

```
predicted.data$rank<-1:nrow(predicted.data)
```

In [55]:

```
ggplot(data=predicted.data,aes(x=rank,y=probability.of.credit))+geom_point(aes(color=credit)
```



Conclusions

We recall that 0 = Good and 1 = Bad. . It will cost more for the bank to give loans to individuals with bad credit. To ensure that, I will make the threshold for classification to be less than 0.50.