Homework 4 - Week 4

Akylas Stratigakos

12 Jun 2018

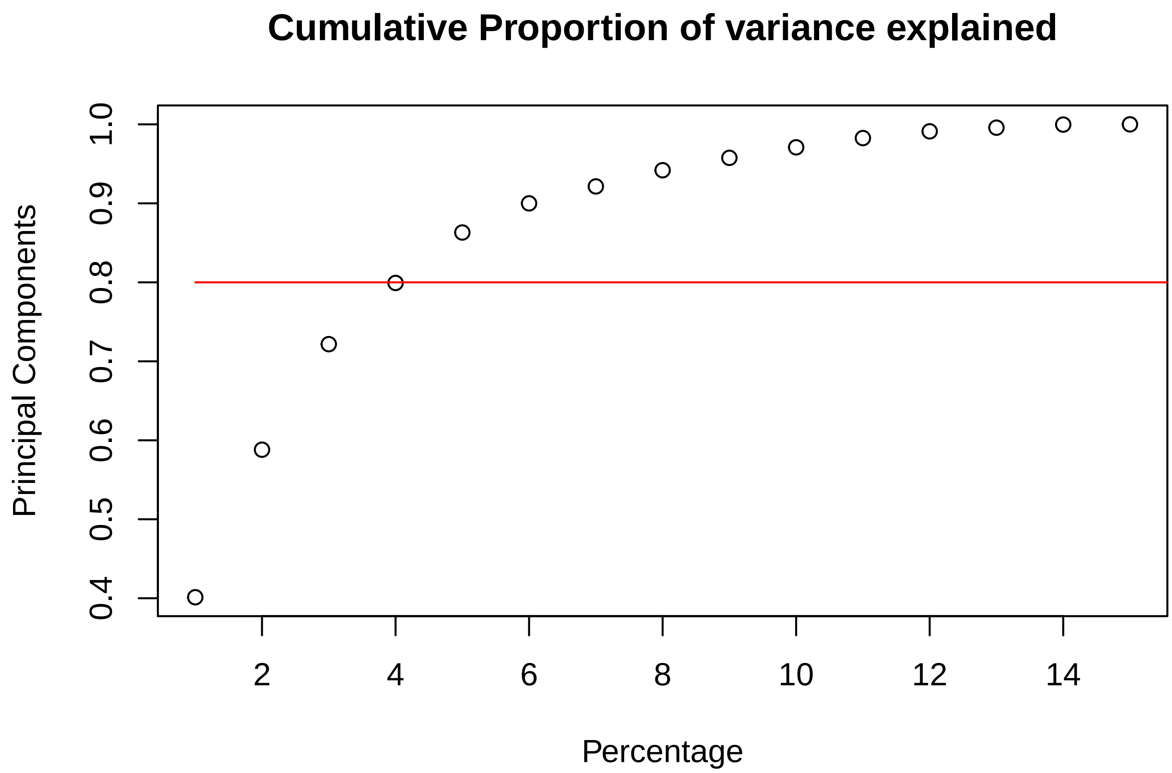
Question 9.1

In this task a linear regression model will be applied on the "uscrime" data, after reducing the dimensions of the predictors via Principal Component Analysis. Besides dimensionality reduction, PCA will also ensure no colinearity between the predictors. First, I will apply PCA on the dataset (without the response collumn, i.e. crime). In order to select a reasonable number of PCs, I plot the cumulative variance explained by the PCs (which depends on the eigenvalues of each component), and will select a number that explains around 80% of the total variance in the predictors.

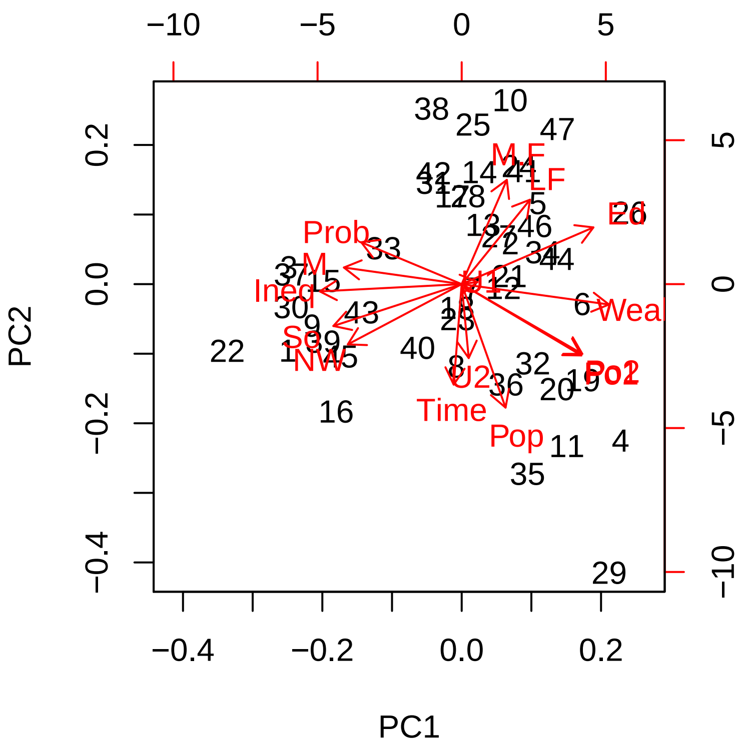
```
datapath=paste("C:/Users/a.stratigakos/Desktop/edx/Introduction to Analytics Modelling"
               ,"/Week 2/UScrime_data.txt",sep="")
uscrime<-(read.table(datapath,header =TRUE))

#PCA without the response variable
PC=prcomp(uscrime[,-16], center = TRUE, scale. = TRUE)
summary(PC)
```

```
Importance of components:
                          PC1    PC2    PC3     PC4     PC5     PC6
Standard deviation     2.4534 1.6739 1.4160 1.07806 0.97893 0.74377
Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688
Cumulative Proportion  0.4013 0.5880 0.7217 0.79920 0.86308 0.89996
                           PC7     PC8     PC9    PC10    PC11    PC12
Standard deviation     0.56729 0.55444 0.48493 0.44708 0.41915 0.35804
Proportion of Variance 0.02145 0.02049 0.01568 0.01333 0.01171 0.00855
Cumulative Proportion  0.92142 0.94191 0.95759 0.97091 0.98263 0.99117
                          PC13   PC14    PC15
Standard deviation     0.26333 0.2418 0.06793
Proportion of Variance 0.00462 0.0039 0.00031
Cumulative Proportion  0.99579 0.9997 1.00000
```

```
Threshold=matrix(0.8,nrow=16)
VarExpl=cumsum((PC$sdev)^2)/sum(PC$sdev^2)
plot(VarExpl,main='Cumulative Proportion of variance explained',
     ylab='Principal Components',
     xlab="Percentage")
lines(Threshold,type="l",col="red")
```




```
biplot(PC,choices = 1:2,scale = 1)
```




```
PCcoeff=PC$rotation
scores=PC$x
```

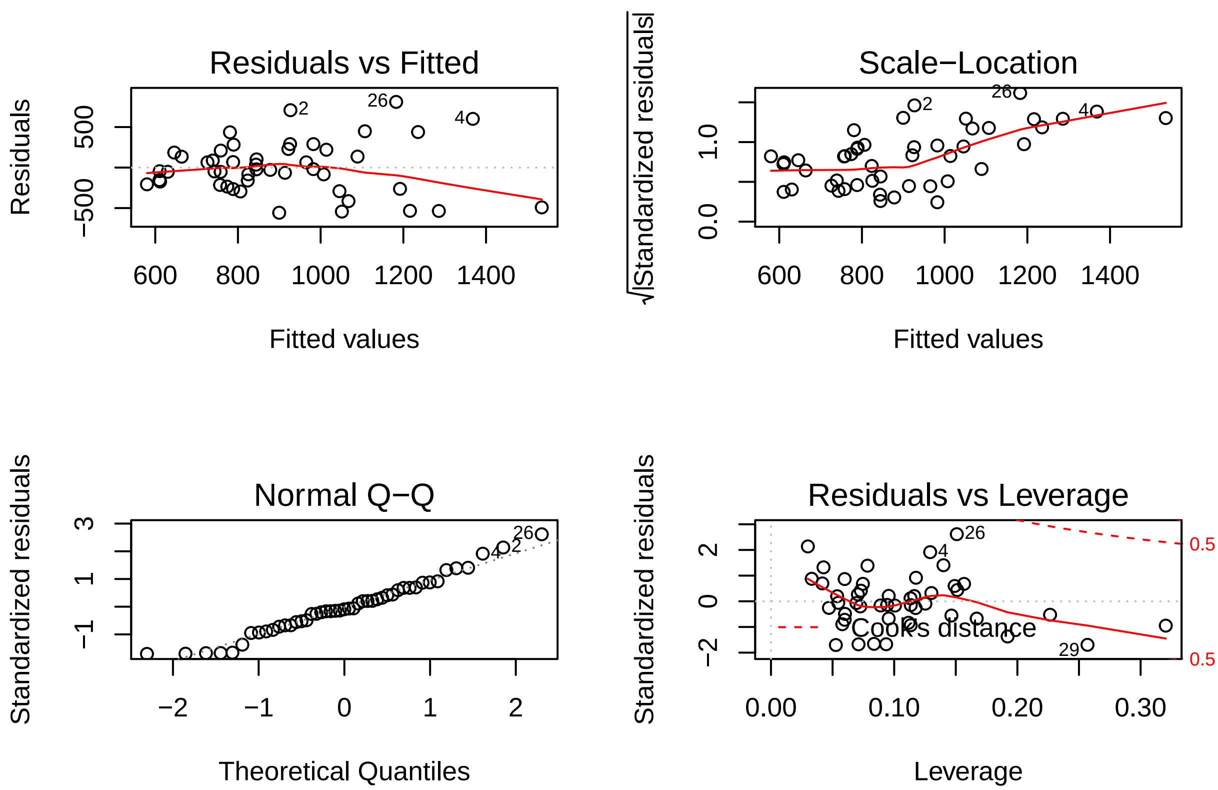
As can be seen above, the first 4 Components explain around 80% of the variance, so I'll select them as the predictors. Additionally, the biplot of the first 2 PCs is shown, which shows the **coefficients** (eigenvectors or weights) of each variable and it's usefull for qualitatevely analysis. Afterwards, I proceed with applying a linear regression using the Crime collumn as response and the **scores** (i.e. original data points projected on the new dimensions) as the predictors. This methodology will be referred to as "PCA regression".

```
Crime=uscrime[,16];Sc1=scores[,1];Sc2=scores[,2]
Sc3=scores[,3];Sc4=scores[,4]
PCdf=data.frame(Crime,Sc1,Sc2,Sc3,Sc4)
PCmodel=lm(Crime~.,data=PCdf)
summary(PCmodel)
```

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

Residuals:
    Min      1Q  Median      3Q     Max
-557.76 -210.91  -29.08  197.26  810.35

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   905.09      49.07  18.443  < 2e-16 ***
Sc1            65.22      20.22   3.225  0.00244 **
```

```
Sc2           -70.08      29.63  -2.365  0.02273 *
Sc3            25.19      35.03   0.719  0.47602
Sc4            69.45      46.01   1.509  0.13872
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 336.4 on 42 degrees of freedom
Multiple R-squared:  0.3091,    Adjusted R-squared:  0.2433
F-statistic: 4.698 on 4 and 42 DF,  p-value: 0.003178
```

```r
layout(matrix(c(1,2,3,4),2,2)) # optional 4 graphs/page
plot(PCmodel)
```




```r
a=summary(PCmodel)$coefficients
Rsq=summary(PCmodel)$r.squared
adRsq=summary(PCmodel)$adj.r.squared
Fstat=summary(PCmodel)$fstatistic
```

Above we can see some diagnostics about the PCA regression such as F-statistic, R squared and adjusted R square. Since PCA is a linear combination of the original dataset, the PCA regression will be also be this transformation multiplied by the regression's coefficients, which in turn results again in a linear combination of the original variables. Note the variables **scaled**. Due to space limitations I will only show the first coefficient and first Principal Component in the equation:

$$Y = 905.0851064 + 65.2159301 * X_{np} * \begin{pmatrix} -0.3037119 \\ -0.3308813 \\ 0.3396215 \\ 0.3086341 \\ 0.3109929 \\ 0.1761776 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \\ -0.3037119 \end{pmatrix} + ...$$

where:

$$X = \begin{pmatrix} M & So & Ed & Po1 & ... & Ineq & Prob & Time \end{pmatrix}$$

I also fitted a simple linear regression model, with the whole dataset as a predictor and plotted the diagnostics. Afterwards I used the two models to make a prediction based on the new datapoint given in Question 8.2. Note that in order to make a prediction with the PCA regression I need to map the new data point into the Principal Components. After **scaling** them, I use matrix multiplication with the corresponding eigenvectors (coeffcients), and then use them for prediction.

```r
#Simple linear reg
fit <- lm( Crime~., data=uscrime)
summary(fit)
```

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

Residuals:
    Min      1Q  Median      3Q     Max 
-395.74  -98.09   -6.69  112.99  512.67 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
M            8.783e+01  4.171e+01   2.106 0.043443 *  
So          -3.803e+00  1.488e+02  -0.026 0.979765    
Ed           1.883e+02  6.209e+01   3.033 0.004861 ** 
Po1          1.928e+02  1.061e+02   1.817 0.078892 .  
Po2         -1.094e+02  1.175e+02  -0.931 0.358830    
LF          -6.638e+02  1.470e+03  -0.452 0.654654    
M.F          1.741e+01  2.035e+01   0.855 0.398995    
Pop         -7.330e-01  1.290e+00  -0.568 0.573845    
NW           4.204e+00  6.481e+00   0.649 0.521279    
U1          -5.827e+03  4.210e+03  -1.384 0.176238    
U2           1.678e+02  8.234e+01   2.038 0.050161 .  
```

```
Wealth       9.617e-02  1.037e-01   0.928 0.360754
Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
Time        -3.479e+00  7.165e+00  -0.486 0.630708
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 209.1 on 31 degrees of freedom
Multiple R-squared:  0.8031,    Adjusted R-squared:  0.7078
F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

```
lmRsq=summary(fit)$r.squared
lmadRsq=summary(fit)$adj.r.squared
lmFstat=summary(fit)$fstatistic;
#Predict
M = 14.0;So = 0;Ed = 10.0;Po1 = 12.0;Po2 = 15.5;LF = 0.640
M.F = 94.0;Pop = 150;NW = 1.1;U1 = 0.120;U2 = 3.6;Wealth = 3200
Ineq = 20.1;Prob = 0.04;Time = 39.0
new.df <- data.frame(1,M,So, Ed, Po1, Po2, LF, M.F, Pop, NW, U1, U2, Wealth, Ineq, Prob, Time)
y=matrix(c(M,So, Ed, Po1, Po2, LF, M.F, Pop, NW, U1, U2, Wealth, Ineq, Prob, Time),nrow=1,ncol=15)
#Map data frame into the first 2 PC
newScores=((y-PC$center)/t(PC$scale))%*%PCcoeff[,1:4]
new.PCdf <- data.frame(1,Sc1=newScores[,1],Sc2=newScores[,2],Sc3=newScores[,3],Sc4=newScores[,4])
PCpred=predict(PCmodel, new.PCdf) #Scaling needs fixing
#LM pred
LMpred=predict(fit, new.df)
PCpred
```

```
     PC1
1112.678
```

```
LMpred
```

```
       1
155.4349
```

Question 10.1

In this Question the focus is to find a good model for the “uscrime” data using (a) a regression tree and (b) a random forest, while also provide some qualitative takeaways from analyzing the results.

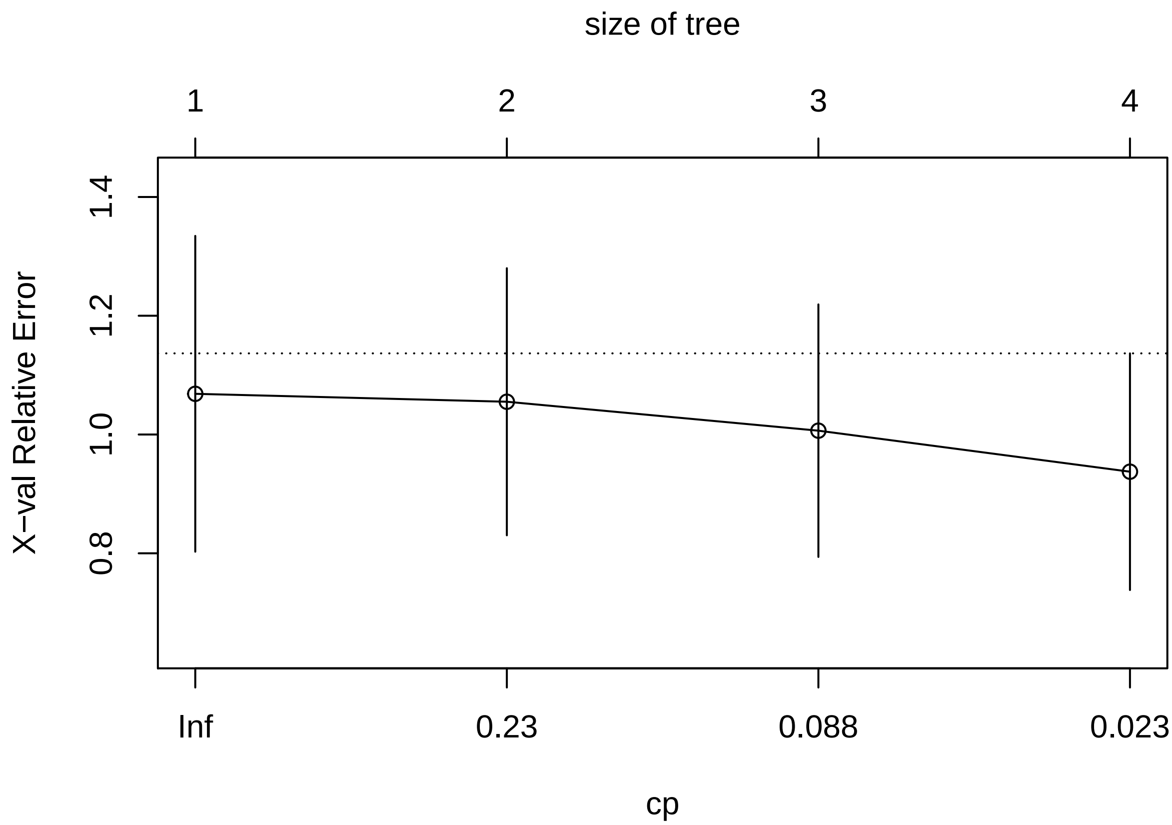
Part a: First, a regression tree model will be fitted in the dataset, by using the standard recursive binary split. For this purpose the **tree** package is utilized.

```
datapath=paste("C:/Users/a.stratigakos/Desktop/edx/Introduction to Analytics Modelling"
               ,"/Week 2/UScrime_data.txt",sep="")
uscrime<-(read.table(datapath,header =TRUE))
library(rpart)
rtree=rpart(Crime~., data=uscrime, method="anova")

printcp(rtree) # display the results
```

```
Regression tree:
rpart(formula = Crime ~ ., data = uscrime, 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 1.06854 0.26591
2 0.148143      1   0.63704 1.05518 0.22490
3 0.051732      2   0.48889 1.00648 0.21263
4 0.010000      3   0.43716 0.93739 0.19926
```

```r
plotcp(rtree) # visualize cross-validation results
```




```r
summary(rtree) # detailed summary of splits
```

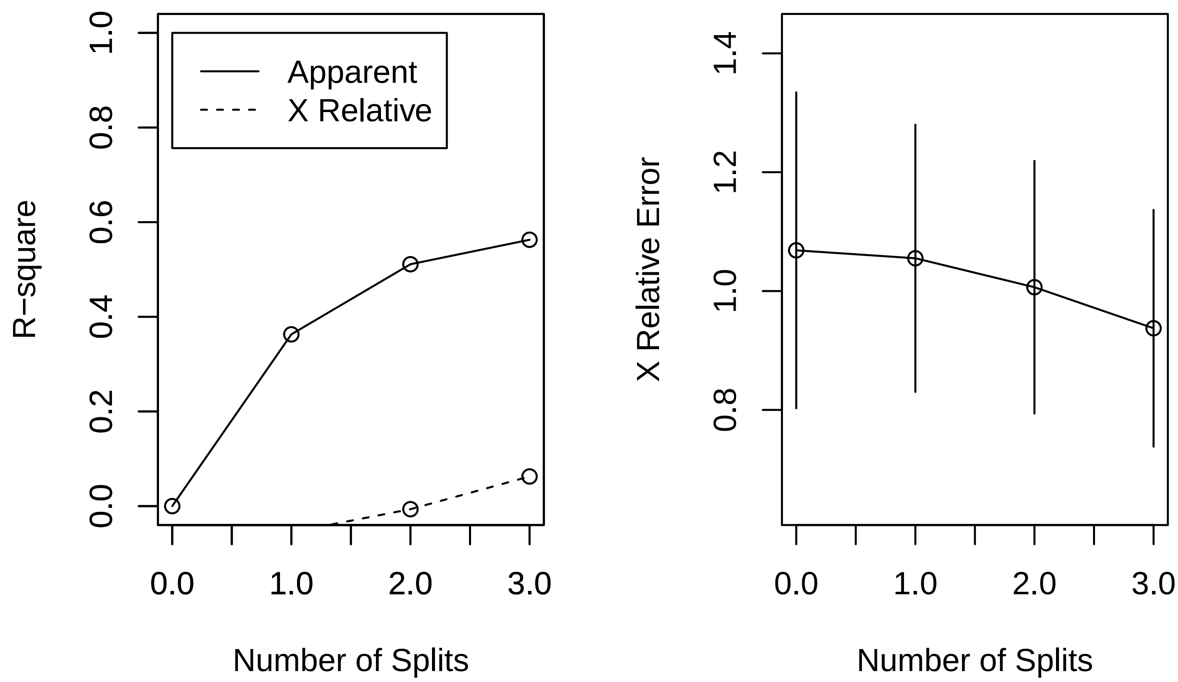
```
Call:
rpart(formula = Crime ~ ., data = uscrime, method = "anova")
  n= 47

          CP nsplit rel error    xerror      xstd
1 0.36296293      0 1.0000000 1.0685435 0.2659092
2 0.14814320      1 0.6370371 1.0551764 0.2248958
3 0.05173165      2 0.4888939 1.0064809 0.2126293
4 0.01000000      3 0.4371622 0.9373906 0.1992577

Variable importance
   Po1    Po2 Wealth   Ineq   Prob      M     NW    Pop   Time     Ed
```

```
   17     17     11     11     10     10      9      5      4      4
   LF     So
    1      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
```

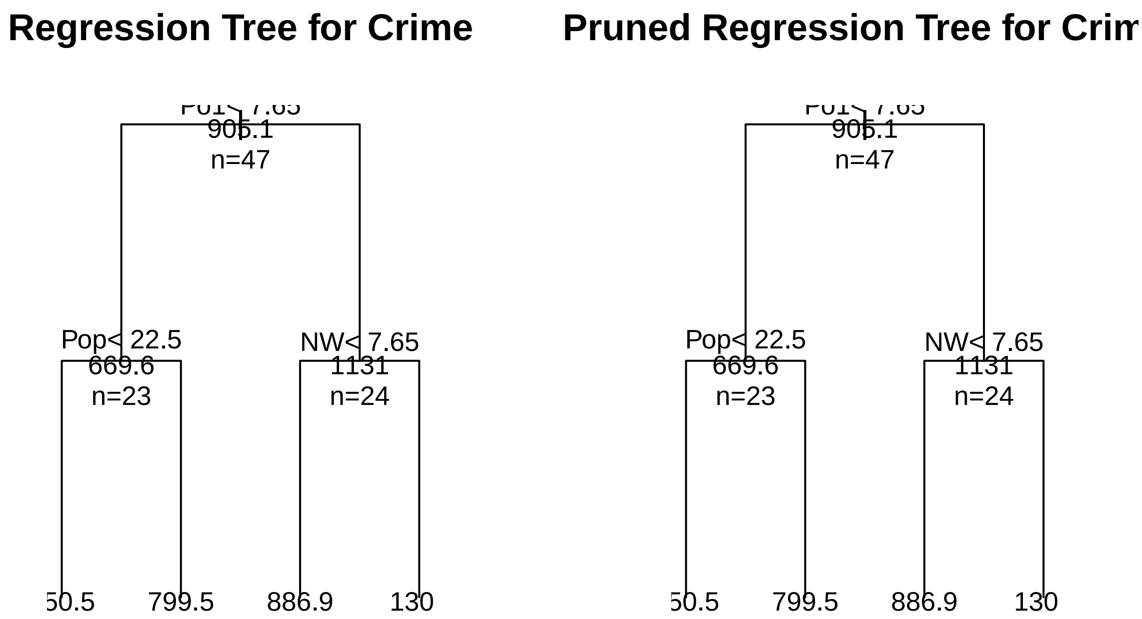
```
# create additional plots
par(mfrow=c(1,2)) # two plots on one page
rsq.rpart(rtree) # visualize cross-validation results
```

```
Regression tree:
rpart(formula = Crime ~ ., data = uscrime, 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 1.06854 0.26591
2 0.148143      1   0.63704 1.05518 0.22490
3 0.051732      2   0.48889 1.00648 0.21263
4 0.010000      3   0.43716 0.93739 0.19926
```



The function automatically selects the regression tree which minimizes the cross-validation error. A variety of diagnostics such as the cross-validation results vs number splits, importance of the variables and the improvement of the model for each split are plotted above. The most important are Police expenditure in 1960, expenditure in 1959, wealth and income inequality. We can plot the regression tree in order to visualize the results better. Additionally, we can prune the tree in order to avoid overfitting the data.

```
pfit<- prune(rtree, cp=0.01160389) # from cptable
# plot the pruned tree
par(mfrow=c(1,2)) # two plots on one page
plot(rtree, uniform=TRUE,
     main="Regression Tree for Crime")
text(rtree, use.n=TRUE, all=TRUE, cex=.8)
plot(pfit, uniform=TRUE,
     main="Pruned Regression Tree for Crime")
text(rtree, use.n=TRUE, all=TRUE, cex=.8)
```



The prune method results in the same model as the initial regression. The first split is made for **Police expenditure in 1960<7.65**. If this inequality is true, the second split is made for **State Population<22.5**, while if its not true we check whether the **Number of non-whites per 1000 people is <7.65**. Overall the tree has 3 nodes, and 4 leaves.

Part b: Now we apply a random forest algorithm in the dataset. Generally, random forests improve significantly over the simple regression trees, but they are harder to interpret. The **randomForest** package was used.

```
library(randomForest)
rf<- randomForest(Crime~.,data=uscrime,ntree=1000)
print(rf) # view results
```

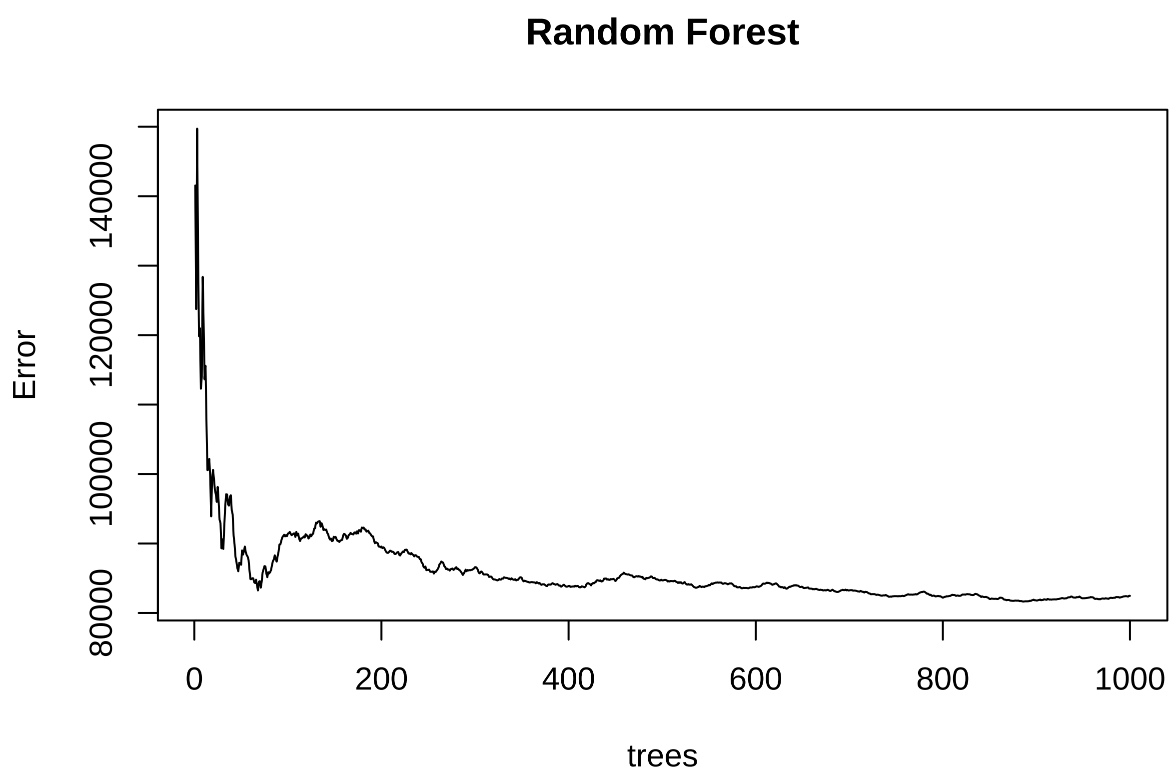
```

Call:
 randomForest(formula = Crime ~ ., data = uscrime, ntree = 1000)
               Type of random forest: regression
                     Number of trees: 1000
No. of variables tried at each split: 5

          Mean of squared residuals: 82462.27
                    % Var explained: 43.67
```

```
importance(rf) # importance of each predictor
```

```
       IncNodePurity
M          204293.55
So          18358.69
```

```
Ed          258121.29
Po1        1234820.69
Po2        1072613.32
LF          270494.90
M.F         274824.77
Pop         335363.36
NW          536890.06
U1          138162.18
U2          175420.12
Wealth      692032.60
Ineq        213707.90
Prob        712482.08
Time        222096.78
```

```r
plot(rf,main="Random Forest")
```



In the plot above we can see the relative error vs the number of trees used. For each tree 5 predictors were utilized. The **importance** function shows that the most important variables are Police expenditure in 1960, Police expenditure in 1959, probability of imprisonment and wealth.

Question 10.2

Logistic regression is a go-to method for binary classification problems and for calculating probability of events. For example it could be used to calculate the probability of penalty kick success in soccer. List of predictors could include height of the player, whether he uses right or left foot to shoot the ball, strength of his kick, arm span of the goalkeeper etc.

Question 10.3

Part 1: In this task I'll use a logistic regression in order to find a good predictive model for whether credit applicants are "good" or "bad". I define as positive state in my model the **"bad"** state. First I'll load the data and do some manipulation in order to produce more presentable results (create binary output by setting "good" as 0 and "bad" as 1). I use a 80/20 split in my dataset, for training and test set. I use the package **caret** in order to produce the confusion matrix and some helpfull diagnostics, such as accuracy, sensitivity and specificity. I'll also plot some diagnostics of the logistic regression such as z-stat of the coefficients.

```
datapath=paste("C:/Users/a.stratigakos/Desktop/edx/Introduction to Analytics Modelling"
               ,"/Week 4/German_credit.txt",sep="")
library(caret)
German_credit<-(read.table(datapath,header =TRUE))
str(German_credit)
```

```
'data.frame':   999 obs. of  21 variables:
 $ A11  : Factor w/ 4 levels "A11","A12","A13",..: 2 4 1 1 4 4 2 4 2 2 ...
 $ X6   : int  48 12 42 24 36 24 36 12 30 12 ...
 $ A34  : Factor w/ 5 levels "A30","A31","A32",..: 3 5 3 4 3 3 3 3 5 3 ...
 $ A43  : Factor w/ 10 levels "A40","A41","A410",..: 5 8 4 1 8 4 2 5 1 1 ...
 $ X1169: int  5951 2096 7882 4870 9055 2835 6948 3059 5234 1295 ...
 $ A65  : Factor w/ 5 levels "A61","A62","A63",..: 1 1 1 1 5 3 1 4 1 1 ...
 $ A75  : Factor w/ 5 levels "A71","A72","A73",..: 3 4 4 3 3 5 3 4 1 2 ...
 $ X4   : int  2 2 2 3 2 3 2 2 4 3 ...
 $ A93  : Factor w/ 4 levels "A91","A92","A93",..: 2 3 3 3 3 3 3 1 4 2 ...
 $ A101 : Factor w/ 3 levels "A101","A102",..: 1 1 3 1 1 1 1 1 1 1 ...
 $ X4.1 : int  2 3 4 4 4 4 2 4 2 1 ...
 $ A121 : Factor w/ 4 levels "A121","A122",..: 1 1 2 4 4 2 3 1 3 3 ...
 $ X67  : int  22 49 45 53 35 53 35 61 28 25 ...
 $ A143 : Factor w/ 3 levels "A141","A142",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ A152 : Factor w/ 3 levels "A151","A152",..: 2 2 3 3 3 2 1 2 2 1 ...
 $ X2   : int  1 1 1 2 1 1 1 1 2 1 ...
 $ A173 : Factor w/ 4 levels "A171","A172",..: 3 2 3 3 2 3 4 2 4 3 ...
 $ X1   : int  1 2 2 2 2 1 1 1 1 1 ...
 $ A192 : Factor w/ 2 levels "A191","A192": 1 1 1 1 2 1 2 1 1 1 ...
 $ A201 : Factor w/ 2 levels "A201","A202": 1 1 1 1 1 1 1 1 1 1 ...
 $ X1.1 : int  2 1 1 2 1 1 1 1 2 2 ...
```

```
Actual<-ifelse(German_credit[,21]==1,"GOOD", "BAD")
German_credit[which(German_credit[,21]==1),21]=0  #GOOD
German_credit[which(German_credit[,21]==2),21]=1  #BAD
spec = c(train = .80, test = .20)
g = sample(cut(seq(nrow(German_credit)), nrow(German_credit)*cumsum(c(0,spec)),labels = names(spec)))
traind=as.data.frame(German_credit[g=="train",])
testd=as.data.frame(German_credit[g=="test",])
#transform to X=0 GOOD, X=1 BAD
#Predictive model with logistic regression
logitMod <- glm(X1.1~., data=traind, family=binomial(link="logit"))
summary(logitMod)
```

```

Call:
glm(formula = X1.1 ~ ., family = binomial(link = "logit"), data = traind)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2575  -0.6780  -0.3515   0.6963   2.6243
```

```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  7.630e-01  1.240e+00   0.615 0.538278
A11A12      -5.980e-01  2.492e-01  -2.399 0.016420 *
A11A13      -1.268e+00  4.339e-01  -2.923 0.003471 **
A11A14      -1.891e+00  2.667e-01  -7.092 1.32e-12 ***
X6           2.505e-02  1.053e-02   2.379 0.017337 *
A34A31       2.680e-01  6.256e-01   0.428 0.668424
A34A32      -5.607e-01  5.029e-01  -1.115 0.264931
A34A33      -4.490e-01  5.396e-01  -0.832 0.405387
A34A34      -1.421e+00  5.091e-01  -2.791 0.005261 **
A43A41      -1.614e+00  4.162e-01  -3.878 0.000105 ***
A43A410     -1.901e+00  8.391e-01  -2.265 0.023497 *
A43A42      -9.697e-01  2.948e-01  -3.289 0.001004 **
A43A43      -1.018e+00  2.853e-01  -3.568 0.000359 ***
A43A44      -6.799e-01  8.065e-01  -0.843 0.399214
A43A45       1.809e-02  6.082e-01   0.030 0.976273
A43A46      -2.735e-01  4.528e-01  -0.604 0.545860
A43A48      -2.080e+00  1.306e+00  -1.592 0.111307
A43A49      -7.535e-01  3.797e-01  -1.984 0.047220 *
X1169        1.213e-04  4.969e-05   2.441 0.014627 *
A65A62      -2.976e-01  3.180e-01  -0.936 0.349359
A65A63      -4.643e-01  4.299e-01  -1.080 0.280094
A65A64      -1.456e+00  5.937e-01  -2.453 0.014173 *
A65A65      -1.097e+00  3.108e-01  -3.529 0.000417 ***
A75A72       2.139e-01  4.715e-01   0.454 0.650027
A75A73       3.307e-01  4.481e-01   0.738 0.460518
A75A74      -5.892e-01  4.838e-01  -1.218 0.223213
A75A75      -8.321e-02  4.533e-01  -0.184 0.854350
X4           3.515e-01  1.016e-01   3.460 0.000540 ***
A93A92      -7.354e-02  4.294e-01  -0.171 0.864030
A93A93      -4.435e-01  4.191e-01  -1.058 0.289925
A93A94      -2.787e-01  5.258e-01  -0.530 0.596049
A101A102     3.098e-01  4.678e-01   0.662 0.507772
A101A103    -1.097e+00  5.004e-01  -2.191 0.028437 *
X4.1         5.084e-02  9.911e-02   0.513 0.607978
A121A122     2.133e-01  2.928e-01   0.728 0.466371
A121A123     2.344e-01  2.702e-01   0.867 0.385786
A121A124     8.491e-01  4.776e-01   1.778 0.075432 .
X67         -1.367e-02  1.049e-02  -1.303 0.192427
A143A142    -6.222e-01  4.828e-01  -1.289 0.197547
A143A143    -8.858e-01  2.736e-01  -3.238 0.001204 **
A152A152    -4.019e-01  2.681e-01  -1.499 0.133877
A152A153    -6.747e-01  5.389e-01  -1.252 0.210574
X2           1.571e-01  2.256e-01   0.697 0.486016
A173A172     3.819e-01  7.684e-01   0.497 0.619145
A173A173     3.090e-01  7.359e-01   0.420 0.674548
A173A174     4.466e-01  7.625e-01   0.586 0.558038
X1          -2.826e-02  2.919e-01  -0.097 0.922868
A192A192    -3.959e-01  2.305e-01  -1.717 0.085902 .
A201A202    -1.224e+00  6.569e-01  -1.864 0.062358 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

    Null deviance: 980.03  on 798  degrees of freedom
Residual deviance: 700.80  on 750  degrees of freedom
AIC: 798.8

Number of Fisher Scoring iterations: 5
```

```r
predicted <- predict(logitMod, testd, type="response")  # predicted scores
Prediction=ifelse(predicted>0.5,'BAD',"GOOD")
cM=confusionMatrix(table(Prediction,Actual[g=="test"]))
confusionMatrix(table(Prediction,Actual[g=="test"]))
```

```
Confusion Matrix and Statistics


Prediction BAD GOOD
      BAD   26   22
      GOOD  32  120

               Accuracy : 0.73
                 95% CI : (0.6628, 0.7902)
    No Information Rate : 0.71
    P-Value [Acc > NIR] : 0.2954

                  Kappa : 0.3091
 Mcnemar's Test P-Value : 0.2207

            Sensitivity : 0.4483
            Specificity : 0.8451
         Pos Pred Value : 0.5417
         Neg Pred Value : 0.7895
             Prevalence : 0.2900
         Detection Rate : 0.1300
   Detection Prevalence : 0.2400
      Balanced Accuracy : 0.6467

       'Positive' Class : BAD
```

```r
Acc=round(cM$overall[1],digits=2)
Sens=round(cM$byClass[1],digits=2)
Spec=cM$byClass[2]
```

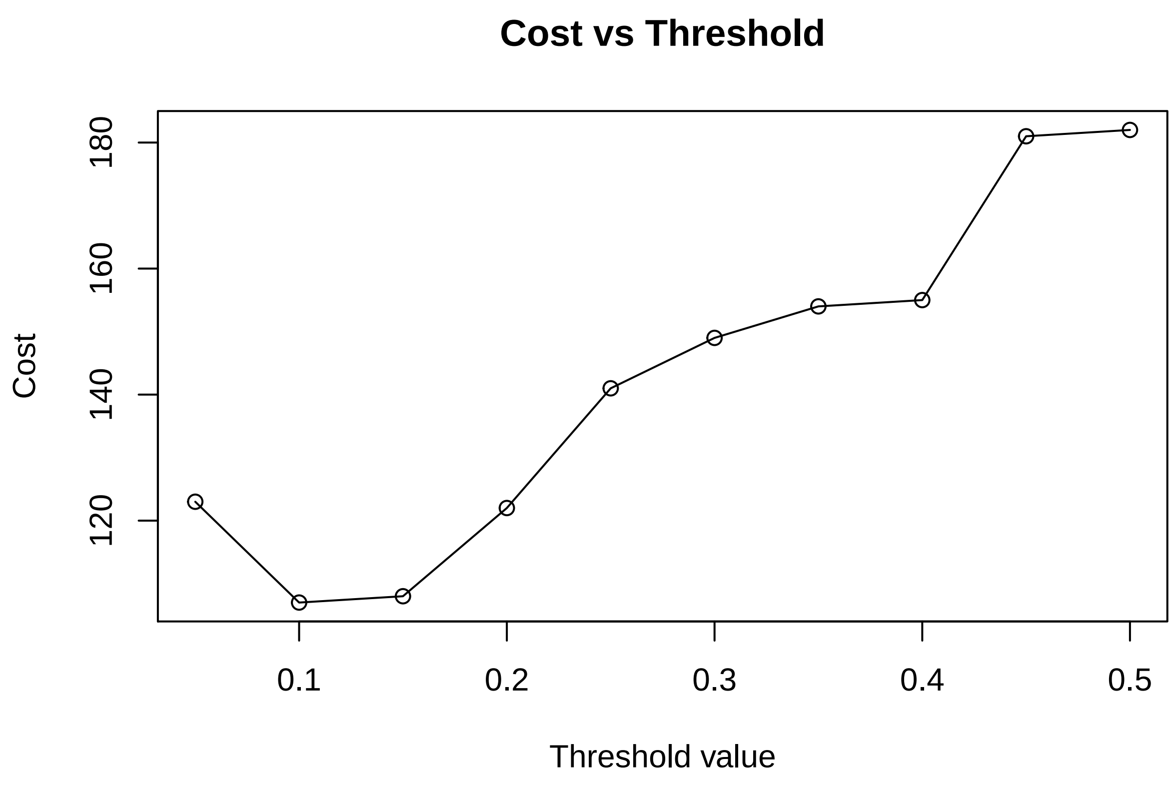
The coefficients are not all important as evident from the diagnostics, which means the model could be improved via subset selection or some other method. In the above example the **Threshold** for selecting one of the states (i.e. classifying) of the response **0.5**. The accuracy of the model was **0.73** which is pretty good. If we did not know any more specifics about the problem at hand this would effective. In the case that the bad valued differently the various errors, for example a "bad" creditor given a loan is more costly than a "good" one being denied a loan, we would like to check the sensitive and specifity of the model. In this case the model has sensitivity of **0.45**, which is depentant on the True positive and False negative errors ("bad" creditors classified as "good"). Based on this the model does not perform well and we shall see in the following part how to optimize it.

Part 2: Given that the **false negative errors** ("bad" creditors classified as "good") is 5 times more costly than the false positive errors ("good" creditors identifyied as "bad") we would like to have a our model to minimize the **Total cost function**. In this case the cost function is:

$$Cost = FN * 5 * C + FP * C$$

where C= positive constant. We could have plotted a **ROC** curve and select an appropriate cut-off point, but since we have the specific cost function I have decided to follow a different approach. This is equivalent to saying that we would like a model with higher **sensitivity**. In the following code I tried to create a simple optimization with a "for" loop, rather than use a premade package. For this task, I trained the model with 80% of the data, since the model remains the same for different Thresholds, calculated the cost function based on the perfomance on the test set for various Thresholds and selected the model which **minimizes** the total cost. I do not think there is need for validation set here, since the model remains the same.

```
Cost=matrix(0L,nrow=10,ncol=1)
range=seq(0.05,0.5,0.05)
for (i in 1:10){
  Pred2=ifelse(predicted>range[i],'BAD',"GOOD")
  temp=confusionMatrix(table(Pred2,Actual[g=="test"]))
  Cost[i]=temp$table[2,1]*5+temp$table[1,2]
}
plot(range,Cost,xlab="Threshold value",ylab="Cost",main="Cost vs Threshold",type="o")
```




```
OptTh=range[which(Cost==min(Cost))]
final=confusionMatrix(table(ifelse(predicted>OptTh,
                                   'BAD',"GOOD"),Actual[g=="test"]))
Sens=round(final$byClass[1],digits=2)
```

In the above example I used various lower Thresholds in order to increase the sensitivity of the model. The **minimum Cost** was found for **Threshold value equal to 0.1**. In this case the sensitivity of the model was **0.93**, which is pretty good.