fraud_model

June 8, 2018

1 Earnings Manipulation

1.1 By Kumar Rahul

The analysis is on company financial manipulations and devise algorithm to identify a manipulater from a non manipulater based on the financial ratios reported by the companies. There are a total of 1239 observations in the data set. Out of these 1239 observations, there are 1200 non manipulaters and 39 manipulaters.

1. Look for different types of model which can be built using R. Also has a guideline for fine tuning paramters
2. Refer link to know random forest and Refer to know about OOB error
3. Demonstration of some of the bagging and boosting algorithm
4. Understand the logic for bagging in logistic regression
5. Interpret the tree structure generated out of random forest model

---

Not all the packages are available for installation through anaconda r-essentials. To install the packages which are not available through anaconda framework, use the below code chunk:

```
In [1]: #install.packages("inTrees", "/Users/Rahul/anaconda3/lib/R/library")
        #install.packages("DMwR", "/Users/Rahul/anaconda3/lib/R/library")
        #install.packages("UBL", "/Users/Rahul/anaconda3/lib/R/library")
        #install.packages("adabag", "/Users/Rahul/anaconda3/lib/R/library")
        #install.packages("tictoc", "/Users/Rahul/anaconda3/lib/R/library")
        #install.packages("doMC", "/Users/Rahul/anaconda3/lib/R/library")
```

```
In [2]: library(caret)          #for split and model accuracy
        library(DMwR)           #for SMOTE Sampling
        library(randomForest)
        library(ROCR)           #for ROC Plot
        library(e1071)
        library(xgboost)        #to implement xgbTree
        #library(rattle)        #print the business rules for the model
        library(inTrees)        #to extract the business rules from rf model
        library(UBL)
        library(tictoc)         #to record the time elapsed
```

```
        library(parallel)
        library(doParallel)
        library(doMC)
        setwd("/Users/Rahul/Documents/Rahul Office/IIMB/Work @ IIMB/Company Fraud")
```

```
Loading required package: lattice
Loading required package: ggplot2
Loading required package: grid
randomForest 4.6-12
Type rfNews() to see new features/changes/bug fixes.

Attaching package: randomForest

The following object is masked from package:ggplot2:

    margin

Loading required package: gplots

Attaching package: gplots

The following object is masked from package:stats:

    lowess

Loading required package: MBA
Loading required package: gstat
Loading required package: automap
Loading required package: sp
Loading required package: foreach
Loading required package: iterators
```

1.2 Preparing data

Read data from a specified location

```
In [3]: raw_data <- read.csv("/Users/Rahul/Documents/Rahul Office/IIMB/Work @ IIMB/Company Frau
                             head=TRUE,na.strings=c("", " ", "NA"), sep=",")

        filter_data <- raw_data[,-c(1)]
```

Define an 70%/30% train/test split of the dataset

```
In [4]: set.seed(4121)
        trainIndex <- createDataPartition(filter_data$Manipulater, p = 0.70, list=FALSE)
        train_df <- filter_data[ trainIndex,]
        test_df <- filter_data[-trainIndex,]
```

Prepare and run numerical summaries

```
In [5]: summary(train_df) #summary of the data
        train_df <- na.omit(train_df) # listwise deletion of missing
        test_df <- na.omit(test_df) # listwise deletion of missing
```

```
      DSRI              GMI                AQI                SGI          
 Min.   : 0.0000   Min.   :-20.8118   Min.   :-21.7338   Min.   : 0.06454  
 1st Qu.: 0.8876   1st Qu.:  0.9253   1st Qu.:  0.7856   1st Qu.: 0.97341  
 Median : 1.0200   Median :  1.0000   Median :  1.0079   Median : 1.09614  
 Mean   : 1.1387   Mean   :  0.9778   Mean   :  1.0763   Mean   : 1.13740  
 3rd Qu.: 1.1872   3rd Qu.:  1.0507   3rd Qu.:  1.2110   3rd Qu.: 1.20608  
 Max.   :15.3435   Max.   : 46.4667   Max.   : 52.8867   Max.   :13.06465  
      DEPI              SGAI              ACCR               LEVI        
 Min.   :0.06882   Min.   : 0.0000   Min.   :-0.68226   Min.   : 0.0000  
 1st Qu.:0.93554   1st Qu.: 0.9008   1st Qu.:-0.07631   1st Qu.: 0.9232  
 Median :1.00000   Median : 1.0002   Median :-0.03004   Median : 1.0133  
 Mean   :1.02915   Mean   : 1.1073   Mean   :-0.03045   Mean   : 1.0574  
 3rd Qu.:1.07637   3rd Qu.: 1.1290   3rd Qu.: 0.02016   3rd Qu.: 1.1154  
 Max.   :5.39387   Max.   :49.3018   Max.   : 0.95989   Max.   :13.0586  
 Manipulater C.MANIPULATOR    
 No :840     Min.   :0.00000  
 Yes: 28     1st Qu.:0.00000  
             Median :0.00000  
             Mean   :0.03226  
             3rd Qu.:0.00000  
             Max.   :1.00000  
```

Train and test dataset with needed variables

```
In [6]: model_df <- as.data.frame(filter_data[,c(#"DSRI",
                                                  #"GMI",
                                                  "AQI",
                                                  #"SGI",
                                                  "DEPI",
                                                  "SGAI",
                                                  "ACCR",
                                                  "LEVI",
                                                  "Manipulater"
        )])

        model_train_df <- as.data.frame(train_df[,c(#"DSRI",
                                                     #"GMI",
                                                     "AQI",
                                                     #"SGI",
                                                     "DEPI",
                                                     "SGAI",
                                                     "ACCR",
```

```
                                                    "LEVI",
                                                    "Manipulater"
)])

model_test_df <- as.data.frame(test_df[,c(#"DSRI",
                                                  #"GMI",
                                                  "AQI",
                                                  #"SGI",
                                                  "DEPI",
                                                  "SGAI",
                                                  "ACCR",
                                                  "LEVI",
                                                  "Manipulater"
)])
```

Corelation amongst variable The below chunk of code will show the co-relation if any between the numerical variables. The function **highlyCorelated()** shows the variables which are corelated with an absolute corelation of more than 0.6. In this case there are no variables which are highly corelated.

```
In [7]: correlation_matrix <- cor(model_df[,c(1:5)])
        print(correlation_matrix)
        # find attributes that are highly corrected (ideally >0.7)
        highly_correlated <- findCorrelation(correlation_matrix, cutoff = 0.6, names = TRUE)
        print(highly_correlated)
```

```
              AQI        DEPI         SGAI        ACCR        LEVI
AQI   1.000000000 -0.02124161  0.003712316 -0.04542383  0.07027302
DEPI -0.021241615  1.00000000 -0.067247329 -0.01661336 -0.01271157
SGAI  0.003712316 -0.06724733  1.000000000 -0.09066795  0.02174950
ACCR -0.045423827 -0.01661336 -0.090667950  1.00000000 -0.01163113
LEVI  0.070273016 -0.01271157  0.021749500 -0.01163113  1.00000000
character(0)
```

1.3 Caret Package

caret is a useful and a robust package which helps to set a generic framework to implement any kind of model in R. Some of the algorithm's which can be implemented using caret package are:

```
In [8]: names(getModelInfo())

        #getModelInfo()$glm
```

1. 'ada' 2. 'AdaBag' 3. 'AdaBoost.M1' 4. 'adaboost' 5. 'amdai' 6. 'ANFIS' 7. 'avNNet' 8. 'awnb' 9. 'awtan' 10. 'bag' 11. 'bagEarth' 12. 'bagEarthGCV' 13. 'bagFDA' 14. 'bagFDAGCV' 15. 'bam' 16. 'bartMachine' 17. 'bayesglm' 18. 'binda' 19. 'blackboost' 20. 'blasso' 21. 'blassoAveraged' 22. 'bridge' 23. 'brnn' 24. 'BstLm' 25. 'bstSm' 26. 'bstTree' 27. 'C5.0' 28. 'C5.0Cost' 29. 'C5.0Rules'

30. 'C5.0Tree' 31. 'cforest' 32. 'chaid' 33. 'CSimca' 34. 'ctree' 35. 'ctree2' 36. 'cubist' 37. 'dda' 38. 'deepboost' 39. 'DENFIS' 40. 'dnn' 41. 'dwdLinear' 42. 'dwdPoly' 43. 'dwdRadial' 44. 'earth' 45. 'elm' 46. 'enet' 47. 'evtree' 48. 'extraTrees' 49. 'fda' 50. 'FH.GBML' 51. 'FIR.DM' 52. 'foba' 53. 'FRBCS.CHI' 54. 'FRBCS.W' 55. 'FS.HGD' 56. 'gam' 57. 'gamboost' 58. 'gamLoess' 59. 'gamSpline' 60. 'gaussprLinear' 61. 'gaussprPoly' 62. 'gaussprRadial' 63. 'gbm_h2o' 64. 'gbm' 65. 'gcvEarth' 66. 'GFS.FR.MOGUL' 67. 'GFS.LT.RS' 68. 'GFS.THRIFT' 69. 'glm.nb' 70. 'glm' 71. 'glmboost' 72. 'glmnet_h2o' 73. 'glmnet' 74. 'glmStepAIC' 75. 'gpls' 76. 'hda' 77. 'hdda' 78. 'hdrda' 79. 'HYFIS' 80. 'icr' 81. 'J48' 82. 'JRip' 83. 'kernelpls' 84. 'kknn' 85. 'knn' 86. 'krlsPoly' 87. 'krlsRadial' 88. 'lars' 89. 'lars2' 90. 'lasso' 91. 'lda' 92. 'lda2' 93. 'leapBackward' 94. 'leapForward' 95. 'leapSeq' 96. 'Linda' 97. 'lm' 98. 'lmStepAIC' 99. 'LMT' 100. 'loclda' 101. 'logicBag' 102. 'LogitBoost' 103. 'logreg' 104. 'lssvmLinear' 105. 'lssvmPoly' 106. 'lssvmRadial' 107. 'lvq' 108. 'M5' 109. 'M5Rules' 110. 'manb' 111. 'mda' 112. 'Mlda' 113. 'mlp' 114. 'mlpKerasDecay' 115. 'mlpKerasDecayCost' 116. 'mlpKerasDropout' 117. 'mlpKerasDropoutCost' 118. 'mlpML' 119. 'mlpSGD' 120. 'mlpWeightDecay' 121. 'mlpWeightDecayML' 122. 'monmlp' 123. 'msaenet' 124. 'multinom' 125. 'mxnet' 126. 'mxnetAdam' 127. 'naive_bayes' 128. 'nb' 129. 'nbDiscrete' 130. 'nbSearch' 131. 'neuralnet' 132. 'nnet' 133. 'nnls' 134. 'nodeHarvest' 135. 'null' 136. 'OneR' 137. 'ordinalNet' 138. 'ORFlog' 139. 'ORFpls' 140. 'ORFridge' 141. 'ORFsvm' 142. 'ownn' 143. 'pam' 144. 'parRF' 145. 'PART' 146. 'partDSA' 147. 'pcaNNet' 148. 'pcr' 149. 'pda' 150. 'pda2' 151. 'penalized' 152. 'PenalizedLDA' 153. 'plr' 154. 'pls' 155. 'plsRglm' 156. 'polr' 157. 'ppr' 158. 'PRIM' 159. 'protoclass' 160. 'pythonKnnReg' 161. 'qda' 162. 'QdaCov' 163. 'qrf' 164. 'qrnn' 165. 'randomGLM' 166. 'ranger' 167. 'rbf' 168. 'rbfDDA' 169. 'Rborist' 170. 'rda' 171. 'regLogistic' 172. 'relaxo' 173. 'rf' 174. 'rFerns' 175. 'RFlda' 176. 'rfRules' 177. 'ridge' 178. 'rlda' 179. 'rlm' 180. 'rmda' 181. 'rocc' 182. 'rotationForest' 183. 'rotationForestCp' 184. 'rpart' 185. 'rpart1SE' 186. 'rpart2' 187. 'rpartCost' 188. 'rpartScore' 189. 'rqlasso' 190. 'rqnc' 191. 'RRF' 192. 'RRFglobal' 193. 'rrlda' 194. 'RSimca' 195. 'rvmLinear' 196. 'rvmPoly' 197. 'rvmRadial' 198. 'SBC' 199. 'sda' 200. 'sdwd' 201. 'simpls' 202. 'SLAVE' 203. 'slda' 204. 'smda' 205. 'snn' 206. 'sparseLDA' 207. 'spikeslab' 208. 'spls' 209. 'stepLDA' 210. 'stepQDA' 211. 'superpc' 212. 'svmBoundrangeString' 213. 'svmExpoString' 214. 'svmLinear' 215. 'svmLinear2' 216. 'svmLinear3' 217. 'svmLinearWeights' 218. 'svmLinearWeights2' 219. 'svmPoly' 220. 'svmRadial' 221. 'svmRadialCost' 222. 'svmRadialSigma' 223. 'svmRadialWeights' 224. 'svmSpectrumString' 225. 'tan' 226. 'tanSearch' 227. 'treebag' 228. 'vbmpRadial' 229. 'vglmAdjCat' 230. 'vglmContRatio' 231. 'vglmCumulative' 232. 'widekernelpls' 233. 'WM' 234. 'wsrf' 235. 'xgbDART' 236. 'xgbLinear' 237. 'xgbTree' 238. 'xyf'

1.4 Bagging Model

Bagging is the process of taking bootstrap sample and then aggreagting the model learned on each sample. Each of the models are trained independently on the N observations picked randomly from N observations in the original dataset (with replacement). The models can be trained parallely as the training is based on independent samples. Since models are trained on different but overlapping samples of the original data, the predictions from different models will be different.

1.4.1 Bagging models in R

The algorithms in bagging are:

1. Bagged Adaboost: ***adabag()*** Required Package is **adabag, plyr**
2. Bagged CART: ***treebag()*** Required Package is **ipred, e1071, plyr**

3. Bagged Flexible Discriminant Analysis: ***bagFDA()*** Required Package is **earth, mda**
4. Bagged Logic Regression: ***logicBag()*** Required Package is **logicFS**
5. Bagged MARS: ***bagEarth()*** Required Package is **earth**
6. Bagged Model: ***bag()*** Required Package is **caret**
7. Ensemble of Generalized Linear Models: ***randomGLM()*** Required Package is **randomGLM**
8. Model Averaged Neural Network: ***avNNET()*** Required Package is **nnet**
9. Quantile Regression Neural Network: ***qrnn()*** Required Package is **qrnn**
10. Random Ferns: ***rFerns()*** Required Package is **rFerns**

The below methods are all applicable to implement random forest as a bagging algorithm:

11. Parallel Random Forest: ***parRF()*** Required Package is **e1071, randomForest, foreach**
12. Quantile Random Forest: ***qrf()*** Required Package is **quantregForest**
13. Conditional Inference Random Forest: ***cforest()*** Required Package is **party**
14. Random Forest: ***ranger()*** Required Package is **e1071, ranger**
15. Random Forest: ***Rborist()*** Required Package is **Rborist**
16. Random Forest: ***rf()*** Required Package is **randomForest**
17. Random Forest by Randomization: ***extraTrees()*** Required Package is **extraTrees**
18. Random Forest rule based Model: ***rfRules()*** Required Package is **randomForest, inTrees, plyr**
19. Regularized Random Forest: ***RRF()*** Required Package is **randomForest, RRF**
20. Regularized Random Forest: ***RRFglobal()*** Required Package is **RRF**
21. Weighted Subspace Random Forest: ***wsrf()*** Required Package is **wsrf**

###Random Forest with bootstrap sampling Random forests is one of the algorithm which uses bagging as a technique. In the below code chunk we will use bootstrap sampling to implement bagging using rf method. This means that if there are 100 observations in a training dataset the resulting sample will select 100 samples with replacement.

The below code chunk sets some of the control parameters

```
In [9]: tic("Total Time for Bagging and Boosting")

In [10]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='none',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE,allowParallel=TRUE)
```

After setting the control paramters, the model is run

```
In [11]: num_cores <- makeCluster(detectCores()-5)
         num_cores

socket cluster with 3 nodes on host localhost
```

```
In [12]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("RF Bagging with Bootstrap Sample")

         set.seed(4121)
         rf_bootstrap_model <- train(model_train_df[,1:5], model_train_df[,6],
                            method='rf',
                            trControl=objControl, ntree = 500,
                            metric = "ROC")
         stopCluster(num_cores)
         toc()
```

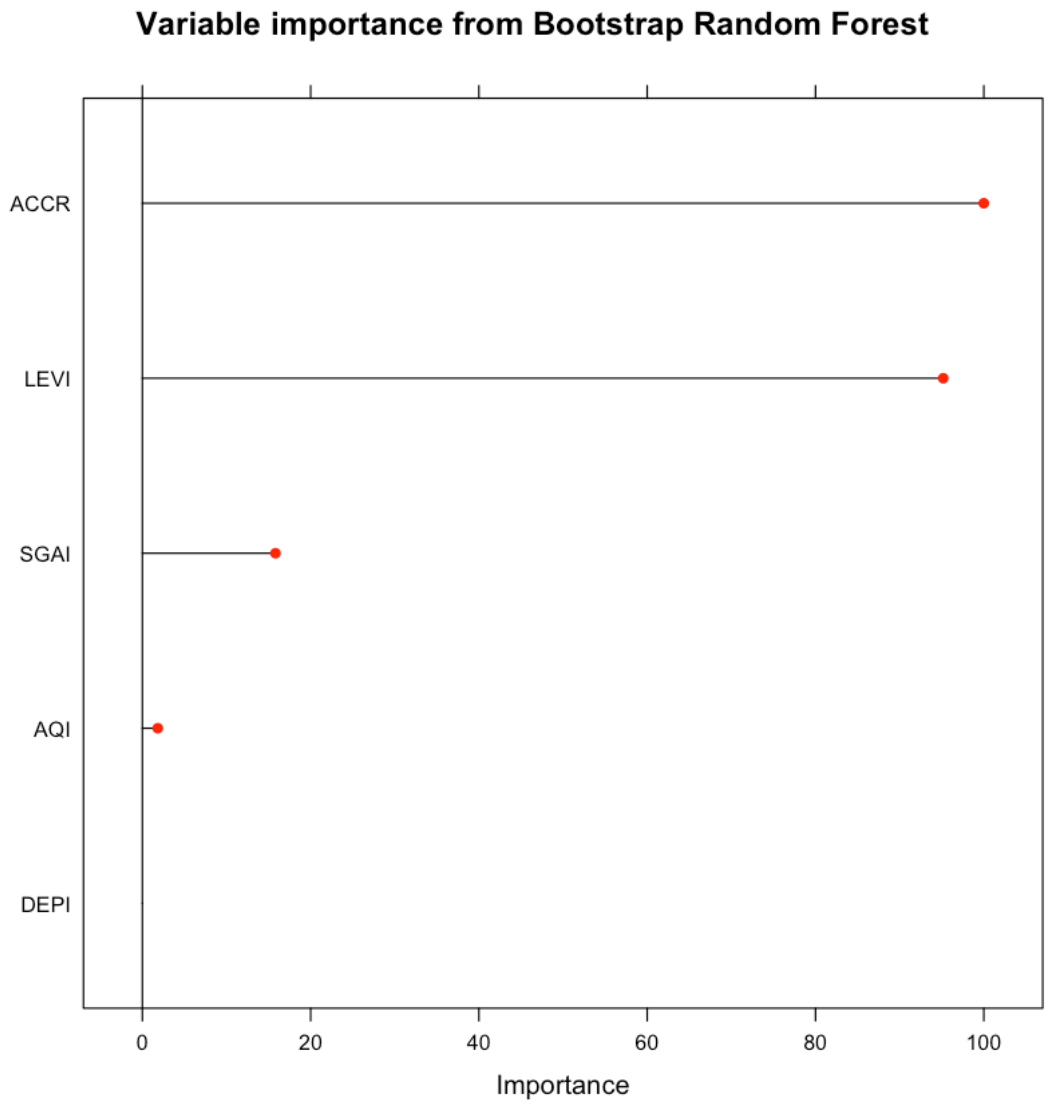
```
RF Bagging with Bootstrap Sample: 5.247 sec elapsed
```

Confusion Matrix for bootstrap sampling on train set

```
In [13]: #rf_bootstrap_model$finalModel #rf_bootstrap_model$results
         print(rf_bootstrap_model)
         confusionMatrix.train(rf_bootstrap_model)
         plot(varImp(rf_bootstrap_model), main = "Variable importance from Bootstrap Random Fo
```

```
Random Forest

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Resampling results across tuning parameters:

  mtry  ROC        Sens       Spec
  2     0.9036885  0.9967213  0
  3     0.9180328  0.9934426  0
  5     0.9206967  0.9901639  0

ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 5.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  96.5  2.6
```

```
      Yes  1.0  0.0

 Accuracy (average) : 0.9649
```



Confusion Matrix for bootstrap sampling on test set

In [14]:
```
caretPredictedClass <- predict(rf_bootstrap_model, model_test_df, type = "raw")
confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
```

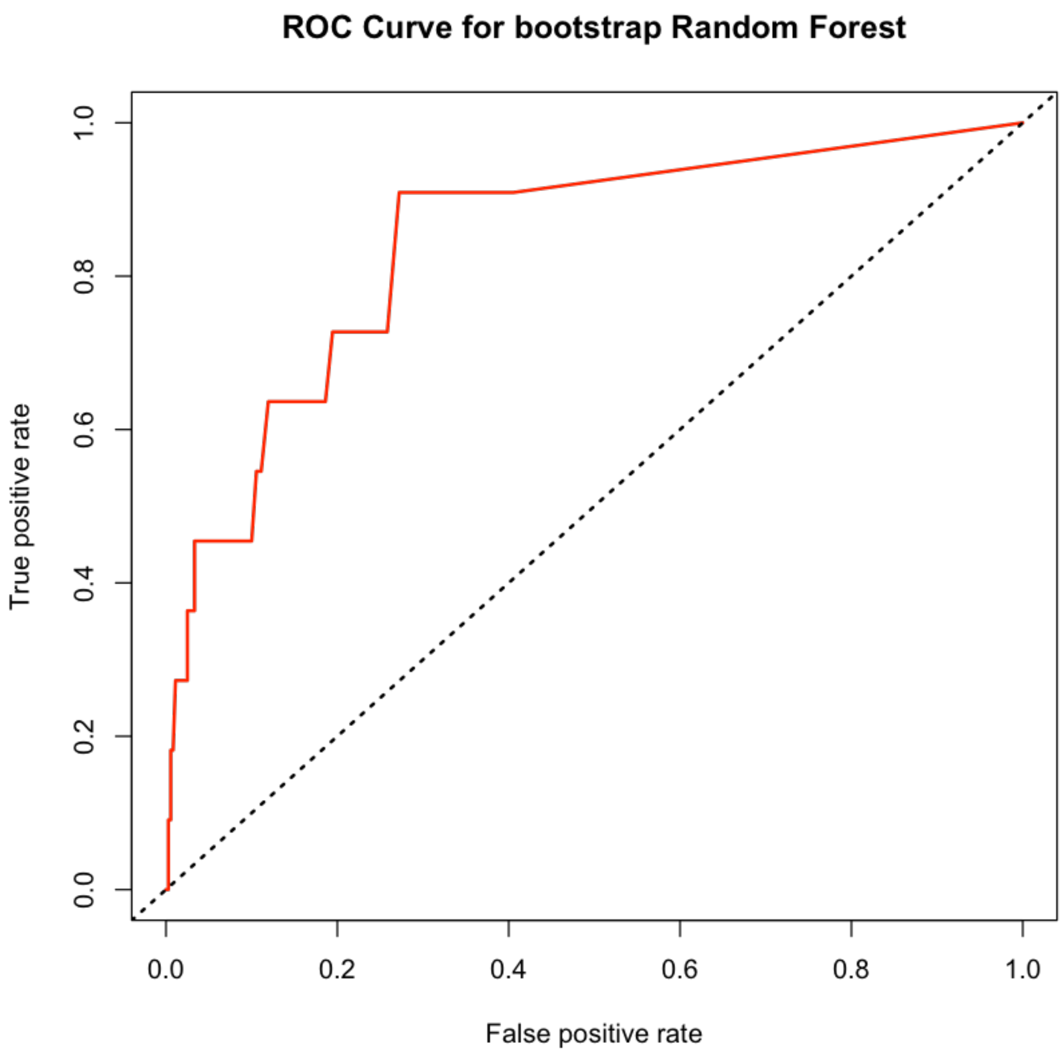
```
Prediction  No Yes
       No  358  10
       Yes   2   1

               Accuracy : 0.9677
                 95% CI : (0.9442, 0.9832)
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.69036

                  Kappa : 0.1318
 Mcnemar's Test P-Value : 0.04331

            Sensitivity : 0.99444
            Specificity : 0.09091
         Pos Pred Value : 0.97283
         Neg Pred Value : 0.33333
             Prevalence : 0.97035
         Detection Rate : 0.96496
   Detection Prevalence : 0.99191
      Balanced Accuracy : 0.54268

       'Positive' Class : No
```

ROC plot for bootstrap random forest on test set

In [15]:
```
rf_bootstrap_pred <- predict(rf_bootstrap_model, model_test_df, type = "prob")[,2]
rf_bootstrap_prediction <- prediction(rf_bootstrap_pred,model_test_df$Manipulater)
rf_bootstrap_perf <- performance(rf_bootstrap_prediction, "tpr","fpr")

plot(rf_bootstrap_perf,main="ROC Curve for bootstrap Random Forest",col=2,lwd=2)
abline(a=0,b=1,lwd=2,lty=3,col="black")

#AUC for the ROC plot
performance(rf_bootstrap_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
```

```
list()

Slot "y.values":
[[1]]
[1] 0.8438131


Slot "alpha.values":
list()
```



The best model was

```
In [16]: rf_bootstrap_model$bestTune
```

	mtry
3	5

Visulaizing the rules coming out of random forest. We can loop and print all the trees built using up sampling. For simplicity, printing just one of the trees

```
In [17]: getTree(rf_bootstrap_model$finalModel,3)
```

	left daughter	right daughter	split var	split point	status	prediction
1	2	3	5	0.337891381	1	0
2	4	5	4	-0.003885957	1	0
3	6	7	5	9.654494271	1	0
4	0	0	0	0.000000000	-1	1
5	0	0	0	0.000000000	-1	2
6	8	9	3	26.834578311	1	0
7	0	0	0	0.000000000	-1	2
8	10	11	4	0.471948375	1	0
9	0	0	0	0.000000000	-1	2
10	12	13	3	0.129103965	1	0
11	0	0	0	0.000000000	-1	2
12	14	15	4	0.011196943	1	0
13	16	17	1	32.886056170	1	0
14	0	0	0	0.000000000	-1	1
15	0	0	0	0.000000000	-1	2
16	18	19	4	-0.013528032	1	0
17	0	0	0	0.000000000	-1	2
18	20	21	5	0.534575428	1	0
19	22	23	4	-0.013412714	1	0
20	0	0	0	0.000000000	-1	2
21	24	25	1	7.003424847	1	0
22	0	0	0	0.000000000	-1	2
23	26	27	3	1.154956261	1	0
24	0	0	0	0.000000000	-1	1
25	28	29	3	1.555515027	1	0
26	30	31	5	2.470942669	1	0
27	32	33	3	1.159900667	1	0
28	0	0	0	0.000000000	-1	1
29	0	0	0	0.000000000	-1	2
30	34	35	4	-0.004324249	1	0
31	0	0	0	0.000000000	-1	2
32	0	0	0	0.000000000	-1	2
33	36	37	1	0.964475340	1	0
34	38	39	4	-0.005291429	1	0
35	0	0	0	0.000000000	-1	1
36	40	41	1	0.905780112	1	0
37	0	0	0	0.000000000	-1	1
38	0	0	0	0.000000000	-1	1
39	0	0	0	0.000000000	-1	2
40	42	43	4	0.138495443	1	0
41	0	0	0	0.000000000	-1	2
42	0	0	0	0.000000000	-1	1
43	44	45	3	1.548379099	1	0
44	0	0	0	0.000000000	-1	1
45	0	0	0	0.000000000	-1	2

1.4.2 Random Forest with up sampling

To incorporate up-sampling (sample the minority class to make their frequencies closer to the majority class.), random forest can use an upsampling strategy

The below code chunk sets some of the control parameters

```
In [18]: objControl <- trainControl(method='boot', number = 1,
                                     returnResamp='final',
                                     summaryFunction = twoClassSummary,
                                     savePredictions = TRUE,
                                     classProbs = TRUE,
                                     sampling="up",allowParallel=TRUE)
```

Parallel processing using doMC needs the below setup: > * num_cores <- (detectCores()-1) * registerDoMC(num_cores)

doMC may give added benefit but is OS dependent. May not work on Windows.

The below code chunk uses doParallel library for parallel processing. After setting the control paramters, the model is run

```
In [19]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("RF Bagging with Up Sample")

         set.seed(4121)
         rf_up_model <- train(model_train_df[,1:5], model_train_df[,6],
                              method='rf',
                              trControl=objControl,
                              metric = "ROC",
                              prox=TRUE)
         stopCluster(num_cores)
         toc()
```

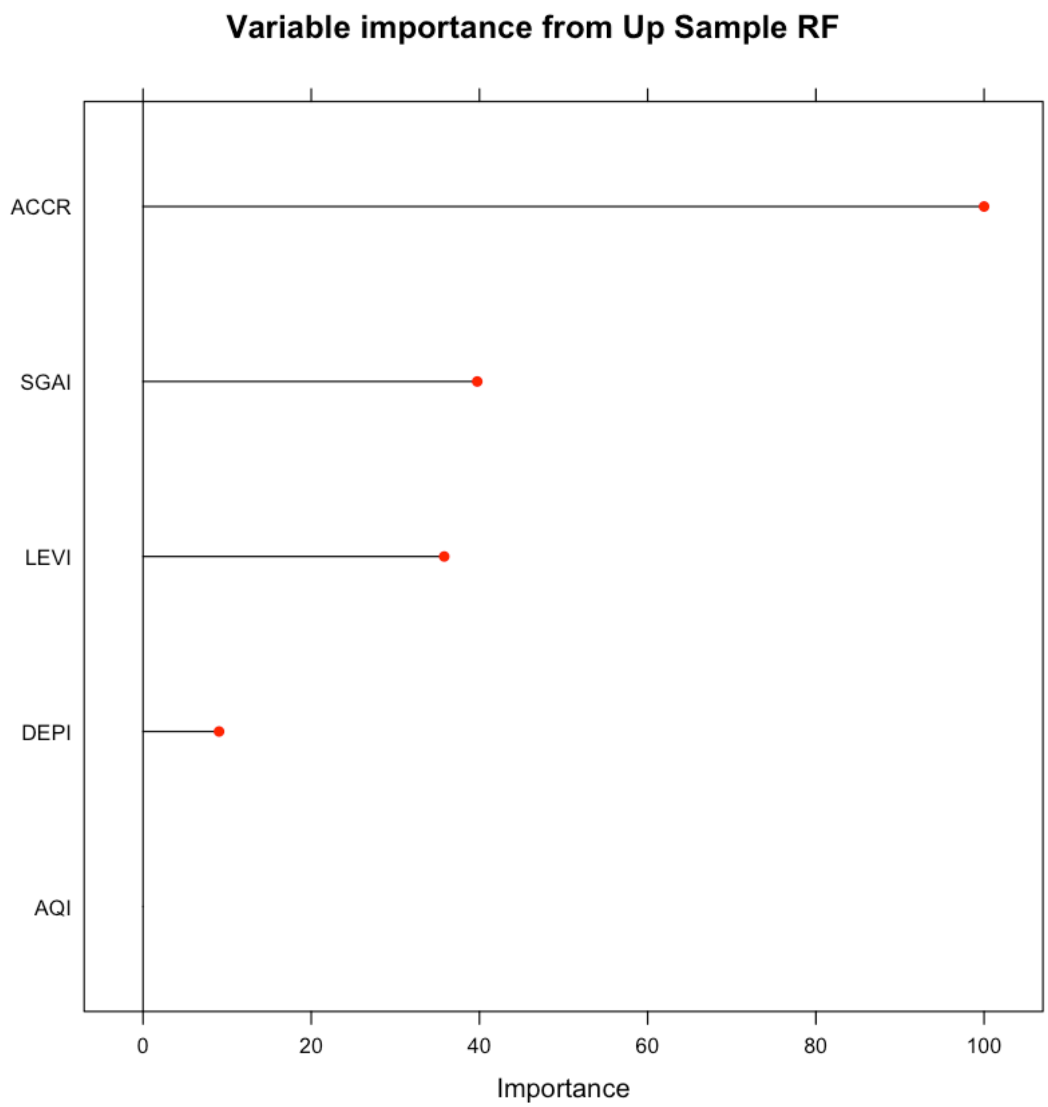
```
RF Bagging with Up Sample: 9.747 sec elapsed
```

Confusion Matrix for upsampling on train set

```
In [20]: #rf_up_model$finalModel #rf_up_model$results

         print(rf_up_model)
         confusionMatrix.train(rf_up_model)
         plot(varImp(rf_up_model), main = "Variable importance from Up Sample RF", col = 2, lwd
```

```
Random Forest

868 samples
  5 predictor
  2 classes: 'No', 'Yes'
```

```
No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using up-sampling

Resampling results across tuning parameters:

  mtry  ROC        Sens       Spec
  2     0.8698770  0.9967213  0
  3     0.8858607  0.9934426  0
  5     0.7725410  0.9803279  0

ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 3.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  96.8  2.6
       Yes  0.6  0.0

 Accuracy (average) : 0.9681
```



Confusion Matrix for upsampling on test set

```
In [21]: caretPredictedClass <- predict(rf_up_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  357  10
       Yes   3   1

               Accuracy : 0.965
                 95% CI : (0.9408, 0.9812)
```

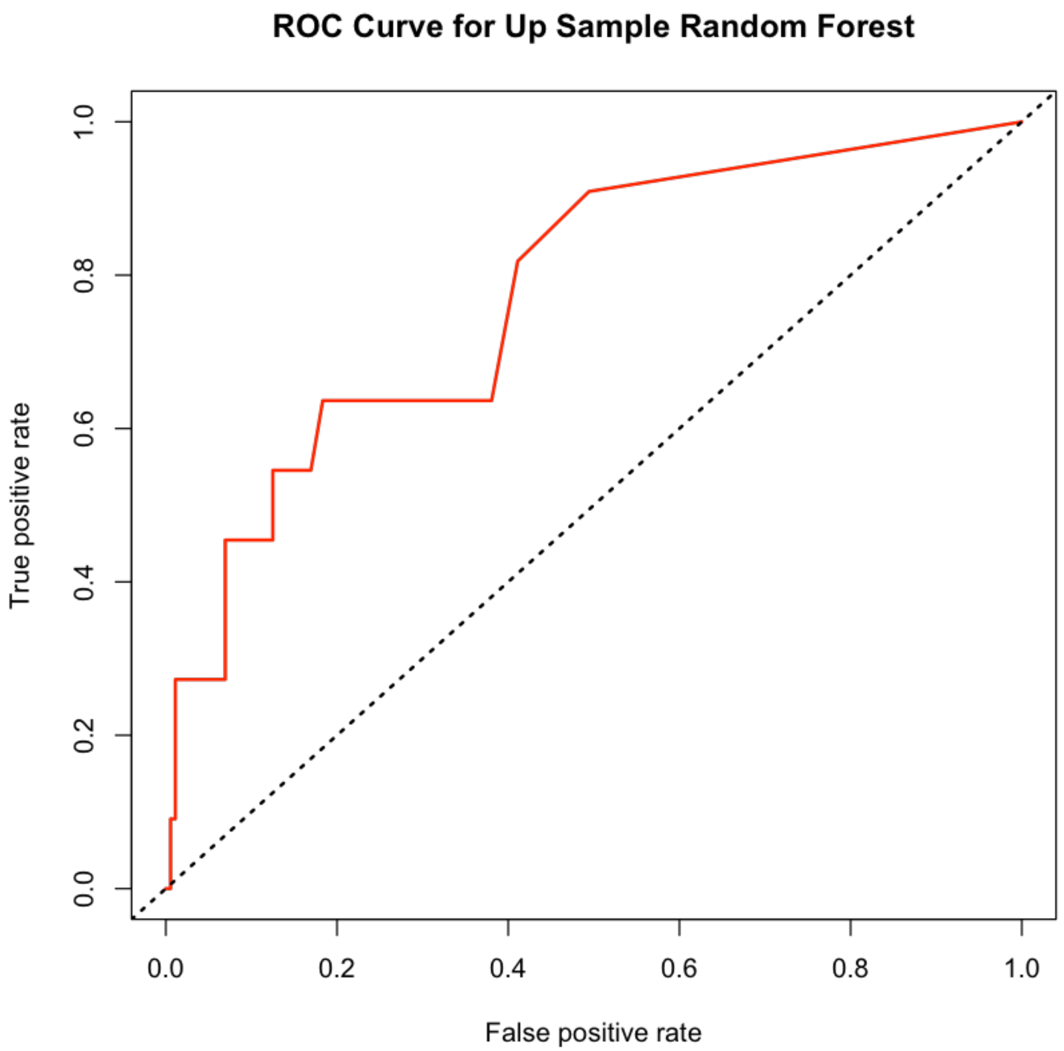
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.78410

                  Kappa : 0.1194
 Mcnemar's Test P-Value : 0.09609

            Sensitivity : 0.99167
            Specificity : 0.09091
         Pos Pred Value : 0.97275
         Neg Pred Value : 0.25000
             Prevalence : 0.97035
         Detection Rate : 0.96226
   Detection Prevalence : 0.98922
      Balanced Accuracy : 0.54129

       'Positive' Class : No
```

ROC plot for upsample random forest on test set

```
In [22]: rf_up_pred <- predict(rf_up_model, model_test_df, type = "prob")[,2]
         rf_up_prediction <- prediction(rf_up_pred,model_test_df$Manipulater)
         rf_up_perf <- performance(rf_up_prediction, "tpr","fpr")

         plot(rf_up_perf,main="ROC Curve for Up Sample Random Forest",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(rf_up_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.7763889
```

```
Slot "alpha.values":
list()
```



Extracting all the rules from the trees built using random forest

```
In [23]: rf_up_treelist <- RF2List(rf_up_model$finalModel)
         rf_up_rules <- extractRules(rf_up_treelist,model_train_df[,c(1:5)], ntree = 10)
         rf_up_rules_metric <- getRuleMetric(rf_up_rules,model_train_df[,c(1:5)],model_train_d
         rf_up_rules_metric <- pruneRule(rf_up_rules_metric,model_train_df[,c(1:5)],model_trai
         rf_up_rules_metric <- selectRuleRRF(rf_up_rules_metric,model_train_df[,c(1:5)],model_
```

```r
#readable rules
print(presentRules(rf_up_rules_metric, colnames(model_train_df[,c(1:5)])))
#rf.up.learner <- buildLearner(rf.up.rules.metric,model_df[,c(1:6)],model_df[,7])
```

```
229 rules (length<=6) were extracted from the first 10 trees.
      len freq    err
 [1,] "2" "0.007" "0.167"
 [2,] "5" "0.007" "0.167"
 [3,] "2" "0.002" "0"
 [4,] "2" "0.002" "0"
 [5,] "1" "0.794" "0.033"
 [6,] "3" "0.002" "0"
 [7,] "2" "0.001" "0"
 [8,] "2" "0.001" "0"
 [9,] "4" "0.003" "0.333"
[10,] "1" "0.679" "0.017"
[11,] "1" "0.594" "0.01"
[12,] "1" "0.143" "0.048"
[13,] "1" "0.893" "0.025"
      condition
 [1,] "ACCR>-0.0137397365 & LEVI<=0.405980794"
 [2,] "DEPI<=1.32222965 & DEPI>0.9754958525 & SGAI<=0.672654396 & ACCR>-0.0137397365 & LEVI>1.0
 [3,] "ACCR<=-0.545217622 & ACCR>-0.637260803"
 [4,] "AQI>6.559077466 & ACCR<=-0.1654674745"
 [5,] "DEPI<=1.09348791"
 [6,] "SGAI<=0.661013693 & SGAI>0.6425966655 & LEVI>1.070554392"
 [7,] "ACCR>-0.036607684 & ACCR<=-0.0363856715"
 [8,] "ACCR>0.3937475925 & LEVI<=0.537491711"
 [9,] "AQI>1.183785029 & DEPI<=1.085872075 & SGAI>2.399295019 & ACCR>-0.013528032"
[10,] "ACCR<=0.001618745"
[11,] "ACCR<=-0.013528032"
[12,] "LEVI>1.1831149185"
[13,] "SGAI<=1.3562452445"
      pred  impRRF
 [1,] "Yes" "1"
 [2,] "Yes" "0.961093022902289"
 [3,] "Yes" "0.474948322651951"
 [4,] "Yes" "0.468119662217227"
 [5,] "No"  "0.307865365223766"
 [6,] "Yes" "0.289905653533515"
 [7,] "Yes" "0.227946850114761"
 [8,] "Yes" "0.219275994918584"
 [9,] "Yes" "0.116948887047343"
[10,] "No"  "0.0973335668127188"
[11,] "No"  "0.0340600883621492"
[12,] "No"  "0.0182289479010012"
[13,] "No"  "0.018123514100372"
```

1.4.3 Random Forest with down sampling - First Approach

To incorporate down-sampling (sample the majority class to make their frequencies closer to the minority class.), random forest can use an downsampling strategy

The below code chunk sets some of the control parameters

```
In [24]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE,
                                    sampling="down")
```

After setting the control parameters, the model is run

```
In [25]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("RF Bagging with Down Sample")

         set.seed(4121)
         rf_down1_model <- train(model_train_df[,1:5], model_train_df[,6],
                             method='rf',
                             trControl=objControl,
                             metric = "ROC")
         stopCluster(num_cores)
         toc()
```

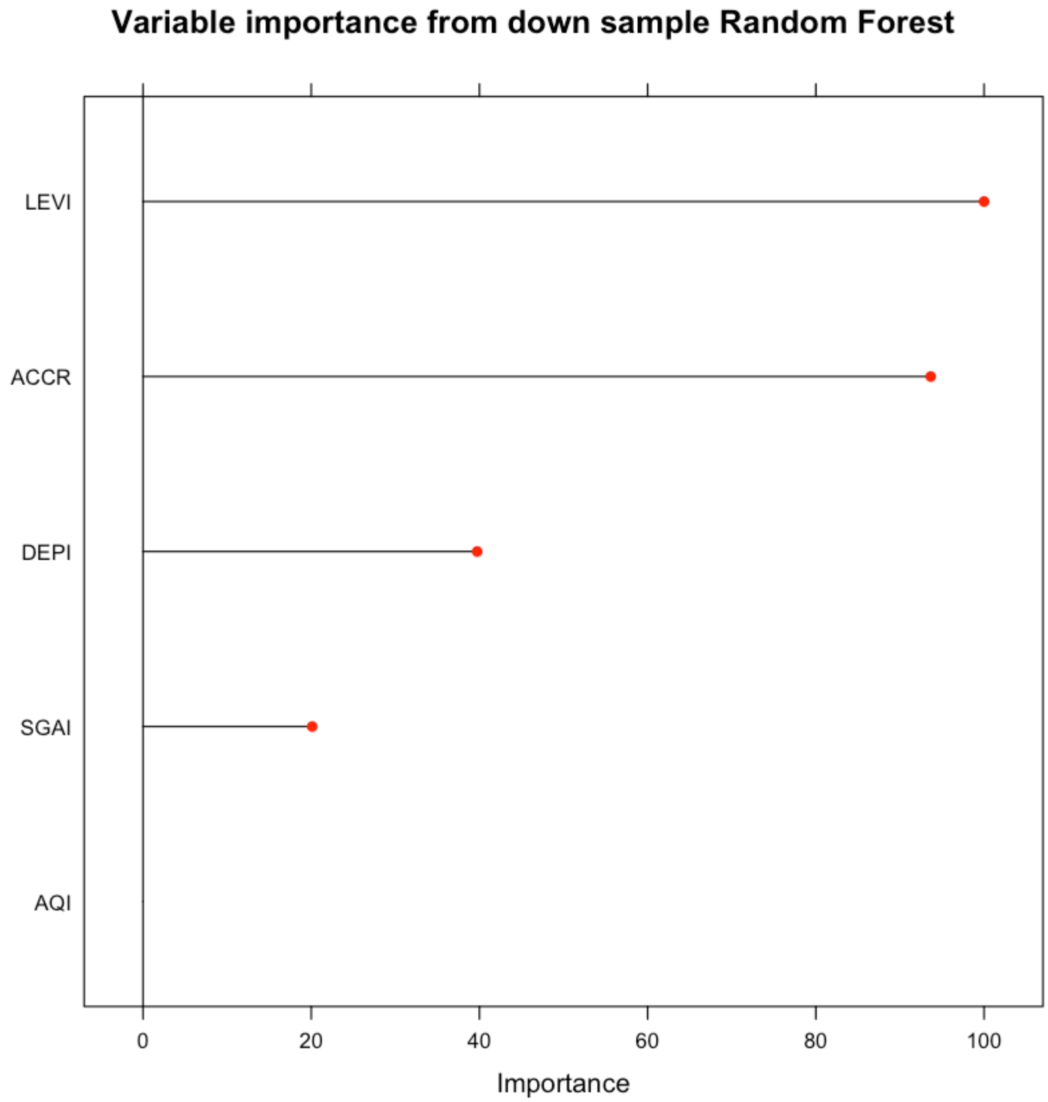
```
RF Bagging with Down Sample: 3.122 sec elapsed
```

Confusion Matrix for down sampling RF on train set

```
In [26]: #rf_down1_model$finalModel #rf_down1_model$results
         print(rf_down1_model)
         confusionMatrix.train(rf_down1_model)
         plot(varImp(rf_down1_model), main = "Variable importance from down sample Random Fore
```

```
Random Forest

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using down-sampling

Resampling results across tuning parameters:
```

```
  mtry  ROC        Sens       Spec
  2     0.7186475  0.8327869  0.375
  3     0.6973361  0.7967213  0.125
  5     0.6540984  0.7639344  0.250

ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  81.2  1.6
       Yes 16.3  1.0

 Accuracy (average) : 0.8211
```



Confusion Matrix for down sampling RF on test set

```
In [27]: caretPredictedClass <- predict(rf_down1_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  251   3
       Yes 109   8

               Accuracy : 0.6981
                 95% CI : (0.6486, 0.7444)
```

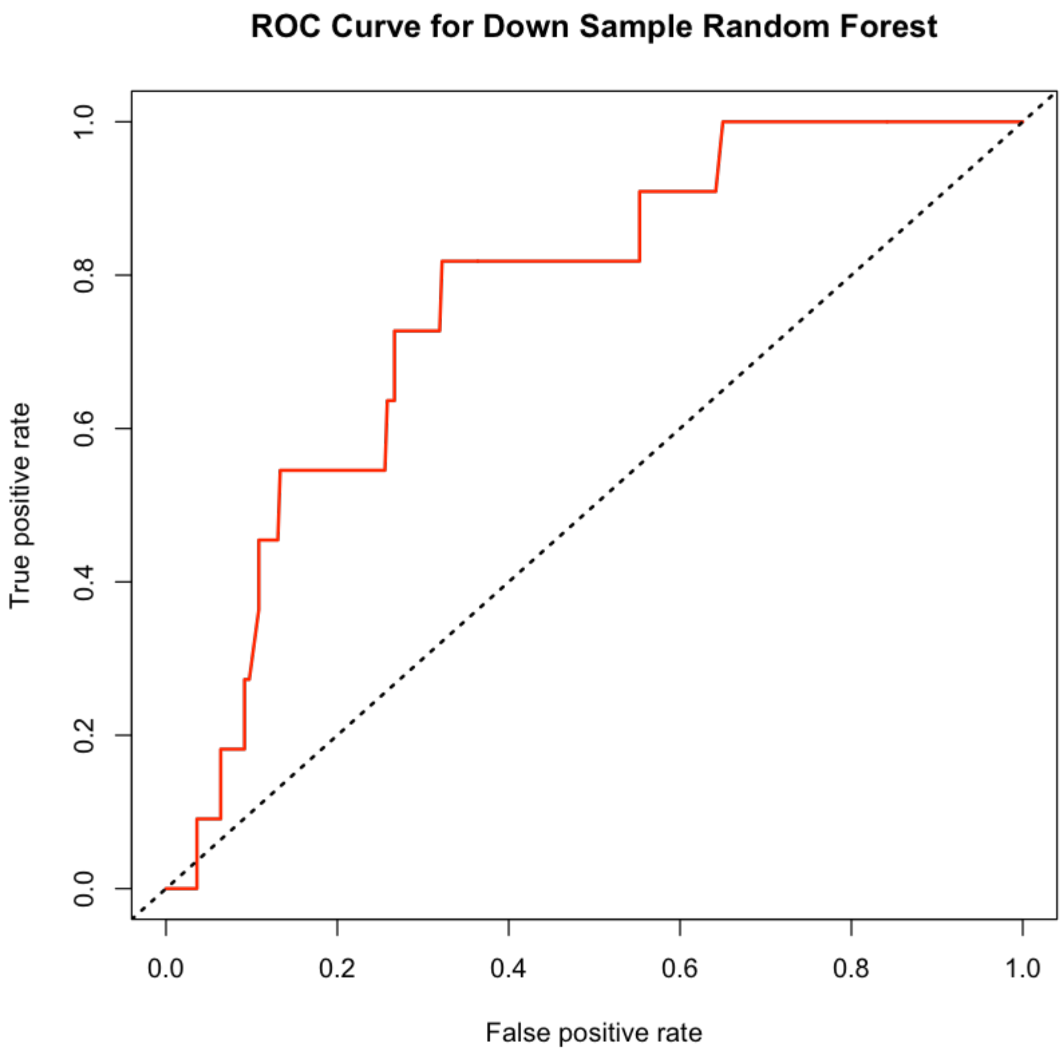
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0749
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.69722
            Specificity : 0.72727
         Pos Pred Value : 0.98819
         Neg Pred Value : 0.06838
             Prevalence : 0.97035
         Detection Rate : 0.67655
   Detection Prevalence : 0.68464
      Balanced Accuracy : 0.71225

       'Positive' Class : No
```

ROC plot for down sample random forest on test set

```
In [28]: rf_down1_pred <- predict(rf_down1_model, model_test_df, type = "prob")[,2]
         rf_down1_prediction <- prediction(rf_down1_pred,model_test_df$Manipulater)
         rf_down1_perf <- performance(rf_down1_prediction, "tpr","fpr")

         plot(rf_down1_perf,main="ROC Curve for Down Sample Random Forest",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(rf_down1_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.7656566
```

```
Slot "alpha.values":
list()
```



1.4.4 Random Forest with down sampling - Second Approach

To incorporate down-sampling (sample the majority class to make their frequencies closer to the rarest class.), random forest can take a random sample of size c*nmin, where c is the number of classes and nmin is the number of samples in the minority class.

THIS IMPLEMENTATION IS WITHOUT CARET PACKAGE

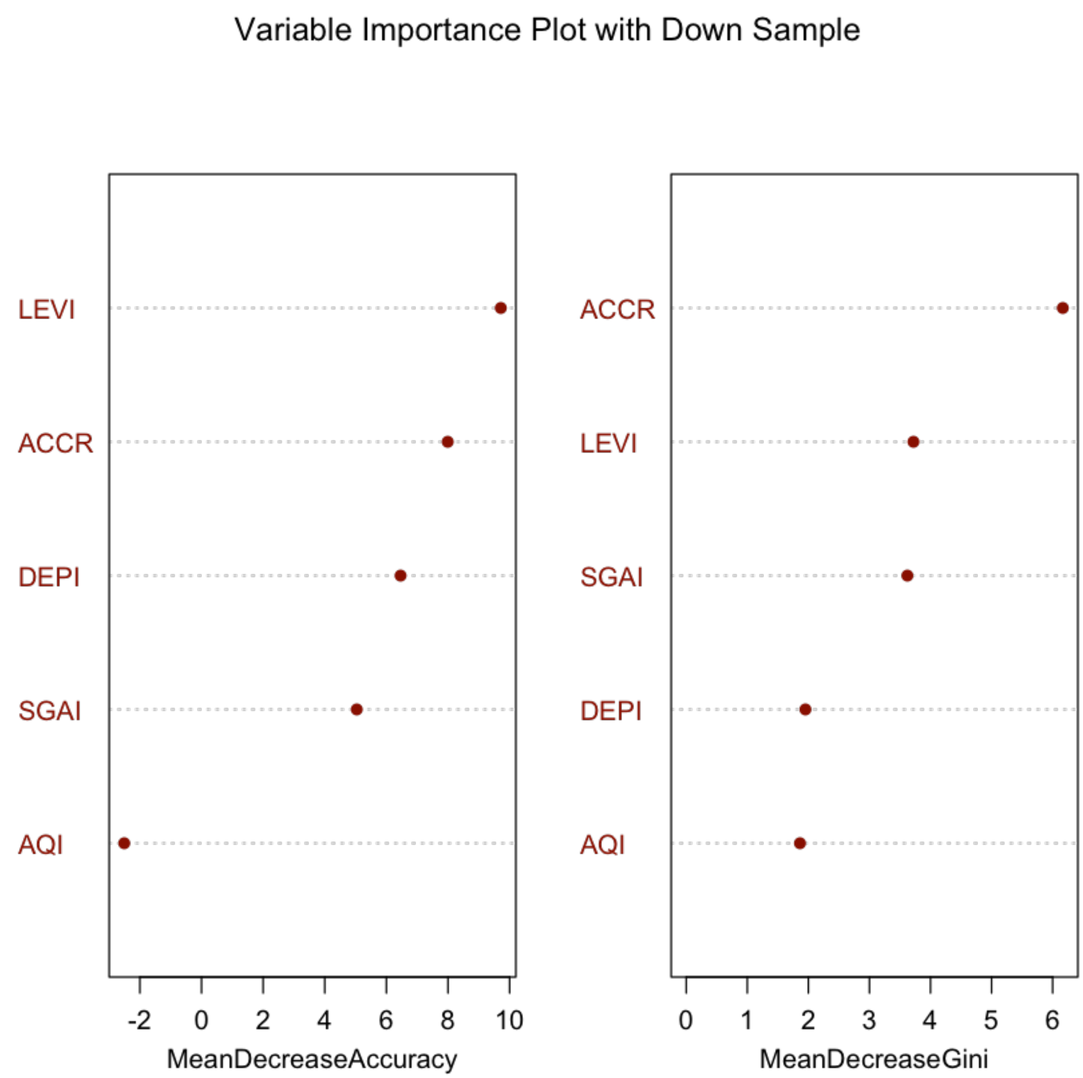
```
In [29]: nmin <- sum(model_train_df$Manipulater == "Yes") #total minority cases
         set.seed(4121)
         tic("RF Bagging with Down")
         rf_down2_model <- randomForest(Manipulater ~ .,
                                  data=model_train_df, importance=TRUE, mtry = 2,
                                  #if strata is not defined RF does bootstrap sample
                                  strata = model_train_df$Manipulater,
                                  #selecting nmin cases from positive and negative class
                                  sampsize = rep(nmin,2),
                                  #cutoff: winning class for an observation is the one
                                  #with the maximum ratio of proportion of votes to cutoff.
                                  cutoff = c(1/2, 1/2),ntree=1024,  nodesize = 10,
                                  keep.forest = TRUE)#, xtest = model_test_df[,-12])
         toc()
```

```
RF Bagging with Down: 0.162 sec elapsed
```

Variable importance and Confusion matrix on downsample random forest on train set

```
In [30]: #To plot the error rate.
         #plot(rf_down1_model, main = "Error rate vs. number of trees (RF with downsample", ty

         #To know the legends, type rf_down1_model to get the confusion matrix and #see the er

         print(rf_down2_model)

         varImpPlot(rf_down2_model, main = "Variable Importance Plot with Down Sample", pch = 
```

```
Call:
 randomForest(formula = Manipulater ~ ., data = model_train_df,      importance = TRUE, mtry =
               Type of random forest: classification
                     Number of trees: 1024
No. of variables tried at each split: 2

        OOB estimate of  error rate: 21.54%
Confusion matrix:
     No Yes class.error
No  663 177   0.2107143
Yes  10  18   0.3571429
```



Variable importance and Confusion matrix on downsample random forest on test set

In [31]:
```
testPredictedClass <- predict(rf_down2_model, model_test_df, type = "response")
confusionMatrix(testPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  275   1
       Yes  85  10

               Accuracy : 0.7682
                 95% CI : (0.7219, 0.8102)
```

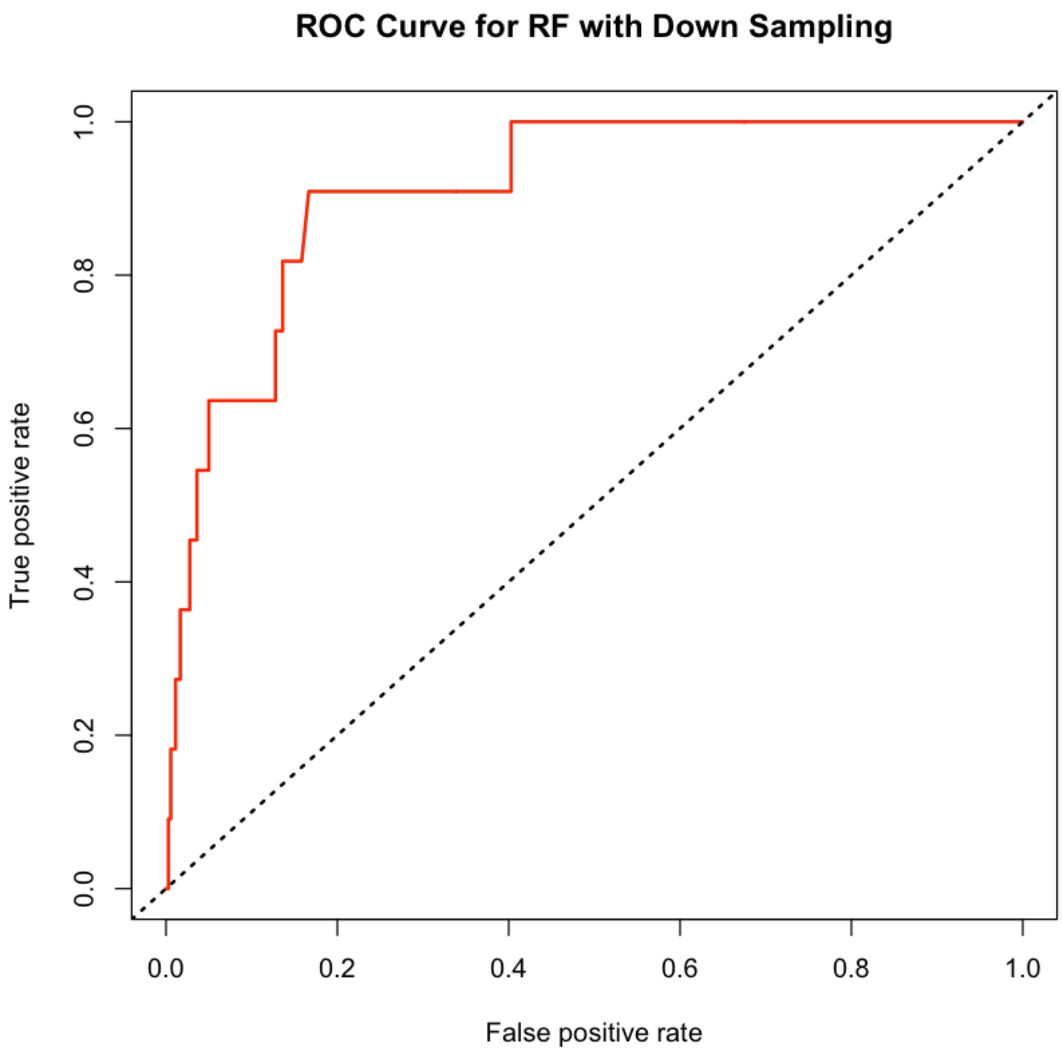
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1431
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.7639
            Specificity : 0.9091
         Pos Pred Value : 0.9964
         Neg Pred Value : 0.1053
             Prevalence : 0.9704
         Detection Rate : 0.7412
   Detection Prevalence : 0.7439
      Balanced Accuracy : 0.8365

       'Positive' Class : No
```

ROC plot for Random Forest with downsampling on test set

In [32]:
```
rf_down2_pred <- predict(rf_down2_model, model_test_df, type = "prob")[,2]
rf_down2_prediction <- prediction(rf_down2_pred,model_test_df$Manipulater)
rf_down2_perf <- performance(rf_down2_prediction, "tpr","fpr")

plot(rf_down2_perf,main="ROC Curve for RF with Down Sampling",col=2,lwd=2)
abline(a=0,b=1,lwd=2,lty=3,col="black")

#AUC for the ROC plot
performance(rf_down2_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.9109848
```

```
Slot "alpha.values":
list()
```



1.4.5 Random Forest with SMOTE

Synthetic minority oversampling technique (SMOTE) blends under-sampling of the majority class with a special form of over-sampling the minority class. SMOTE oversamples the rare event by using bootstrapping and k-nearest neighbor to synthetically create additional observations of that event.

The below code chunk sets some of the control parameters

```
In [33]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE,
                                    sampling="smote")
```

After setting the control parameters, the model is run

```
In [34]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("RF Bagging with SMOTE Sample")

         set.seed(4121)
         rf_smote_model <- train(model_train_df[,1:5], model_train_df[,6],
                                 method='rf',
                                 trControl=objControl,
                                 metric = "ROC",
                                 prox=TRUE,allowParallel=TRUE)
         stopCluster(num_cores)
         toc()
```

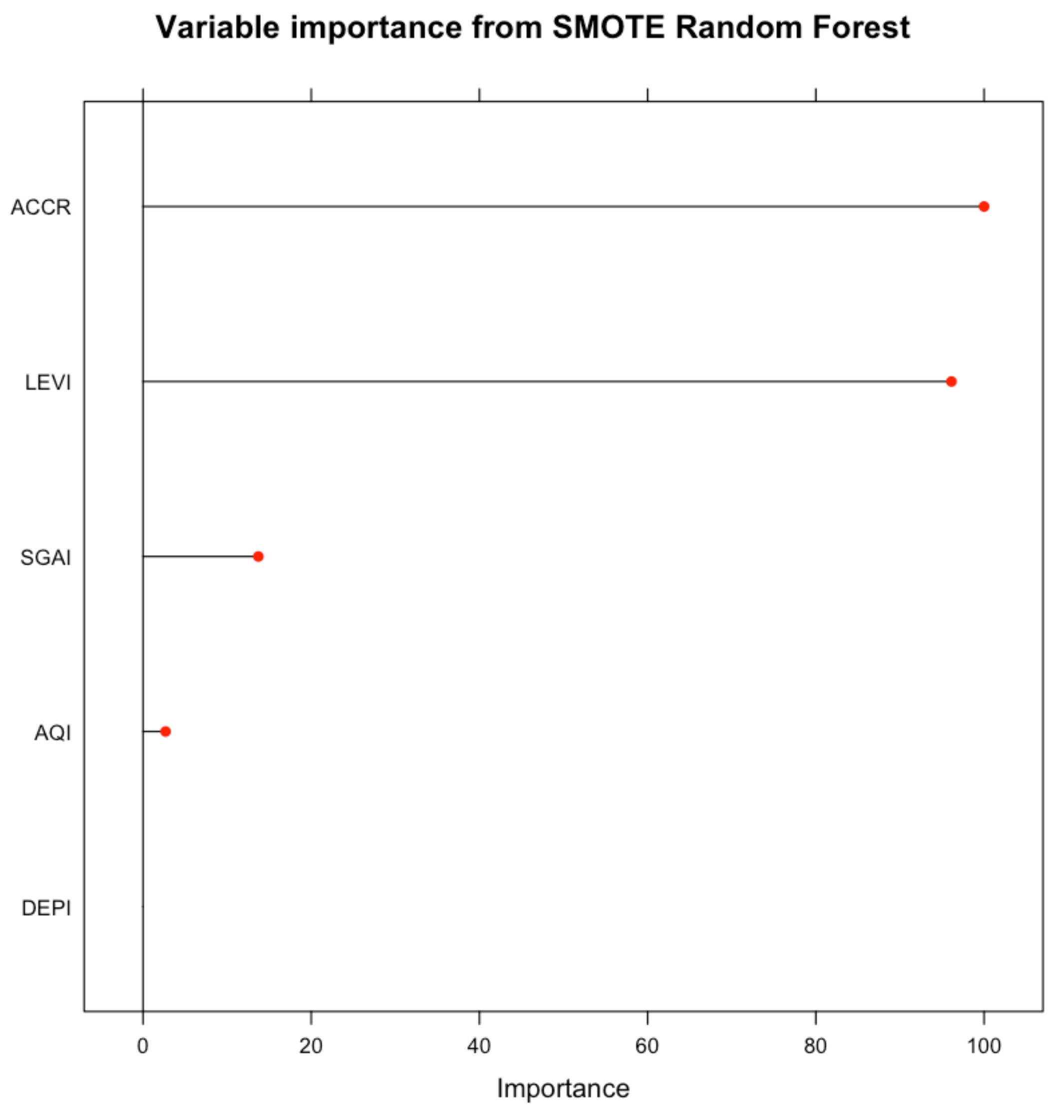
```
RF Bagging with SMOTE Sample: 3.496 sec elapsed
```

Confusion Matrix for RF on train set

```
In [35]: #rf_smote_model$finalModel #rf_smote_model$results
         print(rf_smote_model)
         confusionMatrix.train(rf_smote_model)
         plot(varImp(rf_smote_model), main = "Variable importance from SMOTE Random Forest", co
```

```
Random Forest

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using SMOTE

Resampling results across tuning parameters:

  mtry  ROC        Sens       Spec
  2     0.7354508  0.8819672  0.125
  3     0.6907787  0.8852459  0.125
  5     0.7579918  0.8852459  0.250
```

```
ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 5.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  86.3  1.9
       Yes 11.2  0.6

 Accuracy (average) : 0.869
```



Confusion Matrix for RF on test set

```
In [36]: caretPredictedClass <- predict(rf_smote_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  292   5
       Yes  68   6

               Accuracy : 0.8032
                 95% CI : (0.7591, 0.8425)
```

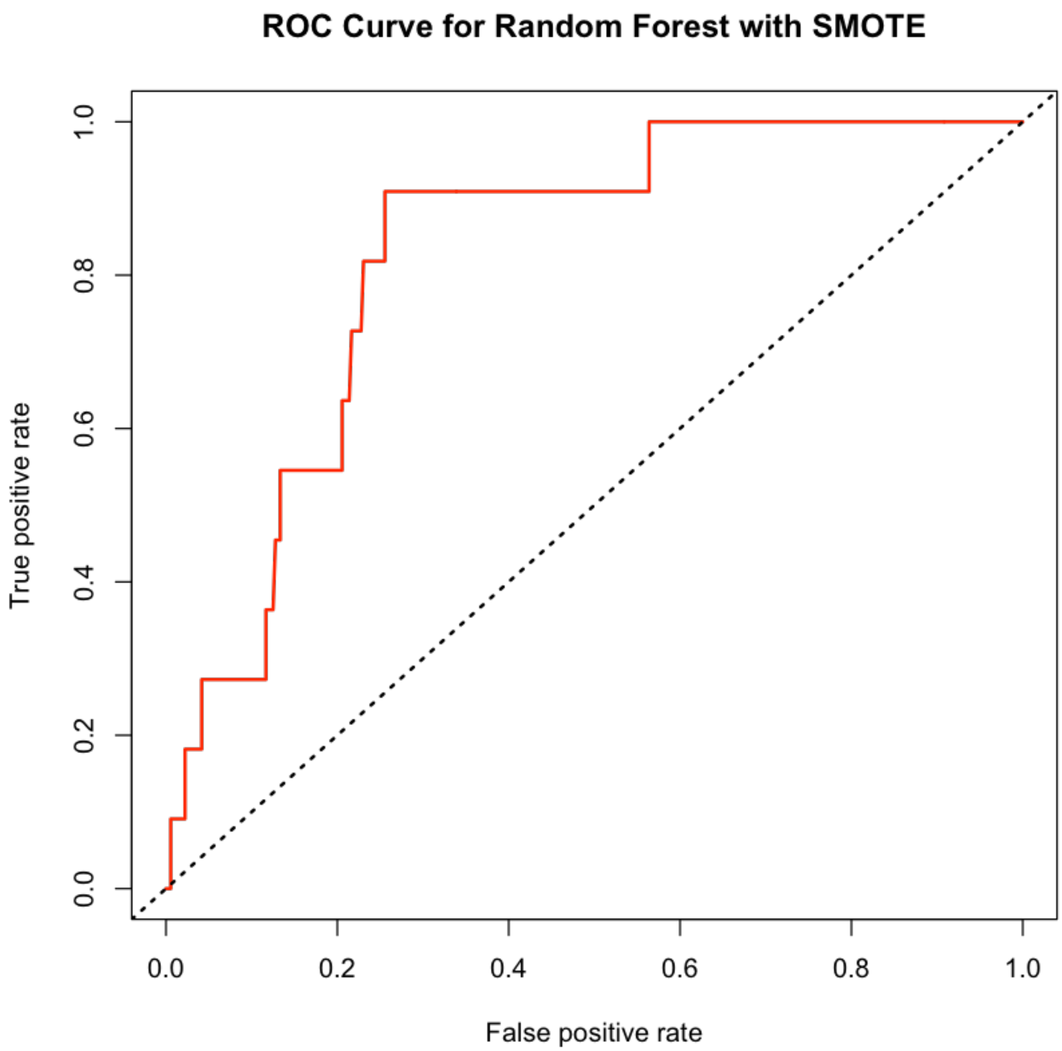
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0944
 Mcnemar's Test P-Value : 3.971e-13

            Sensitivity : 0.81111
            Specificity : 0.54545
         Pos Pred Value : 0.98316
         Neg Pred Value : 0.08108
             Prevalence : 0.97035
         Detection Rate : 0.78706
   Detection Prevalence : 0.80054
      Balanced Accuracy : 0.67828

       'Positive' Class : No
```

ROC plot for random forest on test set

```
In [37]: rf_smote_pred <- predict(rf_smote_model, model_test_df, type = "prob")[,2]
         rf_smote_prediction <- prediction(rf_smote_pred,model_test_df$Manipulater)
         rf_smote_perf <- performance(rf_smote_prediction, "tpr","fpr")

         plot(rf_smote_perf,main="ROC Curve for Random Forest with SMOTE",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(rf_smote_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8258838
```

```
Slot "alpha.values":
list()
```



1.5 Boosting

Boosting is an ensemble technique which tries to create a strong classifier from several weak classifier. The model buidling through boosting is sequential. 1. The first model is build based on the random sample on N observations picked from original dataset (with replacement). Equal weight is assigned to each observation. These weights decide the probability of observations which will be picked up in the training set. 2. In the second step, all the original dataset is passed through the model. For regressor model, the observations whose predicted value differs the most from the actual value is defined to be most in error. 3. The sampling probabilities of the observations which are most in error, is adjusted such that their chance of getting picked up for the second model is higher. 4. As the model buidling progresses, in each of the sequence of models, the pattern which are more difficult are picked up. Different models are better in different part of the observation space. 5. Rgeressors are combined using weighted median. Models which are more confident about their predictions are weighted more heavily.

1.5.1 Boosting algorithms in R

Adaboost is one of the ways to boost the performance of decision trees on binary classification problems. The decision trees with just one level will mostly be a weak learner. These weak learners will achieve an accuracy just above random chance on a classification problem.

Adaboost is also referred to as discrete AdaBoost as it is used for classification rather than regression. The algorithms in boosting are:

1. Adaboost classification trees: ***adaboost()*** Required Package is **fastAdaboost**
2. Adaboost.M1: ***AdaBoost.M1()*** Required Package is **adabag, plyr**
3. Boosted Classification Trees: ***ada()*** Required Package is **adabag, plyr**
4. Boosted Generalized Additive Model: ***gamBoost()*** Required Package is **mboost, plyr**
5. Boosted Generalized Linear Model: ***glmboost()*** Required Package is **mboost, plyr**
6. Boosted Linear Model: ***Bstlm()*** Required Package is **bst, plyr**
7. Boosted Logistic Regression: ***LogitBoost()*** Required Package is **caTools**
8. Boosted Smoothing Spline: ***bstSm()*** Required Package is **bst, plyr**
9. Boosted Tree: ***blackboost()*** Required Package is **party, mboost, plyr**
10. Boosted Tree: ***bstTree()*** Required Package is **bst, plyr**
11. C5.0: ***C5.0()*** Required Package is **C50, plyr**
12. Cost Sensitive C5.0: ***C5.0Cost()*** Required Package is **C50, plyr**
13. Cubist: ***glmboost()*** Required Package is **cubist**
14. DeepBoost: ***deepboost()*** Required Package is **deepboost**
15. eXtreme Gradient Boosting: ***xgbLinear()*** Required Package is **xgboost**
16. eXtreme Gradient Boosting: ***xgbTree()*** Required Package is **xgboost, plyr**
17. Stochastic Gradient Boosting: ***gbm()*** Required Package is **gbm, plyr**

1.5.2 Boosting with adaboost (normal)

The below code chunk sets some of the control parameters for adaboost

```
In [38]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='all',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE)#, p = 0.70) #in case method = #"LGOCV"
```

```
In [39]: search_grid <- expand.grid(mfinal = c(20:100), maxdepth = c(2:4),
                             coeflearn = c("Breiman", "Freund", "Zhu"))
```

Look for the documentation of library **adabag**. The **boosting()** function of adabag implments 'AdaBoost.M1'. The *boos* paramter of boosting function is set to TRUE by default. This meand a bootstrap sample of the training set is drawn using the weights for each observation on that iteration. If FALSE, every observation is used with its weights.

After setting the control paramters, the model is run

```
In [40]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Adaptive Boosting with Bootstrap Sample")

         set.seed(4121)
         ada_model <- train(model_train_df[,1:5], model_train_df[,6],
                            method='AdaBoost.M1',
                            trControl=objControl,
                            tuneGrid = search_grid,
                            metric = "ROC")
         stopCluster(num_cores)
         toc()
```

```
Adaptive Boosting with Bootstrap Sample: 121.42 sec elapsed
```

Confusion Matrix for adaboost on train set

```
In [41]: #ada_model$finalModel #ada_model$results
         print(ada_model)
         confusionMatrix.train(ada_model)
         plot(varImp(ada_model), main = "Variable importance from Adaboost with Bootstrap", co
```

```
AdaBoost.M1

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Resampling results across tuning parameters:

  coeflearn  maxdepth  mfinal  ROC        Sens       Spec
  Breiman    2          20     0.6795082  1.0000000  0.000
  Breiman    2          21     0.6827869  1.0000000  0.000
  Breiman    2          22     0.6774590  1.0000000  0.000
  Breiman    2          23     0.6676230  1.0000000  0.000
  Breiman    2          24     0.7036885  1.0000000  0.000
  Breiman    2          25     0.6889344  1.0000000  0.000
  Breiman    2          26     0.6885246  1.0000000  0.000
  Breiman    2          27     0.6905738  1.0000000  0.000
```

Breiman	2	28	0.6913934	0.9967213	0.000
Breiman	2	29	0.6942623	1.0000000	0.000
Breiman	2	30	0.6729508	0.9967213	0.000
Breiman	2	31	0.6729508	0.9967213	0.000
Breiman	2	32	0.6729508	1.0000000	0.000
Breiman	2	33	0.6643443	1.0000000	0.000
Breiman	2	34	0.6631148	1.0000000	0.000
Breiman	2	35	0.6704918	1.0000000	0.000
Breiman	2	36	0.6844262	1.0000000	0.000
Breiman	2	37	0.6766393	1.0000000	0.000
Breiman	2	38	0.6918033	0.9967213	0.000
Breiman	2	39	0.6918033	1.0000000	0.000
Breiman	2	40	0.6926230	0.9967213	0.000
Breiman	2	41	0.6885246	0.9967213	0.000
Breiman	2	42	0.6881148	0.9934426	0.000
Breiman	2	43	0.6913934	0.9934426	0.000
Breiman	2	44	0.6959016	0.9934426	0.000
Breiman	2	45	0.6905738	0.9934426	0.000
Breiman	2	46	0.6967213	0.9934426	0.000
Breiman	2	47	0.6967213	0.9967213	0.000
Breiman	2	48	0.6918033	0.9967213	0.000
Breiman	2	49	0.6774590	0.9901639	0.000
Breiman	2	50	0.6950820	0.9967213	0.000
Breiman	2	51	0.6926230	0.9967213	0.000
Breiman	2	52	0.6909836	1.0000000	0.000
Breiman	2	53	0.6725410	1.0000000	0.000
Breiman	2	54	0.6725410	1.0000000	0.000
Breiman	2	55	0.6725410	1.0000000	0.000
Breiman	2	56	0.6778689	1.0000000	0.000
Breiman	2	57	0.6713115	1.0000000	0.000
Breiman	2	58	0.6717213	1.0000000	0.000
Breiman	2	59	0.6639344	1.0000000	0.000
Breiman	2	60	0.6696721	1.0000000	0.000
Breiman	2	61	0.6696721	1.0000000	0.000
Breiman	2	62	0.6733607	1.0000000	0.000
Breiman	2	63	0.6524590	0.9967213	0.000
Breiman	2	64	0.6553279	0.9967213	0.000
Breiman	2	65	0.6540984	0.9967213	0.000
Breiman	2	66	0.6536885	0.9967213	0.000
Breiman	2	67	0.6532787	0.9967213	0.000
Breiman	2	68	0.6397541	1.0000000	0.000
Breiman	2	69	0.6512295	1.0000000	0.000
Breiman	2	70	0.6573770	1.0000000	0.000
Breiman	2	71	0.6577869	1.0000000	0.000
Breiman	2	72	0.6553279	1.0000000	0.000
Breiman	2	73	0.6553279	1.0000000	0.000
Breiman	2	74	0.6561475	1.0000000	0.000
Breiman	2	75	0.6561475	1.0000000	0.000

Breiman	2	76	0.6508197	1.0000000	0.000
Breiman	2	77	0.6614754	0.9967213	0.000
Breiman	2	78	0.6643443	1.0000000	0.000
Breiman	2	79	0.6655738	0.9967213	0.000
Breiman	2	80	0.6713115	1.0000000	0.000
Breiman	2	81	0.6655738	1.0000000	0.000
Breiman	2	82	0.6655738	1.0000000	0.000
Breiman	2	83	0.6659836	1.0000000	0.000
Breiman	2	84	0.6659836	1.0000000	0.000
Breiman	2	85	0.6655738	1.0000000	0.000
Breiman	2	86	0.6651639	1.0000000	0.000
Breiman	2	87	0.6635246	0.9967213	0.000
Breiman	2	88	0.6745902	1.0000000	0.000
Breiman	2	89	0.6565574	1.0000000	0.000
Breiman	2	90	0.6508197	0.9967213	0.000
Breiman	2	91	0.6508197	1.0000000	0.000
Breiman	2	92	0.6586066	0.9967213	0.000
Breiman	2	93	0.6586066	1.0000000	0.000
Breiman	2	94	0.6356557	1.0000000	0.000
Breiman	2	95	0.6471311	1.0000000	0.000
Breiman	2	96	0.6471311	1.0000000	0.000
Breiman	2	97	0.6446721	1.0000000	0.000
Breiman	2	98	0.6422131	1.0000000	0.000
Breiman	2	99	0.6635246	1.0000000	0.000
Breiman	2	100	0.6586066	1.0000000	0.000
Breiman	3	20	0.7979508	0.9901639	0.000
Breiman	3	21	0.7901639	0.9901639	0.000
Breiman	3	22	0.7934426	0.9868852	0.000
Breiman	3	23	0.7891393	0.9901639	0.000
Breiman	3	24	0.7825820	0.9901639	0.000
Breiman	3	25	0.7793033	0.9901639	0.000
Breiman	3	26	0.7739754	0.9868852	0.000
Breiman	3	27	0.7735656	0.9901639	0.000
Breiman	3	28	0.7727459	0.9868852	0.000
Breiman	3	29	0.7850410	0.9868852	0.000
Breiman	3	30	0.7911885	0.9901639	0.000
Breiman	3	31	0.7977459	0.9901639	0.000
Breiman	3	32	0.7821721	0.9901639	0.000
Breiman	3	33	0.7809426	0.9868852	0.000
Breiman	3	34	0.7846311	0.9868852	0.000
Breiman	3	35	0.7993852	0.9901639	0.000
Breiman	3	36	0.8053279	0.9901639	0.000
Breiman	3	37	0.8016393	0.9934426	0.000
Breiman	3	38	0.8010246	0.9967213	0.000
Breiman	3	39	0.7774590	0.9934426	0.000
Breiman	3	40	0.8032787	0.9934426	0.000
Breiman	3	41	0.8061475	0.9934426	0.000
Breiman	3	42	0.8102459	0.9934426	0.000

Breiman	3	43	0.8094262	0.9967213	0.000
Breiman	3	44	0.8086066	0.9967213	0.000
Breiman	3	45	0.7934426	0.9967213	0.000
Breiman	3	46	0.8016393	0.9934426	0.000
Breiman	3	47	0.8000000	0.9967213	0.000
Breiman	3	48	0.7811475	0.9934426	0.000
Breiman	3	49	0.7938525	0.9934426	0.000
Breiman	3	50	0.8008197	0.9934426	0.000
Breiman	3	51	0.7938525	0.9934426	0.000
Breiman	3	52	0.7721311	0.9934426	0.000
Breiman	3	53	0.7729508	0.9934426	0.000
Breiman	3	54	0.7680328	0.9901639	0.000
Breiman	3	55	0.7598361	0.9868852	0.000
Breiman	3	56	0.7647541	0.9901639	0.000
Breiman	3	57	0.7618852	0.9868852	0.000
Breiman	3	58	0.7670082	0.9868852	0.000
Breiman	3	59	0.7596311	0.9934426	0.000
Breiman	3	60	0.7514344	0.9901639	0.000
Breiman	3	61	0.7411885	0.9901639	0.000
Breiman	3	62	0.7485656	0.9868852	0.000
Breiman	3	63	0.7452869	0.9901639	0.000
Breiman	3	64	0.7452869	0.9901639	0.000
Breiman	3	65	0.7428279	0.9901639	0.000
Breiman	3	66	0.7588115	0.9901639	0.000
Breiman	3	67	0.7588115	0.9934426	0.000
Breiman	3	68	0.7588115	0.9934426	0.000
Breiman	3	69	0.7563525	0.9901639	0.000
Breiman	3	70	0.7563525	0.9934426	0.000
Breiman	3	71	0.7547131	0.9934426	0.000
Breiman	3	72	0.7678279	0.9934426	0.000
Breiman	3	73	0.7715164	0.9934426	0.000
Breiman	3	74	0.7731557	0.9934426	0.000
Breiman	3	75	0.7637295	0.9901639	0.000
Breiman	3	76	0.7625000	0.9934426	0.000
Breiman	3	77	0.7620902	0.9934426	0.000
Breiman	3	78	0.7764344	0.9901639	0.000
Breiman	3	79	0.7764344	0.9934426	0.000
Breiman	3	80	0.7756148	0.9934426	0.000
Breiman	3	81	0.7711066	0.9934426	0.000
Breiman	3	82	0.7723361	0.9901639	0.000
Breiman	3	83	0.7731557	0.9934426	0.000
Breiman	3	84	0.7706967	0.9901639	0.000
Breiman	3	85	0.7674180	0.9934426	0.000
Breiman	3	86	0.7637295	0.9934426	0.000
Breiman	3	87	0.7588115	0.9934426	0.000
Breiman	3	88	0.7604508	0.9934426	0.000
Breiman	3	89	0.7625000	0.9934426	0.000
Breiman	3	90	0.7633197	0.9934426	0.000

Breiman	3	91	0.7588115	0.9934426	0.000
Breiman	3	92	0.7584016	0.9901639	0.000
Breiman	3	93	0.7584016	0.9934426	0.000
Breiman	3	94	0.7584016	0.9934426	0.000
Breiman	3	95	0.7764344	0.9967213	0.000
Breiman	3	96	0.7784836	0.9934426	0.000
Breiman	3	97	0.7743852	0.9967213	0.000
Breiman	3	98	0.7809426	0.9967213	0.000
Breiman	3	99	0.7661885	0.9967213	0.000
Breiman	3	100	0.7661885	0.9967213	0.000
Breiman	4	20	0.5399590	0.9934426	0.000
Breiman	4	21	0.5311475	0.9934426	0.000
Breiman	4	22	0.5227459	0.9934426	0.000
Breiman	4	23	0.5329918	0.9934426	0.000
Breiman	4	24	0.5284836	0.9934426	0.000
Breiman	4	25	0.5252049	0.9934426	0.000
Breiman	4	26	0.5495902	0.9934426	0.000
Breiman	4	27	0.5551230	0.9934426	0.000
Breiman	4	28	0.5698770	0.9934426	0.000
Breiman	4	29	0.5915984	0.9934426	0.000
Breiman	4	30	0.6030738	0.9934426	0.000
Breiman	4	31	0.5731557	0.9934426	0.000
Breiman	4	32	0.5639344	0.9934426	0.000
Breiman	4	33	0.5602459	0.9934426	0.000
Breiman	4	34	0.5770492	0.9901639	0.000
Breiman	4	35	0.5741803	0.9934426	0.000
Breiman	4	36	0.5655738	0.9967213	0.000
Breiman	4	37	0.5573770	0.9934426	0.000
Breiman	4	38	0.5581967	0.9934426	0.000
Breiman	4	39	0.5553279	0.9934426	0.000
Breiman	4	40	0.5508197	0.9934426	0.000
Breiman	4	41	0.5336066	0.9934426	0.000
Breiman	4	42	0.5704918	0.9901639	0.000
Breiman	4	43	0.5622951	0.9901639	0.000
Breiman	4	44	0.5811475	0.9934426	0.000
Breiman	4	45	0.5790984	0.9934426	0.000
Breiman	4	46	0.5729508	0.9967213	0.000
Breiman	4	47	0.5717213	0.9967213	0.000
Breiman	4	48	0.5942623	0.9967213	0.000
Breiman	4	49	0.5926230	0.9967213	0.000
Breiman	4	50	0.6250000	0.9934426	0.000
Breiman	4	51	0.6444672	0.9967213	0.000
Breiman	4	52	0.6358607	0.9934426	0.000
Breiman	4	53	0.6440574	0.9934426	0.000
Breiman	4	54	0.6399590	0.9934426	0.000
Breiman	4	55	0.6366803	0.9934426	0.000
Breiman	4	56	0.6247951	0.9901639	0.000
Breiman	4	57	0.6297131	0.9901639	0.000

Breiman	4	58	0.6383197	0.9901639	0.000
Breiman	4	59	0.6436475	0.9901639	0.000
Breiman	4	60	0.6362705	0.9901639	0.000
Breiman	4	61	0.6706967	0.9901639	0.000
Breiman	4	62	0.6815574	0.9934426	0.000
Breiman	4	63	0.6721311	0.9901639	0.000
Breiman	4	64	0.6795082	0.9901639	0.000
Breiman	4	65	0.6877049	0.9901639	0.000
Breiman	4	66	0.6807377	0.9901639	0.000
Breiman	4	67	0.6778689	0.9901639	0.000
Breiman	4	68	0.6762295	0.9901639	0.000
Breiman	4	69	0.6868852	0.9901639	0.000
Breiman	4	70	0.6848361	0.9901639	0.000
Breiman	4	71	0.6983607	0.9901639	0.000
Breiman	4	72	0.6979508	0.9901639	0.000
Breiman	4	73	0.6901639	0.9901639	0.000
Breiman	4	74	0.6897541	0.9901639	0.000
Breiman	4	75	0.6905738	0.9901639	0.000
Breiman	4	76	0.6868852	0.9901639	0.000
Breiman	4	77	0.6864754	0.9901639	0.000
Breiman	4	78	0.6807377	0.9901639	0.000
Breiman	4	79	0.6905738	0.9901639	0.000
Breiman	4	80	0.6942623	0.9901639	0.000
Breiman	4	81	0.6983607	0.9901639	0.000
Breiman	4	82	0.6901639	0.9901639	0.000
Breiman	4	83	0.6897541	0.9901639	0.000
Breiman	4	84	0.6881148	0.9901639	0.000
Breiman	4	85	0.6827869	0.9901639	0.000
Breiman	4	86	0.6696721	0.9901639	0.000
Breiman	4	87	0.6758197	0.9901639	0.000
Breiman	4	88	0.6631148	0.9901639	0.000
Breiman	4	89	0.6627049	0.9901639	0.000
Breiman	4	90	0.6618852	0.9901639	0.000
Breiman	4	91	0.6454918	0.9901639	0.000
Breiman	4	92	0.6512295	0.9934426	0.000
Breiman	4	93	0.6483607	0.9934426	0.000
Breiman	4	94	0.6467213	0.9934426	0.000
Breiman	4	95	0.6475410	0.9934426	0.000
Breiman	4	96	0.6487705	0.9934426	0.000
Breiman	4	97	0.6491803	0.9934426	0.000
Breiman	4	98	0.6450820	0.9934426	0.000
Breiman	4	99	0.6340164	0.9934426	0.000
Breiman	4	100	0.6397541	0.9934426	0.000
Freund	2	20	0.7725410	0.9934426	0.000
Freund	2	21	0.7709016	0.9967213	0.000
Freund	2	22	0.7694672	0.9934426	0.000
Freund	2	23	0.7694672	0.9967213	0.000
Freund	2	24	0.7672131	0.9934426	0.000

Freund	2	25	0.7647541	0.9934426	0.000
Freund	2	26	0.7756148	0.9934426	0.000
Freund	2	27	0.7557377	0.9868852	0.000
Freund	2	28	0.7323770	0.9868852	0.000
Freund	2	29	0.7422131	0.9901639	0.000
Freund	2	30	0.7719262	0.9901639	0.000
Freund	2	31	0.7706967	0.9934426	0.000
Freund	2	32	0.7555328	0.9967213	0.000
Freund	2	33	0.7772541	0.9934426	0.000
Freund	2	34	0.7604508	0.9901639	0.000
Freund	2	35	0.7649590	0.9967213	0.000
Freund	2	36	0.7727459	0.9901639	0.000
Freund	2	37	0.7694672	0.9967213	0.000
Freund	2	38	0.7625000	0.9967213	0.000
Freund	2	39	0.7596311	0.9967213	0.000
Freund	2	40	0.7469262	0.9967213	0.000
Freund	2	41	0.7334016	0.9934426	0.000
Freund	2	42	0.7081967	0.9967213	0.000
Freund	2	43	0.7196721	1.0000000	0.000
Freund	2	44	0.7286885	0.9967213	0.000
Freund	2	45	0.7286885	1.0000000	0.000
Freund	2	46	0.7590164	0.9967213	0.000
Freund	2	47	0.7639344	0.9967213	0.000
Freund	2	48	0.7647541	0.9967213	0.000
Freund	2	49	0.7434426	1.0000000	0.000
Freund	2	50	0.7282787	1.0000000	0.000
Freund	2	51	0.7282787	1.0000000	0.000
Freund	2	52	0.7389344	1.0000000	0.000
Freund	2	53	0.7393443	1.0000000	0.000
Freund	2	54	0.7340164	1.0000000	0.000
Freund	2	55	0.7303279	1.0000000	0.000
Freund	2	56	0.7032787	1.0000000	0.000
Freund	2	57	0.7028689	1.0000000	0.000
Freund	2	58	0.7159836	1.0000000	0.000
Freund	2	59	0.7147541	1.0000000	0.000
Freund	2	60	0.7049180	1.0000000	0.000
Freund	2	61	0.7045082	1.0000000	0.000
Freund	2	62	0.7122951	1.0000000	0.000
Freund	2	63	0.7122951	1.0000000	0.000
Freund	2	64	0.7098361	1.0000000	0.000
Freund	2	65	0.7053279	1.0000000	0.000
Freund	2	66	0.6942623	0.9967213	0.000
Freund	2	67	0.6967213	1.0000000	0.000
Freund	2	68	0.7045082	0.9967213	0.000
Freund	2	69	0.6979508	0.9967213	0.000
Freund	2	70	0.6848361	0.9934426	0.000
Freund	2	71	0.7094262	0.9967213	0.000
Freund	2	72	0.7061475	0.9967213	0.000

Freund	2	73	0.7049180	0.9967213	0.000
Freund	2	74	0.6971311	0.9967213	0.000
Freund	2	75	0.6864754	0.9967213	0.000
Freund	2	76	0.6823770	0.9967213	0.000
Freund	2	77	0.6856557	0.9967213	0.000
Freund	2	78	0.6729508	0.9967213	0.000
Freund	2	79	0.6782787	0.9967213	0.000
Freund	2	80	0.6762295	0.9967213	0.000
Freund	2	81	0.6758197	0.9967213	0.000
Freund	2	82	0.6762295	0.9967213	0.000
Freund	2	83	0.6762295	0.9967213	0.000
Freund	2	84	0.6704918	0.9967213	0.000
Freund	2	85	0.6831967	0.9967213	0.000
Freund	2	86	0.6946721	0.9967213	0.000
Freund	2	87	0.6946721	0.9967213	0.000
Freund	2	88	0.6946721	0.9967213	0.000
Freund	2	89	0.6942623	0.9967213	0.000
Freund	2	90	0.7004098	0.9967213	0.000
Freund	2	91	0.7049180	0.9967213	0.000
Freund	2	92	0.7147541	1.0000000	0.000
Freund	2	93	0.7049180	1.0000000	0.000
Freund	2	94	0.6819672	1.0000000	0.000
Freund	2	95	0.6754098	0.9967213	0.000
Freund	2	96	0.6754098	0.9967213	0.000
Freund	2	97	0.6733607	0.9967213	0.000
Freund	2	98	0.6782787	0.9967213	0.000
Freund	2	99	0.6663934	0.9967213	0.000
Freund	2	100	0.6786885	0.9967213	0.000
Freund	3	20	0.8172131	0.9934426	0.000
Freund	3	21	0.8262295	0.9934426	0.000
Freund	3	22	0.8069672	0.9901639	0.000
Freund	3	23	0.7819672	0.9901639	0.000
Freund	3	24	0.7770492	0.9901639	0.000
Freund	3	25	0.7803279	0.9868852	0.000
Freund	3	26	0.7561475	0.9901639	0.000
Freund	3	27	0.7200820	0.9901639	0.000
Freund	3	28	0.7200820	0.9901639	0.000
Freund	3	29	0.6930328	0.9967213	0.000
Freund	3	30	0.6852459	0.9934426	0.000
Freund	3	31	0.7200820	0.9934426	0.000
Freund	3	32	0.7004098	0.9868852	0.000
Freund	3	33	0.7098361	0.9868852	0.000
Freund	3	34	0.7086066	0.9868852	0.000
Freund	3	35	0.7598361	0.9868852	0.000
Freund	3	36	0.7573770	0.9868852	0.000
Freund	3	37	0.8147541	0.9901639	0.000
Freund	3	38	0.7741803	0.9934426	0.000
Freund	3	39	0.7836066	0.9901639	0.000

Freund	3	40	0.7729508	0.9901639	0.000
Freund	3	41	0.7840164	0.9901639	0.000
Freund	3	42	0.7827869	0.9901639	0.000
Freund	3	43	0.7754098	0.9901639	0.000
Freund	3	44	0.7635246	0.9901639	0.000
Freund	3	45	0.7696721	0.9901639	0.000
Freund	3	46	0.7696721	0.9934426	0.000
Freund	3	47	0.7725410	0.9901639	0.000
Freund	3	48	0.7639344	0.9868852	0.000
Freund	3	49	0.7618852	0.9901639	0.000
Freund	3	50	0.7573770	0.9934426	0.000
Freund	3	51	0.7327869	0.9901639	0.000
Freund	3	52	0.7319672	0.9901639	0.000
Freund	3	53	0.7336066	0.9901639	0.000
Freund	3	54	0.7200820	0.9967213	0.000
Freund	3	55	0.7327869	0.9901639	0.000
Freund	3	56	0.7397541	0.9934426	0.000
Freund	3	57	0.7229508	0.9934426	0.000
Freund	3	58	0.7237705	0.9934426	0.000
Freund	3	59	0.7491803	0.9934426	0.000
Freund	3	60	0.7442623	0.9934426	0.000
Freund	3	61	0.7385246	0.9934426	0.000
Freund	3	62	0.7295082	0.9934426	0.000
Freund	3	63	0.7200820	0.9934426	0.000
Freund	3	64	0.7090164	0.9934426	0.000
Freund	3	65	0.7196721	0.9934426	0.000
Freund	3	66	0.7188525	0.9934426	0.000
Freund	3	67	0.7286885	0.9934426	0.000
Freund	3	68	0.7090164	0.9967213	0.000
Freund	3	69	0.7200820	0.9967213	0.000
Freund	3	70	0.7204918	0.9967213	0.000
Freund	3	71	0.7200820	0.9967213	0.000
Freund	3	72	0.7118852	0.9967213	0.000
Freund	3	73	0.7229508	0.9934426	0.000
Freund	3	74	0.7278689	0.9934426	0.000
Freund	3	75	0.7250000	0.9934426	0.000
Freund	3	76	0.7188525	0.9967213	0.000
Freund	3	77	0.7196721	0.9934426	0.000
Freund	3	78	0.7135246	0.9934426	0.000
Freund	3	79	0.7118852	0.9934426	0.000
Freund	3	80	0.7065574	0.9934426	0.000
Freund	3	81	0.7073770	0.9934426	0.000
Freund	3	82	0.7127049	0.9901639	0.000
Freund	3	83	0.7032787	0.9934426	0.000
Freund	3	84	0.7016393	0.9901639	0.000
Freund	3	85	0.6959016	0.9934426	0.000
Freund	3	86	0.7045082	0.9934426	0.000
Freund	3	87	0.7213115	0.9934426	0.000

Freund	3	88	0.7192623	0.9934426	0.000
Freund	3	89	0.7155738	0.9934426	0.000
Freund	3	90	0.7344262	0.9934426	0.000
Freund	3	91	0.7393443	0.9967213	0.000
Freund	3	92	0.7327869	0.9934426	0.000
Freund	3	93	0.7336066	0.9967213	0.000
Freund	3	94	0.7536885	0.9934426	0.000
Freund	3	95	0.7598361	0.9934426	0.000
Freund	3	96	0.7569672	0.9934426	0.000
Freund	3	97	0.7479508	0.9901639	0.000
Freund	3	98	0.7422131	0.9934426	0.000
Freund	3	99	0.7549180	0.9934426	0.000
Freund	3	100	0.7491803	0.9934426	0.000
Freund	4	20	0.5987705	0.9868852	0.125
Freund	4	21	0.5963115	0.9901639	0.125
Freund	4	22	0.6028689	0.9901639	0.125
Freund	4	23	0.5524590	0.9901639	0.000
Freund	4	24	0.5868852	0.9901639	0.000
Freund	4	25	0.5655738	0.9901639	0.000
Freund	4	26	0.5516393	0.9868852	0.000
Freund	4	27	0.5500000	0.9868852	0.000
Freund	4	28	0.5336066	0.9868852	0.000
Freund	4	29	0.5393443	0.9901639	0.000
Freund	4	30	0.5256148	0.9868852	0.000
Freund	4	31	0.5202869	0.9901639	0.000
Freund	4	32	0.5137295	0.9901639	0.000
Freund	4	33	0.4805328	0.9901639	0.000
Freund	4	34	0.4911885	0.9901639	0.000
Freund	4	35	0.4965164	0.9901639	0.000
Freund	4	36	0.5375000	0.9901639	0.000
Freund	4	37	0.5239754	0.9901639	0.000
Freund	4	38	0.5145492	0.9901639	0.000
Freund	4	39	0.5088115	0.9901639	0.000
Freund	4	40	0.5473361	0.9901639	0.000
Freund	4	41	0.5784836	0.9901639	0.000
Freund	4	42	0.5891393	0.9901639	0.000
Freund	4	43	0.5862705	0.9901639	0.000
Freund	4	44	0.5946721	0.9901639	0.000
Freund	4	45	0.6135246	0.9934426	0.000
Freund	4	46	0.6241803	0.9901639	0.000
Freund	4	47	0.6180328	0.9901639	0.000
Freund	4	48	0.6135246	0.9901639	0.000
Freund	4	49	0.6036885	0.9934426	0.000
Freund	4	50	0.6049180	0.9934426	0.000
Freund	4	51	0.5950820	0.9934426	0.000
Freund	4	52	0.5967213	0.9901639	0.000
Freund	4	53	0.6163934	0.9934426	0.000
Freund	4	54	0.6131148	0.9901639	0.000

Freund	4	55	0.6081967	0.9934426	0.000
Freund	4	56	0.6102459	0.9934426	0.000
Freund	4	57	0.6184426	0.9934426	0.000
Freund	4	58	0.6106557	0.9934426	0.000
Freund	4	59	0.6196721	0.9934426	0.000
Freund	4	60	0.6290984	0.9901639	0.000
Freund	4	61	0.6356557	0.9934426	0.000
Freund	4	62	0.6385246	0.9934426	0.000
Freund	4	63	0.6336066	0.9901639	0.000
Freund	4	64	0.6348361	0.9868852	0.000
Freund	4	65	0.6344262	0.9868852	0.000
Freund	4	66	0.6413934	0.9868852	0.000
Freund	4	67	0.6647541	0.9868852	0.000
Freund	4	68	0.6655738	0.9868852	0.000
Freund	4	69	0.6688525	0.9868852	0.000
Freund	4	70	0.6811475	0.9934426	0.000
Freund	4	71	0.6991803	0.9868852	0.000
Freund	4	72	0.7012295	0.9868852	0.000
Freund	4	73	0.7032787	0.9868852	0.000
Freund	4	74	0.7016393	0.9868852	0.000
Freund	4	75	0.7045082	0.9868852	0.000
Freund	4	76	0.7020492	0.9868852	0.000
Freund	4	77	0.6844262	0.9868852	0.000
Freund	4	78	0.6918033	0.9868852	0.000
Freund	4	79	0.6926230	0.9901639	0.000
Freund	4	80	0.6991803	0.9901639	0.000
Freund	4	81	0.7118852	0.9901639	0.000
Freund	4	82	0.7151639	0.9901639	0.000
Freund	4	83	0.7024590	0.9901639	0.000
Freund	4	84	0.6930328	0.9901639	0.000
Freund	4	85	0.7012295	0.9901639	0.000
Freund	4	86	0.6975410	0.9901639	0.000
Freund	4	87	0.6975410	0.9901639	0.000
Freund	4	88	0.6971311	0.9901639	0.000
Freund	4	89	0.6856557	0.9934426	0.000
Freund	4	90	0.6815574	0.9901639	0.000
Freund	4	91	0.6774590	0.9934426	0.000
Freund	4	92	0.6696721	0.9934426	0.000
Freund	4	93	0.6790984	0.9934426	0.000
Freund	4	94	0.6901639	0.9934426	0.000
Freund	4	95	0.6852459	0.9934426	0.000
Freund	4	96	0.6823770	0.9934426	0.000
Freund	4	97	0.6840164	0.9934426	0.000
Freund	4	98	0.6819672	0.9934426	0.000
Freund	4	99	0.6881148	0.9934426	0.000
Freund	4	100	0.6827869	0.9934426	0.000
Zhu	2	20	0.6838115	0.9967213	0.000
Zhu	2	21	0.6838115	1.0000000	0.000

Zhu	2	22	0.6977459	1.0000000	0.000
Zhu	2	23	0.6899590	1.0000000	0.000
Zhu	2	24	0.6899590	1.0000000	0.000
Zhu	2	25	0.6842213	1.0000000	0.000
Zhu	2	26	0.6793033	1.0000000	0.000
Zhu	2	27	0.6651639	1.0000000	0.000
Zhu	2	28	0.6637295	1.0000000	0.000
Zhu	2	29	0.6522541	1.0000000	0.000
Zhu	2	30	0.6424180	0.9967213	0.000
Zhu	2	31	0.6407787	1.0000000	0.000
Zhu	2	32	0.6526639	1.0000000	0.000
Zhu	2	33	0.6538934	0.9967213	0.000
Zhu	2	34	0.6538934	1.0000000	0.000
Zhu	2	35	0.6518443	0.9967213	0.000
Zhu	2	36	0.6547131	1.0000000	0.000
Zhu	2	37	0.6522541	0.9967213	0.000
Zhu	2	38	0.6645492	0.9934426	0.000
Zhu	2	39	0.6645492	0.9934426	0.000
Zhu	2	40	0.6645492	0.9934426	0.000
Zhu	2	41	0.6825820	0.9934426	0.000
Zhu	2	42	0.6850410	0.9934426	0.000
Zhu	2	43	0.6555328	0.9934426	0.000
Zhu	2	44	0.6555328	0.9934426	0.000
Zhu	2	45	0.6723361	0.9934426	0.000
Zhu	2	46	0.6682377	0.9934426	0.000
Zhu	2	47	0.6727459	0.9934426	0.000
Zhu	2	48	0.6657787	0.9934426	0.000
Zhu	2	49	0.6797131	0.9967213	0.000
Zhu	2	50	0.6633197	0.9934426	0.000
Zhu	2	51	0.6629098	0.9934426	0.000
Zhu	2	52	0.6198770	0.9967213	0.000
Zhu	2	53	0.6346311	0.9967213	0.000
Zhu	2	54	0.6370902	0.9967213	0.000
Zhu	2	55	0.6108607	0.9967213	0.000
Zhu	2	56	0.6096311	0.9967213	0.000
Zhu	2	57	0.6211066	0.9967213	0.000
Zhu	2	58	0.6444672	0.9967213	0.000
Zhu	2	59	0.6444672	0.9967213	0.000
Zhu	2	60	0.6750000	0.9967213	0.000
Zhu	2	61	0.6750000	0.9967213	0.000
Zhu	2	62	0.6577869	0.9967213	0.000
Zhu	2	63	0.6471311	0.9967213	0.000
Zhu	2	64	0.6805328	0.9967213	0.000
Zhu	2	65	0.7002049	0.9967213	0.000
Zhu	2	66	0.6600410	0.9967213	0.000
Zhu	2	67	0.6670082	0.9967213	0.000
Zhu	2	68	0.6780738	0.9967213	0.000
Zhu	2	69	0.6784836	0.9967213	0.000

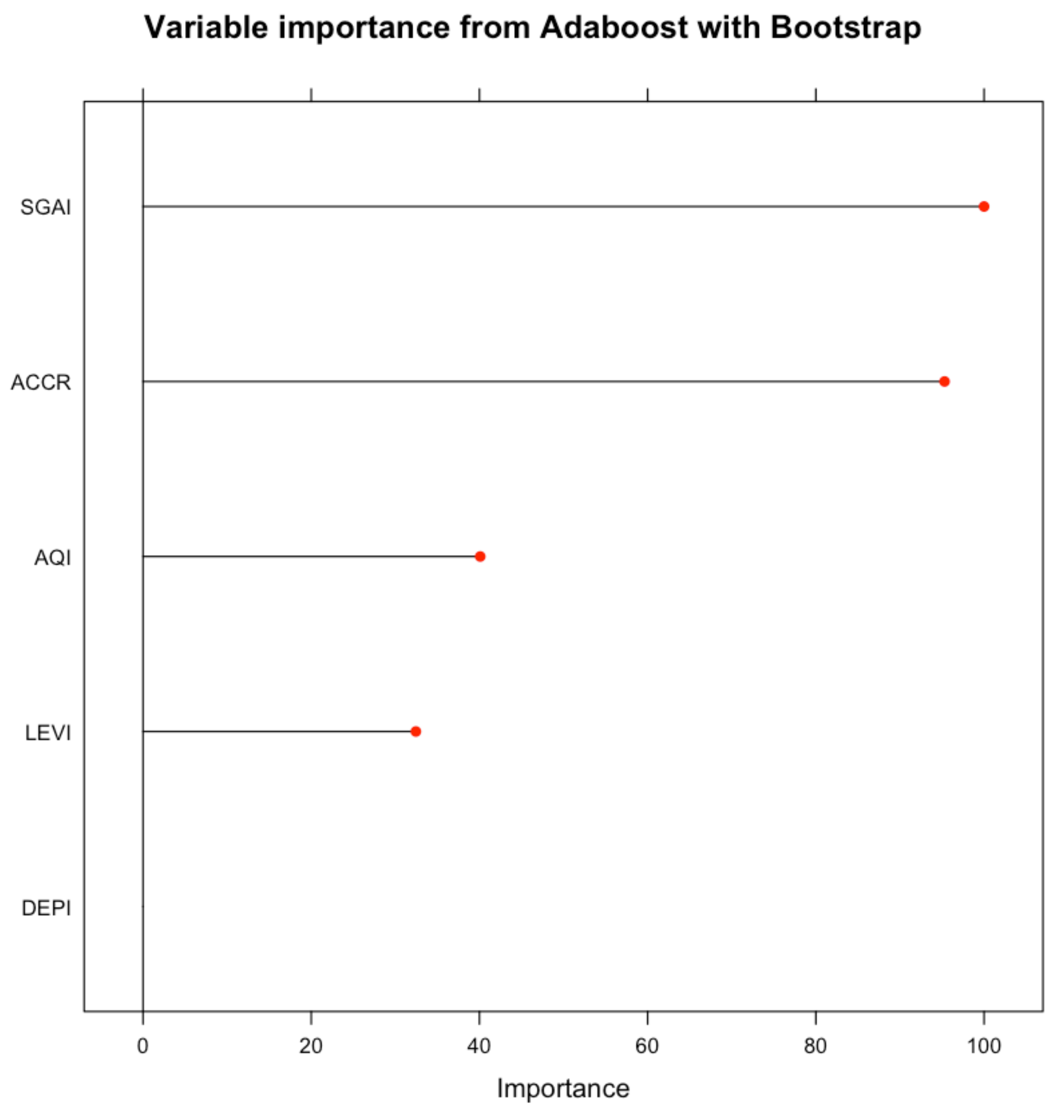
Zhu	2	70	0.6768443	0.9967213	0.000
Zhu	2	71	0.6715164	0.9967213	0.000
Zhu	2	72	0.6719262	0.9967213	0.000
Zhu	2	73	0.6903689	0.9967213	0.000
Zhu	2	74	0.6838115	0.9967213	0.000
Zhu	2	75	0.6838115	0.9967213	0.000
Zhu	2	76	0.6838115	0.9967213	0.000
Zhu	2	77	0.6747951	0.9967213	0.000
Zhu	2	78	0.6747951	0.9967213	0.000
Zhu	2	79	0.6801230	0.9967213	0.000
Zhu	2	80	0.6801230	0.9967213	0.000
Zhu	2	81	0.6772541	0.9967213	0.000
Zhu	2	82	0.6772541	1.0000000	0.000
Zhu	2	83	0.6719262	0.9967213	0.000
Zhu	2	84	0.6719262	1.0000000	0.000
Zhu	2	85	0.6686475	0.9967213	0.000
Zhu	2	86	0.6702869	0.9967213	0.000
Zhu	2	87	0.6731557	0.9967213	0.000
Zhu	2	88	0.6731557	1.0000000	0.000
Zhu	2	89	0.6633197	1.0000000	0.000
Zhu	2	90	0.6489754	1.0000000	0.000
Zhu	2	91	0.6756148	0.9967213	0.000
Zhu	2	92	0.6616803	0.9967213	0.000
Zhu	2	93	0.6608607	0.9967213	0.000
Zhu	2	94	0.6424180	0.9967213	0.000
Zhu	2	95	0.6538934	0.9967213	0.000
Zhu	2	96	0.6670082	0.9967213	0.000
Zhu	2	97	0.6649590	0.9967213	0.000
Zhu	2	98	0.6616803	0.9967213	0.000
Zhu	2	99	0.6653689	0.9967213	0.000
Zhu	2	100	0.6321721	0.9967213	0.000
Zhu	3	20	0.5063525	0.9934426	0.000
Zhu	3	21	0.5034836	0.9967213	0.000
Zhu	3	22	0.4805328	0.9934426	0.000
Zhu	3	23	0.5112705	0.9967213	0.000
Zhu	3	24	0.5100410	0.9967213	0.000
Zhu	3	25	0.5264344	0.9967213	0.000
Zhu	3	26	0.5055328	1.0000000	0.000
Zhu	3	27	0.5319672	1.0000000	0.000
Zhu	3	28	0.5217213	1.0000000	0.000
Zhu	3	29	0.4959016	0.9934426	0.000
Zhu	3	30	0.4893443	0.9967213	0.000
Zhu	3	31	0.4868852	0.9967213	0.000
Zhu	3	32	0.5024590	1.0000000	0.000
Zhu	3	33	0.4889344	1.0000000	0.000
Zhu	3	34	0.4803279	1.0000000	0.000
Zhu	3	35	0.5020492	1.0000000	0.000
Zhu	3	36	0.5139344	1.0000000	0.000

Zhu	3	37	0.4987705	1.0000000	0.000
Zhu	3	38	0.5311475	1.0000000	0.000
Zhu	3	39	0.5254098	1.0000000	0.000
Zhu	3	40	0.5258197	1.0000000	0.000
Zhu	3	41	0.5254098	1.0000000	0.000
Zhu	3	42	0.5090164	1.0000000	0.000
Zhu	3	43	0.5209016	1.0000000	0.000
Zhu	3	44	0.5176230	1.0000000	0.000
Zhu	3	45	0.5184426	1.0000000	0.000
Zhu	3	46	0.5364754	1.0000000	0.000
Zhu	3	47	0.5348361	1.0000000	0.000
Zhu	3	48	0.5254098	1.0000000	0.000
Zhu	3	49	0.5303279	1.0000000	0.000
Zhu	3	50	0.5356557	1.0000000	0.000
Zhu	3	51	0.5352459	1.0000000	0.000
Zhu	3	52	0.5491803	1.0000000	0.000
Zhu	3	53	0.5479508	1.0000000	0.000
Zhu	3	54	0.5454918	1.0000000	0.000
Zhu	3	55	0.5454918	1.0000000	0.000
Zhu	3	56	0.5430328	1.0000000	0.000
Zhu	3	57	0.5286885	1.0000000	0.000
Zhu	3	58	0.5487705	1.0000000	0.000
Zhu	3	59	0.5352459	1.0000000	0.000
Zhu	3	60	0.5413934	1.0000000	0.000
Zhu	3	61	0.5516393	0.9967213	0.000
Zhu	3	62	0.5565574	0.9967213	0.000
Zhu	3	63	0.5836066	0.9967213	0.000
Zhu	3	64	0.5831967	0.9967213	0.000
Zhu	3	65	0.5897541	0.9967213	0.000
Zhu	3	66	0.5909836	0.9967213	0.000
Zhu	3	67	0.5950820	1.0000000	0.000
Zhu	3	68	0.5823770	0.9967213	0.000
Zhu	3	69	0.6163934	0.9967213	0.000
Zhu	3	70	0.6069672	0.9967213	0.000
Zhu	3	71	0.6065574	0.9967213	0.000
Zhu	3	72	0.5721311	0.9967213	0.000
Zhu	3	73	0.5721311	0.9934426	0.000
Zhu	3	74	0.5635246	0.9934426	0.000
Zhu	3	75	0.5635246	0.9934426	0.000
Zhu	3	76	0.5762295	0.9967213	0.000
Zhu	3	77	0.5840164	0.9967213	0.000
Zhu	3	78	0.5709016	0.9967213	0.000
Zhu	3	79	0.5647541	0.9967213	0.000
Zhu	3	80	0.5704918	0.9967213	0.000
Zhu	3	81	0.5737705	0.9967213	0.000
Zhu	3	82	0.6229508	0.9967213	0.000
Zhu	3	83	0.6229508	0.9967213	0.000
Zhu	3	84	0.6245902	0.9967213	0.000

Zhu	3	85	0.6299180	0.9967213	0.000
Zhu	3	86	0.6491803	0.9967213	0.000
Zhu	3	87	0.6532787	0.9967213	0.000
Zhu	3	88	0.6532787	0.9967213	0.000
Zhu	3	89	0.6442623	0.9967213	0.000
Zhu	3	90	0.6495902	0.9967213	0.000
Zhu	3	91	0.6508197	0.9967213	0.000
Zhu	3	92	0.6426230	1.0000000	0.000
Zhu	3	93	0.6319672	1.0000000	0.000
Zhu	3	94	0.6278689	1.0000000	0.000
Zhu	3	95	0.6204918	0.9967213	0.000
Zhu	3	96	0.6274590	0.9967213	0.000
Zhu	3	97	0.6319672	0.9967213	0.000
Zhu	3	98	0.6327869	0.9967213	0.000
Zhu	3	99	0.6172131	0.9967213	0.000
Zhu	3	100	0.6172131	0.9967213	0.000
Zhu	4	20	0.6866803	0.9967213	0.000
Zhu	4	21	0.7364754	0.9967213	0.000
Zhu	4	22	0.7467213	0.9967213	0.000
Zhu	4	23	0.7276639	0.9934426	0.000
Zhu	4	24	0.7329918	0.9901639	0.000
Zhu	4	25	0.7293033	0.9934426	0.000
Zhu	4	26	0.7051230	0.9967213	0.000
Zhu	4	27	0.7047131	0.9967213	0.000
Zhu	4	28	0.6944672	0.9967213	0.000
Zhu	4	29	0.7047131	0.9934426	0.125
Zhu	4	30	0.7071721	0.9934426	0.125
Zhu	4	31	0.6575820	0.9934426	0.125
Zhu	4	32	0.6661885	0.9901639	0.125
Zhu	4	33	0.6653689	0.9934426	0.125
Zhu	4	34	0.6506148	0.9934426	0.125
Zhu	4	35	0.6530738	0.9934426	0.125
Zhu	4	36	0.6682377	0.9901639	0.125
Zhu	4	37	0.6526639	0.9934426	0.125
Zhu	4	38	0.6485656	0.9901639	0.125
Zhu	4	39	0.6569672	0.9934426	0.125
Zhu	4	40	0.6557377	0.9934426	0.000
Zhu	4	41	0.6737705	0.9934426	0.125
Zhu	4	42	0.6774590	0.9901639	0.125
Zhu	4	43	0.6815574	0.9901639	0.125
Zhu	4	44	0.6750000	0.9901639	0.125
Zhu	4	45	0.6733607	0.9901639	0.125
Zhu	4	46	0.6659836	0.9901639	0.125
Zhu	4	47	0.6569672	0.9901639	0.125
Zhu	4	48	0.6684426	0.9901639	0.000
Zhu	4	49	0.6622951	0.9901639	0.125
Zhu	4	50	0.6602459	0.9901639	0.125
Zhu	4	51	0.6663934	0.9901639	0.125

Zhu	4	52	0.6717213	0.9901639	0.000
Zhu	4	53	0.6909836	0.9901639	0.125
Zhu	4	54	0.6995902	0.9901639	0.125
Zhu	4	55	0.6942623	0.9901639	0.000
Zhu	4	56	0.7180328	0.9901639	0.125
Zhu	4	57	0.7241803	0.9901639	0.125
Zhu	4	58	0.7303279	0.9901639	0.125
Zhu	4	59	0.7418033	0.9901639	0.125
Zhu	4	60	0.7413934	0.9901639	0.125
Zhu	4	61	0.7327869	0.9901639	0.125
Zhu	4	62	0.7245902	0.9901639	0.125
Zhu	4	63	0.7213115	0.9901639	0.125
Zhu	4	64	0.7049180	0.9901639	0.125
Zhu	4	65	0.7286885	0.9901639	0.125
Zhu	4	66	0.7299180	0.9901639	0.125
Zhu	4	67	0.7155738	0.9901639	0.125
Zhu	4	68	0.7061475	0.9901639	0.125
Zhu	4	69	0.7098361	0.9901639	0.125
Zhu	4	70	0.7090164	0.9901639	0.125
Zhu	4	71	0.7118852	0.9901639	0.125
Zhu	4	72	0.7163934	0.9901639	0.125
Zhu	4	73	0.7151639	0.9901639	0.125
Zhu	4	74	0.7229508	0.9901639	0.125
Zhu	4	75	0.7168033	0.9901639	0.125
Zhu	4	76	0.7172131	0.9901639	0.125
Zhu	4	77	0.7090164	0.9901639	0.125
Zhu	4	78	0.7188525	0.9901639	0.125
Zhu	4	79	0.7180328	0.9934426	0.125
Zhu	4	80	0.7233607	0.9901639	0.125
Zhu	4	81	0.7245902	0.9901639	0.000
Zhu	4	82	0.7209016	0.9901639	0.000
Zhu	4	83	0.7196721	0.9901639	0.000
Zhu	4	84	0.7057377	0.9901639	0.000
Zhu	4	85	0.7024590	0.9901639	0.125
Zhu	4	86	0.7000000	0.9901639	0.125
Zhu	4	87	0.6897541	0.9901639	0.125
Zhu	4	88	0.6860656	0.9901639	0.125
Zhu	4	89	0.6889344	0.9901639	0.125
Zhu	4	90	0.6959016	0.9901639	0.125
Zhu	4	91	0.6959016	0.9901639	0.125
Zhu	4	92	0.7081967	0.9901639	0.125
Zhu	4	93	0.7069672	0.9901639	0.125
Zhu	4	94	0.7061475	0.9901639	0.125
Zhu	4	95	0.7004098	0.9901639	0.125
Zhu	4	96	0.6905738	0.9901639	0.125
Zhu	4	97	0.6930328	0.9901639	0.125
Zhu	4	98	0.7004098	0.9901639	0.125
Zhu	4	99	0.6942623	0.9901639	0.125

```
  Zhu       4         100     0.6905738  0.9901639  0.125

ROC was used to select the optimal model using the largest value.
The final values used for the model were mfinal = 21, maxdepth = 3
 and coeflearn = Freund.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  96.8  2.6
       Yes  0.6  0.0

 Accuracy (average) : 0.9681
```



Confusion Matrix for adaboost on test set

```
In [42]: caretPredictedClass <- predict(ada_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  356   9
       Yes   4   2

               Accuracy : 0.965
                 95% CI : (0.9408, 0.9812)
```

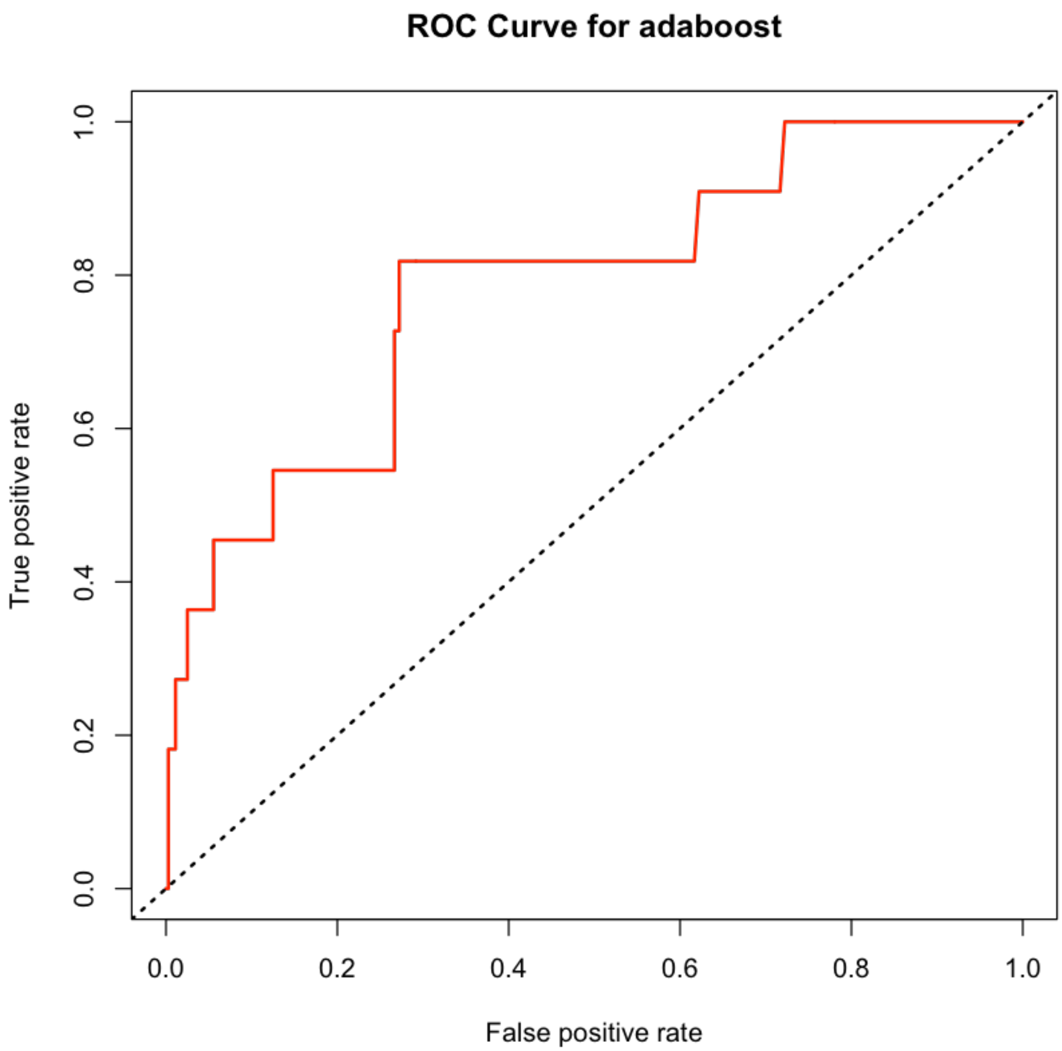
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.7841

                  Kappa : 0.2189
 Mcnemar's Test P-Value : 0.2673

            Sensitivity : 0.9889
            Specificity : 0.1818
         Pos Pred Value : 0.9753
         Neg Pred Value : 0.3333
             Prevalence : 0.9704
         Detection Rate : 0.9596
   Detection Prevalence : 0.9838
      Balanced Accuracy : 0.5854

       'Positive' Class : No
```

ROC plot for adaboost on test set

```
In [43]: ada_pred <- predict(ada_model, model_test_df, type = "prob")[,2]
         ada_prediction <- prediction(ada_pred,model_test_df$Manipulater)
         ada_perf <- performance(ada_prediction, "tpr","fpr")

         plot(ada_perf,main="ROC Curve for adaboost",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(ada_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.7848485
```

```
Slot "alpha.values":
list()
```



Visulaizing the rules coming out of ada boost. We can loop and print all the trees which was built using boosting. For simplicity, we are printing just one of the trees

To retrieve the understand any model specific attribute, we have to call the **$finalmodel** of the train object created using caret package. This is a generic way to use functions which are model specific. Here **get_tree()** is a function of **fastadaboost** package which cannot be used unless the the object returned is not of adaboost class.

```
In [44]: #listTreesAda(ada_model$finalModel,3) #this is a function with rattle package
         #get_tree(ada_model$finalModel,2)
```

1.5.3 Boosting with adaboost (upsample)

The below code chunk sets some of the control parameters for adaboost

```
In [45]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='all',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE,
                                    sampling = "up")#, p = 0.70) #in case method = #"LGOCV"
```

```
In [46]: search_grid <- expand.grid(mfinal = c(20:100), maxdepth = c(2:4),
                             coeflearn = c("Breiman", "Freund", "Zhu"))
```

After setting the control paramters, the model is run

```
In [47]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Adaptive Boosting with UP Sample")

         set.seed(4121)
         ada_up_model <- train(model_train_df[,1:5], model_train_df[,6],
                                  method='AdaBoost.M1',
                                  trControl=objControl,
                                  tuneGrid = search_grid,
                                  metric = "ROC")
         stopCluster(num_cores)
         toc()
```

```
Adaptive Boosting with UP Sample: 116.933 sec elapsed
```

Confusion Matrix for adaboost on train set

```
In [48]: #ada_up_model$finalModel #ada_up_model$results
         print(ada_up_model)
         confusionMatrix.train(ada_up_model)
         plot(varImp(ada_up_model), main = "Variable importance from Adaboost with Up Sample",
```

```
AdaBoost.M1

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using up-sampling
```

```
Resampling results across tuning parameters:

  coeflearn  maxdepth  mfinal  ROC        Sens       Spec
  Breiman    2          20     0.7114754  0.9245902  0.000
  Breiman    2          21     0.7135246  0.9278689  0.125
  Breiman    2          22     0.7135246  0.9344262  0.000
  Breiman    2          23     0.6938525  0.9311475  0.000
  Breiman    2          24     0.6938525  0.9311475  0.000
  Breiman    2          25     0.6821721  0.9344262  0.000
  Breiman    2          26     0.6903689  0.9409836  0.000
  Breiman    2          27     0.6934426  0.9409836  0.000
  Breiman    2          28     0.6926230  0.9344262  0.000
  Breiman    2          29     0.6926230  0.9475410  0.000
  Breiman    2          30     0.6868852  0.9442623  0.000
  Breiman    2          31     0.6868852  0.9475410  0.000
  Breiman    2          32     0.6889344  0.9442623  0.000
  Breiman    2          33     0.7040984  0.9409836  0.000
  Breiman    2          34     0.6844262  0.9409836  0.000
  Breiman    2          35     0.6795082  0.9475410  0.000
  Breiman    2          36     0.6795082  0.9475410  0.000
  Breiman    2          37     0.6856557  0.9508197  0.000
  Breiman    2          38     0.6950820  0.9475410  0.000
  Breiman    2          39     0.6950820  0.9475410  0.000
  Breiman    2          40     0.7073770  0.9475410  0.000
  Breiman    2          41     0.7065574  0.9442623  0.000
  Breiman    2          42     0.7020492  0.9442623  0.000
  Breiman    2          43     0.7020492  0.9442623  0.000
  Breiman    2          44     0.7147541  0.9508197  0.000
  Breiman    2          45     0.7094262  0.9475410  0.000
  Breiman    2          46     0.7057377  0.9442623  0.000
  Breiman    2          47     0.7020492  0.9475410  0.000
  Breiman    2          48     0.7090164  0.9475410  0.000
  Breiman    2          49     0.7090164  0.9508197  0.000
  Breiman    2          50     0.7090164  0.9540984  0.000
  Breiman    2          51     0.6875000  0.9540984  0.000
  Breiman    2          52     0.6948770  0.9508197  0.000
  Breiman    2          53     0.7129098  0.9540984  0.000
  Breiman    2          54     0.7227459  0.9540984  0.000
  Breiman    2          55     0.7252049  0.9606557  0.000
  Breiman    2          56     0.7096311  0.9573770  0.000
  Breiman    2          57     0.7137295  0.9606557  0.000
  Breiman    2          58     0.7129098  0.9606557  0.000
  Breiman    2          59     0.7137295  0.9606557  0.000
  Breiman    2          60     0.7141393  0.9573770  0.000
  Breiman    2          61     0.7141393  0.9639344  0.000
  Breiman    2          62     0.7174180  0.9606557  0.000
  Breiman    2          63     0.7174180  0.9639344  0.000
```

Breiman	2	64	0.7092213	0.9606557	0.000
Breiman	2	65	0.7004098	0.9606557	0.000
Breiman	2	66	0.7258197	0.9606557	0.000
Breiman	2	67	0.7356557	0.9606557	0.000
Breiman	2	68	0.7307377	0.9573770	0.000
Breiman	2	69	0.7295082	0.9573770	0.000
Breiman	2	70	0.7254098	0.9540984	0.000
Breiman	2	71	0.7254098	0.9573770	0.000
Breiman	2	72	0.7245902	0.9573770	0.000
Breiman	2	73	0.7245902	0.9606557	0.000
Breiman	2	74	0.7254098	0.9639344	0.000
Breiman	2	75	0.7299180	0.9639344	0.000
Breiman	2	76	0.7352459	0.9639344	0.000
Breiman	2	77	0.7340164	0.9672131	0.000
Breiman	2	78	0.7413934	0.9704918	0.000
Breiman	2	79	0.7413934	0.9639344	0.000
Breiman	2	80	0.7413934	0.9704918	0.000
Breiman	2	81	0.7413934	0.9704918	0.000
Breiman	2	82	0.7397541	0.9704918	0.000
Breiman	2	83	0.7381148	0.9672131	0.000
Breiman	2	84	0.7430328	0.9704918	0.000
Breiman	2	85	0.7545082	0.9737705	0.000
Breiman	2	86	0.7495902	0.9737705	0.000
Breiman	2	87	0.7532787	0.9737705	0.000
Breiman	2	88	0.7495902	0.9704918	0.000
Breiman	2	89	0.7495902	0.9704918	0.000
Breiman	2	90	0.7463115	0.9737705	0.000
Breiman	2	91	0.7336066	0.9672131	0.000
Breiman	2	92	0.7331967	0.9672131	0.000
Breiman	2	93	0.7331967	0.9737705	0.000
Breiman	2	94	0.7295082	0.9672131	0.000
Breiman	2	95	0.7295082	0.9672131	0.000
Breiman	2	96	0.7360656	0.9672131	0.000
Breiman	2	97	0.7360656	0.9704918	0.000
Breiman	2	98	0.7434426	0.9672131	0.000
Breiman	2	99	0.7372951	0.9672131	0.000
Breiman	2	100	0.7430328	0.9672131	0.000
Breiman	3	20	0.7313525	0.9475410	0.000
Breiman	3	21	0.7245902	0.9540984	0.000
Breiman	3	22	0.6766393	0.9573770	0.000
Breiman	3	23	0.6795082	0.9639344	0.000
Breiman	3	24	0.6741803	0.9573770	0.000
Breiman	3	25	0.6827869	0.9606557	0.000
Breiman	3	26	0.6959016	0.9704918	0.000
Breiman	3	27	0.7069672	0.9606557	0.000
Breiman	3	28	0.6959016	0.9704918	0.000
Breiman	3	29	0.6959016	0.9672131	0.000
Breiman	3	30	0.6836066	0.9704918	0.000

Breiman	3	31	0.6573770	0.9672131	0.000
Breiman	3	32	0.6676230	0.9672131	0.000
Breiman	3	33	0.6879098	0.9672131	0.000
Breiman	3	34	0.7047131	0.9639344	0.000
Breiman	3	35	0.7112705	0.9704918	0.000
Breiman	3	36	0.7116803	0.9672131	0.000
Breiman	3	37	0.7252049	0.9704918	0.000
Breiman	3	38	0.7223361	0.9737705	0.000
Breiman	3	39	0.7258197	0.9770492	0.000
Breiman	3	40	0.7254098	0.9770492	0.000
Breiman	3	41	0.7200820	0.9737705	0.000
Breiman	3	42	0.7135246	0.9737705	0.000
Breiman	3	43	0.7024590	0.9803279	0.000
Breiman	3	44	0.7155738	0.9770492	0.000
Breiman	3	45	0.7127049	0.9770492	0.000
Breiman	3	46	0.7094262	0.9770492	0.000
Breiman	3	47	0.7061475	0.9803279	0.000
Breiman	3	48	0.7172131	0.9803279	0.000
Breiman	3	49	0.7028689	0.9836066	0.000
Breiman	3	50	0.7016393	0.9836066	0.000
Breiman	3	51	0.6963115	0.9836066	0.000
Breiman	3	52	0.6950820	0.9836066	0.000
Breiman	3	53	0.6971311	0.9803279	0.000
Breiman	3	54	0.6959016	0.9836066	0.000
Breiman	3	55	0.6983607	0.9836066	0.000
Breiman	3	56	0.6897541	0.9836066	0.000
Breiman	3	57	0.6868852	0.9803279	0.000
Breiman	3	58	0.6864754	0.9770492	0.000
Breiman	3	59	0.6860656	0.9770492	0.000
Breiman	3	60	0.6680328	0.9803279	0.000
Breiman	3	61	0.6622951	0.9803279	0.000
Breiman	3	62	0.6741803	0.9803279	0.000
Breiman	3	63	0.6799180	0.9836066	0.000
Breiman	3	64	0.6877049	0.9836066	0.000
Breiman	3	65	0.6868852	0.9836066	0.000
Breiman	3	66	0.6909836	0.9836066	0.000
Breiman	3	67	0.6913934	0.9836066	0.000
Breiman	3	68	0.6913934	0.9836066	0.000
Breiman	3	69	0.6831967	0.9836066	0.000
Breiman	3	70	0.6913934	0.9836066	0.000
Breiman	3	71	0.6897541	0.9836066	0.000
Breiman	3	72	0.6893443	0.9836066	0.000
Breiman	3	73	0.6844262	0.9836066	0.000
Breiman	3	74	0.6717213	0.9836066	0.000
Breiman	3	75	0.6737705	0.9836066	0.000
Breiman	3	76	0.6897541	0.9836066	0.000
Breiman	3	77	0.6901639	0.9868852	0.000
Breiman	3	78	0.6959016	0.9901639	0.000

Breiman	3	79	0.6926230	0.9868852	0.000
Breiman	3	80	0.6926230	0.9901639	0.000
Breiman	3	81	0.6881148	0.9901639	0.000
Breiman	3	82	0.6827869	0.9868852	0.000
Breiman	3	83	0.6807377	0.9868852	0.000
Breiman	3	84	0.6807377	0.9868852	0.000
Breiman	3	85	0.6881148	0.9901639	0.000
Breiman	3	86	0.6872951	0.9901639	0.000
Breiman	3	87	0.6852459	0.9901639	0.000
Breiman	3	88	0.6840164	0.9868852	0.000
Breiman	3	89	0.6840164	0.9868852	0.000
Breiman	3	90	0.6889344	0.9868852	0.000
Breiman	3	91	0.6950820	0.9868852	0.000
Breiman	3	92	0.6930328	0.9868852	0.000
Breiman	3	93	0.6885246	0.9868852	0.000
Breiman	3	94	0.6655738	0.9868852	0.000
Breiman	3	95	0.6577869	0.9868852	0.000
Breiman	3	96	0.6586066	0.9868852	0.000
Breiman	3	97	0.6586066	0.9868852	0.000
Breiman	3	98	0.6557377	0.9868852	0.000
Breiman	3	99	0.6651639	0.9868852	0.000
Breiman	3	100	0.6647541	0.9868852	0.000
Breiman	4	20	0.7075820	0.9868852	0.000
Breiman	4	21	0.7057377	0.9901639	0.000
Breiman	4	22	0.6741803	0.9901639	0.000
Breiman	4	23	0.6733607	0.9934426	0.000
Breiman	4	24	0.6696721	0.9901639	0.000
Breiman	4	25	0.6762295	0.9868852	0.000
Breiman	4	26	0.6733607	0.9901639	0.000
Breiman	4	27	0.6504098	0.9934426	0.000
Breiman	4	28	0.6846311	0.9934426	0.000
Breiman	4	29	0.6735656	0.9901639	0.000
Breiman	4	30	0.6649590	0.9967213	0.000
Breiman	4	31	0.6579918	0.9934426	0.000
Breiman	4	32	0.6784836	0.9967213	0.000
Breiman	4	33	0.6817623	0.9967213	0.000
Breiman	4	34	0.6866803	0.9967213	0.000
Breiman	4	35	0.6645492	0.9934426	0.000
Breiman	4	36	0.6686475	0.9967213	0.000
Breiman	4	37	0.6645492	0.9967213	0.000
Breiman	4	38	0.6784836	0.9967213	0.000
Breiman	4	39	0.6735656	0.9934426	0.000
Breiman	4	40	0.6801230	0.9967213	0.000
Breiman	4	41	0.6805328	0.9934426	0.000
Breiman	4	42	0.6686475	0.9901639	0.000
Breiman	4	43	0.6571721	0.9934426	0.000
Breiman	4	44	0.6571721	0.9934426	0.000
Breiman	4	45	0.6518443	0.9967213	0.000

Breiman	4	46	0.6440574	0.9967213	0.000
Breiman	4	47	0.6493852	0.9967213	0.000
Breiman	4	48	0.6543033	0.9967213	0.000
Breiman	4	49	0.6522541	0.9901639	0.000
Breiman	4	50	0.6485656	0.9934426	0.000
Breiman	4	51	0.6502049	0.9934426	0.000
Breiman	4	52	0.6592213	0.9934426	0.000
Breiman	4	53	0.6678279	0.9934426	0.000
Breiman	4	54	0.6838115	0.9934426	0.000
Breiman	4	55	0.6665984	0.9934426	0.000
Breiman	4	56	0.6670082	0.9934426	0.000
Breiman	4	57	0.6415984	0.9934426	0.000
Breiman	4	58	0.6411885	0.9934426	0.000
Breiman	4	59	0.6297131	0.9934426	0.000
Breiman	4	60	0.6309426	0.9934426	0.000
Breiman	4	61	0.6375000	0.9934426	0.000
Breiman	4	62	0.6448770	0.9967213	0.000
Breiman	4	63	0.6555328	0.9901639	0.000
Breiman	4	64	0.6657787	0.9901639	0.000
Breiman	4	65	0.6612705	0.9901639	0.000
Breiman	4	66	0.6665984	0.9901639	0.000
Breiman	4	67	0.6575820	0.9901639	0.000
Breiman	4	68	0.6620902	0.9901639	0.000
Breiman	4	69	0.6620902	0.9901639	0.000
Breiman	4	70	0.6772541	0.9901639	0.000
Breiman	4	71	0.6686475	0.9901639	0.000
Breiman	4	72	0.6682377	0.9901639	0.000
Breiman	4	73	0.6661885	0.9901639	0.000
Breiman	4	74	0.6420082	0.9901639	0.000
Breiman	4	75	0.6432377	0.9901639	0.000
Breiman	4	76	0.6272541	0.9901639	0.000
Breiman	4	77	0.6293033	0.9901639	0.000
Breiman	4	78	0.6411885	0.9901639	0.000
Breiman	4	79	0.6370902	0.9901639	0.000
Breiman	4	80	0.6375000	0.9901639	0.000
Breiman	4	81	0.6559426	0.9901639	0.000
Breiman	4	82	0.6547131	0.9901639	0.000
Breiman	4	83	0.6735656	0.9901639	0.000
Breiman	4	84	0.6706967	0.9901639	0.000
Breiman	4	85	0.6600410	0.9901639	0.000
Breiman	4	86	0.6715164	0.9901639	0.000
Breiman	4	87	0.6690574	0.9901639	0.000
Breiman	4	88	0.6661885	0.9901639	0.000
Breiman	4	89	0.6686475	0.9901639	0.000
Breiman	4	90	0.6686475	0.9901639	0.000
Breiman	4	91	0.6711066	0.9901639	0.000
Breiman	4	92	0.6727459	0.9901639	0.000
Breiman	4	93	0.6797131	0.9901639	0.000

Breiman	4	94	0.6911885	0.9901639	0.000
Breiman	4	95	0.6965164	0.9901639	0.000
Breiman	4	96	0.7014344	0.9901639	0.000
Breiman	4	97	0.6965164	0.9901639	0.000
Breiman	4	98	0.6969262	0.9901639	0.000
Breiman	4	99	0.6961066	0.9901639	0.000
Breiman	4	100	0.6739754	0.9901639	0.000
Freund	2	20	0.7979508	0.9344262	0.125
Freund	2	21	0.7993852	0.9409836	0.125
Freund	2	22	0.7989754	0.9475410	0.000
Freund	2	23	0.7924180	0.9508197	0.000
Freund	2	24	0.7858607	0.9573770	0.000
Freund	2	25	0.7797131	0.9606557	0.000
Freund	2	26	0.7838115	0.9508197	0.000
Freund	2	27	0.7829918	0.9606557	0.000
Freund	2	28	0.7747951	0.9606557	0.000
Freund	2	29	0.7584016	0.9508197	0.000
Freund	2	30	0.7446721	0.9475410	0.000
Freund	2	31	0.7413934	0.9508197	0.000
Freund	2	32	0.7508197	0.9540984	0.000
Freund	2	33	0.7508197	0.9606557	0.000
Freund	2	34	0.7680328	0.9639344	0.000
Freund	2	35	0.7741803	0.9704918	0.000
Freund	2	36	0.7799180	0.9639344	0.000
Freund	2	37	0.7831967	0.9672131	0.000
Freund	2	38	0.7635246	0.9573770	0.000
Freund	2	39	0.7586066	0.9672131	0.000
Freund	2	40	0.7610656	0.9704918	0.000
Freund	2	41	0.7622951	0.9672131	0.000
Freund	2	42	0.7581967	0.9672131	0.000
Freund	2	43	0.7651639	0.9704918	0.125
Freund	2	44	0.7672131	0.9672131	0.125
Freund	2	45	0.7598361	0.9672131	0.125
Freund	2	46	0.7487705	0.9737705	0.125
Freund	2	47	0.7471311	0.9737705	0.125
Freund	2	48	0.7471311	0.9737705	0.125
Freund	2	49	0.7483607	0.9737705	0.125
Freund	2	50	0.7340164	0.9737705	0.125
Freund	2	51	0.7340164	0.9737705	0.000
Freund	2	52	0.7434426	0.9704918	0.000
Freund	2	53	0.7454918	0.9704918	0.000
Freund	2	54	0.7454918	0.9704918	0.000
Freund	2	55	0.7581967	0.9737705	0.125
Freund	2	56	0.7540984	0.9737705	0.000
Freund	2	57	0.7536885	0.9737705	0.000
Freund	2	58	0.7536885	0.9737705	0.000
Freund	2	59	0.7442623	0.9737705	0.000
Freund	2	60	0.7442623	0.9737705	0.000

Freund	2	61	0.7471311	0.9737705	0.000
Freund	2	62	0.7581967	0.9704918	0.000
Freund	2	63	0.7581967	0.9770492	0.000
Freund	2	64	0.7512295	0.9737705	0.000
Freund	2	65	0.7512295	0.9770492	0.000
Freund	2	66	0.7500000	0.9803279	0.000
Freund	2	67	0.7430328	0.9803279	0.000
Freund	2	68	0.7475410	0.9803279	0.000
Freund	2	69	0.7668033	0.9836066	0.000
Freund	2	70	0.7618852	0.9803279	0.000
Freund	2	71	0.7618852	0.9836066	0.000
Freund	2	72	0.7610656	0.9803279	0.000
Freund	2	73	0.7758197	0.9803279	0.000
Freund	2	74	0.7602459	0.9803279	0.000
Freund	2	75	0.7725410	0.9803279	0.000
Freund	2	76	0.7754098	0.9770492	0.000
Freund	2	77	0.7754098	0.9803279	0.000
Freund	2	78	0.7745902	0.9770492	0.000
Freund	2	79	0.7737705	0.9803279	0.000
Freund	2	80	0.7487705	0.9770492	0.000
Freund	2	81	0.7602459	0.9803279	0.000
Freund	2	82	0.7627049	0.9770492	0.000
Freund	2	83	0.7741803	0.9737705	0.000
Freund	2	84	0.7741803	0.9803279	0.000
Freund	2	85	0.7565574	0.9770492	0.000
Freund	2	86	0.7557377	0.9803279	0.000
Freund	2	87	0.7459016	0.9803279	0.000
Freund	2	88	0.7618852	0.9803279	0.000
Freund	2	89	0.7606557	0.9803279	0.000
Freund	2	90	0.7471311	0.9803279	0.000
Freund	2	91	0.7372951	0.9803279	0.125
Freund	2	92	0.7528689	0.9803279	0.125
Freund	2	93	0.7356557	0.9803279	0.125
Freund	2	94	0.7266393	0.9803279	0.000
Freund	2	95	0.7270492	0.9803279	0.000
Freund	2	96	0.7266393	0.9803279	0.000
Freund	2	97	0.7385246	0.9803279	0.000
Freund	2	98	0.7213115	0.9836066	0.125
Freund	2	99	0.7143443	0.9803279	0.125
Freund	2	100	0.7143443	0.9836066	0.125
Freund	3	20	0.6618852	0.9934426	0.125
Freund	3	21	0.6606557	0.9901639	0.000
Freund	3	22	0.7028689	0.9934426	0.250
Freund	3	23	0.7061475	0.9934426	0.000
Freund	3	24	0.7049180	0.9934426	0.000
Freund	3	25	0.6926230	0.9967213	0.000
Freund	3	26	0.7098361	0.9967213	0.000
Freund	3	27	0.7120902	0.9934426	0.000

Freund	3	28	0.7034836	0.9934426	0.125
Freund	3	29	0.7010246	0.9934426	0.000
Freund	3	30	0.6850410	0.9967213	0.125
Freund	3	31	0.6887295	0.9934426	0.125
Freund	3	32	0.6866803	0.9934426	0.000
Freund	3	33	0.6973361	0.9901639	0.125
Freund	3	34	0.6903689	0.9934426	0.000
Freund	3	35	0.6879098	0.9901639	0.125
Freund	3	36	0.6993852	0.9967213	0.125
Freund	3	37	0.7063525	0.9967213	0.125
Freund	3	38	0.7178279	0.9967213	0.000
Freund	3	39	0.7137295	0.9967213	0.125
Freund	3	40	0.7604508	0.9934426	0.000
Freund	3	41	0.7276639	0.9967213	0.000
Freund	3	42	0.6936475	0.9967213	0.000
Freund	3	43	0.6838115	0.9967213	0.000
Freund	3	44	0.6825820	0.9967213	0.000
Freund	3	45	0.6899590	0.9967213	0.000
Freund	3	46	0.6997951	0.9967213	0.000
Freund	3	47	0.6993852	0.9967213	0.000
Freund	3	48	0.6940574	0.9967213	0.000
Freund	3	49	0.6920082	0.9967213	0.000
Freund	3	50	0.6940574	0.9967213	0.000
Freund	3	51	0.6969262	0.9967213	0.000
Freund	3	52	0.7034836	0.9967213	0.000
Freund	3	53	0.7030738	0.9967213	0.000
Freund	3	54	0.7149590	0.9967213	0.000
Freund	3	55	0.7272541	0.9967213	0.000
Freund	3	56	0.7252049	0.9967213	0.000
Freund	3	57	0.7108607	0.9967213	0.000
Freund	3	58	0.7051230	0.9967213	0.000
Freund	3	59	0.6956967	0.9967213	0.000
Freund	3	60	0.6776639	0.9934426	0.000
Freund	3	61	0.6866803	0.9934426	0.000
Freund	3	62	0.6838115	0.9967213	0.000
Freund	3	63	0.6825820	0.9967213	0.000
Freund	3	64	0.6752049	0.9934426	0.000
Freund	3	65	0.6719262	0.9967213	0.000
Freund	3	66	0.6686475	0.9967213	0.000
Freund	3	67	0.6727459	0.9967213	0.000
Freund	3	68	0.6797131	0.9967213	0.000
Freund	3	69	0.6776639	0.9967213	0.000
Freund	3	70	0.6866803	0.9967213	0.000
Freund	3	71	0.6752049	0.9967213	0.000
Freund	3	72	0.6690574	0.9967213	0.000
Freund	3	73	0.6538934	0.9967213	0.000
Freund	3	74	0.6428279	0.9967213	0.000
Freund	3	75	0.6448770	0.9967213	0.000

Freund	3	76	0.6428279	0.9967213	0.000
Freund	3	77	0.6465164	0.9967213	0.000
Freund	3	78	0.6465164	0.9967213	0.000
Freund	3	79	0.6440574	0.9967213	0.000
Freund	3	80	0.6395492	0.9967213	0.000
Freund	3	81	0.6366803	0.9934426	0.000
Freund	3	82	0.6043033	0.9934426	0.000
Freund	3	83	0.6071721	0.9934426	0.000
Freund	3	84	0.6256148	0.9934426	0.000
Freund	3	85	0.6354508	0.9934426	0.000
Freund	3	86	0.6301230	0.9934426	0.000
Freund	3	87	0.6366803	0.9934426	0.000
Freund	3	88	0.6391393	0.9934426	0.000
Freund	3	89	0.6387295	0.9934426	0.000
Freund	3	90	0.6428279	0.9934426	0.000
Freund	3	91	0.6415984	0.9934426	0.000
Freund	3	92	0.6432377	0.9934426	0.000
Freund	3	93	0.6432377	0.9934426	0.000
Freund	3	94	0.6489754	0.9934426	0.000
Freund	3	95	0.6694672	0.9934426	0.000
Freund	3	96	0.6600410	0.9934426	0.000
Freund	3	97	0.6760246	0.9934426	0.000
Freund	3	98	0.6747951	0.9967213	0.000
Freund	3	99	0.6747951	0.9967213	0.000
Freund	3	100	0.6809426	0.9934426	0.000
Freund	4	20	0.5286885	1.0000000	0.000
Freund	4	21	0.5241803	0.9967213	0.000
Freund	4	22	0.5497951	0.9967213	0.000
Freund	4	23	0.5756148	0.9967213	0.125
Freund	4	24	0.5698770	1.0000000	0.000
Freund	4	25	0.5868852	1.0000000	0.125
Freund	4	26	0.6198770	1.0000000	0.125
Freund	4	27	0.6268443	1.0000000	0.125
Freund	4	28	0.6391393	1.0000000	0.125
Freund	4	29	0.6502049	1.0000000	0.125
Freund	4	30	0.6530738	0.9967213	0.000
Freund	4	31	0.6469262	1.0000000	0.125
Freund	4	32	0.6106557	0.9967213	0.000
Freund	4	33	0.6106557	1.0000000	0.125
Freund	4	34	0.6397541	0.9967213	0.000
Freund	4	35	0.6471311	1.0000000	0.000
Freund	4	36	0.6586066	1.0000000	0.000
Freund	4	37	0.6155738	0.9967213	0.000
Freund	4	38	0.6081967	0.9967213	0.000
Freund	4	39	0.6098361	0.9967213	0.000
Freund	4	40	0.6028689	0.9967213	0.000
Freund	4	41	0.6135246	0.9967213	0.000
Freund	4	42	0.6110656	0.9967213	0.000

Freund	4	43	0.6221311	0.9967213	0.000
Freund	4	44	0.6286885	0.9934426	0.000
Freund	4	45	0.6102459	0.9934426	0.000
Freund	4	46	0.6237705	0.9934426	0.000
Freund	4	47	0.6549180	0.9934426	0.000
Freund	4	48	0.6545082	0.9934426	0.000
Freund	4	49	0.6741803	0.9901639	0.000
Freund	4	50	0.6635246	0.9901639	0.000
Freund	4	51	0.6393443	0.9934426	0.000
Freund	4	52	0.6467213	0.9901639	0.000
Freund	4	53	0.6290984	0.9934426	0.000
Freund	4	54	0.6274590	0.9934426	0.000
Freund	4	55	0.6184426	0.9934426	0.000
Freund	4	56	0.6319672	0.9967213	0.000
Freund	4	57	0.6393443	0.9934426	0.000
Freund	4	58	0.6418033	1.0000000	0.000
Freund	4	59	0.6397541	0.9934426	0.000
Freund	4	60	0.6877049	1.0000000	0.000
Freund	4	61	0.7073770	0.9967213	0.000
Freund	4	62	0.7114754	0.9967213	0.125
Freund	4	63	0.7094262	0.9967213	0.000
Freund	4	64	0.7053279	0.9967213	0.000
Freund	4	65	0.6704918	0.9934426	0.000
Freund	4	66	0.6663934	0.9934426	0.000
Freund	4	67	0.6573770	0.9967213	0.000
Freund	4	68	0.6983607	0.9967213	0.000
Freund	4	69	0.6803279	0.9967213	0.000
Freund	4	70	0.7000000	0.9934426	0.125
Freund	4	71	0.6979508	0.9967213	0.125
Freund	4	72	0.7024590	0.9934426	0.125
Freund	4	73	0.7184426	1.0000000	0.125
Freund	4	74	0.7110656	0.9934426	0.125
Freund	4	75	0.6963115	1.0000000	0.000
Freund	4	76	0.7036885	1.0000000	0.125
Freund	4	77	0.7315574	1.0000000	0.000
Freund	4	78	0.7209016	1.0000000	0.000
Freund	4	79	0.7065574	1.0000000	0.000
Freund	4	80	0.7213115	0.9967213	0.000
Freund	4	81	0.7204918	1.0000000	0.000
Freund	4	82	0.7168033	1.0000000	0.000
Freund	4	83	0.7237705	1.0000000	0.000
Freund	4	84	0.7209016	0.9967213	0.000
Freund	4	85	0.7139344	1.0000000	0.000
Freund	4	86	0.7192623	1.0000000	0.000
Freund	4	87	0.7217213	1.0000000	0.000
Freund	4	88	0.7237705	1.0000000	0.000
Freund	4	89	0.7147541	0.9967213	0.000
Freund	4	90	0.7196721	0.9967213	0.000

Freund	4	91	0.7196721	0.9967213	0.000
Freund	4	92	0.7168033	1.0000000	0.000
Freund	4	93	0.7184426	1.0000000	0.000
Freund	4	94	0.7192623	0.9967213	0.000
Freund	4	95	0.7135246	0.9967213	0.000
Freund	4	96	0.7270492	0.9967213	0.000
Freund	4	97	0.7262295	0.9967213	0.000
Freund	4	98	0.7250000	0.9967213	0.000
Freund	4	99	0.7262295	1.0000000	0.000
Freund	4	100	0.7245902	0.9967213	0.000
Zhu	2	20	0.6668033	0.9508197	0.000
Zhu	2	21	0.6586066	0.9573770	0.000
Zhu	2	22	0.6602459	0.9606557	0.000
Zhu	2	23	0.6565574	0.9639344	0.000
Zhu	2	24	0.6323770	0.9639344	0.000
Zhu	2	25	0.6672131	0.9540984	0.000
Zhu	2	26	0.6877049	0.9639344	0.000
Zhu	2	27	0.7397541	0.9573770	0.000
Zhu	2	28	0.7172131	0.9704918	0.000
Zhu	2	29	0.6971311	0.9704918	0.000
Zhu	2	30	0.7254098	0.9737705	0.000
Zhu	2	31	0.7254098	0.9704918	0.000
Zhu	2	32	0.6959016	0.9573770	0.000
Zhu	2	33	0.6909836	0.9606557	0.000
Zhu	2	34	0.6827869	0.9639344	0.000
Zhu	2	35	0.6536885	0.9606557	0.000
Zhu	2	36	0.6500000	0.9639344	0.000
Zhu	2	37	0.6520492	0.9639344	0.000
Zhu	2	38	0.6356557	0.9639344	0.000
Zhu	2	39	0.6413934	0.9606557	0.000
Zhu	2	40	0.6413934	0.9672131	0.000
Zhu	2	41	0.6270492	0.9606557	0.000
Zhu	2	42	0.6163934	0.9606557	0.000
Zhu	2	43	0.6348361	0.9704918	0.000
Zhu	2	44	0.6258197	0.9639344	0.000
Zhu	2	45	0.6213115	0.9639344	0.000
Zhu	2	46	0.6348361	0.9704918	0.000
Zhu	2	47	0.6348361	0.9704918	0.000
Zhu	2	48	0.6155738	0.9672131	0.000
Zhu	2	49	0.6504098	0.9737705	0.000
Zhu	2	50	0.6413934	0.9606557	0.000
Zhu	2	51	0.6413934	0.9737705	0.000
Zhu	2	52	0.6426230	0.9672131	0.000
Zhu	2	53	0.6581967	0.9737705	0.000
Zhu	2	54	0.6389344	0.9770492	0.000
Zhu	2	55	0.6495902	0.9672131	0.000
Zhu	2	56	0.6565574	0.9770492	0.000
Zhu	2	57	0.6676230	0.9770492	0.000

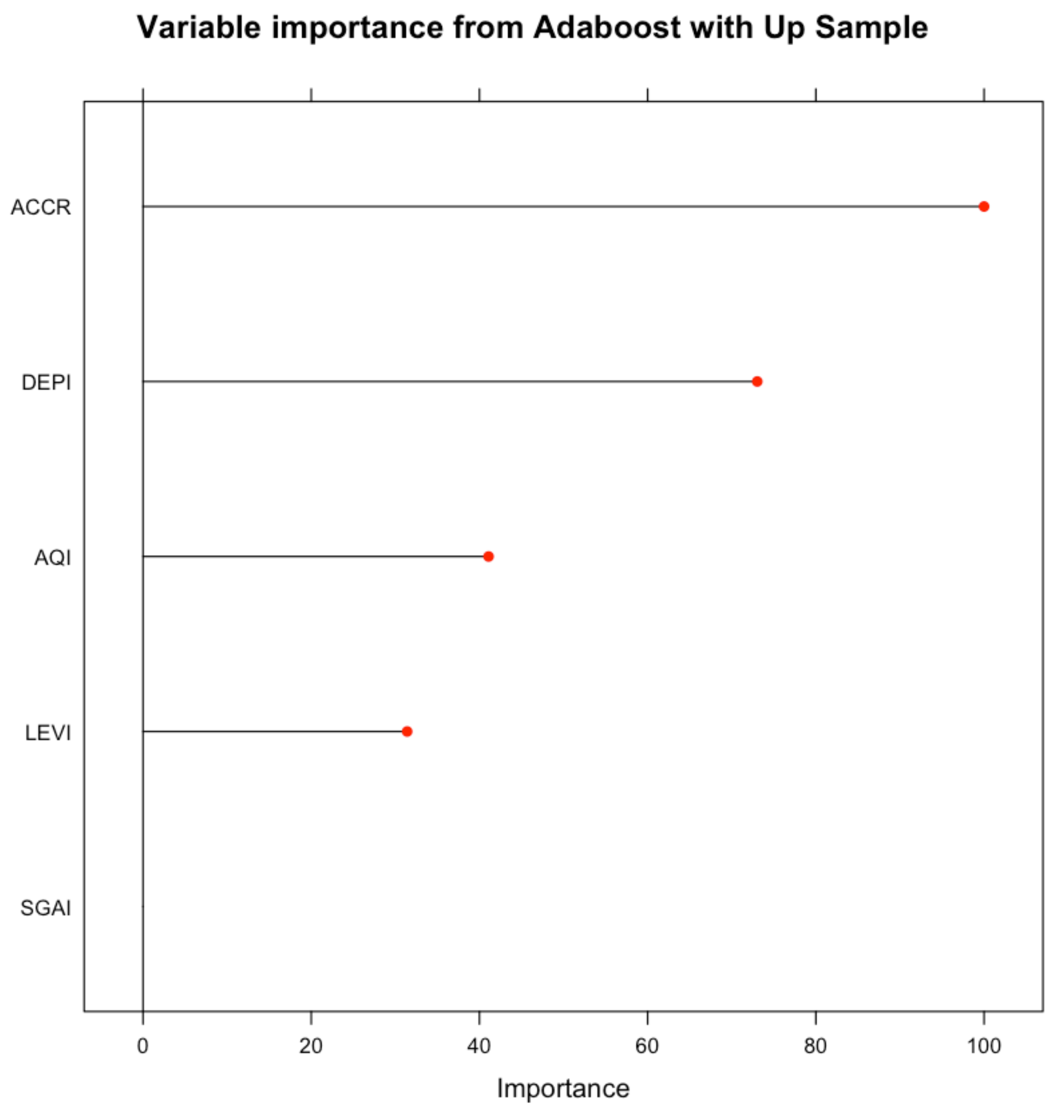
Zhu	2	58	0.6655738	0.9803279	0.000
Zhu	2	59	0.6471311	0.9803279	0.000
Zhu	2	60	0.6463115	0.9803279	0.000
Zhu	2	61	0.6446721	0.9737705	0.000
Zhu	2	62	0.6442623	0.9737705	0.000
Zhu	2	63	0.6454918	0.9737705	0.000
Zhu	2	64	0.6438525	0.9737705	0.000
Zhu	2	65	0.6405738	0.9770492	0.000
Zhu	2	66	0.6405738	0.9770492	0.000
Zhu	2	67	0.6405738	0.9737705	0.000
Zhu	2	68	0.6405738	0.9770492	0.000
Zhu	2	69	0.6372951	0.9737705	0.000
Zhu	2	70	0.6413934	0.9770492	0.000
Zhu	2	71	0.6471311	0.9770492	0.000
Zhu	2	72	0.6471311	0.9770492	0.000
Zhu	2	73	0.6590164	0.9770492	0.000
Zhu	2	74	0.6622951	0.9770492	0.000
Zhu	2	75	0.6622951	0.9770492	0.000
Zhu	2	76	0.6659836	0.9737705	0.000
Zhu	2	77	0.6659836	0.9803279	0.000
Zhu	2	78	0.6659836	0.9737705	0.000
Zhu	2	79	0.6614754	0.9803279	0.000
Zhu	2	80	0.6709016	0.9770492	0.000
Zhu	2	81	0.6704918	0.9803279	0.000
Zhu	2	82	0.6762295	0.9770492	0.000
Zhu	2	83	0.6754098	0.9770492	0.000
Zhu	2	84	0.6750000	0.9803279	0.000
Zhu	2	85	0.6729508	0.9836066	0.000
Zhu	2	86	0.6819672	0.9803279	0.000
Zhu	2	87	0.6733607	0.9836066	0.000
Zhu	2	88	0.6918033	0.9803279	0.000
Zhu	2	89	0.6967213	0.9836066	0.000
Zhu	2	90	0.6971311	0.9836066	0.000
Zhu	2	91	0.7102459	0.9836066	0.000
Zhu	2	92	0.7086066	0.9836066	0.000
Zhu	2	93	0.6987705	0.9836066	0.000
Zhu	2	94	0.6950820	0.9836066	0.000
Zhu	2	95	0.6782787	0.9836066	0.000
Zhu	2	96	0.6782787	0.9836066	0.000
Zhu	2	97	0.6651639	0.9836066	0.000
Zhu	2	98	0.6672131	0.9836066	0.000
Zhu	2	99	0.6647541	0.9836066	0.000
Zhu	2	100	0.6647541	0.9836066	0.000
Zhu	3	20	0.6370902	0.9868852	0.000
Zhu	3	21	0.6290984	0.9868852	0.000
Zhu	3	22	0.6684426	0.9836066	0.000
Zhu	3	23	0.6565574	0.9836066	0.000
Zhu	3	24	0.6815574	0.9868852	0.000

Zhu	3	25	0.6680328	0.9901639	0.000
Zhu	3	26	0.6655738	0.9934426	0.000
Zhu	3	27	0.6254098	0.9868852	0.000
Zhu	3	28	0.6245902	0.9868852	0.000
Zhu	3	29	0.5909836	0.9836066	0.000
Zhu	3	30	0.6204918	0.9868852	0.000
Zhu	3	31	0.6032787	0.9868852	0.000
Zhu	3	32	0.6258197	0.9868852	0.000
Zhu	3	33	0.6598361	0.9803279	0.000
Zhu	3	34	0.6573770	0.9901639	0.000
Zhu	3	35	0.6561475	0.9934426	0.000
Zhu	3	36	0.6639344	0.9901639	0.000
Zhu	3	37	0.7053279	0.9868852	0.000
Zhu	3	38	0.7159836	0.9901639	0.000
Zhu	3	39	0.7139344	0.9868852	0.000
Zhu	3	40	0.7122951	0.9868852	0.000
Zhu	3	41	0.7127049	0.9868852	0.000
Zhu	3	42	0.7098361	0.9901639	0.000
Zhu	3	43	0.7061475	0.9901639	0.000
Zhu	3	44	0.7061475	0.9901639	0.000
Zhu	3	45	0.6905738	0.9901639	0.000
Zhu	3	46	0.7077869	0.9901639	0.000
Zhu	3	47	0.6987705	0.9901639	0.000
Zhu	3	48	0.6946721	0.9901639	0.000
Zhu	3	49	0.6487705	0.9934426	0.000
Zhu	3	50	0.6487705	0.9934426	0.000
Zhu	3	51	0.6672131	0.9934426	0.000
Zhu	3	52	0.6622951	0.9868852	0.000
Zhu	3	53	0.6594262	0.9868852	0.000
Zhu	3	54	0.6905738	0.9868852	0.000
Zhu	3	55	0.6954918	0.9868852	0.125
Zhu	3	56	0.6827869	0.9868852	0.000
Zhu	3	57	0.6655738	0.9868852	0.125
Zhu	3	58	0.6745902	0.9901639	0.000
Zhu	3	59	0.6647541	0.9868852	0.000
Zhu	3	60	0.6639344	0.9901639	0.000
Zhu	3	61	0.6614754	0.9868852	0.000
Zhu	3	62	0.6610656	0.9901639	0.000
Zhu	3	63	0.6790984	0.9901639	0.000
Zhu	3	64	0.6750000	0.9901639	0.000
Zhu	3	65	0.6930328	0.9934426	0.000
Zhu	3	66	0.6618852	0.9868852	0.000
Zhu	3	67	0.6627049	0.9901639	0.000
Zhu	3	68	0.6536885	0.9901639	0.000
Zhu	3	69	0.6577869	0.9901639	0.000
Zhu	3	70	0.6590164	0.9868852	0.000
Zhu	3	71	0.6553279	0.9901639	0.000
Zhu	3	72	0.6409836	0.9868852	0.000

Zhu	3	73	0.6590164	0.9868852	0.000
Zhu	3	74	0.6610656	0.9868852	0.000
Zhu	3	75	0.6586066	0.9868852	0.000
Zhu	3	76	0.6372951	0.9868852	0.000
Zhu	3	77	0.6463115	0.9901639	0.000
Zhu	3	78	0.6463115	0.9901639	0.000
Zhu	3	79	0.6565574	0.9868852	0.000
Zhu	3	80	0.6606557	0.9901639	0.000
Zhu	3	81	0.6594262	0.9934426	0.000
Zhu	3	82	0.6340164	0.9934426	0.000
Zhu	3	83	0.6336066	0.9934426	0.000
Zhu	3	84	0.6467213	0.9934426	0.000
Zhu	3	85	0.6434426	0.9967213	0.000
Zhu	3	86	0.6545082	0.9934426	0.000
Zhu	3	87	0.6434426	0.9901639	0.000
Zhu	3	88	0.6434426	0.9934426	0.000
Zhu	3	89	0.6545082	0.9934426	0.000
Zhu	3	90	0.6471311	0.9934426	0.000
Zhu	3	91	0.6467213	0.9934426	0.000
Zhu	3	92	0.6483607	0.9934426	0.000
Zhu	3	93	0.6540984	0.9934426	0.000
Zhu	3	94	0.6397541	0.9934426	0.000
Zhu	3	95	0.6680328	0.9934426	0.000
Zhu	3	96	0.6836066	0.9934426	0.000
Zhu	3	97	0.6823770	0.9934426	0.000
Zhu	3	98	0.6823770	0.9967213	0.000
Zhu	3	99	0.6631148	0.9934426	0.000
Zhu	3	100	0.6631148	0.9967213	0.000
Zhu	4	20	0.7850410	0.9868852	0.125
Zhu	4	21	0.7864754	0.9868852	0.125
Zhu	4	22	0.7659836	0.9868852	0.125
Zhu	4	23	0.7618852	0.9868852	0.000
Zhu	4	24	0.7286885	0.9868852	0.000
Zhu	4	25	0.7155738	0.9901639	0.000
Zhu	4	26	0.6922131	0.9901639	0.000
Zhu	4	27	0.6823770	0.9901639	0.000
Zhu	4	28	0.6827869	0.9901639	0.000
Zhu	4	29	0.6717213	0.9901639	0.000
Zhu	4	30	0.6745902	0.9901639	0.000
Zhu	4	31	0.6971311	0.9901639	0.000
Zhu	4	32	0.6967213	0.9901639	0.000
Zhu	4	33	0.6610656	0.9934426	0.000
Zhu	4	34	0.6897541	0.9901639	0.000
Zhu	4	35	0.6856557	0.9901639	0.000
Zhu	4	36	0.6922131	0.9901639	0.000
Zhu	4	37	0.6938525	0.9934426	0.000
Zhu	4	38	0.6909836	0.9901639	0.000
Zhu	4	39	0.6905738	0.9934426	0.000

Zhu	4	40	0.7028689	0.9934426	0.000
Zhu	4	41	0.6967213	0.9967213	0.000
Zhu	4	42	0.6967213	0.9934426	0.000
Zhu	4	43	0.7139344	0.9967213	0.000
Zhu	4	44	0.7282787	0.9967213	0.000
Zhu	4	45	0.7258197	0.9967213	0.000
Zhu	4	46	0.7262295	0.9967213	0.000
Zhu	4	47	0.7221311	0.9967213	0.000
Zhu	4	48	0.7180328	0.9934426	0.000
Zhu	4	49	0.7049180	0.9934426	0.000
Zhu	4	50	0.7040984	0.9934426	0.000
Zhu	4	51	0.7446721	0.9934426	0.000
Zhu	4	52	0.7491803	0.9934426	0.000
Zhu	4	53	0.7340164	0.9934426	0.000
Zhu	4	54	0.7323770	0.9934426	0.000
Zhu	4	55	0.7286885	0.9934426	0.000
Zhu	4	56	0.7331967	0.9934426	0.000
Zhu	4	57	0.7450820	0.9934426	0.000
Zhu	4	58	0.7241803	0.9934426	0.000
Zhu	4	59	0.7258197	0.9934426	0.000
Zhu	4	60	0.7405738	0.9934426	0.000
Zhu	4	61	0.7315574	0.9934426	0.000
Zhu	4	62	0.7278689	0.9934426	0.000
Zhu	4	63	0.7217213	0.9934426	0.000
Zhu	4	64	0.7196721	0.9934426	0.000
Zhu	4	65	0.7250000	0.9934426	0.000
Zhu	4	66	0.7127049	0.9934426	0.000
Zhu	4	67	0.7098361	0.9934426	0.000
Zhu	4	68	0.7143443	0.9934426	0.000
Zhu	4	69	0.7336066	0.9934426	0.000
Zhu	4	70	0.7237705	0.9934426	0.000
Zhu	4	71	0.7233607	0.9967213	0.000
Zhu	4	72	0.7241803	0.9934426	0.000
Zhu	4	73	0.7204918	0.9967213	0.000
Zhu	4	74	0.7135246	0.9967213	0.000
Zhu	4	75	0.7094262	0.9967213	0.000
Zhu	4	76	0.7094262	0.9967213	0.000
Zhu	4	77	0.7008197	0.9967213	0.000
Zhu	4	78	0.6983607	0.9967213	0.000
Zhu	4	79	0.6901639	0.9967213	0.000
Zhu	4	80	0.7057377	0.9967213	0.000
Zhu	4	81	0.7061475	0.9967213	0.000
Zhu	4	82	0.6987705	0.9967213	0.000
Zhu	4	83	0.6926230	0.9967213	0.000
Zhu	4	84	0.6983607	0.9967213	0.000
Zhu	4	85	0.6934426	0.9967213	0.000
Zhu	4	86	0.7049180	0.9967213	0.000
Zhu	4	87	0.6950820	0.9967213	0.000

```
  Zhu        4          88     0.7053279  0.9967213  0.000
  Zhu        4          89     0.7024590  0.9967213  0.000
  Zhu        4          90     0.6946721  0.9967213  0.000
  Zhu        4          91     0.6946721  0.9967213  0.000
  Zhu        4          92     0.6926230  0.9967213  0.000
  Zhu        4          93     0.6905738  0.9967213  0.000
  Zhu        4          94     0.6852459  0.9967213  0.000
  Zhu        4          95     0.6844262  0.9967213  0.000
  Zhu        4          96     0.6913934  0.9967213  0.000
  Zhu        4          97     0.6901639  0.9967213  0.000
  Zhu        4          98     0.6856557  0.9967213  0.000
  Zhu        4          99     0.6938525  0.9967213  0.000
  Zhu        4         100     0.6905738  0.9967213  0.000

ROC was used to select the optimal model using the largest value.
The final values used for the model were mfinal = 21, maxdepth = 2
 and coeflearn = Freund.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  91.7  2.2
       Yes  5.8  0.3

 Accuracy (average) : 0.9201
```



Confusion Matrix for adaboost on test set

```
In [49]: caretPredictedClass <- predict(ada_up_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  343   7
       Yes  17   4

               Accuracy : 0.9353
                 95% CI : (0.9053, 0.9581)
```

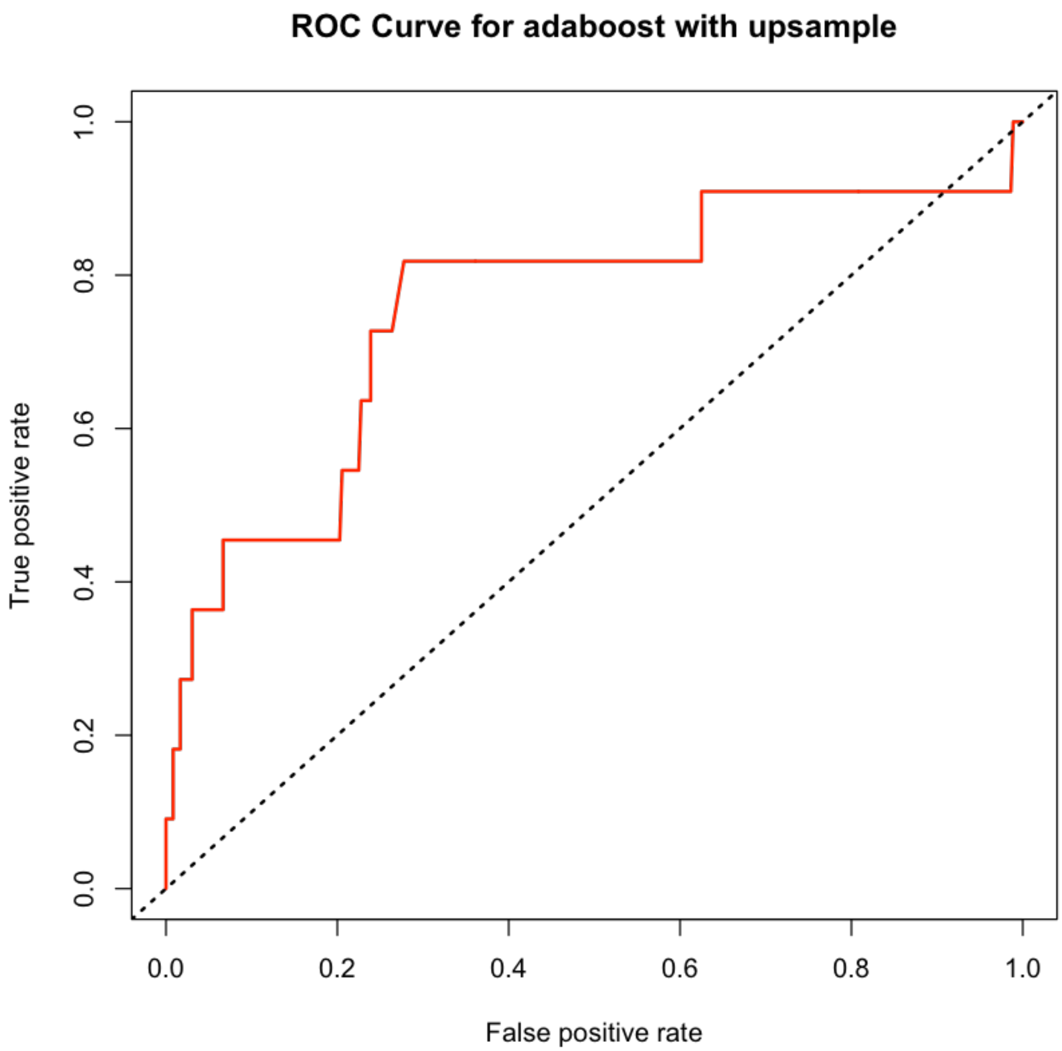
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.99985

                  Kappa : 0.2196
 Mcnemar's Test P-Value : 0.06619

            Sensitivity : 0.9528
            Specificity : 0.3636
         Pos Pred Value : 0.9800
         Neg Pred Value : 0.1905
             Prevalence : 0.9704
         Detection Rate : 0.9245
   Detection Prevalence : 0.9434
      Balanced Accuracy : 0.6582

       'Positive' Class : No
```

ROC plot for adaboost on test set

```
In [50]: ada_pred <- predict(ada_up_model, model_test_df, type = "prob")[,2]
         ada_prediction <- prediction(ada_pred,model_test_df$Manipulater)
         ada_perf <- performance(ada_prediction, "tpr","fpr")

         plot(ada_perf,main="ROC Curve for adaboost with upsample",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(ada_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.7568182
```

```
Slot "alpha.values":
list()
```



1.5.4 Boosting with adaboost (down sample)

The below code chunk sets some of the control parameters for adaboost

```
In [51]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='all',
                                    summaryFunction = twoClassSummary,
```

```
                                    savePredictions = TRUE,
                                    classProbs = TRUE,
                                    sampling = "down")#, p = 0.70) #in case method = #"LGO"
```

```
In [52]: search_grid <- expand.grid(mfinal = c(20:100), maxdepth = c(2:4),
                                    coeflearn = c("Breiman", "Freund", "Zhu"))
```

After setting the control paramters, the model is run

```
In [53]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Adaptive Boosting with Down Sample")

         set.seed(4121)
         ada_down.model <- train(model_train_df[,1:5], model_train_df[,6],
                                 method='AdaBoost.M1',
                                 trControl=objControl,
                                 tuneGrid = search_grid,
                                 metric = "ROC")
         stopCluster(num_cores)
         toc()
```

```
Adaptive Boosting with Down Sample: 137.311 sec elapsed
```

Confusion Matrix for adaboost on train set

```
In [54]: #ada_down.model$finalModel #ada_down.model$results
         print(ada_down.model)
         confusionMatrix.train(ada_down.model)
         plot(varImp(ada_down.model), main = "Variable importance from Adaboost with down sampl
```

```
AdaBoost.M1

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using down-sampling

Resampling results across tuning parameters:

  coeflearn  maxdepth  mfinal  ROC        Sens       Spec
  Breiman    2          20     0.6885246  0.7770492  0.375
  Breiman    2          21     0.6721311  0.7737705  0.375
  Breiman    2          22     0.6817623  0.7836066  0.375
```

Breiman	2	23	0.6764344	0.7836066	0.375
Breiman	2	24	0.6362705	0.7540984	0.375
Breiman	2	25	0.6350410	0.7836066	0.250
Breiman	2	26	0.6399590	0.7803279	0.250
Breiman	2	27	0.6301230	0.7803279	0.250
Breiman	2	28	0.6354508	0.7868852	0.250
Breiman	2	29	0.6481557	0.8032787	0.250
Breiman	2	30	0.6625000	0.7967213	0.250
Breiman	2	31	0.6760246	0.8000000	0.250
Breiman	2	32	0.6782787	0.8295082	0.250
Breiman	2	33	0.6704918	0.7868852	0.250
Breiman	2	34	0.6536885	0.8000000	0.250
Breiman	2	35	0.6602459	0.8131148	0.250
Breiman	2	36	0.6635246	0.8163934	0.250
Breiman	2	37	0.6827869	0.7901639	0.250
Breiman	2	38	0.6848361	0.7934426	0.250
Breiman	2	39	0.6549180	0.7737705	0.250
Breiman	2	40	0.6807377	0.7967213	0.250
Breiman	2	41	0.6823770	0.7803279	0.250
Breiman	2	42	0.6819672	0.7737705	0.250
Breiman	2	43	0.6918033	0.7868852	0.250
Breiman	2	44	0.6860656	0.7967213	0.250
Breiman	2	45	0.6795082	0.8000000	0.250
Breiman	2	46	0.6795082	0.7901639	0.250
Breiman	2	47	0.6983607	0.7770492	0.250
Breiman	2	48	0.7053279	0.7704918	0.250
Breiman	2	49	0.7098361	0.7770492	0.250
Breiman	2	50	0.7106557	0.7836066	0.250
Breiman	2	51	0.6889344	0.7836066	0.250
Breiman	2	52	0.6868852	0.7803279	0.250
Breiman	2	53	0.6766393	0.7868852	0.250
Breiman	2	54	0.6803279	0.8000000	0.250
Breiman	2	55	0.6795082	0.7836066	0.250
Breiman	2	56	0.6795082	0.7803279	0.250
Breiman	2	57	0.6938525	0.7868852	0.250
Breiman	2	58	0.6840164	0.7901639	0.250
Breiman	2	59	0.7000000	0.7704918	0.250
Breiman	2	60	0.6897541	0.7672131	0.250
Breiman	2	61	0.6840164	0.7704918	0.250
Breiman	2	62	0.6918033	0.7672131	0.250
Breiman	2	63	0.6950820	0.7737705	0.250
Breiman	2	64	0.6938525	0.7868852	0.250
Breiman	2	65	0.7131148	0.7836066	0.250
Breiman	2	66	0.7073770	0.7836066	0.250
Breiman	2	67	0.7086066	0.7672131	0.250
Breiman	2	68	0.7024590	0.7672131	0.250
Breiman	2	69	0.7012295	0.7639344	0.250
Breiman	2	70	0.6971311	0.7737705	0.250

Breiman	2	71	0.6913934	0.7704918	0.250
Breiman	2	72	0.6930328	0.7868852	0.250
Breiman	2	73	0.6975410	0.7672131	0.250
Breiman	2	74	0.7061475	0.7934426	0.375
Breiman	2	75	0.7053279	0.7901639	0.375
Breiman	2	76	0.6983607	0.8000000	0.375
Breiman	2	77	0.7086066	0.8229508	0.375
Breiman	2	78	0.7094262	0.8295082	0.375
Breiman	2	79	0.7110656	0.8196721	0.375
Breiman	2	80	0.7020492	0.7934426	0.375
Breiman	2	81	0.7090164	0.7901639	0.375
Breiman	2	82	0.6959016	0.7868852	0.375
Breiman	2	83	0.7049180	0.7737705	0.375
Breiman	2	84	0.7090164	0.7770492	0.375
Breiman	2	85	0.7086066	0.7803279	0.375
Breiman	2	86	0.7295082	0.7934426	0.375
Breiman	2	87	0.7196721	0.7934426	0.375
Breiman	2	88	0.7049180	0.7967213	0.375
Breiman	2	89	0.7065574	0.7967213	0.375
Breiman	2	90	0.7131148	0.7934426	0.375
Breiman	2	91	0.7098361	0.7901639	0.375
Breiman	2	92	0.7012295	0.7868852	0.375
Breiman	2	93	0.7040984	0.7836066	0.375
Breiman	2	94	0.7036885	0.7868852	0.375
Breiman	2	95	0.6995902	0.7836066	0.250
Breiman	2	96	0.6979508	0.7803279	0.250
Breiman	2	97	0.6959016	0.7836066	0.250
Breiman	2	98	0.6954918	0.7868852	0.250
Breiman	2	99	0.7024590	0.7868852	0.250
Breiman	2	100	0.6987705	0.7901639	0.375
Breiman	3	20	0.6106557	0.7770492	0.125
Breiman	3	21	0.6061475	0.7803279	0.125
Breiman	3	22	0.6000000	0.7770492	0.250
Breiman	3	23	0.6491803	0.7770492	0.250
Breiman	3	24	0.6446721	0.7901639	0.250
Breiman	3	25	0.6299180	0.7868852	0.250
Breiman	3	26	0.6475410	0.7934426	0.250
Breiman	3	27	0.6553279	0.7770492	0.250
Breiman	3	28	0.6635246	0.7803279	0.375
Breiman	3	29	0.6545082	0.7967213	0.375
Breiman	3	30	0.6540984	0.7934426	0.375
Breiman	3	31	0.6745902	0.8032787	0.375
Breiman	3	32	0.6811475	0.7836066	0.375
Breiman	3	33	0.6627049	0.7803279	0.375
Breiman	3	34	0.6397541	0.7770492	0.375
Breiman	3	35	0.6770492	0.7836066	0.375
Breiman	3	36	0.6823770	0.7770492	0.375
Breiman	3	37	0.6704918	0.7737705	0.250

Breiman	3	38	0.6602459	0.7737705	0.250
Breiman	3	39	0.6594262	0.7803279	0.250
Breiman	3	40	0.6590164	0.7803279	0.250
Breiman	3	41	0.6610656	0.7967213	0.250
Breiman	3	42	0.6479508	0.7934426	0.250
Breiman	3	43	0.6672131	0.7836066	0.250
Breiman	3	44	0.6668033	0.7901639	0.250
Breiman	3	45	0.6737705	0.7901639	0.250
Breiman	3	46	0.6938525	0.8000000	0.250
Breiman	3	47	0.6995902	0.7967213	0.375
Breiman	3	48	0.6860656	0.8032787	0.250
Breiman	3	49	0.6823770	0.7901639	0.375
Breiman	3	50	0.6692623	0.7901639	0.250
Breiman	3	51	0.6688525	0.7901639	0.250
Breiman	3	52	0.6672131	0.7868852	0.250
Breiman	3	53	0.6668033	0.7934426	0.250
Breiman	3	54	0.6766393	0.7934426	0.250
Breiman	3	55	0.6823770	0.8000000	0.250
Breiman	3	56	0.6770492	0.8032787	0.250
Breiman	3	57	0.6786885	0.7901639	0.250
Breiman	3	58	0.6782787	0.8000000	0.125
Breiman	3	59	0.6754098	0.7868852	0.125
Breiman	3	60	0.6659836	0.7836066	0.375
Breiman	3	61	0.6590164	0.7803279	0.375
Breiman	3	62	0.6479508	0.7639344	0.250
Breiman	3	63	0.6565574	0.7803279	0.250
Breiman	3	64	0.6565574	0.7803279	0.250
Breiman	3	65	0.6512295	0.7770492	0.250
Breiman	3	66	0.6549180	0.7639344	0.375
Breiman	3	67	0.6418033	0.7606557	0.250
Breiman	3	68	0.6475410	0.7639344	0.250
Breiman	3	69	0.6348361	0.7803279	0.250
Breiman	3	70	0.6254098	0.7639344	0.250
Breiman	3	71	0.6262295	0.7770492	0.250
Breiman	3	72	0.6344262	0.7770492	0.250
Breiman	3	73	0.6418033	0.7737705	0.250
Breiman	3	74	0.6504098	0.7770492	0.250
Breiman	3	75	0.6495902	0.7901639	0.250
Breiman	3	76	0.6463115	0.7737705	0.250
Breiman	3	77	0.6360656	0.7672131	0.250
Breiman	3	78	0.6377049	0.7639344	0.250
Breiman	3	79	0.6344262	0.7737705	0.250
Breiman	3	80	0.6327869	0.7704918	0.250
Breiman	3	81	0.6315574	0.7737705	0.250
Breiman	3	82	0.6340164	0.7737705	0.250
Breiman	3	83	0.6307377	0.7836066	0.250
Breiman	3	84	0.6315574	0.7836066	0.250
Breiman	3	85	0.6323770	0.7868852	0.250

Breiman	3	86	0.6319672	0.7868852	0.250
Breiman	3	87	0.6200820	0.7901639	0.250
Breiman	3	88	0.6254098	0.7901639	0.250
Breiman	3	89	0.6233607	0.7836066	0.250
Breiman	3	90	0.6196721	0.7737705	0.250
Breiman	3	91	0.6250000	0.7737705	0.250
Breiman	3	92	0.6168033	0.7704918	0.250
Breiman	3	93	0.6135246	0.7836066	0.250
Breiman	3	94	0.6086066	0.7737705	0.250
Breiman	3	95	0.6040984	0.7737705	0.250
Breiman	3	96	0.6172131	0.7704918	0.250
Breiman	3	97	0.6127049	0.7704918	0.250
Breiman	3	98	0.6081967	0.7737705	0.250
Breiman	3	99	0.6340164	0.7737705	0.250
Breiman	3	100	0.6336066	0.7836066	0.250
Breiman	4	20	0.4651639	0.7344262	0.250
Breiman	4	21	0.4934426	0.7442623	0.125
Breiman	4	22	0.4948770	0.7475410	0.125
Breiman	4	23	0.4801230	0.7508197	0.125
Breiman	4	24	0.5084016	0.7344262	0.125
Breiman	4	25	0.5252049	0.7606557	0.125
Breiman	4	26	0.5186475	0.7606557	0.125
Breiman	4	27	0.5317623	0.7803279	0.125
Breiman	4	28	0.5540984	0.7573770	0.250
Breiman	4	29	0.5524590	0.7803279	0.125
Breiman	4	30	0.5573770	0.7606557	0.250
Breiman	4	31	0.5762295	0.7704918	0.250
Breiman	4	32	0.5889344	0.7639344	0.250
Breiman	4	33	0.5782787	0.7475410	0.125
Breiman	4	34	0.5807377	0.7606557	0.125
Breiman	4	35	0.5950820	0.7573770	0.125
Breiman	4	36	0.6045082	0.7475410	0.250
Breiman	4	37	0.6024590	0.7540984	0.125
Breiman	4	38	0.5971311	0.7508197	0.125
Breiman	4	39	0.6000000	0.7639344	0.125
Breiman	4	40	0.6020492	0.7672131	0.375
Breiman	4	41	0.6024590	0.7639344	0.125
Breiman	4	42	0.6004098	0.7737705	0.250
Breiman	4	43	0.6040984	0.7639344	0.250
Breiman	4	44	0.6028689	0.7639344	0.250
Breiman	4	45	0.5959016	0.7606557	0.125
Breiman	4	46	0.5954918	0.7737705	0.375
Breiman	4	47	0.6020492	0.7704918	0.125
Breiman	4	48	0.6032787	0.7672131	0.125
Breiman	4	49	0.5983607	0.7704918	0.125
Breiman	4	50	0.6008197	0.7704918	0.375
Breiman	4	51	0.6000000	0.7704918	0.375
Breiman	4	52	0.5963115	0.7737705	0.375

Breiman	4	53	0.5922131	0.7737705	0.250
Breiman	4	54	0.6004098	0.7770492	0.375
Breiman	4	55	0.6073770	0.7770492	0.250
Breiman	4	56	0.6151639	0.7672131	0.375
Breiman	4	57	0.6245902	0.7704918	0.250
Breiman	4	58	0.6250000	0.7704918	0.250
Breiman	4	59	0.6254098	0.7639344	0.250
Breiman	4	60	0.6221311	0.7672131	0.250
Breiman	4	61	0.6196721	0.7737705	0.250
Breiman	4	62	0.6127049	0.7803279	0.250
Breiman	4	63	0.6135246	0.7770492	0.250
Breiman	4	64	0.6155738	0.7737705	0.250
Breiman	4	65	0.6094262	0.7770492	0.250
Breiman	4	66	0.6180328	0.7803279	0.250
Breiman	4	67	0.6225410	0.7737705	0.375
Breiman	4	68	0.6188525	0.7704918	0.375
Breiman	4	69	0.6225410	0.7704918	0.375
Breiman	4	70	0.6237705	0.7836066	0.375
Breiman	4	71	0.6295082	0.7737705	0.375
Breiman	4	72	0.6307377	0.7836066	0.250
Breiman	4	73	0.6258197	0.7737705	0.375
Breiman	4	74	0.6139344	0.7803279	0.250
Breiman	4	75	0.6147541	0.7737705	0.250
Breiman	4	76	0.6155738	0.7770492	0.250
Breiman	4	77	0.6127049	0.7704918	0.250
Breiman	4	78	0.6188525	0.7803279	0.250
Breiman	4	79	0.6163934	0.7836066	0.250
Breiman	4	80	0.6217213	0.7803279	0.250
Breiman	4	81	0.6176230	0.7770492	0.250
Breiman	4	82	0.6204918	0.7770492	0.250
Breiman	4	83	0.6204918	0.7770492	0.250
Breiman	4	84	0.6196721	0.7770492	0.250
Breiman	4	85	0.6168033	0.7770492	0.250
Breiman	4	86	0.6180328	0.7770492	0.250
Breiman	4	87	0.6209016	0.7868852	0.250
Breiman	4	88	0.6155738	0.7770492	0.250
Breiman	4	89	0.6159836	0.7836066	0.250
Breiman	4	90	0.6168033	0.7836066	0.250
Breiman	4	91	0.6176230	0.7803279	0.250
Breiman	4	92	0.6143443	0.7803279	0.250
Breiman	4	93	0.6118852	0.7836066	0.250
Breiman	4	94	0.6086066	0.7803279	0.250
Breiman	4	95	0.6045082	0.7737705	0.125
Breiman	4	96	0.6049180	0.7737705	0.250
Breiman	4	97	0.6094262	0.7770492	0.250
Breiman	4	98	0.6110656	0.7770492	0.250
Breiman	4	99	0.6122951	0.7836066	0.250
Breiman	4	100	0.6147541	0.7868852	0.250

Freund	2	20	0.5590164	0.6754098	0.375
Freund	2	21	0.5520492	0.7377049	0.250
Freund	2	22	0.5418033	0.7081967	0.375
Freund	2	23	0.5401639	0.7278689	0.375
Freund	2	24	0.5577869	0.7180328	0.375
Freund	2	25	0.5344262	0.7016393	0.375
Freund	2	26	0.5139344	0.7114754	0.375
Freund	2	27	0.5073770	0.7016393	0.250
Freund	2	28	0.5069672	0.7180328	0.250
Freund	2	29	0.4987705	0.6918033	0.375
Freund	2	30	0.5159836	0.7016393	0.250
Freund	2	31	0.5073770	0.7016393	0.250
Freund	2	32	0.4795082	0.6950820	0.250
Freund	2	33	0.4881148	0.7049180	0.250
Freund	2	34	0.5040984	0.7147541	0.250
Freund	2	35	0.5020492	0.7278689	0.250
Freund	2	36	0.5114754	0.7377049	0.250
Freund	2	37	0.5118852	0.7245902	0.250
Freund	2	38	0.5163934	0.7213115	0.250
Freund	2	39	0.5397541	0.7180328	0.375
Freund	2	40	0.5233607	0.7081967	0.375
Freund	2	41	0.5295082	0.7213115	0.250
Freund	2	42	0.5336066	0.7147541	0.375
Freund	2	43	0.5381148	0.7180328	0.375
Freund	2	44	0.5217213	0.7049180	0.375
Freund	2	45	0.5065574	0.7049180	0.375
Freund	2	46	0.5209016	0.7114754	0.375
Freund	2	47	0.5016393	0.6983607	0.375
Freund	2	48	0.5557377	0.7114754	0.250
Freund	2	49	0.5614754	0.7049180	0.250
Freund	2	50	0.5639344	0.7049180	0.250
Freund	2	51	0.5573770	0.7081967	0.375
Freund	2	52	0.5627049	0.7016393	0.375
Freund	2	53	0.5815574	0.6950820	0.375
Freund	2	54	0.5770492	0.7180328	0.375
Freund	2	55	0.5586066	0.7081967	0.250
Freund	2	56	0.5655738	0.6983607	0.375
Freund	2	57	0.5692623	0.7114754	0.250
Freund	2	58	0.5790984	0.7147541	0.250
Freund	2	59	0.5778689	0.7344262	0.250
Freund	2	60	0.5741803	0.7245902	0.250
Freund	2	61	0.5745902	0.7245902	0.250
Freund	2	62	0.5942623	0.7147541	0.250
Freund	2	63	0.6065574	0.7278689	0.250
Freund	2	64	0.5881148	0.7311475	0.250
Freund	2	65	0.5913934	0.7278689	0.250
Freund	2	66	0.5877049	0.7147541	0.250
Freund	2	67	0.5864754	0.7245902	0.250

Freund	2	68	0.5856557	0.7311475	0.250
Freund	2	69	0.5815574	0.7377049	0.250
Freund	2	70	0.5889344	0.7344262	0.250
Freund	2	71	0.5827869	0.7278689	0.250
Freund	2	72	0.5668033	0.7147541	0.250
Freund	2	73	0.5733607	0.7245902	0.250
Freund	2	74	0.5762295	0.7180328	0.250
Freund	2	75	0.5782787	0.7278689	0.250
Freund	2	76	0.5901639	0.7377049	0.250
Freund	2	77	0.5991803	0.7344262	0.250
Freund	2	78	0.5905738	0.7278689	0.250
Freund	2	79	0.5983607	0.7344262	0.375
Freund	2	80	0.5954918	0.7278689	0.375
Freund	2	81	0.5938525	0.7180328	0.250
Freund	2	82	0.5918033	0.7409836	0.250
Freund	2	83	0.5946721	0.7377049	0.250
Freund	2	84	0.5934426	0.7409836	0.250
Freund	2	85	0.5922131	0.7311475	0.375
Freund	2	86	0.5946721	0.7213115	0.375
Freund	2	87	0.5893443	0.7213115	0.375
Freund	2	88	0.5844262	0.7147541	0.375
Freund	2	89	0.5844262	0.7180328	0.250
Freund	2	90	0.5840164	0.7180328	0.375
Freund	2	91	0.5758197	0.7344262	0.250
Freund	2	92	0.5700820	0.7278689	0.375
Freund	2	93	0.5692623	0.7245902	0.375
Freund	2	94	0.5737705	0.7245902	0.375
Freund	2	95	0.5782787	0.7245902	0.375
Freund	2	96	0.5774590	0.7311475	0.375
Freund	2	97	0.5696721	0.7245902	0.375
Freund	2	98	0.5639344	0.7245902	0.250
Freund	2	99	0.5618852	0.7311475	0.250
Freund	2	100	0.5692623	0.7245902	0.250
Freund	3	20	0.6290984	0.7737705	0.250
Freund	3	21	0.6219262	0.7672131	0.500
Freund	3	22	0.6323770	0.7803279	0.500
Freund	3	23	0.6217213	0.7704918	0.500
Freund	3	24	0.6295082	0.7737705	0.500
Freund	3	25	0.6315574	0.7868852	0.375
Freund	3	26	0.6405738	0.7803279	0.375
Freund	3	27	0.6397541	0.7803279	0.375
Freund	3	28	0.6196721	0.7803279	0.375
Freund	3	29	0.6163934	0.7934426	0.375
Freund	3	30	0.6155738	0.7737705	0.375
Freund	3	31	0.5864754	0.7737705	0.375
Freund	3	32	0.5901639	0.7737705	0.375
Freund	3	33	0.5950820	0.7901639	0.375
Freund	3	34	0.6114754	0.8000000	0.375

Freund	3	35	0.6151639	0.8032787	0.375
Freund	3	36	0.6229508	0.7934426	0.375
Freund	3	37	0.6065574	0.7934426	0.375
Freund	3	38	0.6032787	0.8131148	0.250
Freund	3	39	0.5971311	0.8098361	0.250
Freund	3	40	0.6151639	0.7868852	0.250
Freund	3	41	0.6135246	0.7868852	0.375
Freund	3	42	0.6098361	0.7868852	0.375
Freund	3	43	0.6135246	0.7836066	0.375
Freund	3	44	0.6200820	0.7803279	0.250
Freund	3	45	0.6127049	0.7704918	0.375
Freund	3	46	0.6143443	0.7803279	0.250
Freund	3	47	0.6053279	0.7737705	0.250
Freund	3	48	0.5950820	0.7770492	0.250
Freund	3	49	0.6016393	0.7704918	0.250
Freund	3	50	0.5942623	0.7672131	0.250
Freund	3	51	0.5926230	0.7770492	0.375
Freund	3	52	0.5938525	0.7803279	0.375
Freund	3	53	0.5893443	0.7770492	0.375
Freund	3	54	0.5877049	0.7836066	0.375
Freund	3	55	0.5881148	0.7770492	0.375
Freund	3	56	0.5868852	0.7770492	0.375
Freund	3	57	0.5762295	0.7737705	0.375
Freund	3	58	0.5627049	0.7803279	0.375
Freund	3	59	0.5577869	0.7770492	0.250
Freund	3	60	0.5602459	0.7868852	0.375
Freund	3	61	0.5561475	0.7934426	0.250
Freund	3	62	0.5659836	0.7934426	0.250
Freund	3	63	0.5643443	0.7934426	0.250
Freund	3	64	0.5618852	0.7901639	0.250
Freund	3	65	0.5598361	0.7967213	0.250
Freund	3	66	0.5627049	0.7934426	0.250
Freund	3	67	0.5704918	0.7934426	0.250
Freund	3	68	0.5700820	0.7967213	0.250
Freund	3	69	0.5668033	0.7967213	0.250
Freund	3	70	0.5606557	0.7967213	0.250
Freund	3	71	0.5651639	0.7934426	0.250
Freund	3	72	0.5651639	0.8000000	0.250
Freund	3	73	0.5569672	0.7934426	0.250
Freund	3	74	0.5540984	0.8000000	0.250
Freund	3	75	0.5668033	0.7934426	0.250
Freund	3	76	0.5721311	0.7901639	0.250
Freund	3	77	0.5631148	0.7934426	0.250
Freund	3	78	0.5528689	0.7836066	0.250
Freund	3	79	0.5442623	0.7934426	0.250
Freund	3	80	0.5639344	0.7967213	0.250
Freund	3	81	0.5577869	0.8000000	0.250
Freund	3	82	0.5549180	0.7967213	0.250

Freund	3	83	0.5565574	0.8000000	0.250
Freund	3	84	0.5602459	0.8032787	0.250
Freund	3	85	0.5598361	0.7934426	0.250
Freund	3	86	0.5581967	0.7901639	0.250
Freund	3	87	0.5524590	0.7901639	0.250
Freund	3	88	0.5500000	0.7868852	0.250
Freund	3	89	0.5495902	0.7901639	0.250
Freund	3	90	0.5405738	0.7836066	0.250
Freund	3	91	0.5569672	0.7868852	0.250
Freund	3	92	0.5594262	0.7868852	0.250
Freund	3	93	0.5598361	0.7868852	0.250
Freund	3	94	0.5606557	0.7803279	0.250
Freund	3	95	0.5688525	0.7836066	0.250
Freund	3	96	0.5717213	0.7836066	0.250
Freund	3	97	0.5717213	0.7868852	0.250
Freund	3	98	0.5561475	0.7868852	0.250
Freund	3	99	0.5602459	0.7836066	0.250
Freund	3	100	0.5577869	0.7836066	0.250
Freund	4	20	0.6397541	0.7704918	0.625
Freund	4	21	0.5975410	0.7508197	0.375
Freund	4	22	0.5811475	0.7409836	0.500
Freund	4	23	0.5893443	0.7540984	0.375
Freund	4	24	0.5885246	0.7573770	0.375
Freund	4	25	0.6098361	0.7639344	0.375
Freund	4	26	0.5893443	0.7639344	0.375
Freund	4	27	0.6204918	0.7606557	0.500
Freund	4	28	0.6381148	0.7573770	0.375
Freund	4	29	0.6229508	0.7672131	0.375
Freund	4	30	0.6098361	0.7672131	0.375
Freund	4	31	0.6151639	0.7606557	0.375
Freund	4	32	0.6188525	0.7442623	0.375
Freund	4	33	0.6204918	0.7540984	0.375
Freund	4	34	0.5991803	0.7475410	0.375
Freund	4	35	0.5971311	0.7409836	0.375
Freund	4	36	0.6122951	0.7639344	0.375
Freund	4	37	0.6045082	0.7573770	0.375
Freund	4	38	0.5995902	0.7508197	0.500
Freund	4	39	0.5954918	0.7540984	0.375
Freund	4	40	0.6057377	0.7508197	0.500
Freund	4	41	0.6122951	0.7606557	0.375
Freund	4	42	0.6077869	0.7508197	0.500
Freund	4	43	0.6098361	0.7409836	0.500
Freund	4	44	0.6045082	0.7573770	0.500
Freund	4	45	0.6000000	0.7442623	0.500
Freund	4	46	0.5987705	0.7442623	0.500
Freund	4	47	0.5864754	0.7409836	0.375
Freund	4	48	0.5831967	0.7377049	0.500
Freund	4	49	0.5811475	0.7442623	0.500

Freund	4	50	0.5913934	0.7573770	0.375
Freund	4	51	0.5758197	0.7344262	0.375
Freund	4	52	0.5889344	0.7377049	0.375
Freund	4	53	0.5930328	0.7442623	0.500
Freund	4	54	0.5848361	0.7409836	0.375
Freund	4	55	0.5897541	0.7573770	0.375
Freund	4	56	0.5930328	0.7344262	0.375
Freund	4	57	0.5872951	0.7311475	0.375
Freund	4	58	0.5905738	0.7377049	0.375
Freund	4	59	0.5889344	0.7475410	0.375
Freund	4	60	0.5872951	0.7606557	0.375
Freund	4	61	0.5909836	0.7409836	0.375
Freund	4	62	0.5770492	0.7442623	0.375
Freund	4	63	0.5872951	0.7344262	0.375
Freund	4	64	0.5967213	0.7377049	0.375
Freund	4	65	0.5856557	0.7409836	0.375
Freund	4	66	0.5774590	0.7409836	0.375
Freund	4	67	0.5668033	0.7377049	0.375
Freund	4	68	0.5700820	0.7442623	0.375
Freund	4	69	0.5709016	0.7606557	0.375
Freund	4	70	0.5799180	0.7475410	0.375
Freund	4	71	0.5803279	0.7377049	0.375
Freund	4	72	0.5774590	0.7409836	0.375
Freund	4	73	0.5700820	0.7475410	0.375
Freund	4	74	0.5557377	0.7442623	0.375
Freund	4	75	0.5647541	0.7377049	0.375
Freund	4	76	0.5688525	0.7475410	0.375
Freund	4	77	0.5827869	0.7442623	0.375
Freund	4	78	0.5823770	0.7508197	0.375
Freund	4	79	0.5786885	0.7442623	0.375
Freund	4	80	0.5811475	0.7344262	0.375
Freund	4	81	0.5913934	0.7442623	0.375
Freund	4	82	0.5905738	0.7442623	0.375
Freund	4	83	0.5926230	0.7475410	0.375
Freund	4	84	0.5942623	0.7508197	0.375
Freund	4	85	0.5950820	0.7540984	0.375
Freund	4	86	0.5905738	0.7475410	0.375
Freund	4	87	0.5901639	0.7508197	0.375
Freund	4	88	0.5905738	0.7475410	0.375
Freund	4	89	0.5954918	0.7508197	0.375
Freund	4	90	0.5963115	0.7540984	0.375
Freund	4	91	0.5918033	0.7508197	0.375
Freund	4	92	0.5918033	0.7540984	0.375
Freund	4	93	0.5860656	0.7442623	0.375
Freund	4	94	0.5840164	0.7475410	0.375
Freund	4	95	0.5868852	0.7442623	0.375
Freund	4	96	0.5872951	0.7442623	0.375
Freund	4	97	0.5868852	0.7442623	0.375

Freund	4	98	0.5856557	0.7475410	0.375
Freund	4	99	0.5942623	0.7508197	0.375
Freund	4	100	0.5930328	0.7606557	0.375
Zhu	2	20	0.5663934	0.7770492	0.250
Zhu	2	21	0.5508197	0.7672131	0.125
Zhu	2	22	0.5520492	0.7868852	0.375
Zhu	2	23	0.5377049	0.7836066	0.375
Zhu	2	24	0.5459016	0.7967213	0.375
Zhu	2	25	0.5782787	0.8163934	0.375
Zhu	2	26	0.5684426	0.8065574	0.250
Zhu	2	27	0.5315574	0.7967213	0.250
Zhu	2	28	0.5459016	0.7672131	0.375
Zhu	2	29	0.5393443	0.7901639	0.250
Zhu	2	30	0.5360656	0.8065574	0.375
Zhu	2	31	0.4979508	0.7934426	0.125
Zhu	2	32	0.5168033	0.7803279	0.250
Zhu	2	33	0.5430328	0.7868852	0.250
Zhu	2	34	0.5327869	0.7868852	0.250
Zhu	2	35	0.5520492	0.8000000	0.250
Zhu	2	36	0.5524590	0.8163934	0.250
Zhu	2	37	0.5475410	0.8163934	0.250
Zhu	2	38	0.5877049	0.8262295	0.250
Zhu	2	39	0.5680328	0.8065574	0.250
Zhu	2	40	0.6118852	0.8295082	0.250
Zhu	2	41	0.6118852	0.7934426	0.250
Zhu	2	42	0.6127049	0.8163934	0.250
Zhu	2	43	0.6061475	0.8032787	0.250
Zhu	2	44	0.6061475	0.8032787	0.250
Zhu	2	45	0.5893443	0.8032787	0.250
Zhu	2	46	0.5852459	0.8032787	0.500
Zhu	2	47	0.5717213	0.8000000	0.125
Zhu	2	48	0.5959016	0.7934426	0.375
Zhu	2	49	0.5979508	0.7901639	0.250
Zhu	2	50	0.5942623	0.8032787	0.250
Zhu	2	51	0.6040984	0.7868852	0.375
Zhu	2	52	0.6000000	0.8098361	0.250
Zhu	2	53	0.6213115	0.8098361	0.375
Zhu	2	54	0.5942623	0.8065574	0.375
Zhu	2	55	0.5926230	0.8032787	0.375
Zhu	2	56	0.6008197	0.8032787	0.375
Zhu	2	57	0.5979508	0.8131148	0.250
Zhu	2	58	0.5868852	0.8131148	0.375
Zhu	2	59	0.5868852	0.8131148	0.375
Zhu	2	60	0.5807377	0.8163934	0.250
Zhu	2	61	0.5790984	0.8032787	0.375
Zhu	2	62	0.5672131	0.7967213	0.375
Zhu	2	63	0.5655738	0.8131148	0.250
Zhu	2	64	0.5762295	0.8032787	0.375

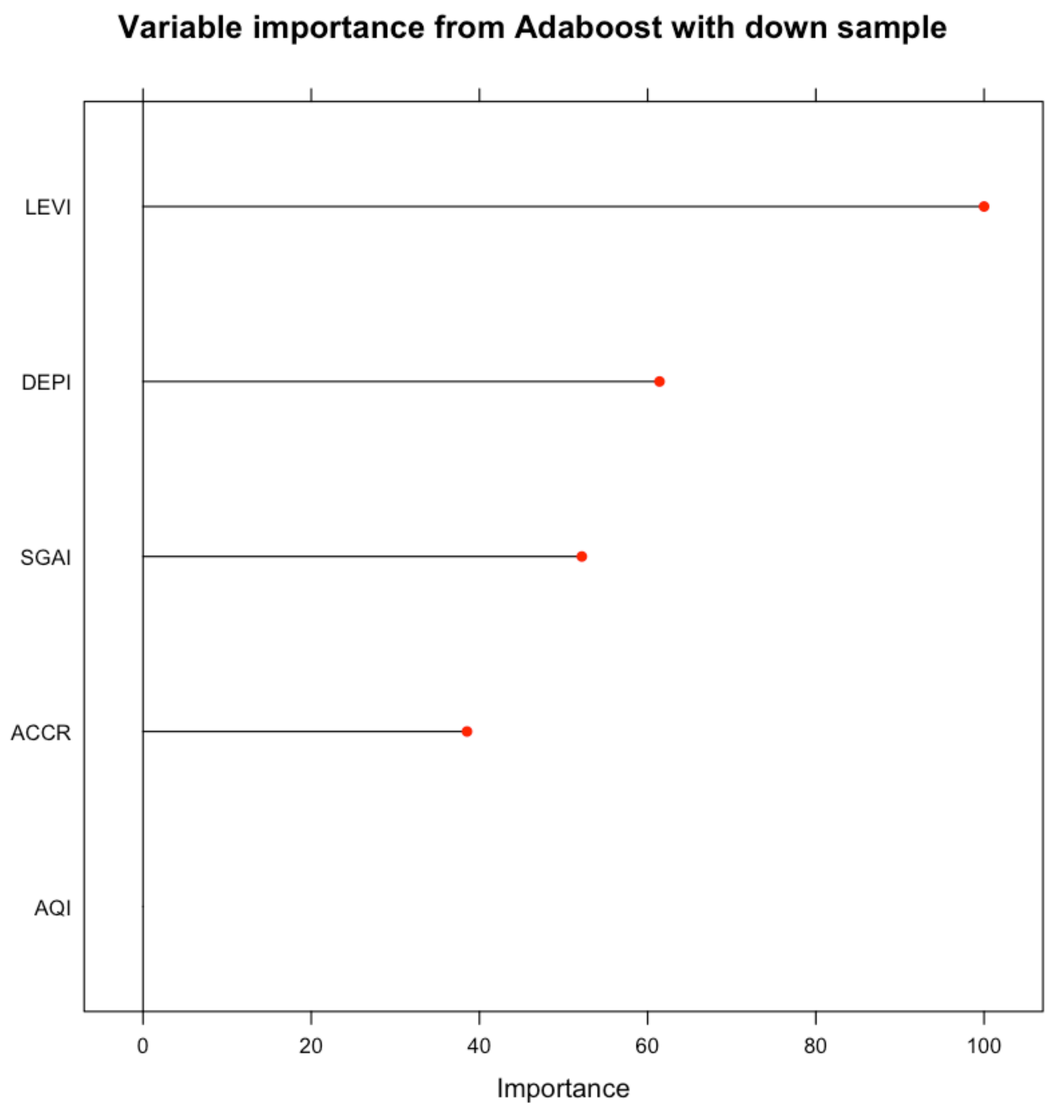
Zhu	2	65	0.5680328	0.8131148	0.250
Zhu	2	66	0.5889344	0.8065574	0.375
Zhu	2	67	0.5852459	0.8032787	0.250
Zhu	2	68	0.5840164	0.8000000	0.375
Zhu	2	69	0.6069672	0.8032787	0.375
Zhu	2	70	0.5954918	0.8065574	0.375
Zhu	2	71	0.6024590	0.7967213	0.375
Zhu	2	72	0.5872951	0.8065574	0.375
Zhu	2	73	0.6090164	0.8032787	0.375
Zhu	2	74	0.5786885	0.8065574	0.375
Zhu	2	75	0.5831967	0.8032787	0.375
Zhu	2	76	0.5803279	0.8131148	0.375
Zhu	2	77	0.5733607	0.8163934	0.375
Zhu	2	78	0.5504098	0.8065574	0.250
Zhu	2	79	0.5360656	0.7967213	0.250
Zhu	2	80	0.5536885	0.8098361	0.250
Zhu	2	81	0.5663934	0.7967213	0.375
Zhu	2	82	0.5618852	0.8000000	0.250
Zhu	2	83	0.5602459	0.7901639	0.375
Zhu	2	84	0.5430328	0.7967213	0.250
Zhu	2	85	0.5602459	0.7901639	0.375
Zhu	2	86	0.5557377	0.7901639	0.375
Zhu	2	87	0.5549180	0.8000000	0.375
Zhu	2	88	0.5487705	0.7901639	0.375
Zhu	2	89	0.5540984	0.8000000	0.375
Zhu	2	90	0.5696721	0.8032787	0.375
Zhu	2	91	0.5704918	0.7967213	0.375
Zhu	2	92	0.5688525	0.7967213	0.375
Zhu	2	93	0.5713115	0.7934426	0.375
Zhu	2	94	0.5737705	0.8065574	0.375
Zhu	2	95	0.5688525	0.8000000	0.375
Zhu	2	96	0.5487705	0.7901639	0.375
Zhu	2	97	0.5512295	0.7803279	0.250
Zhu	2	98	0.5504098	0.7836066	0.250
Zhu	2	99	0.5598361	0.7803279	0.250
Zhu	2	100	0.5631148	0.7967213	0.375
Zhu	3	20	0.5512295	0.6983607	0.375
Zhu	3	21	0.5729508	0.7311475	0.375
Zhu	3	22	0.5704918	0.7377049	0.250
Zhu	3	23	0.5450820	0.7508197	0.250
Zhu	3	24	0.5823770	0.7278689	0.250
Zhu	3	25	0.5848361	0.7475410	0.250
Zhu	3	26	0.6024590	0.7704918	0.500
Zhu	3	27	0.5864754	0.7409836	0.500
Zhu	3	28	0.6258197	0.7508197	0.500
Zhu	3	29	0.6315574	0.7606557	0.500
Zhu	3	30	0.6348361	0.7704918	0.500
Zhu	3	31	0.6356557	0.7770492	0.500

Zhu	3	32	0.6352459	0.7770492	0.500
Zhu	3	33	0.6389344	0.7901639	0.500
Zhu	3	34	0.6159836	0.7737705	0.500
Zhu	3	35	0.6192623	0.7704918	0.500
Zhu	3	36	0.6139344	0.7704918	0.375
Zhu	3	37	0.6299180	0.7770492	0.500
Zhu	3	38	0.6360656	0.7737705	0.500
Zhu	3	39	0.6467213	0.7704918	0.500
Zhu	3	40	0.6315574	0.7672131	0.500
Zhu	3	41	0.6217213	0.7573770	0.500
Zhu	3	42	0.6274590	0.7573770	0.500
Zhu	3	43	0.6340164	0.7606557	0.500
Zhu	3	44	0.6172131	0.7606557	0.500
Zhu	3	45	0.6172131	0.7606557	0.500
Zhu	3	46	0.6122951	0.7475410	0.500
Zhu	3	47	0.6184426	0.7672131	0.500
Zhu	3	48	0.6258197	0.7639344	0.500
Zhu	3	49	0.6204918	0.7639344	0.500
Zhu	3	50	0.6061475	0.7508197	0.500
Zhu	3	51	0.6372951	0.7639344	0.625
Zhu	3	52	0.6381148	0.7737705	0.500
Zhu	3	53	0.6360656	0.7606557	0.625
Zhu	3	54	0.6450820	0.7639344	0.625
Zhu	3	55	0.6479508	0.7606557	0.625
Zhu	3	56	0.6356557	0.7606557	0.500
Zhu	3	57	0.6221311	0.7540984	0.500
Zhu	3	58	0.6184426	0.7475410	0.375
Zhu	3	59	0.6254098	0.7508197	0.625
Zhu	3	60	0.6331967	0.7573770	0.625
Zhu	3	61	0.6372951	0.7672131	0.625
Zhu	3	62	0.6344262	0.7540984	0.625
Zhu	3	63	0.6397541	0.7606557	0.625
Zhu	3	64	0.6430328	0.7606557	0.500
Zhu	3	65	0.6418033	0.7442623	0.625
Zhu	3	66	0.6331967	0.7540984	0.500
Zhu	3	67	0.6336066	0.7409836	0.625
Zhu	3	68	0.6331967	0.7442623	0.625
Zhu	3	69	0.6196721	0.7442623	0.500
Zhu	3	70	0.6155738	0.7475410	0.500
Zhu	3	71	0.6188525	0.7508197	0.500
Zhu	3	72	0.6209016	0.7475410	0.500
Zhu	3	73	0.6143443	0.7540984	0.500
Zhu	3	74	0.6090164	0.7540984	0.500
Zhu	3	75	0.6102459	0.7508197	0.500
Zhu	3	76	0.6069672	0.7540984	0.500
Zhu	3	77	0.6036885	0.7409836	0.500
Zhu	3	78	0.6090164	0.7540984	0.375
Zhu	3	79	0.6168033	0.7508197	0.625

Zhu	3	80	0.6159836	0.7606557	0.375
Zhu	3	81	0.6245902	0.7573770	0.625
Zhu	3	82	0.6262295	0.7540984	0.500
Zhu	3	83	0.6241803	0.7573770	0.625
Zhu	3	84	0.6344262	0.7606557	0.625
Zhu	3	85	0.6303279	0.7639344	0.500
Zhu	3	86	0.6245902	0.7475410	0.625
Zhu	3	87	0.6299180	0.7606557	0.500
Zhu	3	88	0.6262295	0.7606557	0.625
Zhu	3	89	0.6290984	0.7639344	0.500
Zhu	3	90	0.6188525	0.7606557	0.500
Zhu	3	91	0.6229508	0.7475410	0.500
Zhu	3	92	0.6176230	0.7475410	0.500
Zhu	3	93	0.6196721	0.7409836	0.500
Zhu	3	94	0.6176230	0.7475410	0.500
Zhu	3	95	0.6266393	0.7540984	0.625
Zhu	3	96	0.6217213	0.7475410	0.625
Zhu	3	97	0.6250000	0.7540984	0.625
Zhu	3	98	0.6204918	0.7639344	0.500
Zhu	3	99	0.6209016	0.7573770	0.500
Zhu	3	100	0.6168033	0.7573770	0.375
Zhu	4	20	0.6704918	0.7508197	0.375
Zhu	4	21	0.6356557	0.7573770	0.375
Zhu	4	22	0.6299180	0.7672131	0.375
Zhu	4	23	0.6233607	0.7803279	0.375
Zhu	4	24	0.6245902	0.7836066	0.375
Zhu	4	25	0.6213115	0.8065574	0.375
Zhu	4	26	0.6204918	0.7934426	0.375
Zhu	4	27	0.5934426	0.7934426	0.375
Zhu	4	28	0.6012295	0.7803279	0.500
Zhu	4	29	0.5991803	0.7704918	0.375
Zhu	4	30	0.5946721	0.7606557	0.375
Zhu	4	31	0.6016393	0.7836066	0.375
Zhu	4	32	0.6077869	0.7901639	0.375
Zhu	4	33	0.5995902	0.7868852	0.375
Zhu	4	34	0.6000000	0.7934426	0.375
Zhu	4	35	0.6131148	0.8000000	0.375
Zhu	4	36	0.6073770	0.8000000	0.375
Zhu	4	37	0.6020492	0.8032787	0.375
Zhu	4	38	0.6008197	0.7934426	0.375
Zhu	4	39	0.5971311	0.7934426	0.375
Zhu	4	40	0.5856557	0.7868852	0.375
Zhu	4	41	0.5852459	0.7934426	0.375
Zhu	4	42	0.5897541	0.7868852	0.375
Zhu	4	43	0.6036885	0.7901639	0.375
Zhu	4	44	0.6081967	0.7836066	0.375
Zhu	4	45	0.6151639	0.7934426	0.250
Zhu	4	46	0.6094262	0.7934426	0.375

Zhu	4	47	0.6081967	0.8000000	0.375
Zhu	4	48	0.6036885	0.7934426	0.375
Zhu	4	49	0.6020492	0.7967213	0.250
Zhu	4	50	0.6102459	0.7934426	0.375
Zhu	4	51	0.6081967	0.7967213	0.375
Zhu	4	52	0.6110656	0.8032787	0.375
Zhu	4	53	0.6053279	0.8000000	0.375
Zhu	4	54	0.6016393	0.8032787	0.375
Zhu	4	55	0.6122951	0.7967213	0.375
Zhu	4	56	0.6127049	0.8032787	0.375
Zhu	4	57	0.6196721	0.8032787	0.375
Zhu	4	58	0.6040984	0.8065574	0.375
Zhu	4	59	0.5934426	0.8098361	0.375
Zhu	4	60	0.5918033	0.8000000	0.375
Zhu	4	61	0.5868852	0.8098361	0.375
Zhu	4	62	0.6127049	0.8098361	0.375
Zhu	4	63	0.5934426	0.8098361	0.375
Zhu	4	64	0.5872951	0.8098361	0.250
Zhu	4	65	0.6045082	0.7967213	0.375
Zhu	4	66	0.6020492	0.8131148	0.375
Zhu	4	67	0.6053279	0.8163934	0.375
Zhu	4	68	0.6147541	0.8196721	0.250
Zhu	4	69	0.6057377	0.8229508	0.375
Zhu	4	70	0.6032787	0.8229508	0.375
Zhu	4	71	0.6069672	0.8229508	0.375
Zhu	4	72	0.5975410	0.8196721	0.375
Zhu	4	73	0.5922131	0.8196721	0.375
Zhu	4	74	0.6024590	0.8163934	0.250
Zhu	4	75	0.5975410	0.8065574	0.375
Zhu	4	76	0.6098361	0.8098361	0.375
Zhu	4	77	0.6225410	0.8131148	0.375
Zhu	4	78	0.6241803	0.8229508	0.375
Zhu	4	79	0.6213115	0.8229508	0.375
Zhu	4	80	0.6188525	0.8131148	0.375
Zhu	4	81	0.6163934	0.8196721	0.375
Zhu	4	82	0.6237705	0.8131148	0.375
Zhu	4	83	0.6266393	0.8131148	0.375
Zhu	4	84	0.6266393	0.8065574	0.375
Zhu	4	85	0.6336066	0.8032787	0.375
Zhu	4	86	0.6204918	0.8032787	0.375
Zhu	4	87	0.6196721	0.8032787	0.375
Zhu	4	88	0.6204918	0.8032787	0.375
Zhu	4	89	0.6286885	0.8065574	0.375
Zhu	4	90	0.6221311	0.8065574	0.375
Zhu	4	91	0.6237705	0.8098361	0.375
Zhu	4	92	0.6282787	0.8098361	0.375
Zhu	4	93	0.6344262	0.8196721	0.375
Zhu	4	94	0.6274590	0.8196721	0.375

```
  Zhu        4          95     0.6290984  0.8196721  0.375
  Zhu        4          96     0.6319672  0.8098361  0.375
  Zhu        4          97     0.6295082  0.8065574  0.375
  Zhu        4          98     0.6299180  0.8131148  0.250
  Zhu        4          99     0.6381148  0.8098361  0.375
  Zhu        4         100     0.6368852  0.8229508  0.250

ROC was used to select the optimal model using the largest value.
The final values used for the model were mfinal = 86, maxdepth = 2
 and coeflearn = Breiman.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  77.3  1.6
       Yes 20.1  1.0

 Accuracy (average) : 0.7827
```



Confusion Matrix for adaboost on test set

```
In [55]: caretPredictedClass <- predict(ada_down.model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  261   2
       Yes  99   9

               Accuracy : 0.7278
                 95% CI : (0.6794, 0.7724)
```

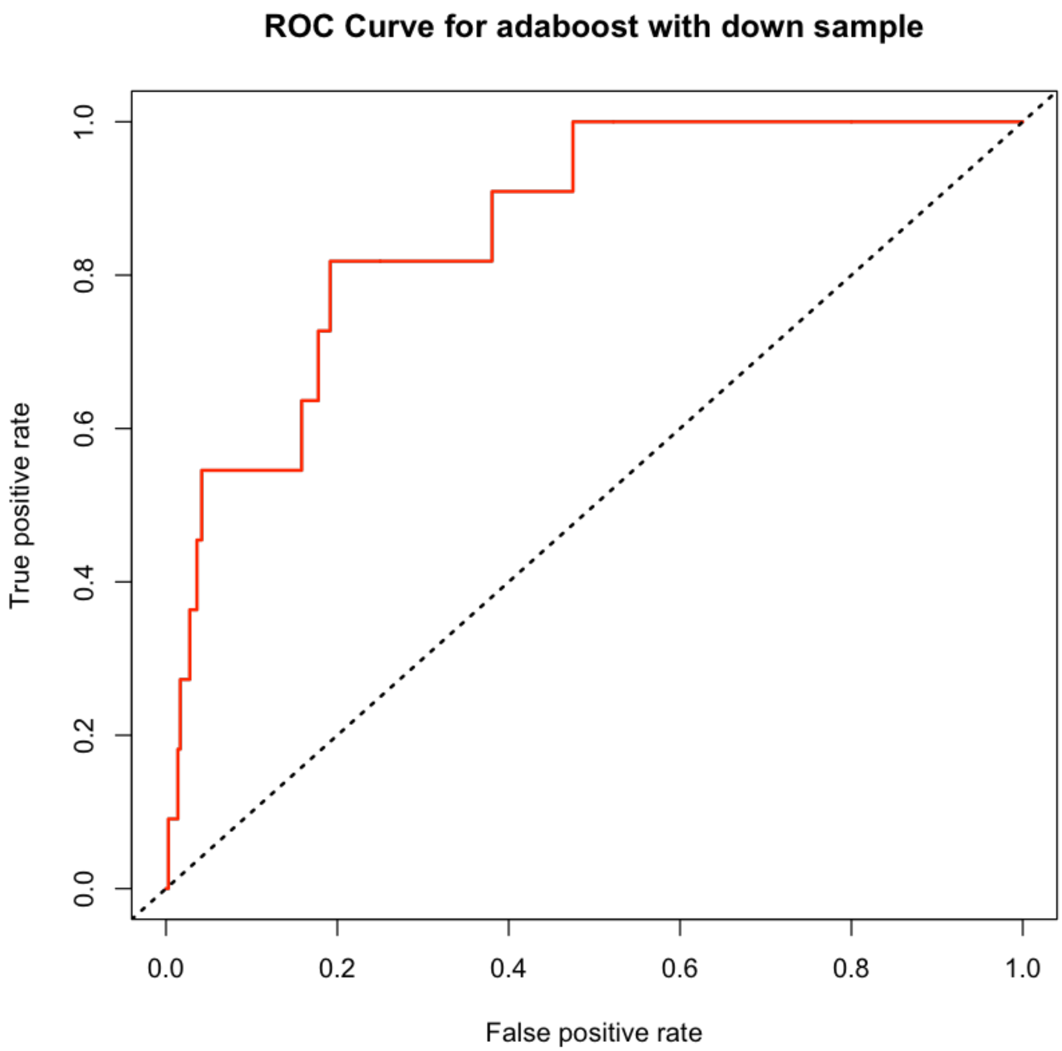
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.103
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.72500
            Specificity : 0.81818
         Pos Pred Value : 0.99240
         Neg Pred Value : 0.08333
             Prevalence : 0.97035
         Detection Rate : 0.70350
   Detection Prevalence : 0.70889
      Balanced Accuracy : 0.77159

       'Positive' Class : No
```

ROC plot for adaboost on test set

```
In [56]: ada_pred <- predict(ada_down.model, model_test_df, type = "prob")[,2]
         ada_prediction <- prediction(ada_pred,model_test_df$Manipulater)
         ada_perf <- performance(ada_prediction, "tpr","fpr")

         plot(ada_perf,main="ROC Curve for adaboost with down sample",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(ada_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8616162
```

```
Slot "alpha.values":
list()
```



1.5.5 Boosting with adaboost (SMOTE)

The below code chunk sets some of the control parameters for adaboost

```
In [57]: objControl <- trainControl(method='boot', number = 1,
                                     returnResamp='all',
                                     summaryFunction = twoClassSummary,
```

```
                                        savePredictions = TRUE,
                                        classProbs = TRUE,
                                        sampling = "smote")#, p = 0.70) #in case method = #"LGO"
```

```
In [58]: search_grid <- expand.grid(mfinal = c(20:100), maxdepth = c(2:4),
                                    coeflearn = c("Breiman", "Freund", "Zhu"))
```

After setting the control paramters, the model is run

```
In [59]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Adaptive Boosting with SMOTE")

         set.seed(4121)
         ada_smote_model <- train(model_train_df[,1:5], model_train_df[,6],
                                  method='AdaBoost.M1',
                                  trControl=objControl,
                                  tuneGrid = search_grid,
                                  metric = "ROC")
         stopCluster(num_cores)
         toc()

Adaptive Boosting with SMOTE: 123.87 sec elapsed
```

Confusion Matrix for adaboost on train set

```
In [60]: #ada_smote_model$finalModel #ada_smote_model$results
         print(ada_smote_model)
         confusionMatrix.train(ada_smote_model)
         plot(varImp(ada_smote_model), main = "Variable importance from Adaboost with SMOTE", 

AdaBoost.M1

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Addtional sampling using SMOTE

Resampling results across tuning parameters:

  coeflearn  maxdepth  mfinal  ROC        Sens       Spec
  Breiman    2          20     0.6805328  0.8262295  0.250
  Breiman    2          21     0.6838115  0.8590164  0.125
  Breiman    2          22     0.6547131  0.8590164  0.125
```

Breiman	2	23	0.6555328	0.8622951	0.000
Breiman	2	24	0.6629098	0.8721311	0.000
Breiman	2	25	0.6612705	0.8852459	0.000
Breiman	2	26	0.6657787	0.8754098	0.000
Breiman	2	27	0.6764344	0.8459016	0.125
Breiman	2	28	0.6784836	0.8622951	0.000
Breiman	2	29	0.6522541	0.8491803	0.125
Breiman	2	30	0.6534836	0.8688525	0.000
Breiman	2	31	0.6532787	0.8688525	0.125
Breiman	2	32	0.6491803	0.8688525	0.000
Breiman	2	33	0.6252049	0.8622951	0.125
Breiman	2	34	0.6362705	0.8459016	0.125
Breiman	2	35	0.6317623	0.8459016	0.125
Breiman	2	36	0.6247951	0.8590164	0.125
Breiman	2	37	0.6235656	0.8622951	0.125
Breiman	2	38	0.6227459	0.8590164	0.125
Breiman	2	39	0.6319672	0.8622951	0.125
Breiman	2	40	0.6319672	0.8688525	0.125
Breiman	2	41	0.6299180	0.8786885	0.125
Breiman	2	42	0.6319672	0.8655738	0.125
Breiman	2	43	0.6225410	0.8557377	0.125
Breiman	2	44	0.6118852	0.8622951	0.125
Breiman	2	45	0.6200820	0.8557377	0.125
Breiman	2	46	0.6196721	0.8622951	0.125
Breiman	2	47	0.6217213	0.8688525	0.125
Breiman	2	48	0.6422131	0.8688525	0.125
Breiman	2	49	0.6229508	0.8622951	0.125
Breiman	2	50	0.6102459	0.8557377	0.125
Breiman	2	51	0.6106557	0.8688525	0.125
Breiman	2	52	0.5918033	0.8590164	0.125
Breiman	2	53	0.5823770	0.8459016	0.125
Breiman	2	54	0.5823770	0.8590164	0.000
Breiman	2	55	0.5913934	0.8557377	0.125
Breiman	2	56	0.5963115	0.8590164	0.125
Breiman	2	57	0.5975410	0.8459016	0.125
Breiman	2	58	0.5967213	0.8590164	0.125
Breiman	2	59	0.5975410	0.8622951	0.125
Breiman	2	60	0.5934426	0.8655738	0.125
Breiman	2	61	0.6159836	0.8688525	0.000
Breiman	2	62	0.5950820	0.8655738	0.125
Breiman	2	63	0.5987705	0.8590164	0.125
Breiman	2	64	0.5987705	0.8557377	0.125
Breiman	2	65	0.5987705	0.8590164	0.125
Breiman	2	66	0.5950820	0.8590164	0.000
Breiman	2	67	0.5893443	0.8590164	0.000
Breiman	2	68	0.5905738	0.8622951	0.000
Breiman	2	69	0.5819672	0.8524590	0.125
Breiman	2	70	0.5790984	0.8491803	0.000

Breiman	2	71	0.5856557	0.8491803	0.000
Breiman	2	72	0.5950820	0.8524590	0.125
Breiman	2	73	0.5918033	0.8655738	0.000
Breiman	2	74	0.6053279	0.8491803	0.125
Breiman	2	75	0.6069672	0.8655738	0.000
Breiman	2	76	0.6061475	0.8622951	0.000
Breiman	2	77	0.6065574	0.8524590	0.125
Breiman	2	78	0.6057377	0.8622951	0.000
Breiman	2	79	0.6110656	0.8524590	0.125
Breiman	2	80	0.6008197	0.8459016	0.125
Breiman	2	81	0.6028689	0.8491803	0.125
Breiman	2	82	0.6147541	0.8426230	0.125
Breiman	2	83	0.6155738	0.8459016	0.125
Breiman	2	84	0.6151639	0.8557377	0.000
Breiman	2	85	0.6262295	0.8491803	0.000
Breiman	2	86	0.6237705	0.8426230	0.000
Breiman	2	87	0.6237705	0.8426230	0.125
Breiman	2	88	0.6069672	0.8393443	0.125
Breiman	2	89	0.6069672	0.8393443	0.125
Breiman	2	90	0.6069672	0.8622951	0.000
Breiman	2	91	0.6241803	0.8491803	0.125
Breiman	2	92	0.6372951	0.8590164	0.000
Breiman	2	93	0.6364754	0.8655738	0.000
Breiman	2	94	0.6290984	0.8688525	0.000
Breiman	2	95	0.6213115	0.8622951	0.000
Breiman	2	96	0.6315574	0.8590164	0.000
Breiman	2	97	0.6344262	0.8491803	0.000
Breiman	2	98	0.6340164	0.8655738	0.000
Breiman	2	99	0.6217213	0.8590164	0.000
Breiman	2	100	0.6184426	0.8590164	0.125
Breiman	3	20	0.6381148	0.8590164	0.125
Breiman	3	21	0.6266393	0.8688525	0.125
Breiman	3	22	0.6245902	0.8721311	0.250
Breiman	3	23	0.6209016	0.8885246	0.125
Breiman	3	24	0.6069672	0.8852459	0.125
Breiman	3	25	0.5959016	0.8819672	0.125
Breiman	3	26	0.6086066	0.8950820	0.125
Breiman	3	27	0.6040984	0.8819672	0.125
Breiman	3	28	0.6081967	0.8754098	0.125
Breiman	3	29	0.6135246	0.8786885	0.125
Breiman	3	30	0.6094262	0.8885246	0.125
Breiman	3	31	0.5852459	0.8819672	0.125
Breiman	3	32	0.6081967	0.8819672	0.125
Breiman	3	33	0.6204918	0.8754098	0.125
Breiman	3	34	0.6204918	0.8721311	0.125
Breiman	3	35	0.6336066	0.8590164	0.125
Breiman	3	36	0.6254098	0.8721311	0.125
Breiman	3	37	0.6192623	0.8721311	0.125

Breiman	3	38	0.6163934	0.8721311	0.125
Breiman	3	39	0.6077869	0.8622951	0.125
Breiman	3	40	0.6028689	0.8622951	0.125
Breiman	3	41	0.6008197	0.8590164	0.125
Breiman	3	42	0.6008197	0.8688525	0.125
Breiman	3	43	0.6008197	0.8721311	0.125
Breiman	3	44	0.5979508	0.8688525	0.125
Breiman	3	45	0.5983607	0.8754098	0.125
Breiman	3	46	0.5926230	0.8622951	0.125
Breiman	3	47	0.5926230	0.8655738	0.125
Breiman	3	48	0.5987705	0.8819672	0.125
Breiman	3	49	0.5811475	0.8655738	0.125
Breiman	3	50	0.5725410	0.8655738	0.125
Breiman	3	51	0.5872951	0.8622951	0.125
Breiman	3	52	0.6004098	0.8655738	0.125
Breiman	3	53	0.6036885	0.8622951	0.125
Breiman	3	54	0.6012295	0.8655738	0.125
Breiman	3	55	0.6053279	0.8590164	0.125
Breiman	3	56	0.5946721	0.8688525	0.125
Breiman	3	57	0.5938525	0.8655738	0.125
Breiman	3	58	0.6155738	0.8655738	0.125
Breiman	3	59	0.6028689	0.8590164	0.125
Breiman	3	60	0.6139344	0.8688525	0.125
Breiman	3	61	0.6286885	0.8655738	0.125
Breiman	3	62	0.6282787	0.8754098	0.125
Breiman	3	63	0.6295082	0.8754098	0.125
Breiman	3	64	0.6336066	0.8655738	0.125
Breiman	3	65	0.6151639	0.8819672	0.125
Breiman	3	66	0.6102459	0.8852459	0.125
Breiman	3	67	0.6061475	0.8852459	0.125
Breiman	3	68	0.6098361	0.8786885	0.125
Breiman	3	69	0.6139344	0.8885246	0.125
Breiman	3	70	0.6274590	0.8885246	0.125
Breiman	3	71	0.6315574	0.8885246	0.125
Breiman	3	72	0.6245902	0.8721311	0.125
Breiman	3	73	0.6385246	0.8721311	0.125
Breiman	3	74	0.6327869	0.8688525	0.125
Breiman	3	75	0.6413934	0.8754098	0.125
Breiman	3	76	0.6348361	0.8688525	0.125
Breiman	3	77	0.6315574	0.8688525	0.125
Breiman	3	78	0.6311475	0.8721311	0.125
Breiman	3	79	0.6397541	0.8786885	0.125
Breiman	3	80	0.6389344	0.8754098	0.125
Breiman	3	81	0.6397541	0.8754098	0.125
Breiman	3	82	0.6430328	0.8721311	0.125
Breiman	3	83	0.6364754	0.8688525	0.125
Breiman	3	84	0.6356557	0.8688525	0.125
Breiman	3	85	0.6356557	0.8721311	0.125

Breiman	3	86	0.6315574	0.8754098	0.125
Breiman	3	87	0.6315574	0.8786885	0.125
Breiman	3	88	0.6377049	0.8819672	0.125
Breiman	3	89	0.6344262	0.8885246	0.125
Breiman	3	90	0.6405738	0.8819672	0.125
Breiman	3	91	0.6397541	0.8885246	0.125
Breiman	3	92	0.6430328	0.8819672	0.125
Breiman	3	93	0.6430328	0.8786885	0.125
Breiman	3	94	0.6401639	0.8852459	0.125
Breiman	3	95	0.6495902	0.8950820	0.125
Breiman	3	96	0.6528689	0.8918033	0.125
Breiman	3	97	0.6397541	0.8885246	0.125
Breiman	3	98	0.6389344	0.8819672	0.125
Breiman	3	99	0.6409836	0.8721311	0.125
Breiman	3	100	0.6446721	0.8852459	0.125
Breiman	4	20	0.6682377	0.8852459	0.125
Breiman	4	21	0.6680328	0.8819672	0.250
Breiman	4	22	0.6725410	0.8819672	0.125
Breiman	4	23	0.6881148	0.8622951	0.250
Breiman	4	24	0.6963115	0.8786885	0.125
Breiman	4	25	0.6934426	0.8918033	0.250
Breiman	4	26	0.6987705	0.8918033	0.125
Breiman	4	27	0.7057377	0.9114754	0.375
Breiman	4	28	0.7086066	0.8950820	0.375
Breiman	4	29	0.7176230	0.9016393	0.375
Breiman	4	30	0.7229508	0.9049180	0.375
Breiman	4	31	0.7237705	0.9114754	0.375
Breiman	4	32	0.7229508	0.9114754	0.375
Breiman	4	33	0.7106557	0.9016393	0.375
Breiman	4	34	0.7237705	0.8885246	0.375
Breiman	4	35	0.7098361	0.8918033	0.375
Breiman	4	36	0.7057377	0.9081967	0.250
Breiman	4	37	0.6934426	0.8983607	0.250
Breiman	4	38	0.6913934	0.8918033	0.250
Breiman	4	39	0.6909836	0.8983607	0.250
Breiman	4	40	0.6950820	0.8950820	0.250
Breiman	4	41	0.6885246	0.8918033	0.250
Breiman	4	42	0.6868852	0.9016393	0.375
Breiman	4	43	0.6774590	0.9049180	0.375
Breiman	4	44	0.6762295	0.9114754	0.375
Breiman	4	45	0.6737705	0.9081967	0.375
Breiman	4	46	0.6745902	0.9016393	0.375
Breiman	4	47	0.6823770	0.9016393	0.375
Breiman	4	48	0.6782787	0.9049180	0.250
Breiman	4	49	0.6766393	0.9049180	0.250
Breiman	4	50	0.6655738	0.9049180	0.250
Breiman	4	51	0.6721311	0.8983607	0.250
Breiman	4	52	0.6610656	0.9049180	0.250

Breiman	4	53	0.6565574	0.9081967	0.375
Breiman	4	54	0.6475410	0.8950820	0.375
Breiman	4	55	0.6553279	0.8983607	0.375
Breiman	4	56	0.6508197	0.8950820	0.375
Breiman	4	57	0.6434426	0.8918033	0.375
Breiman	4	58	0.6397541	0.8950820	0.375
Breiman	4	59	0.6540984	0.8918033	0.250
Breiman	4	60	0.6549180	0.8918033	0.375
Breiman	4	61	0.6557377	0.8950820	0.250
Breiman	4	62	0.6553279	0.8983607	0.250
Breiman	4	63	0.6704918	0.9016393	0.250
Breiman	4	64	0.6762295	0.9016393	0.250
Breiman	4	65	0.6676230	0.9049180	0.250
Breiman	4	66	0.6545082	0.8983607	0.250
Breiman	4	67	0.6553279	0.9016393	0.250
Breiman	4	68	0.6438525	0.8983607	0.250
Breiman	4	69	0.6508197	0.8950820	0.250
Breiman	4	70	0.6508197	0.9016393	0.250
Breiman	4	71	0.6418033	0.8950820	0.125
Breiman	4	72	0.6516393	0.8983607	0.125
Breiman	4	73	0.6471311	0.8983607	0.125
Breiman	4	74	0.6495902	0.8950820	0.250
Breiman	4	75	0.6545082	0.9016393	0.375
Breiman	4	76	0.6450820	0.9016393	0.375
Breiman	4	77	0.6344262	0.8983607	0.375
Breiman	4	78	0.6463115	0.9016393	0.250
Breiman	4	79	0.6426230	0.9016393	0.250
Breiman	4	80	0.6397541	0.9049180	0.250
Breiman	4	81	0.6438525	0.8983607	0.250
Breiman	4	82	0.6401639	0.9016393	0.250
Breiman	4	83	0.6327869	0.8918033	0.250
Breiman	4	84	0.6471311	0.8983607	0.250
Breiman	4	85	0.6483607	0.8950820	0.250
Breiman	4	86	0.6475410	0.9016393	0.250
Breiman	4	87	0.6536885	0.8950820	0.125
Breiman	4	88	0.6360656	0.9016393	0.125
Breiman	4	89	0.6344262	0.8983607	0.125
Breiman	4	90	0.6344262	0.9016393	0.125
Breiman	4	91	0.6315574	0.8983607	0.125
Breiman	4	92	0.6262295	0.9049180	0.125
Breiman	4	93	0.6155738	0.8983607	0.125
Breiman	4	94	0.6245902	0.8983607	0.125
Breiman	4	95	0.6135246	0.8983607	0.125
Breiman	4	96	0.6221311	0.8983607	0.125
Breiman	4	97	0.6241803	0.9016393	0.125
Breiman	4	98	0.6184426	0.9016393	0.125
Breiman	4	99	0.6217213	0.9016393	0.125
Breiman	4	100	0.6352459	0.8983607	0.125

Freund	2	20	0.6084016	0.8360656	0.125
Freund	2	21	0.6297131	0.8327869	0.125
Freund	2	22	0.6385246	0.8327869	0.125
Freund	2	23	0.6688525	0.8393443	0.125
Freund	2	24	0.6520492	0.8557377	0.125
Freund	2	25	0.6356557	0.8360656	0.125
Freund	2	26	0.5954918	0.8360656	0.250
Freund	2	27	0.5954918	0.8459016	0.125
Freund	2	28	0.5983607	0.8524590	0.125
Freund	2	29	0.5827869	0.8327869	0.125
Freund	2	30	0.5852459	0.8295082	0.125
Freund	2	31	0.5766393	0.8459016	0.125
Freund	2	32	0.5721311	0.8327869	0.125
Freund	2	33	0.5725410	0.8459016	0.125
Freund	2	34	0.5696721	0.8459016	0.125
Freund	2	35	0.5618852	0.8393443	0.125
Freund	2	36	0.5668033	0.8295082	0.250
Freund	2	37	0.6237705	0.8491803	0.125
Freund	2	38	0.6377049	0.8393443	0.250
Freund	2	39	0.6368852	0.8393443	0.125
Freund	2	40	0.6438525	0.8327869	0.250
Freund	2	41	0.6463115	0.8426230	0.125
Freund	2	42	0.6422131	0.8262295	0.250
Freund	2	43	0.6418033	0.8295082	0.250
Freund	2	44	0.6446721	0.8295082	0.250
Freund	2	45	0.6438525	0.8327869	0.250
Freund	2	46	0.6098361	0.8327869	0.250
Freund	2	47	0.6581967	0.8327869	0.250
Freund	2	48	0.6434426	0.8393443	0.250
Freund	2	49	0.6418033	0.8426230	0.250
Freund	2	50	0.6430328	0.8393443	0.250
Freund	2	51	0.6512295	0.8360656	0.250
Freund	2	52	0.6389344	0.8295082	0.250
Freund	2	53	0.6393443	0.8360656	0.250
Freund	2	54	0.6610656	0.8459016	0.125
Freund	2	55	0.6659836	0.8393443	0.125
Freund	2	56	0.6745902	0.8393443	0.125
Freund	2	57	0.6668033	0.8360656	0.250
Freund	2	58	0.6586066	0.8360656	0.250
Freund	2	59	0.6536885	0.8360656	0.250
Freund	2	60	0.6618852	0.8327869	0.250
Freund	2	61	0.6647541	0.8360656	0.250
Freund	2	62	0.6684426	0.8426230	0.125
Freund	2	63	0.6606557	0.8459016	0.250
Freund	2	64	0.6450820	0.8393443	0.125
Freund	2	65	0.6536885	0.8393443	0.250
Freund	2	66	0.6553279	0.8426230	0.125
Freund	2	67	0.6504098	0.8622951	0.125

Freund	2	68	0.6553279	0.8524590	0.125
Freund	2	69	0.6413934	0.8459016	0.250
Freund	2	70	0.6446721	0.8426230	0.250
Freund	2	71	0.6479508	0.8524590	0.250
Freund	2	72	0.6483607	0.8360656	0.250
Freund	2	73	0.6512295	0.8557377	0.250
Freund	2	74	0.6385246	0.8491803	0.125
Freund	2	75	0.6389344	0.8426230	0.250
Freund	2	76	0.6389344	0.8590164	0.125
Freund	2	77	0.6467213	0.8524590	0.250
Freund	2	78	0.6467213	0.8459016	0.250
Freund	2	79	0.6545082	0.8557377	0.125
Freund	2	80	0.6573770	0.8557377	0.375
Freund	2	81	0.6545082	0.8524590	0.375
Freund	2	82	0.6528689	0.8557377	0.125
Freund	2	83	0.6491803	0.8524590	0.250
Freund	2	84	0.6438525	0.8491803	0.125
Freund	2	85	0.6278689	0.8557377	0.250
Freund	2	86	0.6311475	0.8557377	0.250
Freund	2	87	0.6401639	0.8524590	0.250
Freund	2	88	0.6393443	0.8557377	0.250
Freund	2	89	0.6483607	0.8524590	0.250
Freund	2	90	0.6487705	0.8426230	0.250
Freund	2	91	0.6454918	0.8459016	0.250
Freund	2	92	0.6491803	0.8393443	0.250
Freund	2	93	0.6516393	0.8459016	0.250
Freund	2	94	0.6446721	0.8491803	0.250
Freund	2	95	0.6278689	0.8491803	0.250
Freund	2	96	0.6274590	0.8557377	0.250
Freund	2	97	0.6393443	0.8459016	0.250
Freund	2	98	0.6389344	0.8491803	0.250
Freund	2	99	0.6352459	0.8524590	0.250
Freund	2	100	0.6368852	0.8557377	0.250
Freund	3	20	0.7442623	0.8819672	0.250
Freund	3	21	0.7395492	0.8590164	0.250
Freund	3	22	0.7479508	0.8688525	0.375
Freund	3	23	0.7668033	0.8655738	0.375
Freund	3	24	0.7565574	0.8721311	0.250
Freund	3	25	0.7471311	0.8688525	0.250
Freund	3	26	0.7241803	0.8590164	0.250
Freund	3	27	0.7213115	0.8819672	0.125
Freund	3	28	0.7161885	0.8786885	0.125
Freund	3	29	0.7247951	0.8655738	0.125
Freund	3	30	0.7360656	0.8819672	0.125
Freund	3	31	0.7237705	0.8721311	0.125
Freund	3	32	0.6852459	0.8655738	0.125
Freund	3	33	0.6610656	0.8721311	0.125
Freund	3	34	0.6479508	0.8655738	0.125

Freund	3	35	0.6524590	0.8721311	0.125
Freund	3	36	0.6524590	0.8655738	0.125
Freund	3	37	0.6565574	0.8786885	0.125
Freund	3	38	0.6393443	0.8754098	0.125
Freund	3	39	0.6442623	0.8786885	0.125
Freund	3	40	0.6639344	0.8557377	0.125
Freund	3	41	0.6680328	0.8557377	0.125
Freund	3	42	0.6807377	0.8655738	0.125
Freund	3	43	0.6745902	0.8655738	0.125
Freund	3	44	0.6786885	0.8622951	0.125
Freund	3	45	0.6709016	0.8491803	0.250
Freund	3	46	0.7127049	0.8622951	0.250
Freund	3	47	0.7122951	0.8688525	0.250
Freund	3	48	0.7094262	0.8819672	0.250
Freund	3	49	0.7258197	0.8786885	0.250
Freund	3	50	0.6954918	0.8721311	0.125
Freund	3	51	0.6950820	0.8688525	0.250
Freund	3	52	0.6959016	0.8754098	0.125
Freund	3	53	0.7118852	0.8786885	0.250
Freund	3	54	0.7143443	0.8721311	0.250
Freund	3	55	0.7077869	0.8786885	0.250
Freund	3	56	0.7176230	0.8721311	0.250
Freund	3	57	0.7200820	0.8786885	0.250
Freund	3	58	0.7135246	0.8786885	0.375
Freund	3	59	0.6983607	0.8754098	0.250
Freund	3	60	0.6922131	0.8655738	0.250
Freund	3	61	0.7016393	0.8655738	0.250
Freund	3	62	0.7036885	0.8754098	0.250
Freund	3	63	0.6872951	0.8688525	0.250
Freund	3	64	0.6795082	0.8688525	0.250
Freund	3	65	0.6729508	0.8557377	0.250
Freund	3	66	0.6831967	0.8622951	0.250
Freund	3	67	0.6729508	0.8524590	0.250
Freund	3	68	0.6663934	0.8557377	0.250
Freund	3	69	0.6520492	0.8524590	0.250
Freund	3	70	0.6520492	0.8688525	0.250
Freund	3	71	0.6504098	0.8524590	0.250
Freund	3	72	0.6450820	0.8590164	0.250
Freund	3	73	0.6385246	0.8524590	0.250
Freund	3	74	0.6430328	0.8524590	0.250
Freund	3	75	0.6442623	0.8655738	0.250
Freund	3	76	0.6118852	0.8557377	0.250
Freund	3	77	0.6061475	0.8622951	0.250
Freund	3	78	0.6094262	0.8557377	0.250
Freund	3	79	0.6065574	0.8590164	0.250
Freund	3	80	0.6143443	0.8622951	0.250
Freund	3	81	0.6077869	0.8557377	0.250
Freund	3	82	0.6081967	0.8655738	0.250

Freund	3	83	0.6106557	0.8622951	0.250
Freund	3	84	0.6098361	0.8655738	0.250
Freund	3	85	0.6172131	0.8590164	0.250
Freund	3	86	0.6196721	0.8524590	0.250
Freund	3	87	0.6188525	0.8590164	0.250
Freund	3	88	0.6168033	0.8655738	0.250
Freund	3	89	0.6176230	0.8655738	0.250
Freund	3	90	0.6180328	0.8590164	0.250
Freund	3	91	0.6225410	0.8557377	0.250
Freund	3	92	0.6204918	0.8622951	0.250
Freund	3	93	0.6217213	0.8590164	0.250
Freund	3	94	0.6139344	0.8557377	0.250
Freund	3	95	0.6139344	0.8590164	0.250
Freund	3	96	0.6163934	0.8557377	0.250
Freund	3	97	0.6237705	0.8524590	0.250
Freund	3	98	0.6254098	0.8557377	0.250
Freund	3	99	0.6250000	0.8491803	0.250
Freund	3	100	0.6237705	0.8590164	0.250
Freund	4	20	0.6028689	0.8524590	0.125
Freund	4	21	0.6024590	0.8655738	0.125
Freund	4	22	0.6131148	0.8819672	0.125
Freund	4	23	0.6200820	0.8655738	0.125
Freund	4	24	0.6131148	0.8622951	0.125
Freund	4	25	0.6114754	0.8819672	0.125
Freund	4	26	0.5909836	0.8852459	0.125
Freund	4	27	0.5901639	0.8819672	0.125
Freund	4	28	0.5782787	0.8622951	0.125
Freund	4	29	0.5807377	0.8754098	0.125
Freund	4	30	0.6118852	0.8754098	0.125
Freund	4	31	0.5860656	0.8754098	0.125
Freund	4	32	0.6045082	0.8918033	0.125
Freund	4	33	0.5766393	0.8721311	0.125
Freund	4	34	0.5860656	0.8754098	0.125
Freund	4	35	0.5918033	0.8786885	0.125
Freund	4	36	0.5811475	0.8918033	0.125
Freund	4	37	0.5758197	0.8754098	0.125
Freund	4	38	0.5778689	0.8852459	0.125
Freund	4	39	0.5659836	0.8754098	0.125
Freund	4	40	0.5799180	0.8786885	0.125
Freund	4	41	0.6016393	0.8721311	0.125
Freund	4	42	0.5971311	0.8819672	0.125
Freund	4	43	0.5950820	0.8721311	0.125
Freund	4	44	0.6012295	0.8721311	0.125
Freund	4	45	0.6237705	0.8754098	0.125
Freund	4	46	0.6327869	0.8819672	0.125
Freund	4	47	0.6237705	0.8786885	0.125
Freund	4	48	0.6254098	0.8721311	0.125
Freund	4	49	0.6139344	0.8786885	0.125

Freund	4	50	0.5938525	0.8852459	0.125
Freund	4	51	0.5877049	0.8918033	0.125
Freund	4	52	0.5655738	0.8983607	0.125
Freund	4	53	0.5762295	0.8885246	0.125
Freund	4	54	0.5864754	0.8918033	0.125
Freund	4	55	0.5954918	0.8819672	0.125
Freund	4	56	0.6131148	0.8983607	0.125
Freund	4	57	0.6118852	0.8754098	0.125
Freund	4	58	0.5922131	0.8983607	0.125
Freund	4	59	0.5885246	0.8754098	0.250
Freund	4	60	0.5823770	0.8819672	0.250
Freund	4	61	0.5881148	0.8819672	0.250
Freund	4	62	0.5938525	0.8950820	0.250
Freund	4	63	0.5926230	0.8852459	0.250
Freund	4	64	0.5860656	0.8852459	0.250
Freund	4	65	0.5963115	0.8885246	0.250
Freund	4	66	0.6000000	0.8819672	0.250
Freund	4	67	0.6118852	0.8918033	0.125
Freund	4	68	0.6102459	0.8918033	0.125
Freund	4	69	0.6069672	0.8918033	0.250
Freund	4	70	0.6073770	0.8852459	0.125
Freund	4	71	0.6106557	0.8918033	0.125
Freund	4	72	0.6053279	0.8885246	0.125
Freund	4	73	0.6102459	0.8885246	0.125
Freund	4	74	0.6106557	0.8950820	0.125
Freund	4	75	0.6057377	0.8918033	0.125
Freund	4	76	0.6073770	0.8918033	0.250
Freund	4	77	0.6118852	0.8950820	0.125
Freund	4	78	0.6106557	0.8983607	0.125
Freund	4	79	0.6045082	0.8950820	0.125
Freund	4	80	0.5954918	0.9049180	0.125
Freund	4	81	0.5901639	0.8950820	0.125
Freund	4	82	0.5918033	0.8983607	0.125
Freund	4	83	0.5897541	0.8950820	0.125
Freund	4	84	0.5918033	0.9016393	0.125
Freund	4	85	0.5995902	0.9016393	0.125
Freund	4	86	0.6004098	0.8983607	0.125
Freund	4	87	0.5987705	0.9049180	0.125
Freund	4	88	0.6045082	0.9016393	0.250
Freund	4	89	0.5979508	0.9049180	0.250
Freund	4	90	0.5913934	0.8983607	0.250
Freund	4	91	0.5963115	0.9081967	0.250
Freund	4	92	0.6053279	0.9049180	0.250
Freund	4	93	0.6028689	0.9049180	0.250
Freund	4	94	0.6036885	0.8983607	0.250
Freund	4	95	0.6016393	0.9081967	0.250
Freund	4	96	0.5995902	0.8950820	0.250
Freund	4	97	0.6040984	0.9081967	0.250

Freund	4	98	0.6106557	0.8983607	0.250
Freund	4	99	0.6163934	0.9147541	0.250
Freund	4	100	0.6209016	0.9016393	0.250
Zhu	2	20	0.5911885	0.8557377	0.125
Zhu	2	21	0.5956967	0.8229508	0.375
Zhu	2	22	0.6010246	0.8557377	0.250
Zhu	2	23	0.6063525	0.8426230	0.250
Zhu	2	24	0.6038934	0.8426230	0.375
Zhu	2	25	0.6116803	0.8327869	0.250
Zhu	2	26	0.6104508	0.8196721	0.375
Zhu	2	27	0.6022541	0.8229508	0.250
Zhu	2	28	0.6063525	0.8426230	0.250
Zhu	2	29	0.6172131	0.8360656	0.375
Zhu	2	30	0.6368852	0.8295082	0.375
Zhu	2	31	0.6303279	0.8163934	0.375
Zhu	2	32	0.6098361	0.8262295	0.375
Zhu	2	33	0.6319672	0.8360656	0.250
Zhu	2	34	0.6454918	0.8327869	0.375
Zhu	2	35	0.6372951	0.8262295	0.250
Zhu	2	36	0.6750000	0.8163934	0.375
Zhu	2	37	0.6803279	0.8262295	0.375
Zhu	2	38	0.6815574	0.8327869	0.375
Zhu	2	39	0.6786885	0.8393443	0.375
Zhu	2	40	0.6852459	0.8262295	0.375
Zhu	2	41	0.6799180	0.8229508	0.375
Zhu	2	42	0.6815574	0.8426230	0.375
Zhu	2	43	0.6848361	0.8426230	0.375
Zhu	2	44	0.6926230	0.8459016	0.375
Zhu	2	45	0.6959016	0.8393443	0.375
Zhu	2	46	0.7045082	0.8360656	0.375
Zhu	2	47	0.6954918	0.8295082	0.375
Zhu	2	48	0.6938525	0.8459016	0.375
Zhu	2	49	0.6995902	0.8393443	0.375
Zhu	2	50	0.6959016	0.8327869	0.375
Zhu	2	51	0.6946721	0.8360656	0.375
Zhu	2	52	0.6967213	0.8295082	0.375
Zhu	2	53	0.7004098	0.8459016	0.375
Zhu	2	54	0.7143443	0.8393443	0.375
Zhu	2	55	0.7188525	0.8491803	0.375
Zhu	2	56	0.7163934	0.8360656	0.375
Zhu	2	57	0.7139344	0.8491803	0.375
Zhu	2	58	0.7209016	0.8491803	0.375
Zhu	2	59	0.7217213	0.8360656	0.500
Zhu	2	60	0.7225410	0.8295082	0.500
Zhu	2	61	0.7176230	0.8393443	0.375
Zhu	2	62	0.7307377	0.8393443	0.500
Zhu	2	63	0.7245902	0.8360656	0.500
Zhu	2	64	0.7168033	0.8360656	0.500

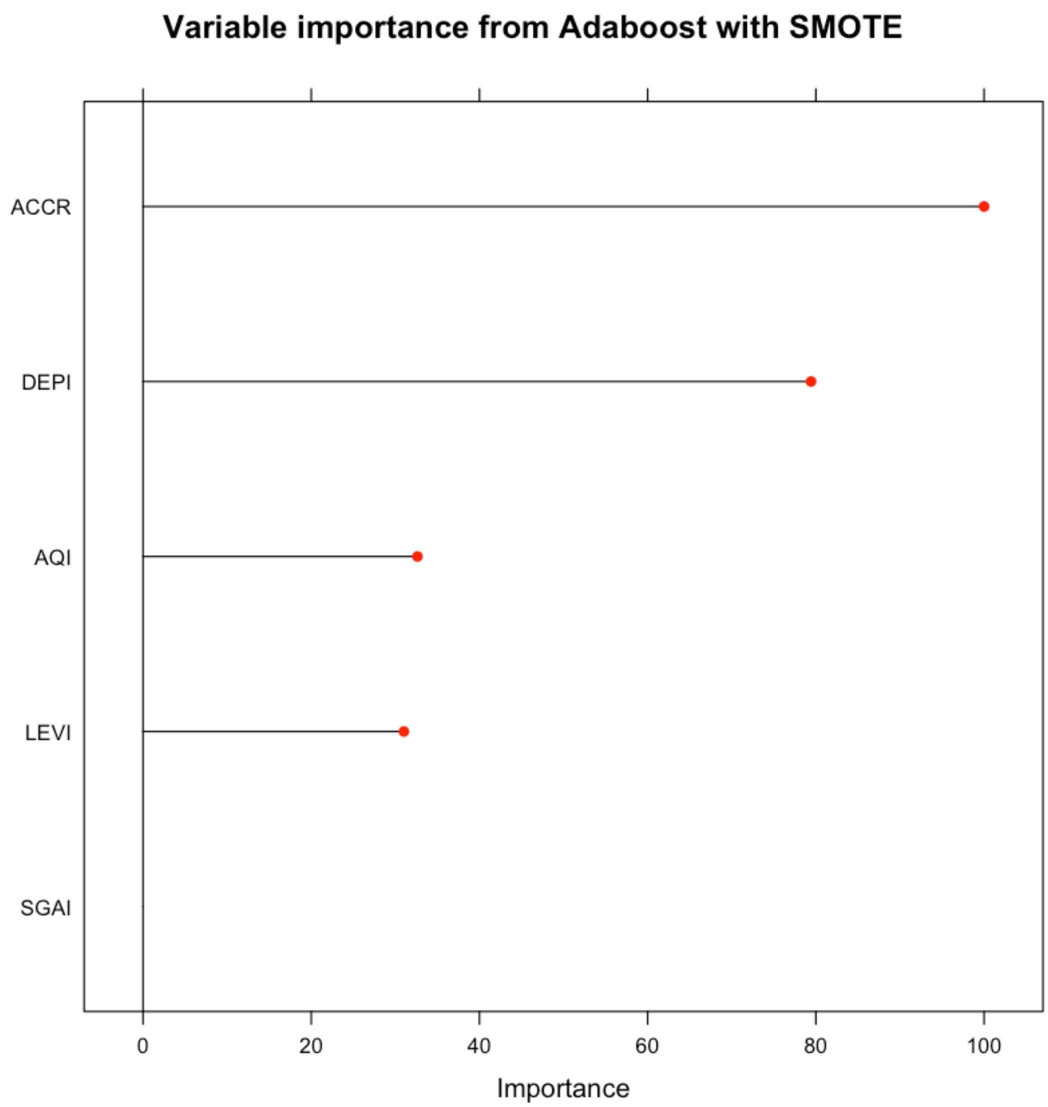
Zhu	2	65	0.6950820	0.8229508	0.375
Zhu	2	66	0.6852459	0.8163934	0.375
Zhu	2	67	0.6889344	0.8196721	0.375
Zhu	2	68	0.7069672	0.8196721	0.375
Zhu	2	69	0.7364754	0.8196721	0.375
Zhu	2	70	0.7397541	0.8163934	0.375
Zhu	2	71	0.7483607	0.8163934	0.500
Zhu	2	72	0.7459016	0.8196721	0.375
Zhu	2	73	0.7446721	0.8327869	0.375
Zhu	2	74	0.7442623	0.8295082	0.500
Zhu	2	75	0.7450820	0.8459016	0.375
Zhu	2	76	0.7327869	0.8196721	0.500
Zhu	2	77	0.7315574	0.8262295	0.500
Zhu	2	78	0.7405738	0.8295082	0.500
Zhu	2	79	0.7385246	0.8327869	0.500
Zhu	2	80	0.7295082	0.8459016	0.500
Zhu	2	81	0.7262295	0.8459016	0.500
Zhu	2	82	0.7229508	0.8459016	0.375
Zhu	2	83	0.7295082	0.8459016	0.500
Zhu	2	84	0.7303279	0.8426230	0.500
Zhu	2	85	0.7299180	0.8524590	0.500
Zhu	2	86	0.7229508	0.8426230	0.500
Zhu	2	87	0.7266393	0.8393443	0.375
Zhu	2	88	0.7352459	0.8229508	0.375
Zhu	2	89	0.7536885	0.8393443	0.500
Zhu	2	90	0.7491803	0.8426230	0.500
Zhu	2	91	0.7364754	0.8360656	0.500
Zhu	2	92	0.7315574	0.8262295	0.500
Zhu	2	93	0.7631148	0.8426230	0.500
Zhu	2	94	0.7622951	0.8557377	0.500
Zhu	2	95	0.7614754	0.8459016	0.500
Zhu	2	96	0.7622951	0.8491803	0.500
Zhu	2	97	0.7594262	0.8360656	0.500
Zhu	2	98	0.7540984	0.8393443	0.500
Zhu	2	99	0.7557377	0.8360656	0.500
Zhu	2	100	0.7553279	0.8590164	0.500
Zhu	3	20	0.6778689	0.8360656	0.375
Zhu	3	21	0.6827869	0.8196721	0.375
Zhu	3	22	0.6823770	0.8524590	0.375
Zhu	3	23	0.6811475	0.8459016	0.375
Zhu	3	24	0.6762295	0.8622951	0.375
Zhu	3	25	0.6737705	0.8557377	0.375
Zhu	3	26	0.6987705	0.8524590	0.375
Zhu	3	27	0.7098361	0.8557377	0.375
Zhu	3	28	0.7303279	0.8491803	0.375
Zhu	3	29	0.7241803	0.8459016	0.375
Zhu	3	30	0.7377049	0.8459016	0.375
Zhu	3	31	0.7397541	0.8491803	0.375

Zhu	3	32	0.7184426	0.8393443	0.375
Zhu	3	33	0.7266393	0.8557377	0.375
Zhu	3	34	0.7200820	0.8491803	0.375
Zhu	3	35	0.6979508	0.8557377	0.375
Zhu	3	36	0.6971311	0.8491803	0.375
Zhu	3	37	0.6901639	0.8557377	0.375
Zhu	3	38	0.6926230	0.8393443	0.375
Zhu	3	39	0.6877049	0.8524590	0.375
Zhu	3	40	0.6938525	0.8557377	0.375
Zhu	3	41	0.7233607	0.8590164	0.375
Zhu	3	42	0.7061475	0.8491803	0.375
Zhu	3	43	0.7233607	0.8524590	0.375
Zhu	3	44	0.7237705	0.8622951	0.375
Zhu	3	45	0.7192623	0.8590164	0.375
Zhu	3	46	0.7135246	0.8557377	0.375
Zhu	3	47	0.7364754	0.8655738	0.375
Zhu	3	48	0.7438525	0.8557377	0.375
Zhu	3	49	0.7471311	0.8590164	0.375
Zhu	3	50	0.7352459	0.8491803	0.375
Zhu	3	51	0.7372951	0.8590164	0.375
Zhu	3	52	0.7381148	0.8655738	0.375
Zhu	3	53	0.7233607	0.8622951	0.375
Zhu	3	54	0.7258197	0.8622951	0.375
Zhu	3	55	0.7290984	0.8557377	0.375
Zhu	3	56	0.7360656	0.8721311	0.375
Zhu	3	57	0.7360656	0.8721311	0.375
Zhu	3	58	0.7385246	0.8622951	0.375
Zhu	3	59	0.7344262	0.8557377	0.375
Zhu	3	60	0.7348361	0.8655738	0.375
Zhu	3	61	0.7274590	0.8655738	0.375
Zhu	3	62	0.7266393	0.8688525	0.375
Zhu	3	63	0.7196721	0.8622951	0.375
Zhu	3	64	0.7155738	0.8590164	0.375
Zhu	3	65	0.7229508	0.8655738	0.375
Zhu	3	66	0.7270492	0.8524590	0.375
Zhu	3	67	0.7258197	0.8622951	0.375
Zhu	3	68	0.7225410	0.8491803	0.375
Zhu	3	69	0.7225410	0.8655738	0.375
Zhu	3	70	0.7168033	0.8524590	0.375
Zhu	3	71	0.7143443	0.8524590	0.375
Zhu	3	72	0.7180328	0.8524590	0.375
Zhu	3	73	0.7163934	0.8524590	0.375
Zhu	3	74	0.7139344	0.8557377	0.375
Zhu	3	75	0.7139344	0.8622951	0.375
Zhu	3	76	0.7114754	0.8622951	0.375
Zhu	3	77	0.7036885	0.8590164	0.375
Zhu	3	78	0.7020492	0.8655738	0.375
Zhu	3	79	0.7053279	0.8590164	0.375

Zhu	3	80	0.7118852	0.8590164	0.375
Zhu	3	81	0.7061475	0.8655738	0.375
Zhu	3	82	0.7188525	0.8557377	0.375
Zhu	3	83	0.7102459	0.8622951	0.375
Zhu	3	84	0.7081967	0.8590164	0.375
Zhu	3	85	0.7192623	0.8524590	0.375
Zhu	3	86	0.7192623	0.8557377	0.375
Zhu	3	87	0.7188525	0.8557377	0.375
Zhu	3	88	0.7069672	0.8557377	0.375
Zhu	3	89	0.7065574	0.8557377	0.375
Zhu	3	90	0.7094262	0.8557377	0.375
Zhu	3	91	0.7151639	0.8491803	0.375
Zhu	3	92	0.7180328	0.8524590	0.375
Zhu	3	93	0.7245902	0.8557377	0.375
Zhu	3	94	0.7245902	0.8557377	0.375
Zhu	3	95	0.7221311	0.8524590	0.375
Zhu	3	96	0.7168033	0.8459016	0.375
Zhu	3	97	0.7168033	0.8459016	0.375
Zhu	3	98	0.7139344	0.8491803	0.375
Zhu	3	99	0.7151639	0.8557377	0.375
Zhu	3	100	0.7196721	0.8491803	0.375
Zhu	4	20	0.6426230	0.8491803	0.250
Zhu	4	21	0.6463115	0.8491803	0.250
Zhu	4	22	0.6557377	0.8393443	0.250
Zhu	4	23	0.6159836	0.8524590	0.250
Zhu	4	24	0.6143443	0.8524590	0.250
Zhu	4	25	0.6225410	0.8622951	0.250
Zhu	4	26	0.6036885	0.8590164	0.250
Zhu	4	27	0.6028689	0.8524590	0.250
Zhu	4	28	0.6086066	0.8590164	0.250
Zhu	4	29	0.6163934	0.8459016	0.250
Zhu	4	30	0.6118852	0.8393443	0.250
Zhu	4	31	0.6139344	0.8491803	0.250
Zhu	4	32	0.6122951	0.8524590	0.250
Zhu	4	33	0.6086066	0.8491803	0.250
Zhu	4	34	0.6090164	0.8491803	0.250
Zhu	4	35	0.5905738	0.8524590	0.250
Zhu	4	36	0.6004098	0.8491803	0.250
Zhu	4	37	0.6032787	0.8393443	0.250
Zhu	4	38	0.6020492	0.8459016	0.250
Zhu	4	39	0.5877049	0.8491803	0.250
Zhu	4	40	0.5815574	0.8491803	0.250
Zhu	4	41	0.5844262	0.8524590	0.250
Zhu	4	42	0.5864754	0.8459016	0.250
Zhu	4	43	0.5807377	0.8524590	0.250
Zhu	4	44	0.5627049	0.8491803	0.250
Zhu	4	45	0.5614754	0.8491803	0.250
Zhu	4	46	0.5540984	0.8491803	0.250

Zhu	4	47	0.5680328	0.8491803	0.250
Zhu	4	48	0.5721311	0.8524590	0.250
Zhu	4	49	0.5762295	0.8491803	0.250
Zhu	4	50	0.5774590	0.8524590	0.250
Zhu	4	51	0.5741803	0.8524590	0.250
Zhu	4	52	0.5704918	0.8557377	0.250
Zhu	4	53	0.5713115	0.8622951	0.250
Zhu	4	54	0.5815574	0.8622951	0.250
Zhu	4	55	0.5815574	0.8590164	0.250
Zhu	4	56	0.5668033	0.8590164	0.250
Zhu	4	57	0.5663934	0.8590164	0.250
Zhu	4	58	0.5782787	0.8590164	0.250
Zhu	4	59	0.5737705	0.8557377	0.250
Zhu	4	60	0.5778689	0.8557377	0.250
Zhu	4	61	0.5754098	0.8557377	0.250
Zhu	4	62	0.5762295	0.8622951	0.250
Zhu	4	63	0.5676230	0.8754098	0.250
Zhu	4	64	0.5655738	0.8688525	0.250
Zhu	4	65	0.5610656	0.8688525	0.250
Zhu	4	66	0.5729508	0.8590164	0.250
Zhu	4	67	0.5774590	0.8491803	0.250
Zhu	4	68	0.5893443	0.8524590	0.250
Zhu	4	69	0.5836066	0.8524590	0.250
Zhu	4	70	0.5676230	0.8557377	0.250
Zhu	4	71	0.5668033	0.8557377	0.250
Zhu	4	72	0.5684426	0.8557377	0.250
Zhu	4	73	0.5819672	0.8590164	0.250
Zhu	4	74	0.5741803	0.8491803	0.250
Zhu	4	75	0.5766393	0.8524590	0.250
Zhu	4	76	0.5581967	0.8557377	0.250
Zhu	4	77	0.5672131	0.8557377	0.250
Zhu	4	78	0.5651639	0.8655738	0.250
Zhu	4	79	0.5602459	0.8590164	0.250
Zhu	4	80	0.5639344	0.8524590	0.250
Zhu	4	81	0.5602459	0.8557377	0.250
Zhu	4	82	0.5631148	0.8590164	0.250
Zhu	4	83	0.5631148	0.8590164	0.250
Zhu	4	84	0.5643443	0.8590164	0.250
Zhu	4	85	0.5643443	0.8622951	0.250
Zhu	4	86	0.5647541	0.8590164	0.250
Zhu	4	87	0.5590164	0.8557377	0.250
Zhu	4	88	0.5590164	0.8491803	0.250
Zhu	4	89	0.5500000	0.8524590	0.250
Zhu	4	90	0.5434426	0.8524590	0.250
Zhu	4	91	0.5651639	0.8524590	0.250
Zhu	4	92	0.5807377	0.8491803	0.250
Zhu	4	93	0.5704918	0.8524590	0.250
Zhu	4	94	0.5745902	0.8491803	0.250

```
  Zhu        4          95     0.5762295  0.8557377  0.250
  Zhu        4          96     0.5754098  0.8491803  0.250
  Zhu        4          97     0.5848361  0.8524590  0.250
  Zhu        4          98     0.5795082  0.8491803  0.250
  Zhu        4          99     0.5676230  0.8524590  0.250
  Zhu        4         100     0.5684426  0.8491803  0.250

ROC was used to select the optimal model using the largest value.
The final values used for the model were mfinal = 23, maxdepth = 3
 and coeflearn = Freund.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  84.3  1.6
       Yes 13.1  1.0

 Accuracy (average) : 0.853
```



Confusion Matrix for adaboost on test set

```
In [61]: caretPredictedClass <- predict(ada_smote_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  289   4
       Yes  71   7

               Accuracy : 0.7978
                 95% CI : (0.7533, 0.8375)
```

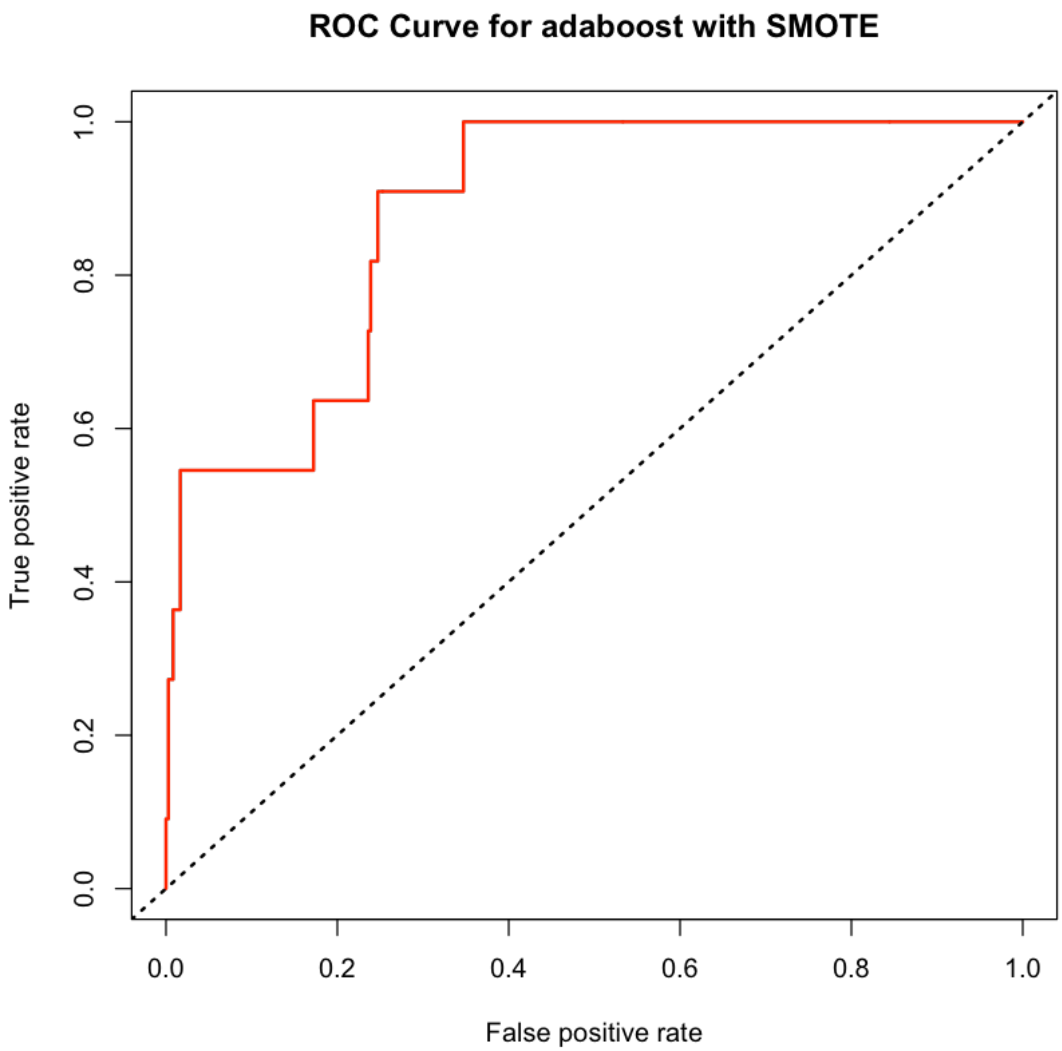
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1111
 Mcnemar's Test P-Value : 2.517e-14

            Sensitivity : 0.80278
            Specificity : 0.63636
         Pos Pred Value : 0.98635
         Neg Pred Value : 0.08974
             Prevalence : 0.97035
         Detection Rate : 0.77898
   Detection Prevalence : 0.78976
      Balanced Accuracy : 0.71957

       'Positive' Class : No
```

ROC plot for adaboost on test set

```
In [62]: ada_pred <- predict(ada_smote_model, model_test_df, type = "prob")[,2]
         ada_prediction <- prediction(ada_pred,model_test_df$Manipulater)
         ada_perf <- performance(ada_prediction, "tpr","fpr")

         plot(ada_perf,main="ROC Curve for adaboost with SMOTE",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(ada_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8828283
```

```
Slot "alpha.values":
list()
```



1.5.6 Boosting with xgboost (normal)

Look for the documentation of library **xgboost**. The **xgb.train()** function of xgboost implments 'xgbTree'(Default) and 'xgbLinear'.

1. Refer to know about the fine tuning parameters.
2. This can also be referred to know about the parameter fine tuning.

For xgbTree the fine tuning paramter consists of:

1. eta control the learning rate: scale the contribution of each tree by a factor of $0 <$ eta < 1 when it is added to the current approximation. Used to prevent overfitting by making the boosting process more conservative. Lower value for eta implies larger value for nrounds: low eta value means model more robust to overfitting but slower to compute. Default: 0.3
2. gamma minimum loss reduction required to make a further partition on a leaf node of the tree. the larger, the more conservative the algorithm will be.
3. max_depth maximum depth of a tree. Default: 6
4. min_child_weight minimum sum of instance weight(hessian) needed in a child. If the tree partition step results in a leaf node with the sum of instance weight less than min_child_weight, then the building process will give up further partitioning. In linear regression mode, this simply corresponds to minimum number of instances needed to be in each node. The larger, the more conservative the algorithm will be. Default: 1
5. subsample subsample ratio of the training instance. Setting it to 0.5 means that xgboost randomly collected half of the data instances to grow trees and this will prevent overfitting. It makes computation shorter (because less data to analyse). It is advised to use this parameter with eta and increase nround. Default: 1
6. colsample_bytree subsample ratio of columns when constructing each tree. Default: 1
7. num_parallel_tree Experimental parameter. number of trees to grow per round. Useful to test Random Forest through Xgboost (set colsample_bytree < 1, subsample < 1 and round $= 1$) accordingly. Default: 1

The below code chunk sets some of the control parameters for adaboost

```
In [63]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE)
```

```
In [64]: search_grid <- expand.grid(nrounds = c(70:150), max_depth = c(2:4),
                             eta = c(0.1,0.3,0.5),
                             gamma = c(0.03,0.09, 0.12),
                             colsample_bytree = c(5:10)/10,
                             min_child_weight = c(1:5),
                             subsample = c(0.5))
```

After setting the control paramters, the model is run

```
In [65]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Xtreme Boosting with Bootstrap Sampling")

         set.seed(4121)
         xg_model <- train(model_train_df[,1:5], model_train_df[,6],
```

```
                          method='xgbTree',
                          trControl=objControl,
                          tuneGrid = search_grid,
                          metric = "ROC")
        stopCluster(num_cores)
        toc()
```

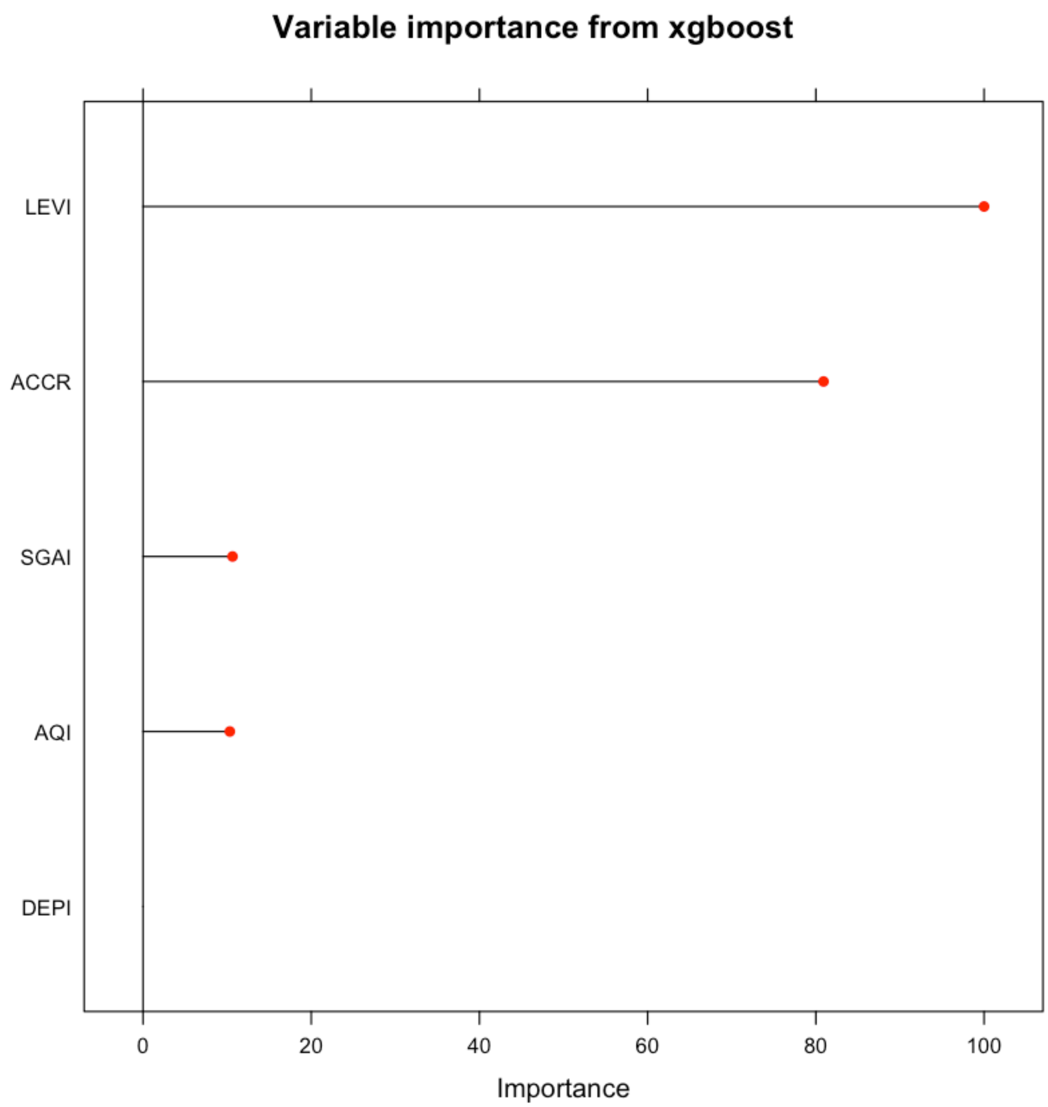
```
Xtreme Boosting with Bootstrap Sampling: 246.938 sec elapsed
```

Confusion Matrix for xgboost on train set

```
In [66]: xg_model$bestTune
         confusionMatrix.train(xg_model)

         plot(varImp(xg_model), main = "Variable importance from xgboost", col = 2, lwd = 2)
```

	nrounds	max_depth	eta	gamma	colsample_bytree	min_child_weight	subsample
16628	92	4	0.1	0.03	1	1	0.5

```
Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  97.1  2.6
       Yes  0.3  0.0

 Accuracy (average) : 0.9712
```



Confusion Matrix for xgboost on test set

```
In [67]: caretPredictedClass <- predict(xg_model, model_test_df[1:5], type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  359  10
       Yes   1   1

               Accuracy : 0.9704
                 95% CI : (0.9476, 0.9851)
```

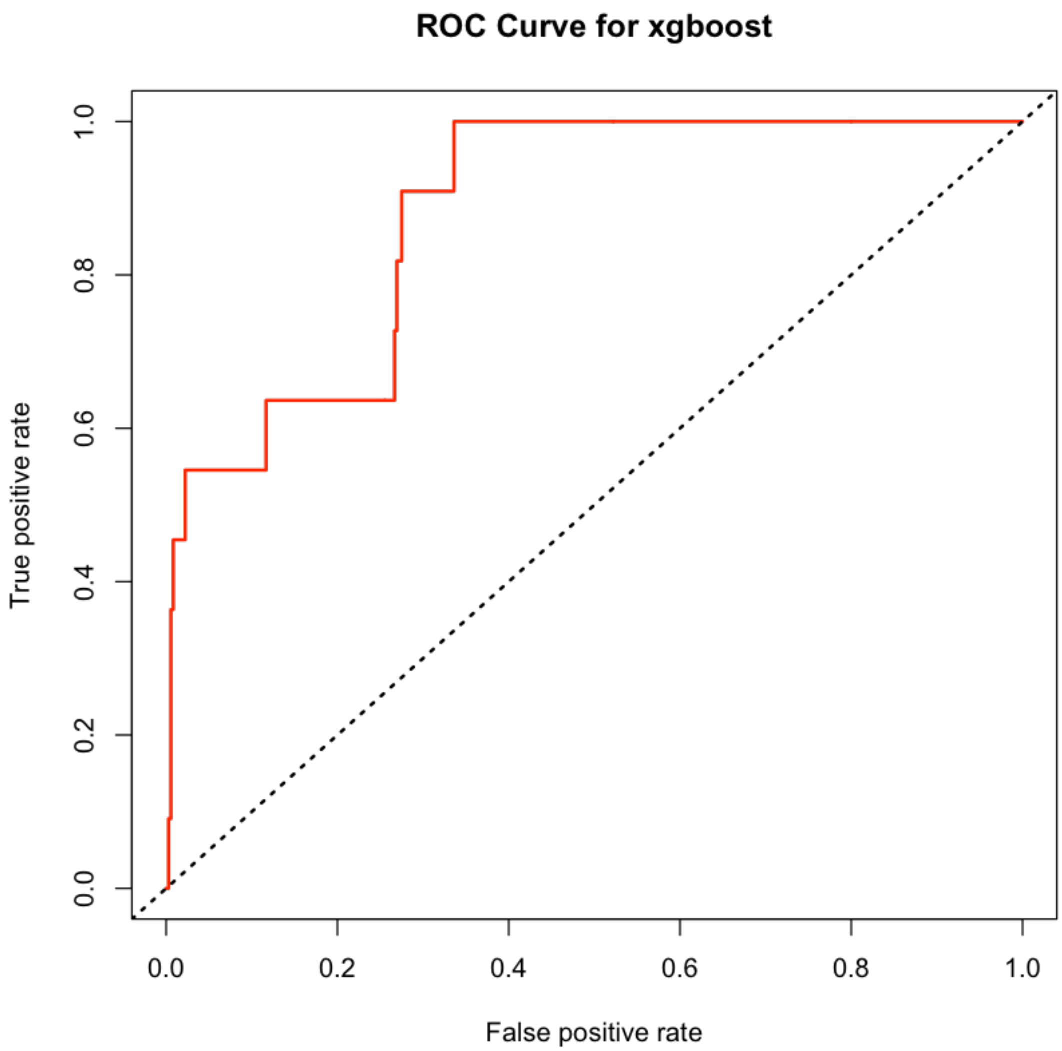
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.57928

                  Kappa : 0.1461
 Mcnemar's Test P-Value : 0.01586

            Sensitivity : 0.99722
            Specificity : 0.09091
         Pos Pred Value : 0.97290
         Neg Pred Value : 0.50000
             Prevalence : 0.97035
         Detection Rate : 0.96765
   Detection Prevalence : 0.99461
      Balanced Accuracy : 0.54407

       'Positive' Class : No
```

ROC plot for xgboost on test set

```
In [68]: xg_pred <- predict(xg_model, model_test_df[1:5], type = "prob")[,2]
         xg_prediction <- prediction(xg_pred,model_test_df$Manipulater)
         xg_perf <- performance(xg_prediction, "tpr","fpr")

         plot(xg_perf,main="ROC Curve for xgboost",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(xg_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8805556
```

```
Slot "alpha.values":
list()
```



1.5.7 Boosting with xgboost (up sample)

The below code chunk sets some of the control parameters for adaboost

```
In [69]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
```

```
                                            savePredictions = TRUE,
                                            classProbs = TRUE, sampling = "up")
```

```
In [70]: search_grid <- expand.grid(nrounds = c(70:150), max_depth = c(2:4),
                                   eta = c(0.1,0.3,0.5),
                                   gamma = c(0.03,0.09, 0.12),
                                   colsample_bytree = c(5:10)/10,
                                   min_child_weight = c(1:5),
                                   subsample = c(0.5))
```

After setting the control paramters, the model is run

```
In [71]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Xtreme Boosting with Up Sampling")

         set.seed(4121)
         xg_up_model <- train(model_train_df[,1:5], model_train_df[,6],
                             method='xgbTree',
                             trControl=objControl,
                             tuneGrid = search_grid,
                             metric = "ROC")
         stopCluster(num_cores)
         toc()
```

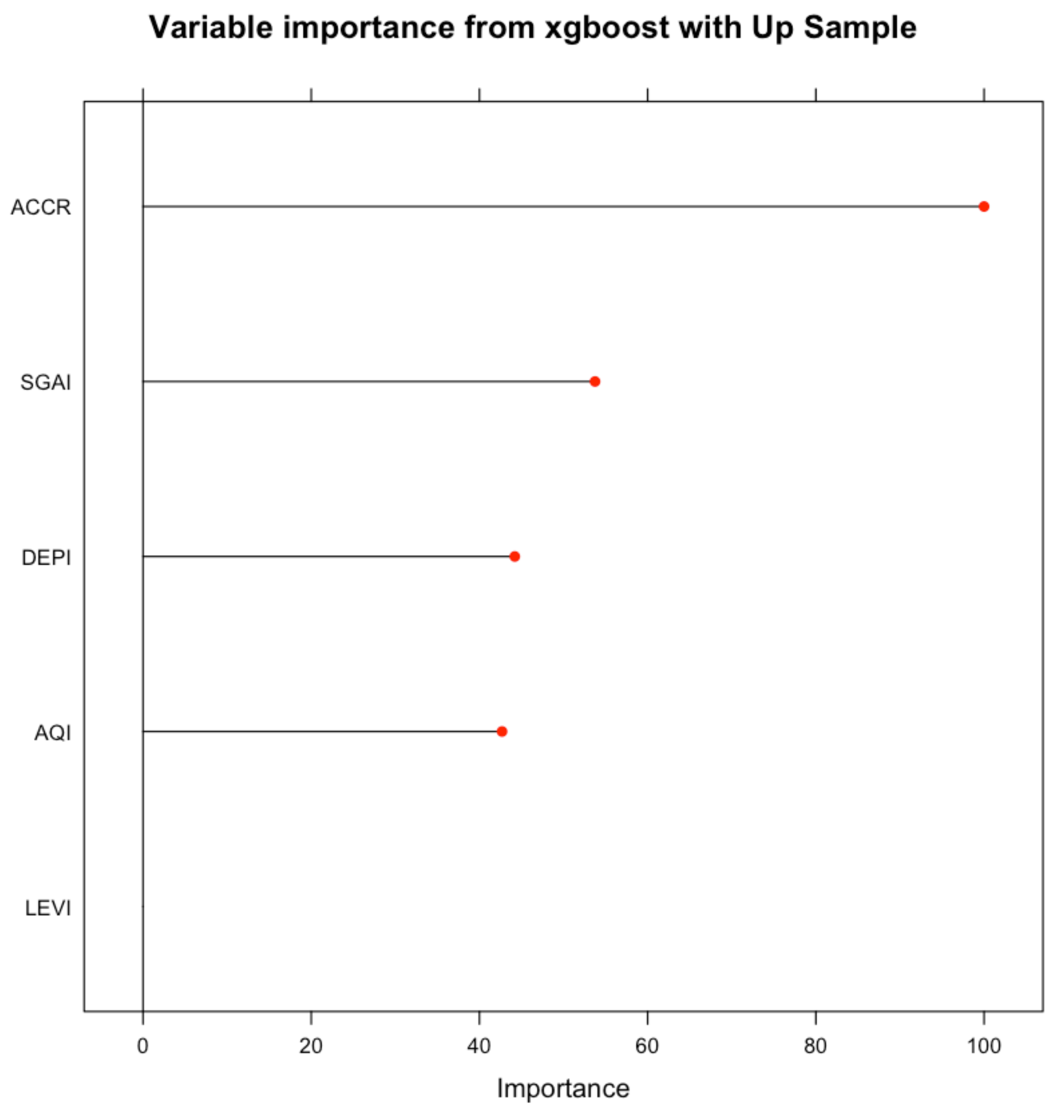
```
Xtreme Boosting with Up Sampling: 309.228 sec elapsed
```

Confusion Matrix for xgboost on train set

```
In [72]: xg_up_model$bestTune
         confusionMatrix.train(xg_up_model)

         plot(varImp(xg_up_model), main = "Variable importance from xgboost with Up Sample", c
```

	nrounds	max_depth	eta	gamma	colsample_bytree	min_child_weight	subsample
28595	71	2	0.3	0.12	0.9	4	0.5

```
Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  96.5  2.6
       Yes  1.0  0.0

 Accuracy (average) : 0.9649
```



Confusion Matrix for xgboost on test set

```
In [73]: caretPredictedClass <- predict(xg_up_model, model_test_df[1:5], type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  339   8
       Yes  21   3

               Accuracy : 0.9218
                 95% CI : (0.8897, 0.947)
```

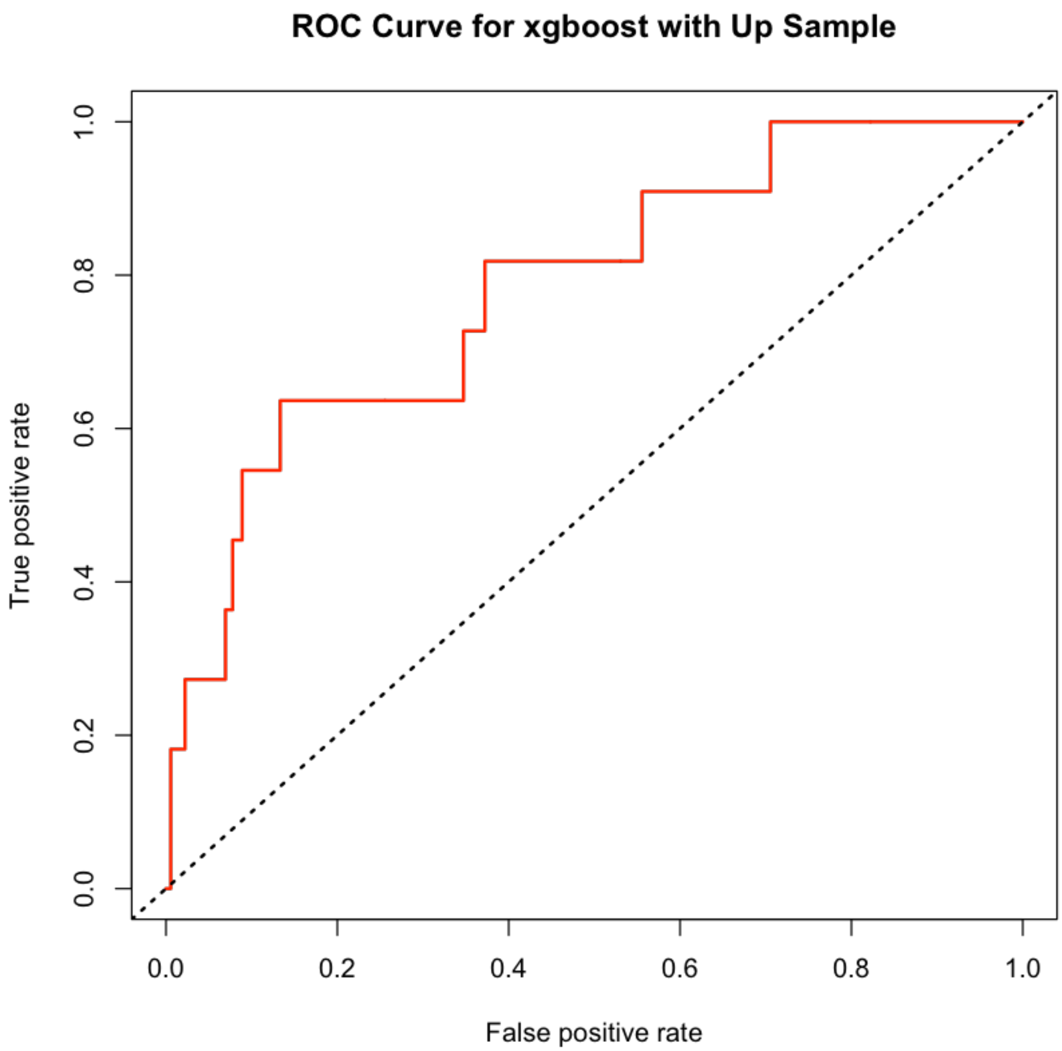
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1.00000

                  Kappa : 0.1363
 Mcnemar's Test P-Value : 0.02586

            Sensitivity : 0.9417
            Specificity : 0.2727
         Pos Pred Value : 0.9769
         Neg Pred Value : 0.1250
             Prevalence : 0.9704
         Detection Rate : 0.9137
   Detection Prevalence : 0.9353
      Balanced Accuracy : 0.6072

       'Positive' Class : No
```

ROC plot for xgboost on test set

```
In [74]: xg_pred <- predict(xg_up_model, model_test_df[1:5], type = "prob")[,2]
         xg_prediction <- prediction(xg_pred,model_test_df$Manipulater)
         xg_perf <- performance(xg_prediction, "tpr","fpr")

         plot(xg_perf,main="ROC Curve for xgboost with Up Sample",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(xg_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.7833333
```

```
Slot "alpha.values":
list()
```



1.5.8 Boosting with xgboost (down sample)

The below code chunk sets some of the control parameters for adaboost

```
In [75]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
```

```
                                          savePredictions = TRUE,
                                          classProbs = TRUE, sampling = "down")
```

```
In [76]: search_grid <- expand.grid(nrounds = c(70:150), max_depth = c(2:4),
                                    eta = c(0.1,0.3,0.5),
                                    gamma = c(0.03,0.09, 0.12),
                                    colsample_bytree = c(5:10)/10,
                                    min_child_weight = c(1:5),
                                    subsample = c(0.5))
```

After setting the control paramters, the model is run

```
In [77]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Xtreme Boosting with Down Sampling")
         set.seed(4121)
         xg_down_model <- train(model_train_df[,1:5], model_train_df[,6],
                                 method='xgbTree',
                                 trControl=objControl,
                                 tuneGrid = search_grid,
                                 metric = "ROC")
         stopCluster(num_cores)
         toc()
```

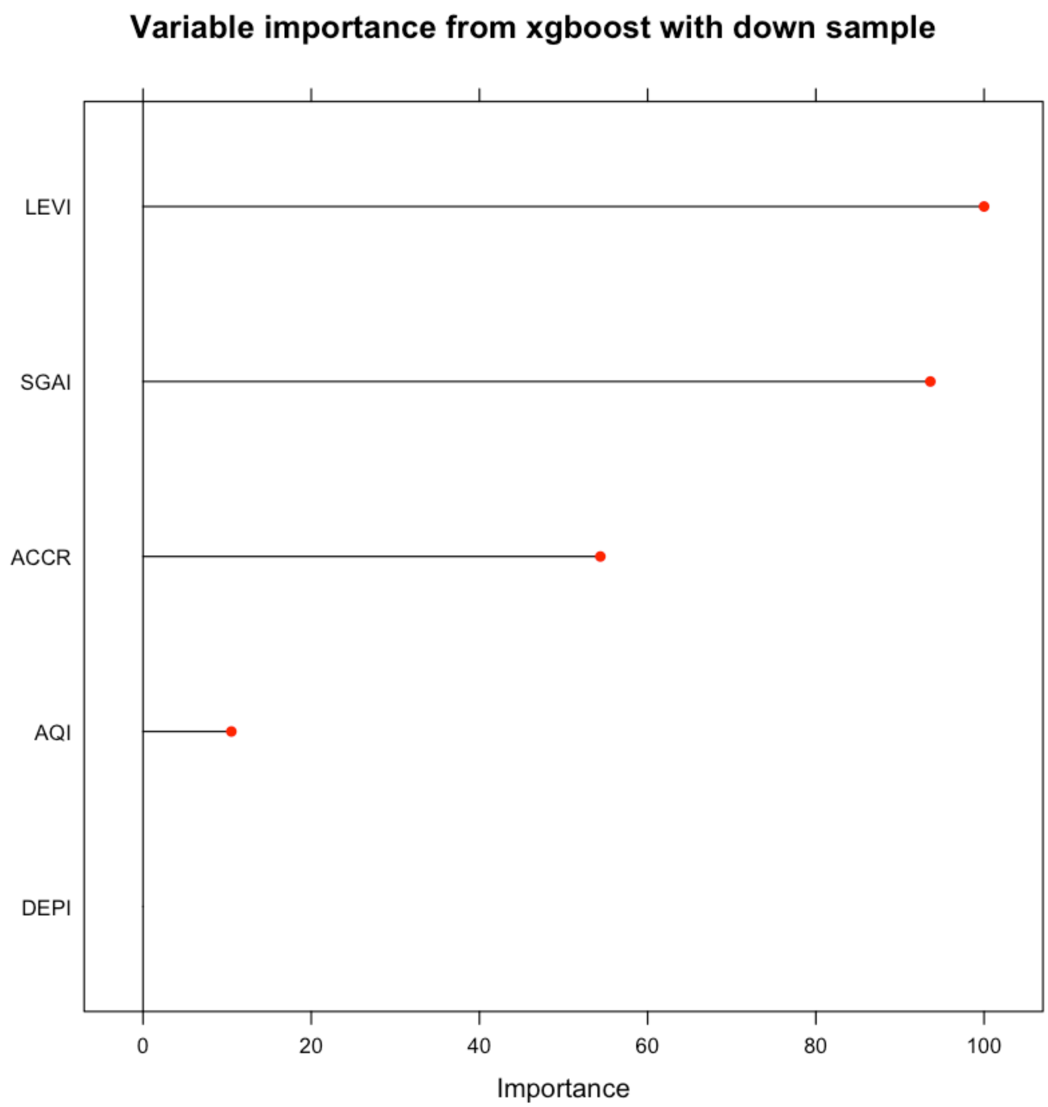
```
Xtreme Boosting with Down Sampling: 221.04 sec elapsed
```

Confusion Matrix for xgboost on train set

```
In [78]: xg_down_model$bestTune
         confusionMatrix.train(xg_down_model)

         plot(varImp(xg_down_model), main = "Variable importance from xgboost with down sample
```

	nrounds	max_depth	eta	gamma	colsample_bytree	min_child_weight	subsample
36099	123	3	0.3	0.12	1	1	0.5

```
Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  70.6  0.6
       Yes 26.8  1.9

 Accuracy (average) : 0.7252
```



Confusion Matrix for xgboost on test set

```
In [79]: caretPredictedClass <- predict(xg_down_model, model_test_df[1:5], type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  232   1
       Yes 128  10

               Accuracy : 0.6523
                 95% CI : (0.6014, 0.7007)
```

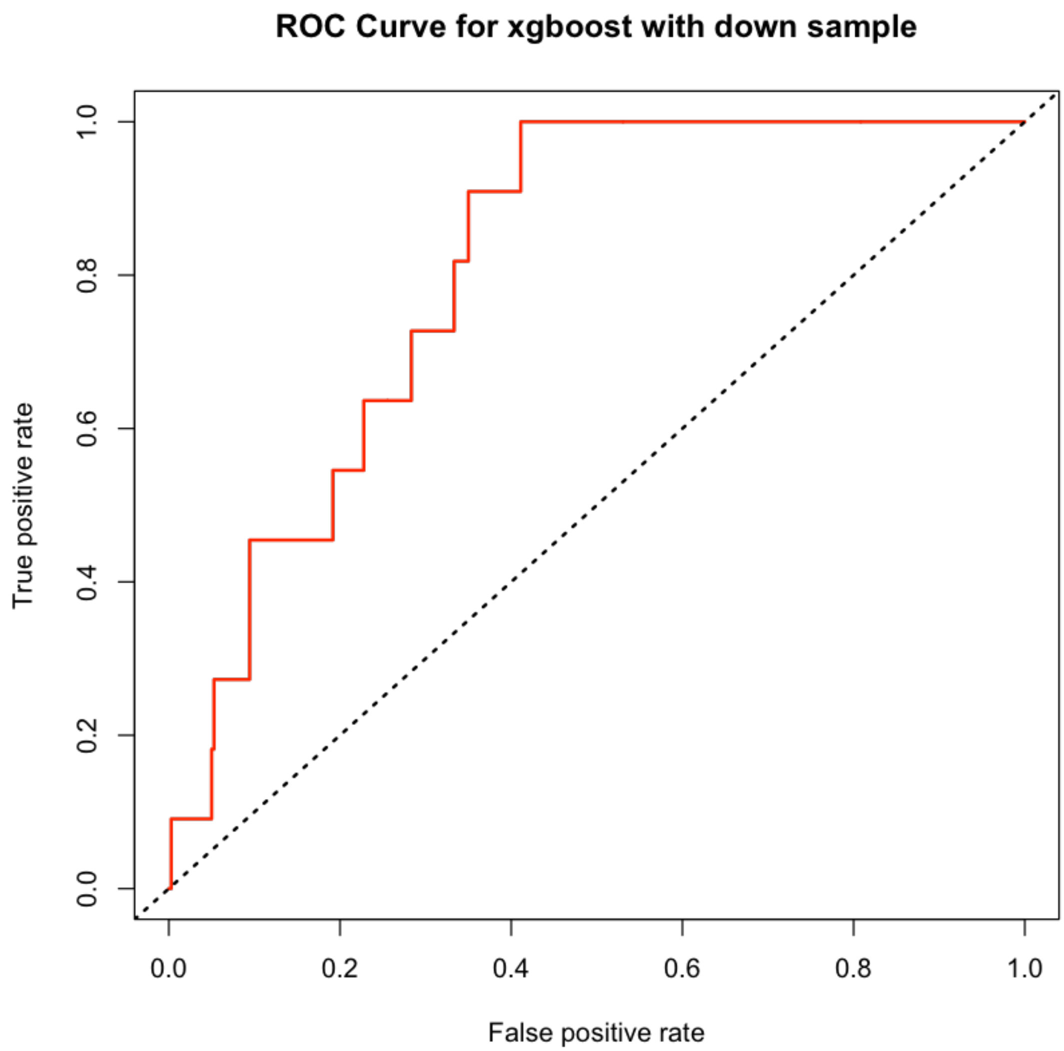
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0839
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.64444
            Specificity : 0.90909
         Pos Pred Value : 0.99571
         Neg Pred Value : 0.07246
             Prevalence : 0.97035
         Detection Rate : 0.62534
   Detection Prevalence : 0.62803
      Balanced Accuracy : 0.77677

       'Positive' Class : No
```

ROC plot for xgboost on test set

```
In [80]: xg_pred <- predict(xg_down_model, model_test_df[1:5], type = "prob")[,2]
         xg_prediction <- prediction(xg_pred,model_test_df$Manipulater)
         xg_perf <- performance(xg_prediction, "tpr","fpr")

         plot(xg_perf,main="ROC Curve for xgboost with down sample",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(xg_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8098485
```

```
Slot "alpha.values":
list()
```



1.5.9 Boosting with xgboost (SMOTE)

The below code chunk sets some of the control parameters for adaboost

```
In [81]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='final',
                                    summaryFunction = twoClassSummary,
```

```
                                    savePredictions = TRUE,
                                    classProbs = TRUE, sampling = "smote")
```

```
In [82]: search_grid <- expand.grid(nrounds = c(70:150), max_depth = c(2:4),
                             eta = c(0.1,0.3,0.5),
                             gamma = c(0.03,0.09, 0.12),
                             colsample_bytree = c(5:10)/10,
                             min_child_weight = c(1:5),
                             subsample = c(0.5))
```

After setting the control paramters, the model is run

```
In [83]: num_cores <- makeCluster(detectCores()-5)
         registerDoParallel(num_cores)
         tic("Xtreme Boosting with SMOTE Sampling")
         set.seed(4121)
         xg_smote_model <- train(model_train_df[,1:5], model_train_df[,6],
                           method='xgbTree',
                           trControl=objControl,
                           tuneGrid = search_grid,
                           metric = "ROC")
         stopCluster(num_cores)
         toc()
```

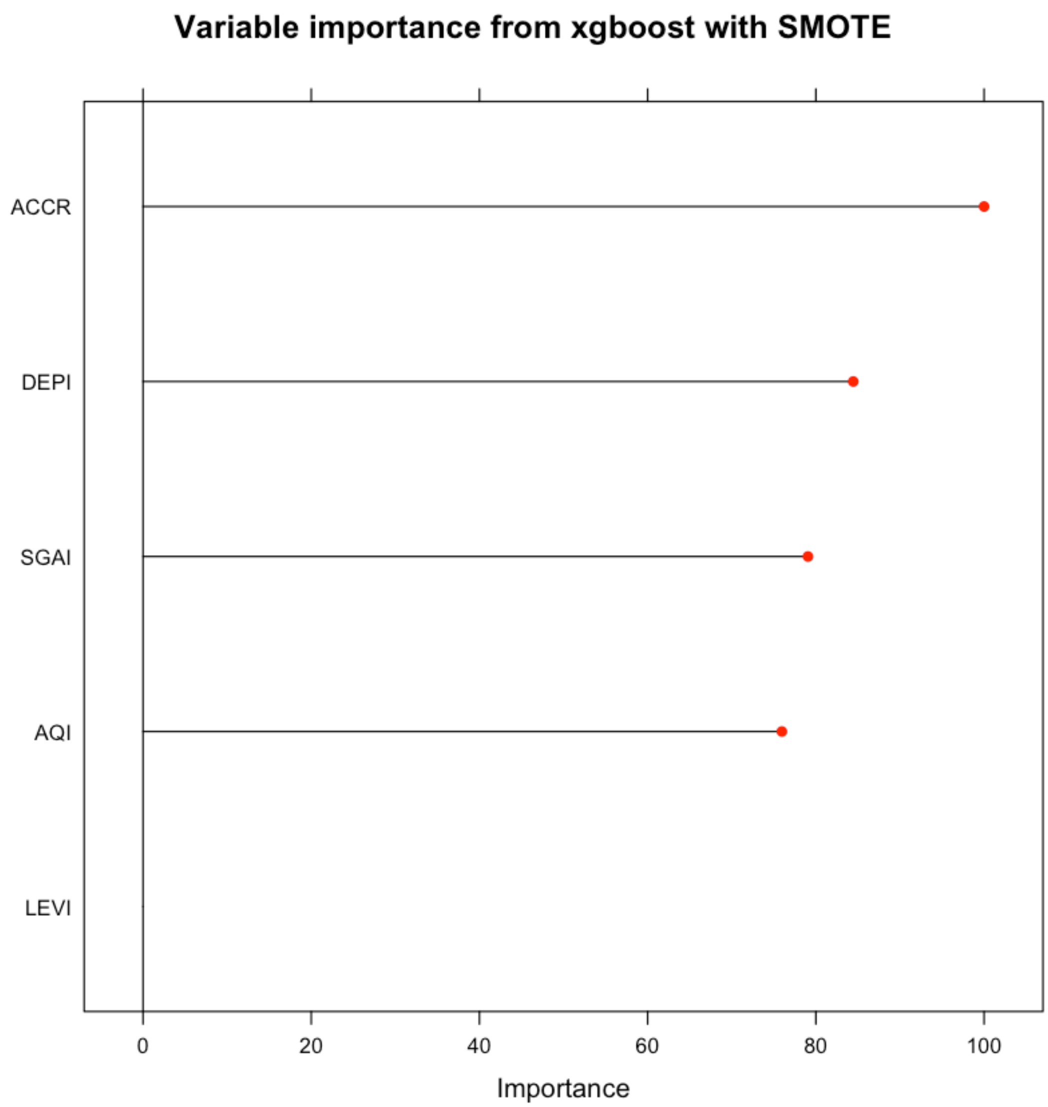
```
Xtreme Boosting with SMOTE Sampling: 261.745 sec elapsed
```

Confusion Matrix for xgboost on train set

```
In [84]: xg_smote_model$bestTune
         confusionMatrix.train(xg_smote_model)

         plot(varImp(xg_smote_model), main = "Variable importance from xgboost with SMOTE", co
```

	nrounds	max_depth	eta	gamma	colsample_bytree	min_child_weight	subsample
45049	82	2	0.5	0.03	0.8	2	0.5

```
Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  85.6  1.6
       Yes 11.8  1.0

 Accuracy (average) : 0.8658
```



Confusion Matrix for xgboost on test set

```
In [85]: caretPredictedClass <- predict(xg_smote_model, model_test_df[1:5], type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  286   4
       Yes  74   7

               Accuracy : 0.7898
                 95% CI : (0.7447, 0.8301)
```

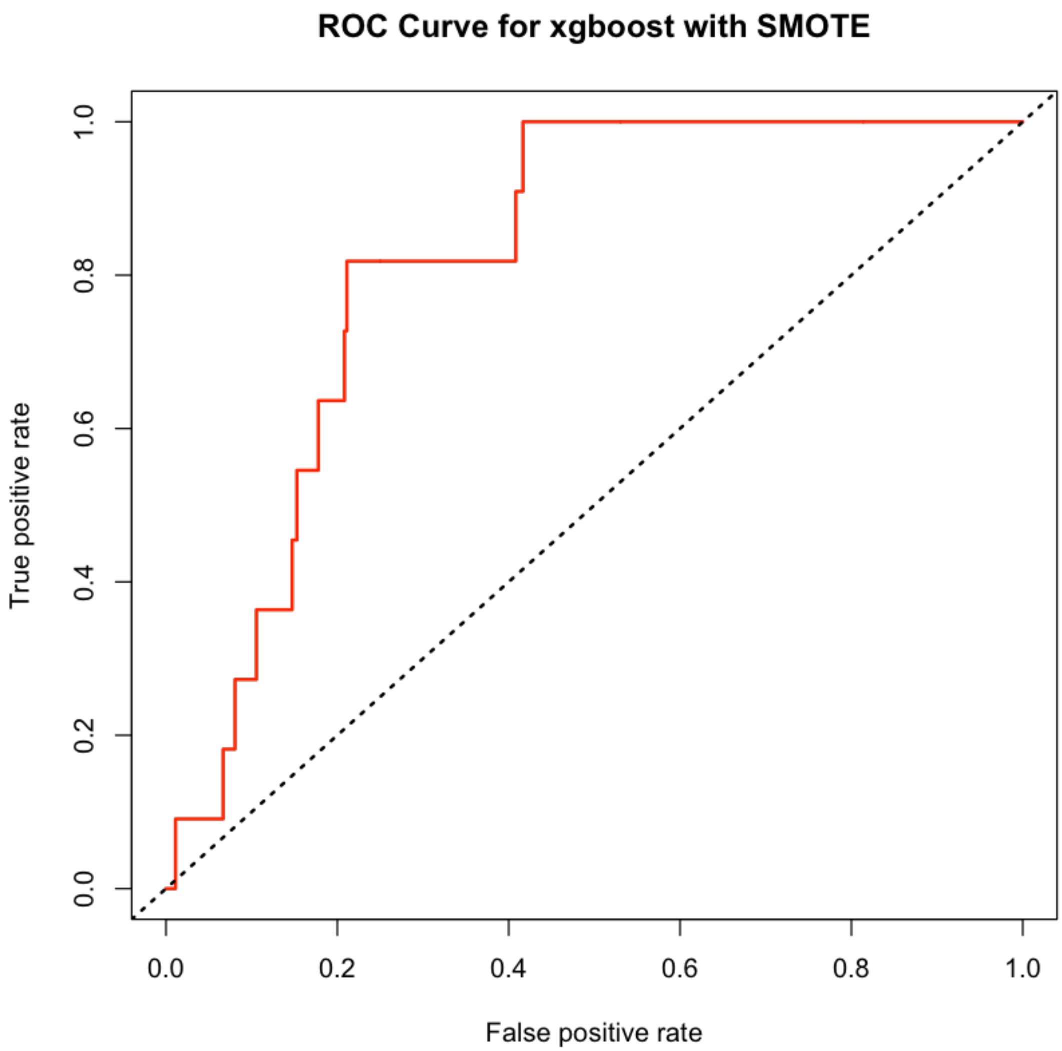
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1055
 Mcnemar's Test P-Value : 5.597e-15

            Sensitivity : 0.79444
            Specificity : 0.63636
         Pos Pred Value : 0.98621
         Neg Pred Value : 0.08642
             Prevalence : 0.97035
         Detection Rate : 0.77089
   Detection Prevalence : 0.78167
      Balanced Accuracy : 0.71540

       'Positive' Class : No
```

ROC plot for xgboost on test set

```
In [86]: xg_pred <- predict(xg_smote_model, model_test_df[1:5], type = "prob")[,2]
         xg_prediction <- prediction(xg_pred,model_test_df$Manipulater)
         xg_perf <- performance(xg_prediction, "tpr","fpr")

         plot(xg_perf,main="ROC Curve for xgboost with SMOTE",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(xg_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8194444
```

```
Slot "alpha.values":
list()
```




```
In [87]: toc()
```

```
Total Time for Bagging and Boosting: 1577.322 sec elapsed
```

1.6 Neural Network

1.6.1 Neural network implementation to find the manipulaters

The below code chunk sets some of the control parameters

```
In [88]: objControl <- trainControl(method='boot', number = 1,
                                    returnResamp='none',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE, allowParallel=FALSE)
```

Using search grid to fine tune the neural network. **Size** fine tunes number of hidden units to tune and **decay** fine tunes weight decay

```
In [89]: search_grid <- expand.grid(.decay = c(0.5, 0.1, 0.05), .size = c(2, 3, 4,5,6,7))
```

After setting the control paramters, the model is run. If we use **linout=TRUE** in **train()** the neural network builds a regression model. **linout=FALSE** will make **nnet** use a sigmodial function and all the predictions will be constrained between **[0,1]**

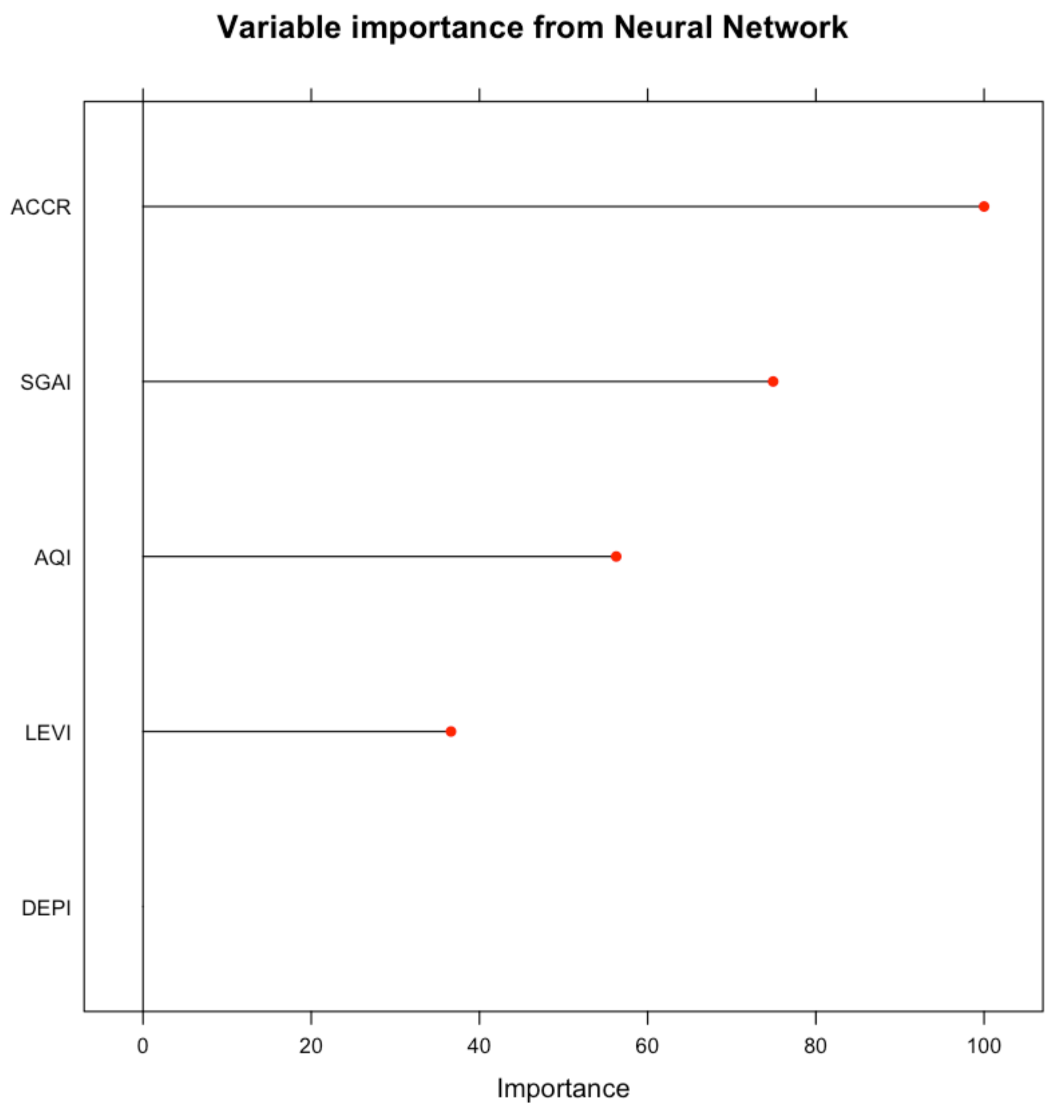
```
In [90]: set.seed(4121)

         nn_model <- train(model_train_df[,1:5], model_train_df[,6],
                           method='nnet',
                           trControl=objControl,
                           metric = "ROC",
                           maxit = 1000,
                           tuneGrid = search_grid,
                           trace = FALSE,
                           linout = FALSE)
```

Confusion Matrix for Neural Network on train set

```
In [91]: #nn_model$finalModel #nn_model$results
         print(nn_model)
         confusionMatrix.train(nn_model)
         plot(varImp(nn_model), main = "Variable importance from Neural Network", col = 2, lwd
```

```
Neural Network

868 samples
  5 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Resampling results across tuning parameters:
```

```
  decay  size  ROC        Sens       Spec
  0.05   2     0.6418033  0.9967213  0
  0.05   3     0.6221311  0.9967213  0
  0.05   4     0.5393443  0.9967213  0
  0.05   5     0.6524590  0.9967213  0
  0.05   6     0.5799180  0.9967213  0
  0.05   7     0.4901639  0.9967213  0
  0.10   2     0.5676230  0.9967213  0
  0.10   3     0.5991803  0.9967213  0
  0.10   4     0.6245902  0.9967213  0
  0.10   5     0.6131148  0.9934426  0
  0.10   6     0.5979508  0.9934426  0
  0.10   7     0.5778689  0.9967213  0
  0.50   2     0.3692623  1.0000000  0
  0.50   3     0.3905738  0.9967213  0
  0.50   4     0.3942623  0.9967213  0
  0.50   5     0.3840164  0.9967213  0
  0.50   6     0.3926230  0.9967213  0
  0.50   7     0.3770492  0.9967213  0

ROC was used to select the optimal model using the largest value.
The final values used for the model were size = 5 and decay = 0.05.


Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  97.1  2.6
       Yes  0.3  0.0

 Accuracy (average) : 0.9712
```



Confusion Matrix for Neural Network on test set

```
In [92]: caretPredictedClass <- predict(nn_model, model_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,model_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  359  11
       Yes   1   0

               Accuracy : 0.9677
                 95% CI : (0.9442, 0.9832)
```

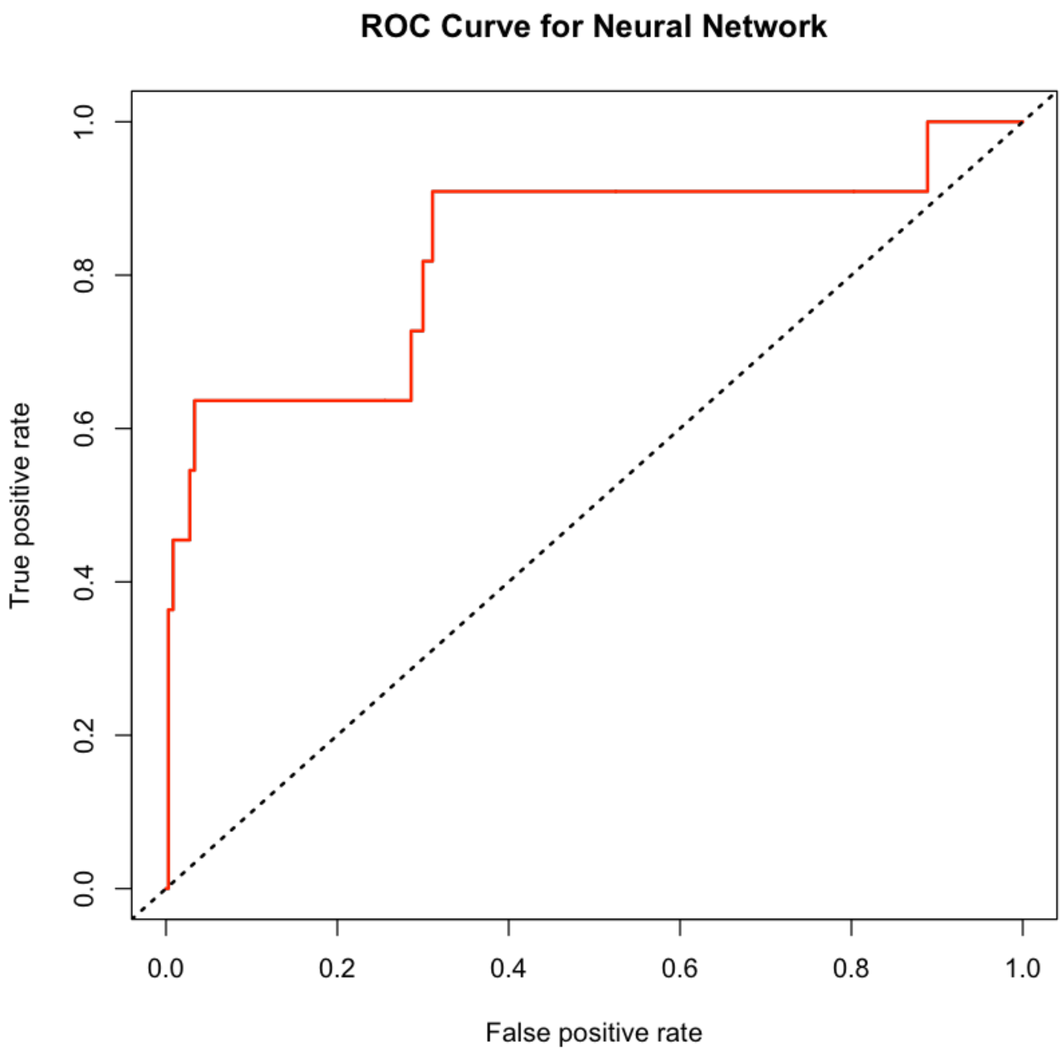
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.690364

                  Kappa : -0.005
 Mcnemar's Test P-Value : 0.009375

            Sensitivity : 0.9972
            Specificity : 0.0000
         Pos Pred Value : 0.9703
         Neg Pred Value : 0.0000
             Prevalence : 0.9704
         Detection Rate : 0.9677
   Detection Prevalence : 0.9973
      Balanced Accuracy : 0.4986

       'Positive' Class : No
```

ROC plot for Neural Network on test set

```
In [93]: nn_pred <- predict(nn_model, model_test_df, type = "prob")[,2]
         nn_prediction <- prediction(nn_pred,model_test_df$Manipulater)
         nn_perf <- performance(nn_prediction, "tpr","fpr")

         plot(nn_perf,main="ROC Curve for Neural Network",col=2,lwd=2)
         abline(a=0,b=1,lwd=2,lty=3,col="black")

         #AUC for the ROC plot
         performance(nn_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.830303
```

```
Slot "alpha.values":
list()
```



1.7 Logistic Regression

The variables DSRI and GMI causes fitted probability to be numerically 0 or 1. Using less number of variables in the logistic regression.

```
In [94]: lg_model_df <- as.data.frame(filter_data[,c(#"DSRI",
                                                     #"GMI",
```

```
                                                  "AQI",
                                                  "SGI",
                                                  "DEPI",
                                                  "SGAI",
                                                  "ACCR",
                                                  "LEVI",
                                                  "Manipulater"
         )])
         lg_train_df <- as.data.frame(train_df[,c(#"DSRI",
                                                  #"GMI",
                                                  "AQI",
                                                  "SGI",
                                                  "DEPI",
                                                  "SGAI",
                                                  "ACCR",
                                                  "LEVI",
                                                  "Manipulater"
         )])
         lg_test_df <- as.data.frame(test_df[,c(#"DSRI",
                                                  #"GMI",
                                                  "AQI",
                                                  "SGI",
                                                  "DEPI",
                                                  "SGAI",
                                                  "ACCR",
                                                  "LEVI",
                                                  "Manipulater"
         )])
```

The below code chunk sets some of the control parameters

```
In [95]: objControl <- trainControl(method='boot', number=1,
                                    returnResamp='none',
                                    summaryFunction = twoClassSummary,
                                    savePredictions = TRUE,
                                    classProbs = TRUE,allowParallel=FALSE)
```

After setting the control paramters, the model is run

```
In [96]: set.seed(4121)
         lg_model <- train(lg_train_df[,1:6], lg_train_df[,7],
                              method='glmStepAIC',
                              trControl=objControl,
                              metric = "ROC")

Start:  AIC=196.08
.outcome ~ AQI + SGI + DEPI + SGAI + ACCR + LEVI


       Df Deviance    AIC
```

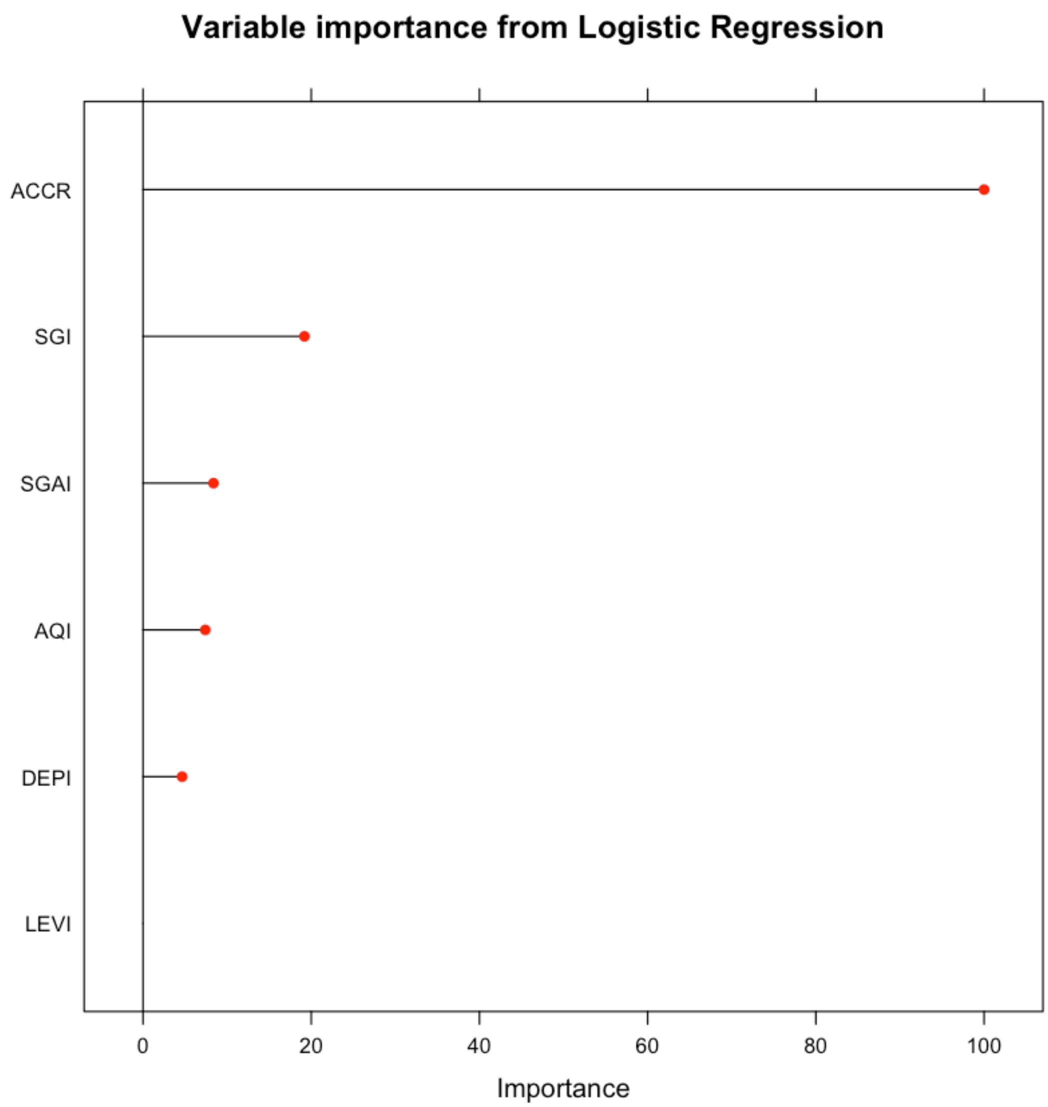
```
- LEVI  1   183.09 195.09
<none>      182.08 196.08
- DEPI  1   184.44 196.44
- AQI   1   186.75 198.75
- SGI   1   189.02 201.02
- SGAI  1   215.58 227.58
- ACCR  1   233.01 245.01

Step:  AIC=195.09
.outcome ~ AQI + SGI + DEPI + SGAI + ACCR

       Df Deviance    AIC
<none>      183.09 195.09
- DEPI  1   185.39 195.39
- AQI   1   187.75 197.75
- SGI   1   189.69 199.69
- SGAI  1   216.46 226.46
- ACCR  1   233.53 243.53
Start:  AIC=216.72
.outcome ~ AQI + SGI + DEPI + SGAI + ACCR + LEVI

       Df Deviance    AIC
- DEPI  1   203.35 215.35
<none>      202.72 216.72
- LEVI  1   206.04 218.04
- SGI   1   209.63 221.63
- AQI   1   214.90 226.90
- SGAI  1   215.20 227.20
- ACCR  1   217.84 229.84

Step:  AIC=215.35
.outcome ~ AQI + SGI + SGAI + ACCR + LEVI

       Df Deviance    AIC
<none>      203.35 215.35
- LEVI  1   206.72 216.72
- SGI   1   210.29 220.29
- AQI   1   217.09 227.09
- SGAI  1   217.61 227.61
- ACCR  1   218.99 228.99
```

Confusion Matrix for logistic regression on train set

In [97]:
```
print(lg_model)
confusionMatrix.train(lg_model)
plot(varImp(lg_model), main = "Variable importance from Logistic Regression", col = 2
```

```
Generalized Linear Model with Stepwise Feature Selection
```

```
868 samples
  6 predictor
  2 classes: 'No', 'Yes'

No pre-processing
Resampling: Bootstrapped (1 reps)
Summary of sample sizes: 868
Resampling results:

  ROC        Sens       Spec
  0.6983607  0.9934426  0.25



Bootstrapped (1 reps) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   No  Yes
       No  96.8  1.9
       Yes  0.6  0.6

 Accuracy (average) : 0.9744
```



Confusion Matrix for logistic regression on test set

```
In [98]: caretPredictedClass <- predict(lg_model, lg_test_df, type = "raw")
         confusionMatrix(caretPredictedClass,lg_test_df$Manipulater)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  360   9
       Yes   0   2

               Accuracy : 0.9757
                 95% CI : (0.9545, 0.9888)
```

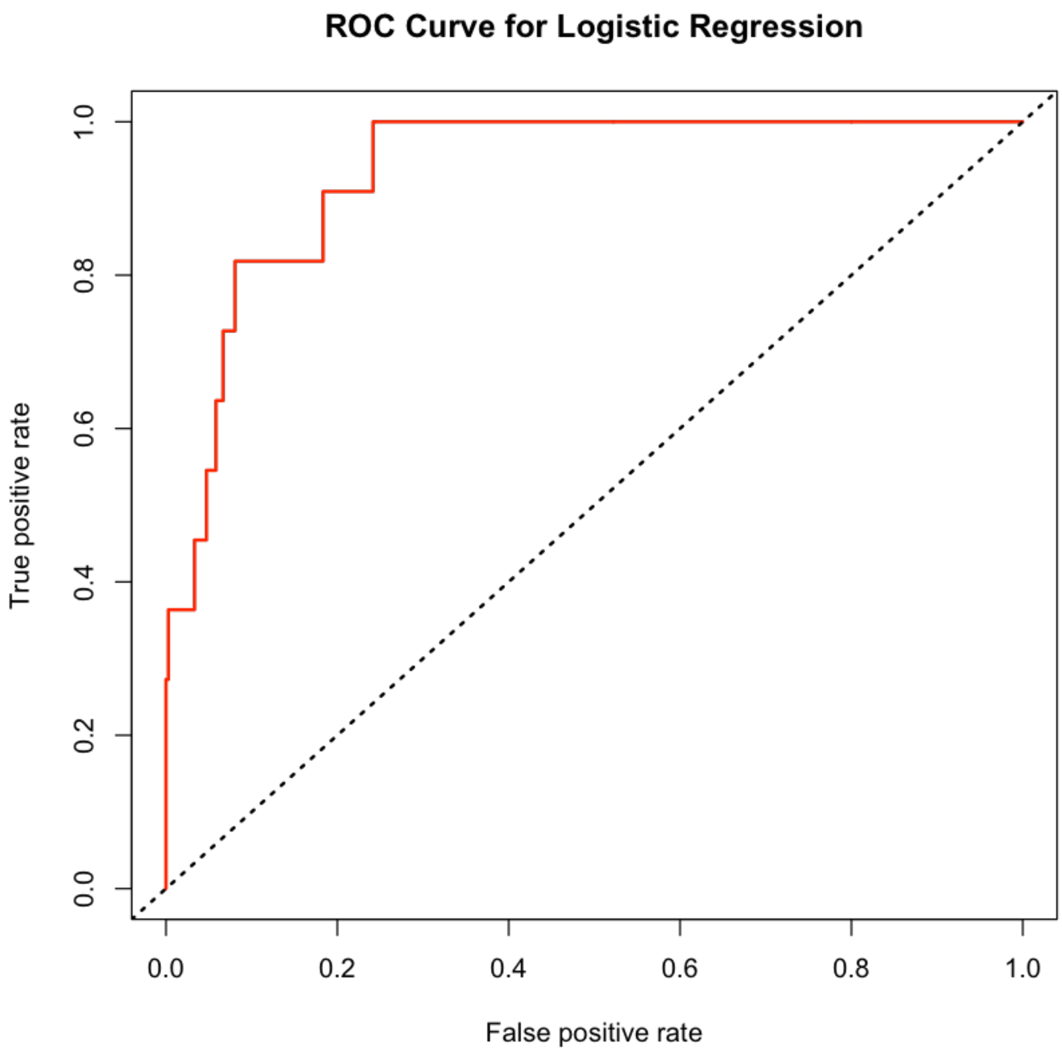
```
    No Information Rate : 0.9704
    P-Value [Acc > NIR] : 0.337237

                  Kappa : 0.3013
 Mcnemar's Test P-Value : 0.007661

            Sensitivity : 1.0000
            Specificity : 0.1818
         Pos Pred Value : 0.9756
         Neg Pred Value : 1.0000
             Prevalence : 0.9704
         Detection Rate : 0.9704
   Detection Prevalence : 0.9946
      Balanced Accuracy : 0.5909

       'Positive' Class : No
```

ROC plot for logistic regression

In [99]:
```
lg_pred <- predict(lg_model, lg_test_df, type = "prob")[,2]
lg_prediction <- prediction(lg_pred,lg_test_df$Manipulater)
lg_perf <- performance(lg_prediction, "tpr","fpr")

plot(lg_perf,main="ROC Curve for Logistic Regression",col=2,lwd=2)
abline(a=0,b=1,lwd=2,lty=3,col="black")

#AUC for the ROC plot
performance(lg_prediction, "auc")
```

```
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.935101
```

```
Slot "alpha.values":
list()
```



End of document