# P3-BostonDataSet

*Jeetendra Gan*

*11/27/2019*

In this problem I will be using the boston data set. In this case our response is 'crim', but crim is a quantative variable. I have replaced crim with TRUE/FALSE, TRUE if the crime rate is greater than the median value, else false. Following are the variable names in the boston data set.

```
##  [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
##  [8] "dis"     "rad"     "tax"     "ptratio" "black"   "lstat"   "medv"
```

The median for crime rate is:

```
## [1] 0.25651
```

Here is the statistic after crim is transformed as a categorical variable.

```
##    Mode   FALSE    TRUE
## logical     253     253
```

I have divided the data set into 75% train and 25% test set.

## Non-ensemble methods

### Logistic regression

Here is the summary after applying logistic regression.

```
##
## Call:
## glm(formula = crim ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7109  -0.1721   0.0001   0.0054   3.5188
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -33.939299   7.488211  -4.532 5.83e-06 ***
## zn           -0.078757   0.038047  -2.070  0.03845 *
## indus        -0.070021   0.052616  -1.331  0.18326
## chas          0.401738   0.853233   0.471  0.63775
## nox          47.398517   8.643372   5.484 4.16e-08 ***
## rm           -0.331606   0.809108  -0.410  0.68192
## age           0.042799   0.015352   2.788  0.00531 **
## dis           0.746777   0.262032   2.850  0.00437 **
## rad           0.586171   0.179218   3.271  0.00107 **
## tax          -0.006918   0.003313  -2.088  0.03678 *
## ptratio       0.250405   0.141495   1.770  0.07678 .
## black        -0.006728   0.005234  -1.285  0.19864
## lstat        -0.011002   0.059629  -0.185  0.85362
## medv          0.120250   0.076044   1.581  0.11380
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 525.38  on 378  degrees of freedom
## Residual deviance: 159.39  on 365  degrees of freedom
## AIC: 187.39
##
## Number of Fisher Scoring iterations: 9
```

For logistic regression following is the order of variable significance - nox > age = dis = rad > tax = zn.

The predictions done by Logistic regression is as follows:

```
##           Reference
## Prediction FALSE TRUE
##      FALSE    62    7
##      TRUE      3   55
```

```
## [1] "The accuracy obtained using logistic regression: 0.92"
```

## KNN

I have fit the knn model using K = 5. Here are the predictions

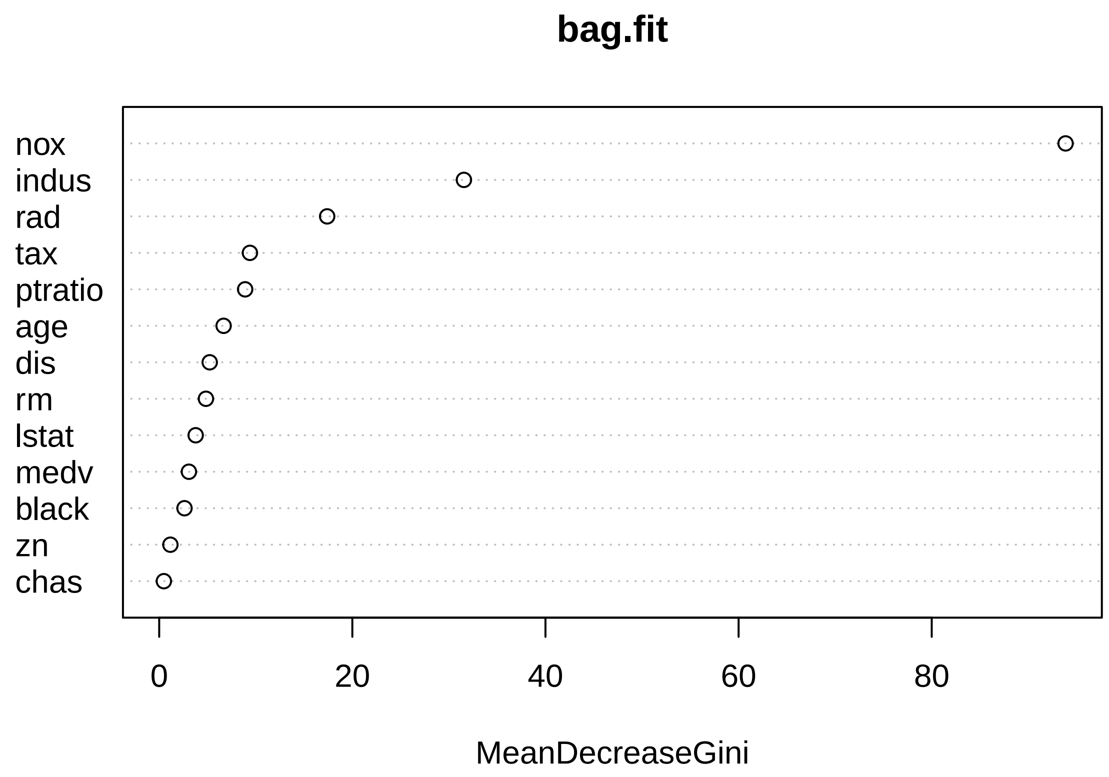
```
##           Reference
## Prediction FALSE TRUE
##      FALSE    64    9
##      TRUE      1   53
```

```
## [1] "The accuracy obtained using KNN is:  0.92"
```

It can be seen that both knn and logistic regression have almost the same accuracy.

## Bagging

The following is the result after applying bagging. The plot shows the importance of variables. The variable importance is close to our findings in logistic regression.



```
##         MeanDecreaseGini
## zn             1.1583406
## indus         31.5538300
## chas           0.4824544
## nox           93.8641614
## rm             4.8409689
## age            6.6695759
## dis            5.2330245
## rad           17.3938645
## tax            9.3943798
## ptratio        8.8989541
## black          2.6187818
## lstat          3.7767721
## medv           3.0753565
```

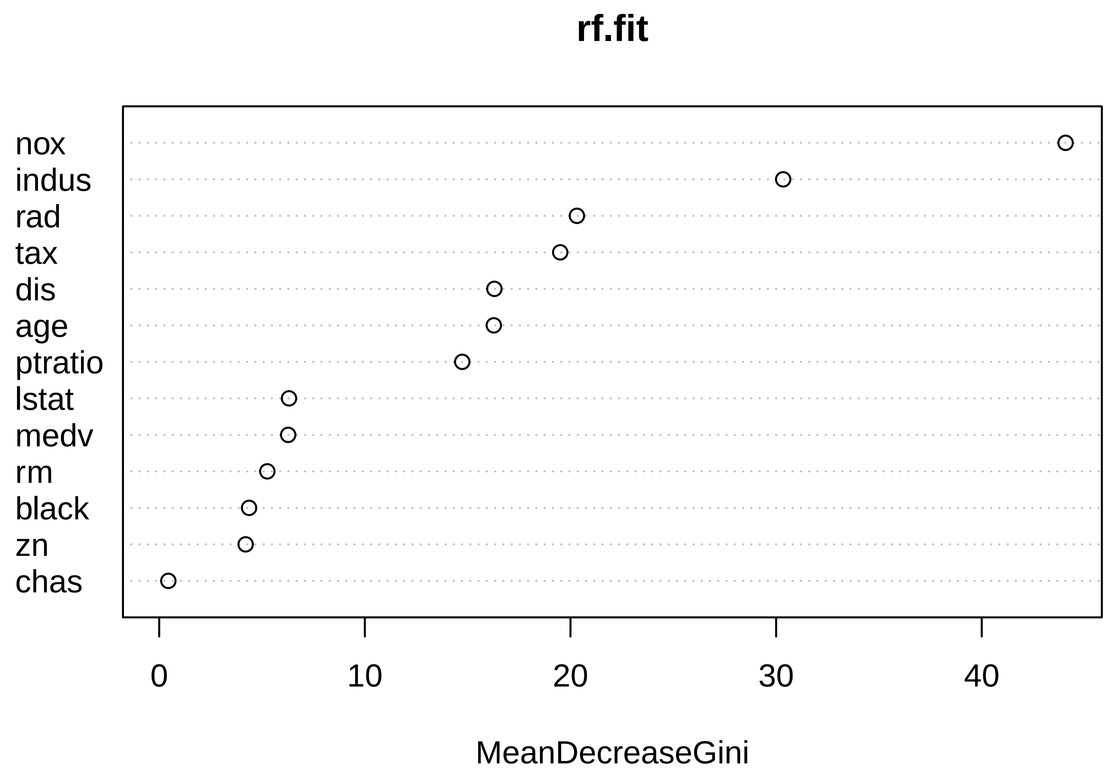
The predictions done by bagging are as follows:

```
##           Reference
## Prediction  0  1
##          0 63  5
##          1  2 57
```

```
## [1] "The accuracy obtained using bagging:  0.945"
```

It can be seen that bagging performs better than KNN and logistic.

## Randomforest

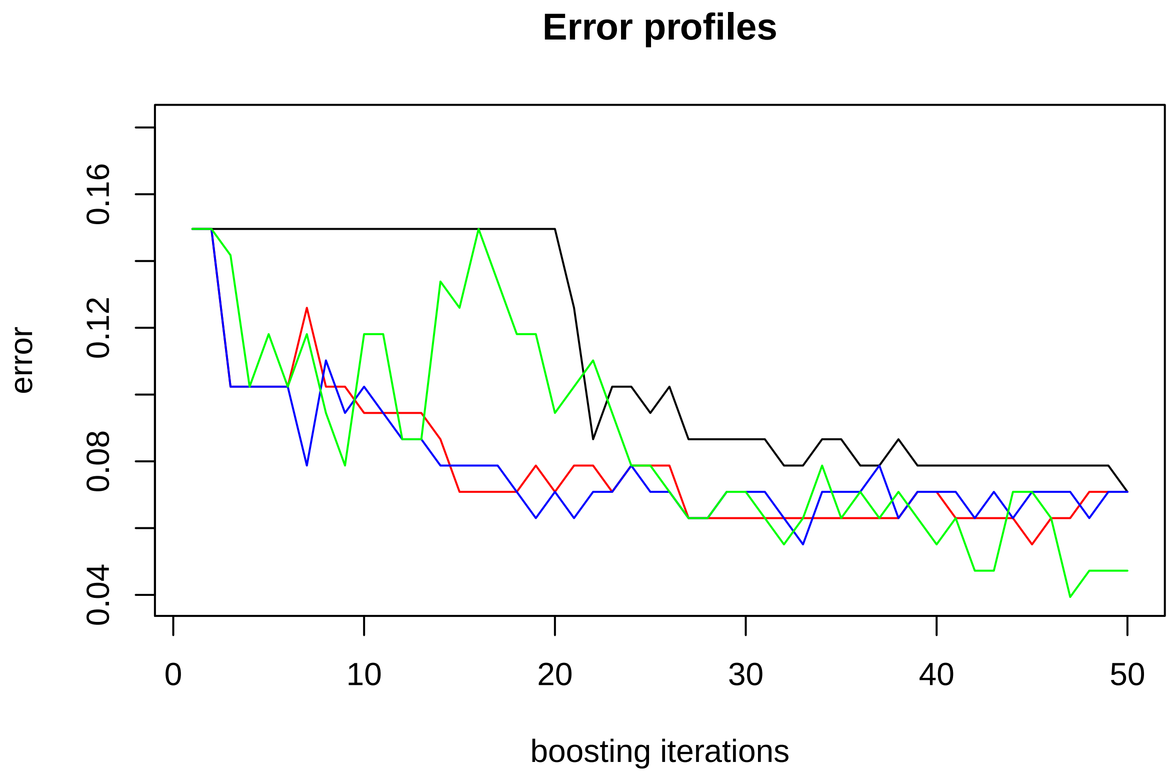


```
##         MeanDecreaseGini
## zn             4.2033699
## indus         30.3419821
## chas           0.4441461
## nox           44.0770111
## rm             5.2576994
## age           16.2716929
## dis           16.2985113
## rad           20.3114567
## tax           19.5009087
## ptratio       14.7359228
## black          4.3710018
## lstat          6.3122084
## medv           6.2717768
```

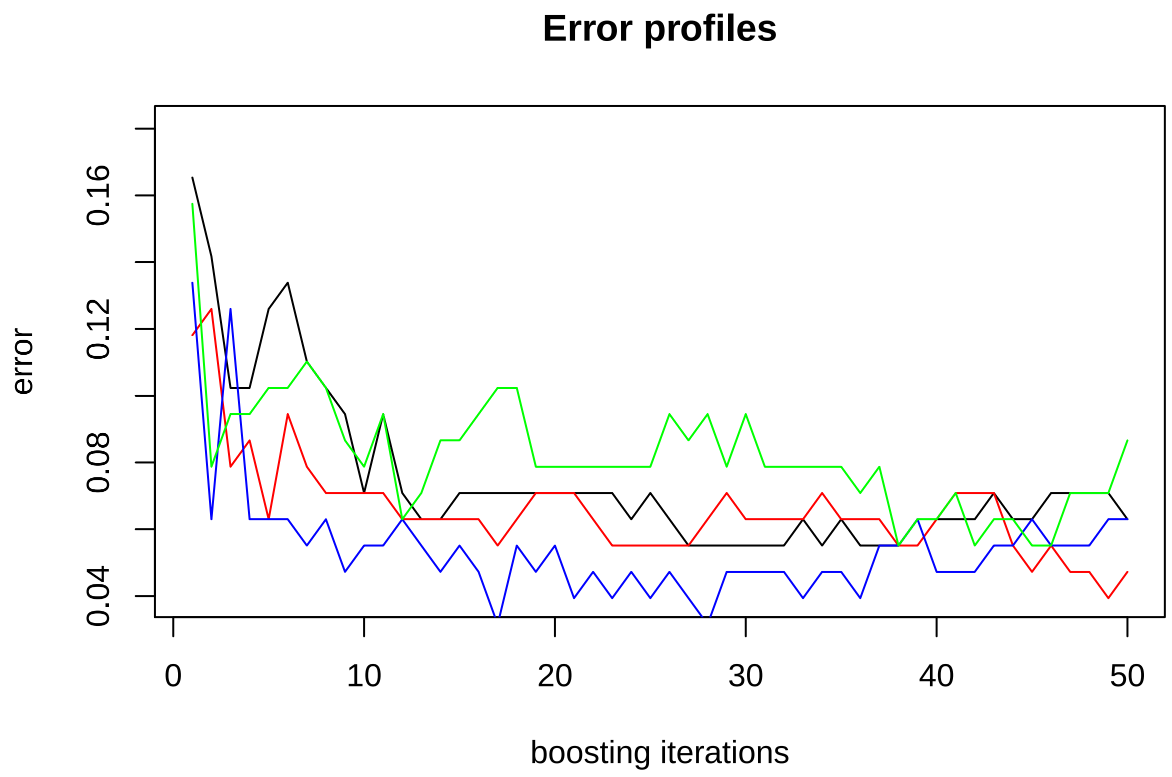
The accuracy obtained using random forest is:

```
## [1] "The accuracy obtained using random forests is:  0.945"
```

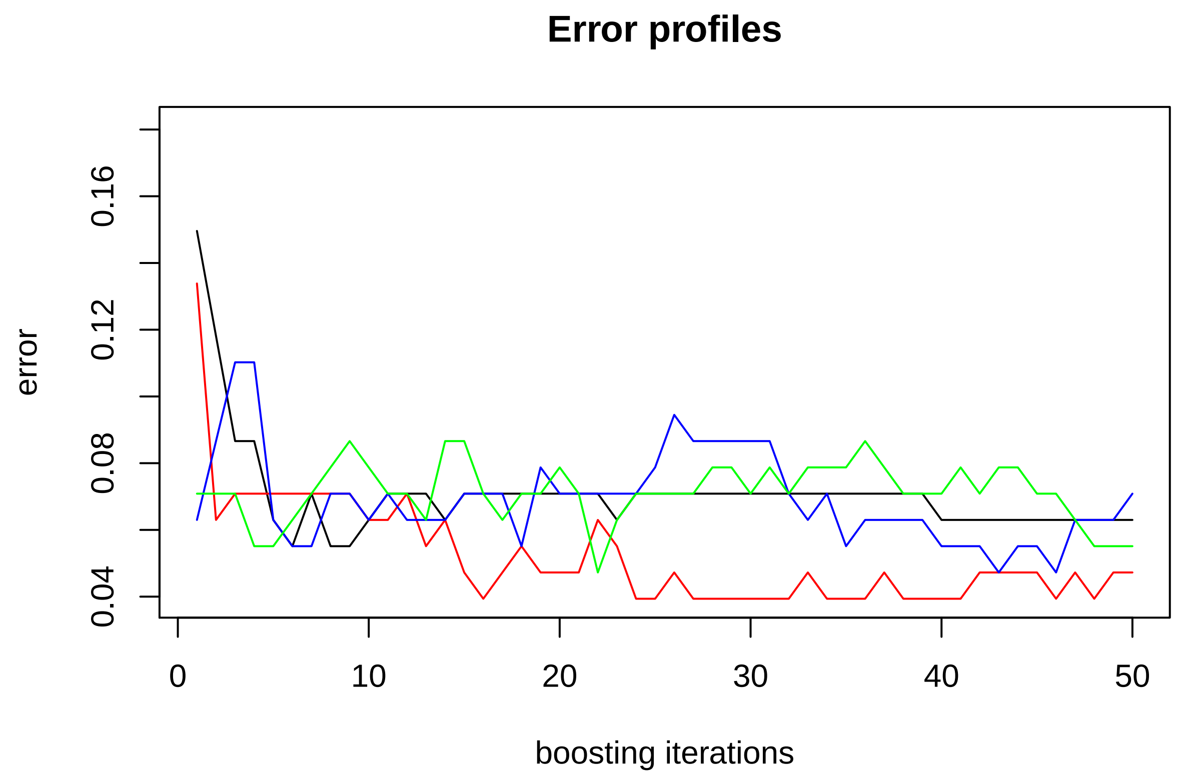
**Boosting**



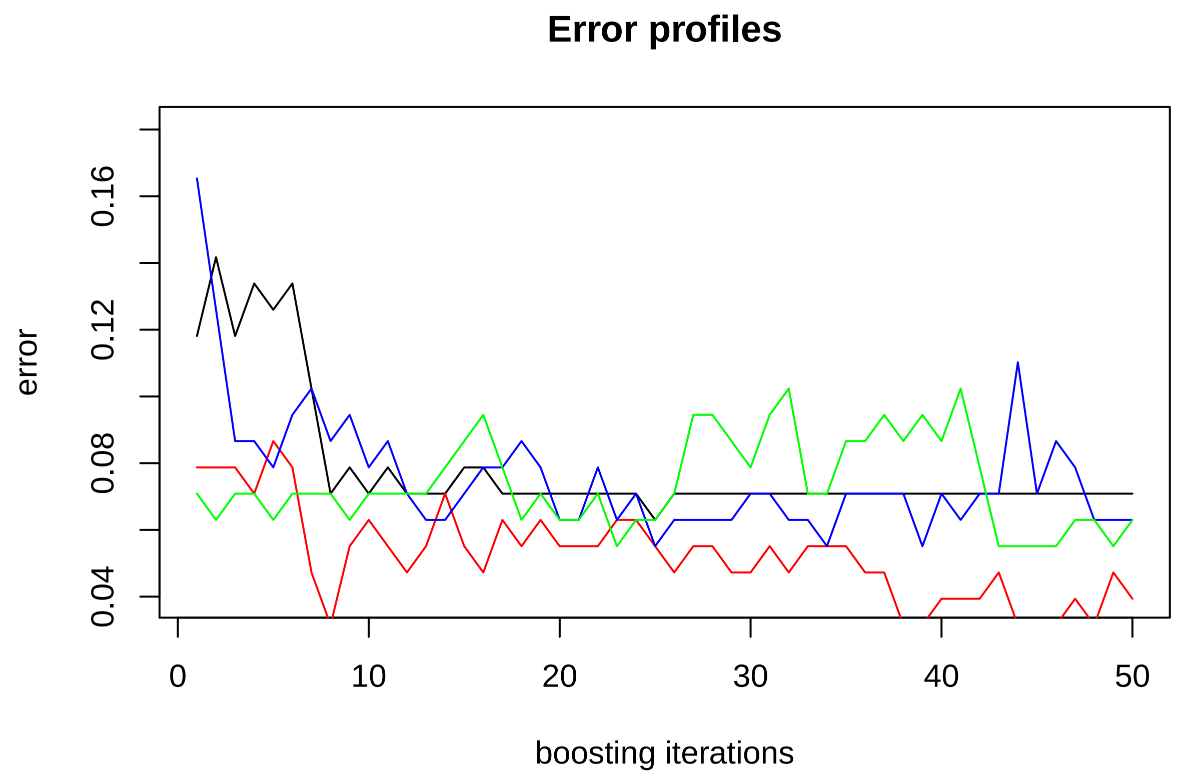
Error Profiles:(Trees, Shrinkage, Interaction):,50,{ 0.1, 0.4, 0.6, 0.8 },1



Error Profiles:(Trees, Shrinkage, Interaction):,50,{ 0.1, 0.4, 0.6, 0.8 },3



Error Profiles:(Trees, Shrinkage, Interaction):,50,{ 0.1, 0.4, 0.6, 0.8 },5



Error Profiles:(Trees, Shrinkage, Interaction):,50,{ 0.1, 0.4, 0.6, 0.8 },7

The graphs are drawn for interaction depths for values - 1, 3, 5, and 7. It can be seen that as the interaction depth increases, the error drops at a faster rate with larger tree count for all the values of lambda. The behaviour of all the models is almost the same if the count of trees is increased.

I have picked interaction depth equal to 5, and lambda equal to 0.4, and tree count equal to 50 so as to create the boosting model. Here is the confusion matrix.

```
##           Reference
## Prediction  0  1
##          0 64  5
##          1  1 57
```

```
## [1] "The accuracy obtained using boosting is: 0.952756"
```

Heighest accuracy is obtained using boosting! Which is closely followed by Random forests and bagging. These three ensamble methods perform better than simple methods like logistic and knn by about 2 percent.

### Advantages of committee machines over simple methods

Bagging averages various trees created using bootstraped data. It simulates the population properties by bootrstrapping. Whereas logistic regression and knn do not do that. Their only reference is the training set. The size of the training set of the current data is 380, which is not much.

Random forests capture the uncorrelated aspects of the data, which is, in general hard to capture using logistic regression and knn.

Boosting uses multiple weak learners which capture unique signals in the data which are all algomerated to form a strong learner. Again this is not possible to capture these signals using logistic and knn, unless we overfit the data. Boosting can also overfit, but does that slowly. Also, the risk of overfitting is reduced because, the majority vote is taken across multiple trees.

KNN may not perform well when the data is sparce in heigher dimensions. Our data set has 14 dimensions and just 379 variables.

Logistic regression considers a linear decision boundary(in our case above). There is a good chance that our data is non-linear in nature.

Disadvantage of boosting is that there are three tuning parameters - tree count, shrinkage, and interaction depth. While simple models do not have that many(knn has only one, i.e k.)