Comparing Linear Regression, Random Forests and Gradient Boosting Machines: A Toy Example

Elio Amicarelli

May 2016

Summary

This report compares the predictive performances of Generalized Linear Models (GLM) with those obtained from Random Forests (RF) and Gradient Boosting Machines (GBM) on the Boston Housing Dataset. In the GLM framework the model for the mean of the response is obtained via automated model selection governed by the second order Akaike Information Criterion (AICc). The variance component is modeled on a Gaussian distribution and the mean-variance relationship specified via an identity link function. Partial residuals are used in order to identify and fix misspecifications in the form of the predictors. The final GLM out-of-sample predictions produce a mean squared error (MSE) of 18.35. RF and GBM are trained and their optimal parameters are selected via 10-fold cross-validation repeated 10 times. RF and GBM out-of-sample predictions produce a MSE of 8.11 and 7.19 respectively.

1 Libraries

```
library(MASS)
library(car)
library(caret)
library(MuMIn)
library(randomForest)
library(xgboost)
```

2 Data

The Boston Housing Dataset contains 506 observations collected by the U.S Census Service concerning housing in the area of Boston Mass. This dataset is often used in toy applications in order to benchmark algorithms. It contains the following variables:

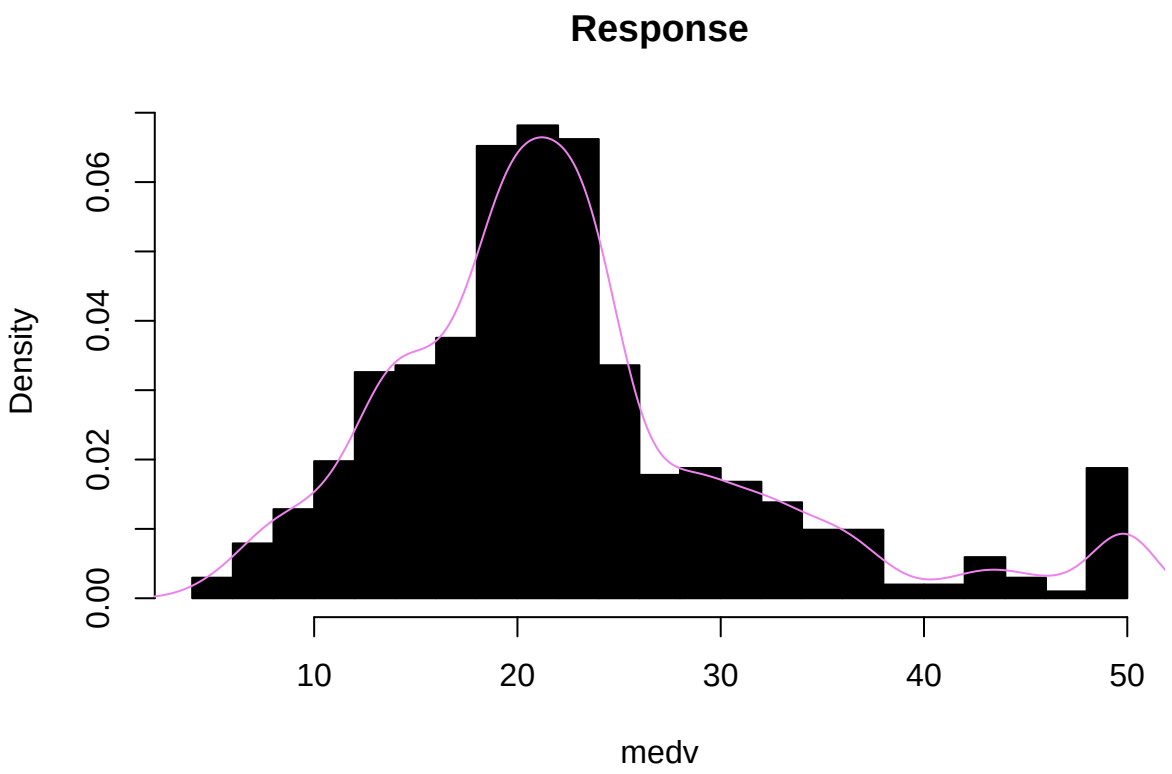
- CRIM - per capita crime rate by town
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.
- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling
- AGE - proportion of owner-occupied units built prior to 1940
- DIS - weighted distances to five Boston employment centres
- RAD - index of accessibility to radial highways
- TAX - full-value property-tax rate per $10,000
- PTRATIO - pupil-teacher ratio by town
- BLACK - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT - % lower status of the population
- MEDV - Median value of owner-occupied homes in $1000's

From a quick description of the variables, it is possible to see that there are no missing values and that the minimum and maximum measurements assume plausible values.

```
data<-data.frame(Boston)
data$chas<-as.factor(data$chas)
summary(data)
```

```
##       crim                zn              indus        chas
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   0:471
##  1st Qu.: 0.08204   1st Qu.:  0.00   1st Qu.: 5.19   1: 35
##  Median : 0.25651   Median :  0.00   Median : 9.69
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74
##       nox               rm              age              dis
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
##       rad              tax           ptratio          black
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
##  Median : 5.000   Median :330.0   Median :19.05   Median :391.44
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90
##      lstat            medv
##  Min.   : 1.73   Min.   : 5.00
##  1st Qu.: 6.95   1st Qu.:17.02
##  Median :11.36   Median :21.20
##  Mean   :12.65   Mean   :22.53
##  3rd Qu.:16.95   3rd Qu.:25.00
##  Max.   :37.97   Max.   :50.00
```

Given the nature and distribution of the response variable, in the next section I will start with a Gaussian specification for the GLM error component.



Before proceeding with the model fitting phase, I operate an 80/20 train/test split on the data.

```
set.seed(1)
j.train<-sample(1:nrow(data), nrow(data)*0.8)
data.train<-data[j.train,]
data.test<-data[setdiff(1:nrow(data),j.train),]
```

3 Modelling

3.1 Generalized Linear Model

Model Selection and Model Fitting

I start by fitting a naive GLM to the train data. The model for the mean is specified using all the available predictors in their linear form and the error component relies on a gaussian-identity specification.

```
glm.naive<-glm(medv ~ ., data=data.train,family=gaussian(link = "identity"))
```

The summary below shows the Variance Inflation Factors (VIFs) for the predictors.

```
##     crim       zn    indus     chas      nox       rm      age      dis
## 1.815883 2.353469 4.077988 1.097070 4.315394 1.909195 3.011277 4.009913
##      rad      tax  ptratio    black    lstat
## 7.756386 9.311437 1.858313 1.295534 2.951501
```

As can be seen, the VIFs for tax and rad are quite high. The summary below shows that after removing the variable with the highest VIF (tax) from the fitting the levels of collinearity among variables drop to less concerning values. This should improve the stability during the model selection and estimation phases.

```
glm.naive<-glm(medv ~ .-tax, data=data.train,family=gaussian(link = "identity"))
```

```
##     crim       zn    indus     chas      nox       rm      age      dis
## 1.815874 2.280458 3.212312 1.085330 4.300008 1.899587 3.007789 4.004604
##      rad  ptratio    black    lstat
## 2.964049 1.852791 1.294632 2.949744
```

Since I neither have substantive knowledge on the Housing market nor I am interested in testing particular hypotheses on the model coefficients, I proceed with an automated model selection governed by a second order Akaike Information Criterion (AICc). As showed in the summary below, given the set of predictors available and their specification the best performing model includes all of them except for *age* and *indus*.

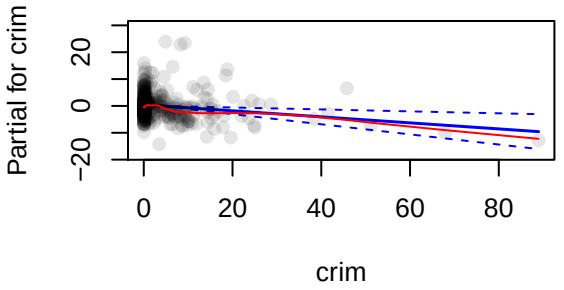
```
## Fixed term is "(Intercept)"
```

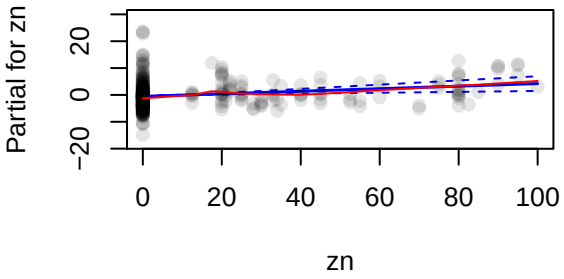
```
## Global model call: glm(formula = medv ~ . - tax, family = gaussian(link = "identity"),
##     data = data.train)
## ---
## Model selection table
##      (Intrc)      age    black chas    crim    dis    indus   lstat    nox
## 4063   45.25          0.007603    + -0.1124 -1.704          -0.5939 -23.12
## 4095   45.34          0.007363    + -0.1144 -1.773 -0.08829 -0.5888 -21.13
## 4064   45.34 0.003304 0.007577    + -0.1124 -1.689          -0.5978 -23.37
## 4096   45.43 0.003705 0.007333    + -0.1144 -1.756 -0.08858 -0.5930 -21.41
## 4059   45.27          0.008017      -0.1172 -1.708          -0.6017 -22.06
## 4091   45.34          0.007823      -0.1192 -1.768 -0.07719 -0.5975 -20.28
##       ptrat    rad    rm      zn df    logLik   AICc delta weight
## 4063 -1.112 0.1604 3.184 0.04759 12 -1207.013 2438.8  0.00  0.337
## 4095 -1.081 0.1701 3.089 0.04695 13 -1206.012 2439.0  0.13  0.315
## 4064 -1.116 0.1613 3.164 0.04795 13 -1206.989 2440.9  2.09  0.119
## 4096 -1.085 0.1712 3.067 0.04735 14 -1205.982 2441.0  2.22  0.111
## 4059 -1.148 0.1629 3.212 0.04571 11 -1209.664 2442.0  3.18  0.069
## 4091 -1.122 0.1715 3.131 0.04507 12 -1208.906 2442.6  3.79  0.051
## Models ranked by AICc(x)
```

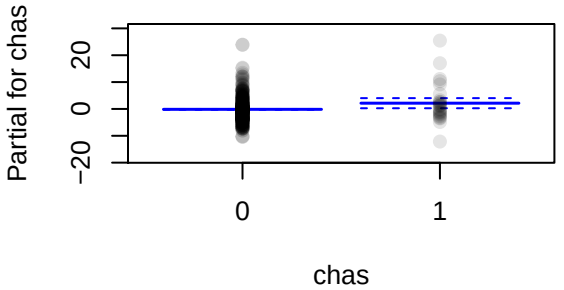
The new GLM specification is:

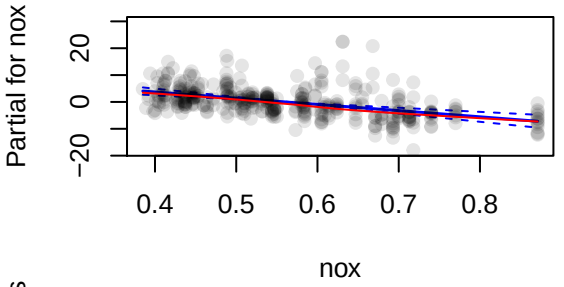
```
model.glm<-glm(medv ~ .-tax -age -indus,data=data.train,family=gaussian(link="identity"))
```

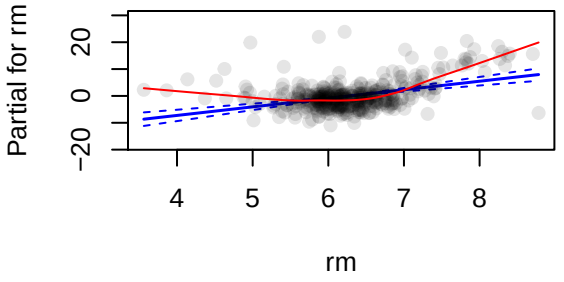
An investigation of the partial residuals from the figure below reveals that for the variables *rm*, *lstat* and *crim* a different specification would be more appropriate.

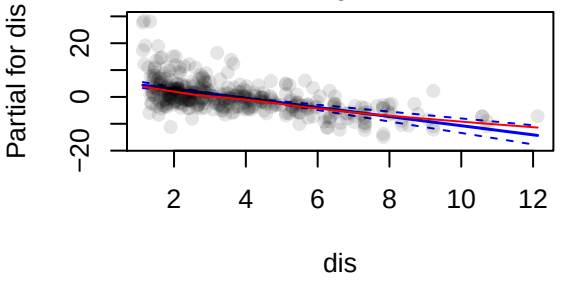

Partial for crim
−20
0
20
0
20
40
60
80
crim

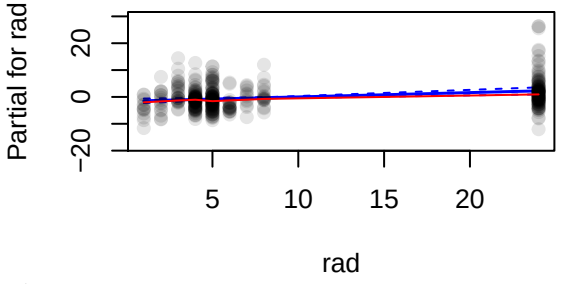

Partial for zn
−20
0
20
0
20
40
60
80
100
zn

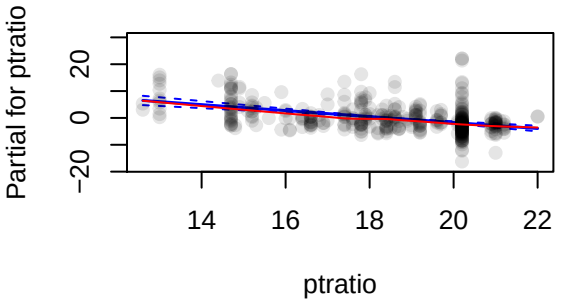

Partial for chas
−20
0
20
0
1
chas

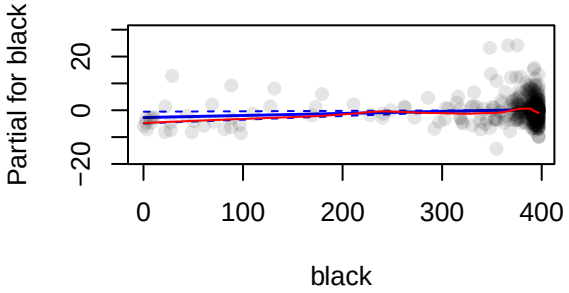

Partial for nox
−20
0
20
0.4
0.5
0.6
0.7
0.8
nox

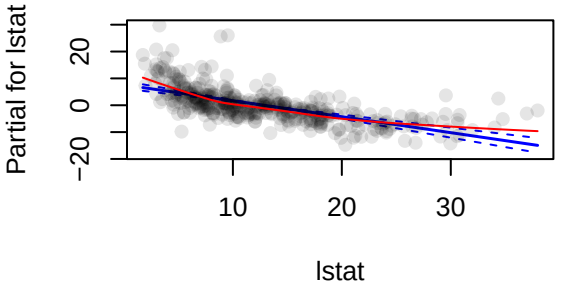

Partial for rm
−20
0
20
4
5
6
7
8
rm


Partial for dis
−20
0
20
2
4
6
8
10
12
dis


Partial for rad
−20
0
20
5
10
15
20
rad


Partial for ptratio
−20
0
20
14
16
18
20
22
ptratio


Partial for black
−20
0
20
0
100
200
300
400
black


Partial for lstat
−20
0
20
10
20
30
lstat

In order to ameliorate the specification for *rm* and *lstat*, the following transformations are applied:

```
data.train$rm2<-(1+1)/(max(data.train$rm)-min(data.train$rm))*
  (data.train$rm-max(data.train$rm))+1
data.train$rm2<-data.train$rm2^2
data.train$lstat2<-log(data.train$lstat)
data.train$crim2<-log(data.train$crim)
```

Interactions between predictors could be valuable but given my lack of knowledge about the housing market I prefer not to model this aspect. The final model is:

```
workingmodel.glm<-glm(medv ~ zn + chas + nox + dis + rad + ptratio + black + lstat2 +
                        rm2 + crim2,data=data.train,family=gaussian(link = "identity"))
```

An AIC-based comparison between the model with all linear terms and the model with transformed predictors confirms the superiority of the latter.

```
AIC(workingmodel.glm, model.glm )
```

```
##                   df      AIC
## workingmodel.glm 12 2332.069
## model.glm        12 2438.025
```

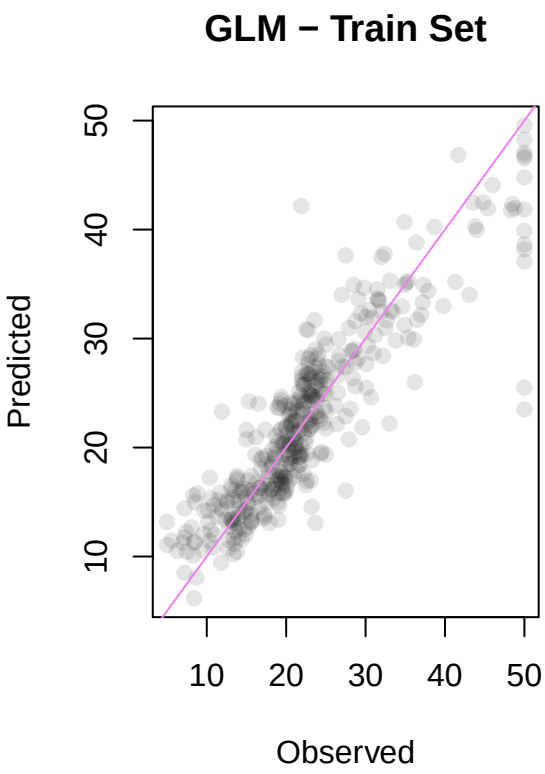
The model with transformed predictors is thus adopted as final glm model.

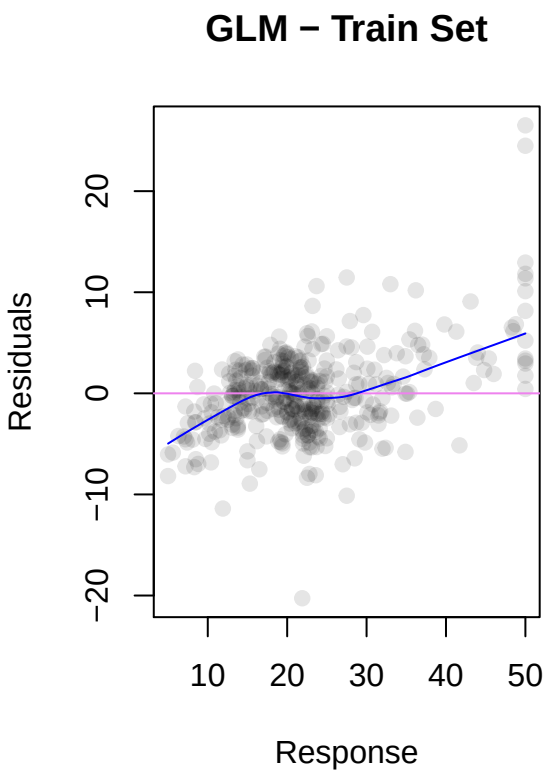
```
finalmodel.glm<-workingmodel.glm
```

Model Checking

The left and right panels in the figure below show the final GLM predicted values against the response values and the residuals against the response respectively. As can be seen, the model is slightly overpredicting the response at its lower values and is underpredicting it only at its maximum value (medv = 50). Considering the fitting behavior across the entire range of the response, the latter aspect seems quite strange and may point to a possible censoring of the *medv* variable in the original dataset.

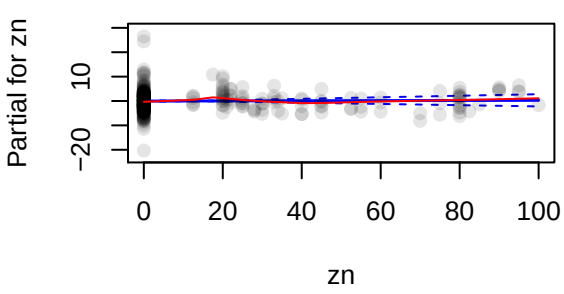
```
par(mfrow=c(1,2))

plot(data.train$medv,finalmodel.glm$fitted.values, main="GLM - Train Set",
     ylab="Predicted",xlab="Observed", pch=19, col=rgb(0,0,0,0.1))
abline(0,1, col="violet")

plot(data.train$medv,finalmodel.glm$residuals, main="GLM - Train Set", ylab="Residuals",
     xlab="Response", pch=19, col=rgb(0,0,0,0.1))
abline(h=0, col="violet")
lines(lowess(data.train$medv,finalmodel.glm$residuals), col="blue")
```

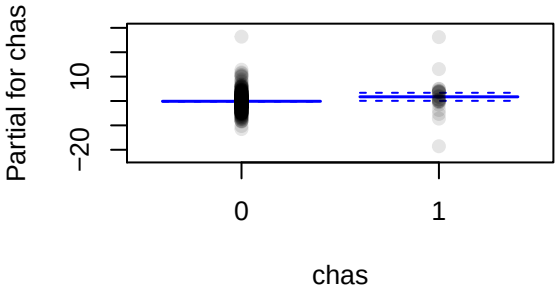


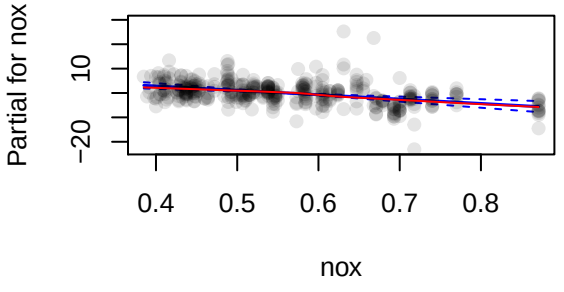


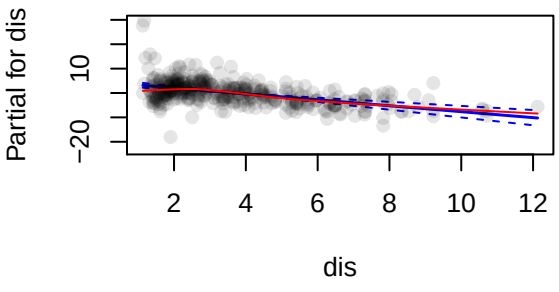
The following graphs show the partial residuals from the final GLM model. No serious problems regarding the specification form of the predictors can be detected.

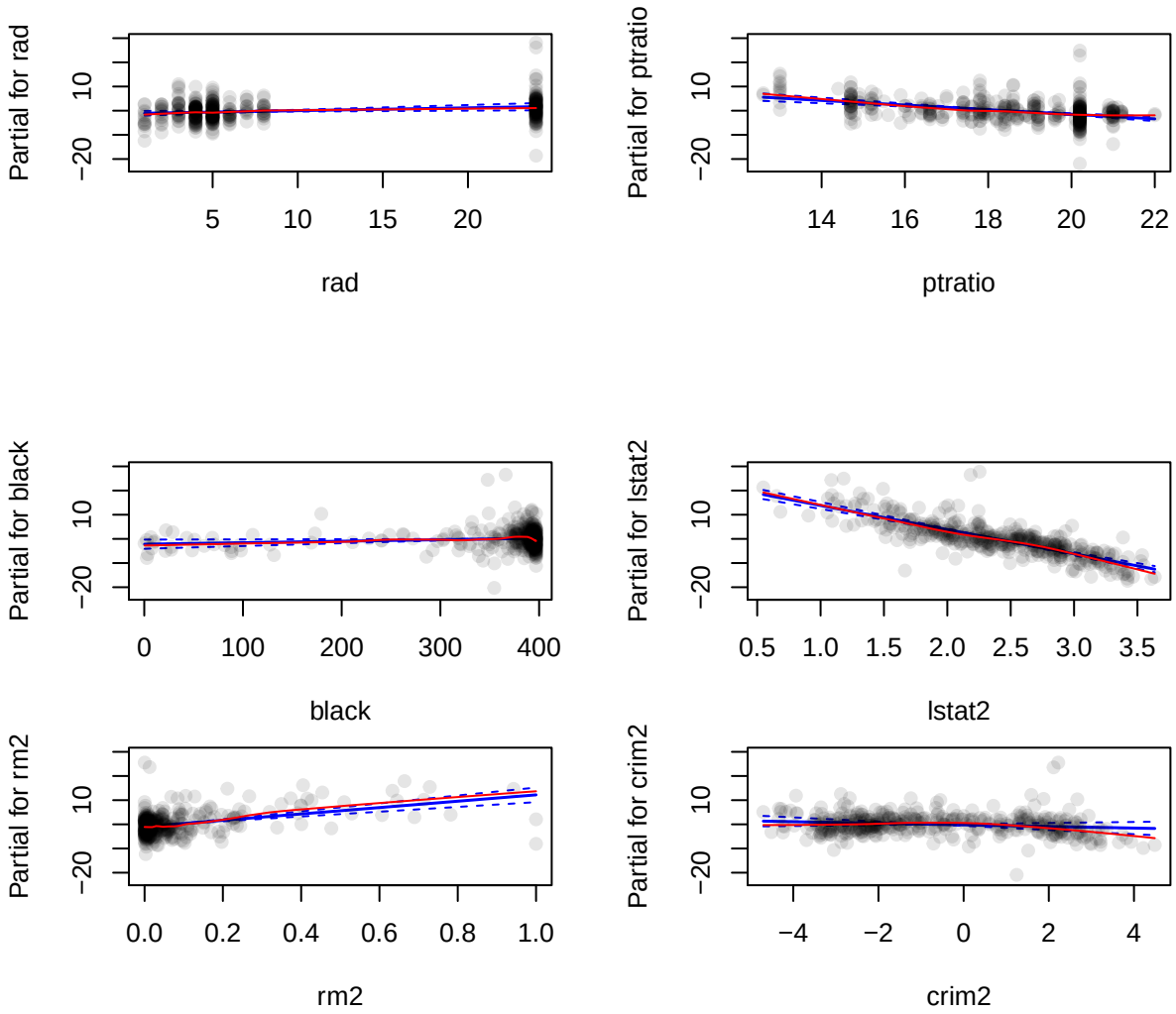
```
par(mfrow=c(2,2))
termplot(finalmodel.glm, partial.resid=TRUE, pch=19,col.res = rgb(0,0,0,0.1),col.term="blue",
         smooth=panel.smooth, col.smth = "red",lty.smth=1,se=T,col.se = "blue")
```





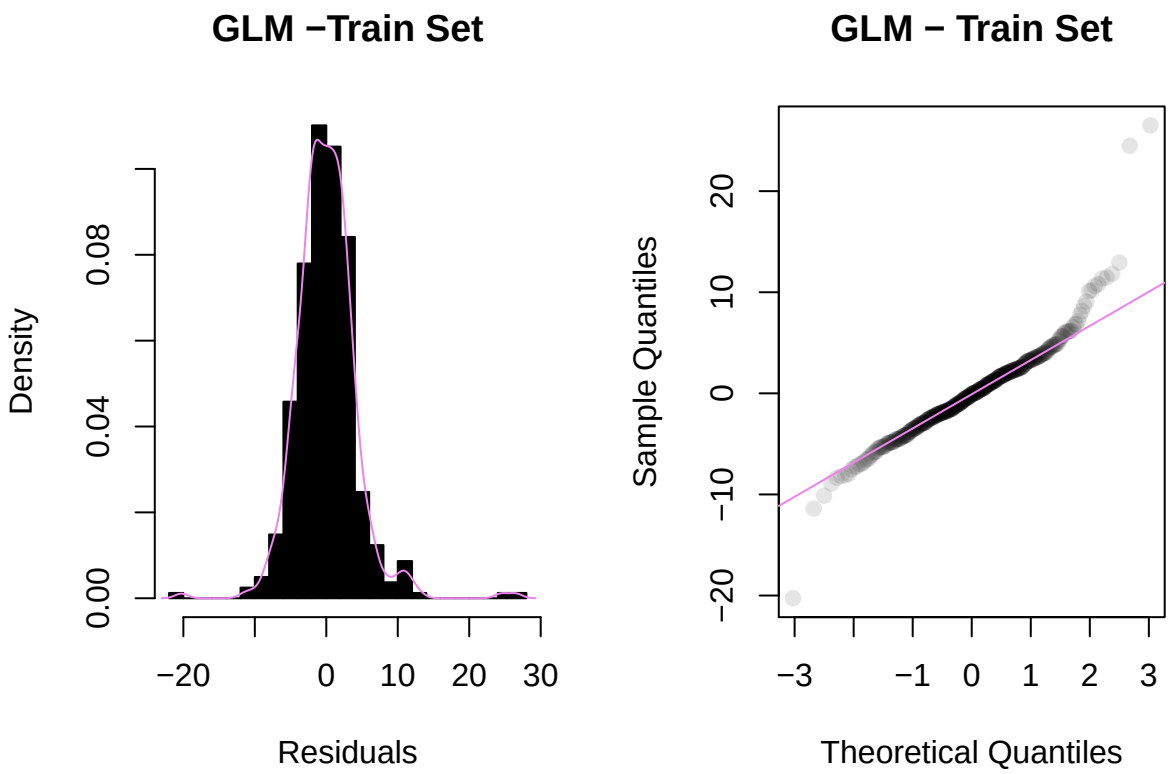






The figure below shows that the regression residuals are approximately normally distributed.

```
par(mfrow=c(1,2))
hist(finalmodel.glm$residuals, main="GLM -Train Set",xlab="Residuals",
     breaks=20,col="black",probability=T)
lines(density(finalmodel.glm$residuals),col="violet")

qqnorm(finalmodel.glm$residuals, main="GLM - Train Set", pch=19, col=rgb(0,0,0,0.1))
qqline(finalmodel.glm$residuals,col="violet")
```



Overall the GLM seems to fit the data well. In the next section I evaluate the GLM out-of-sample predictive performance.

Model Predictive Performances

Here I apply the same variables transformations used in the train set to the test set.

```
data.test$rm2<-(1+1)/(max(data.test$rm)-min(data.test$rm))*
  (data.test$rm-max(data.test$rm))+1
data.test$rm2<-data.test$rm2^2
data.test$lstat2<-log(data.test$lstat)
data.test$crim2<-log(data.test$crim)
```

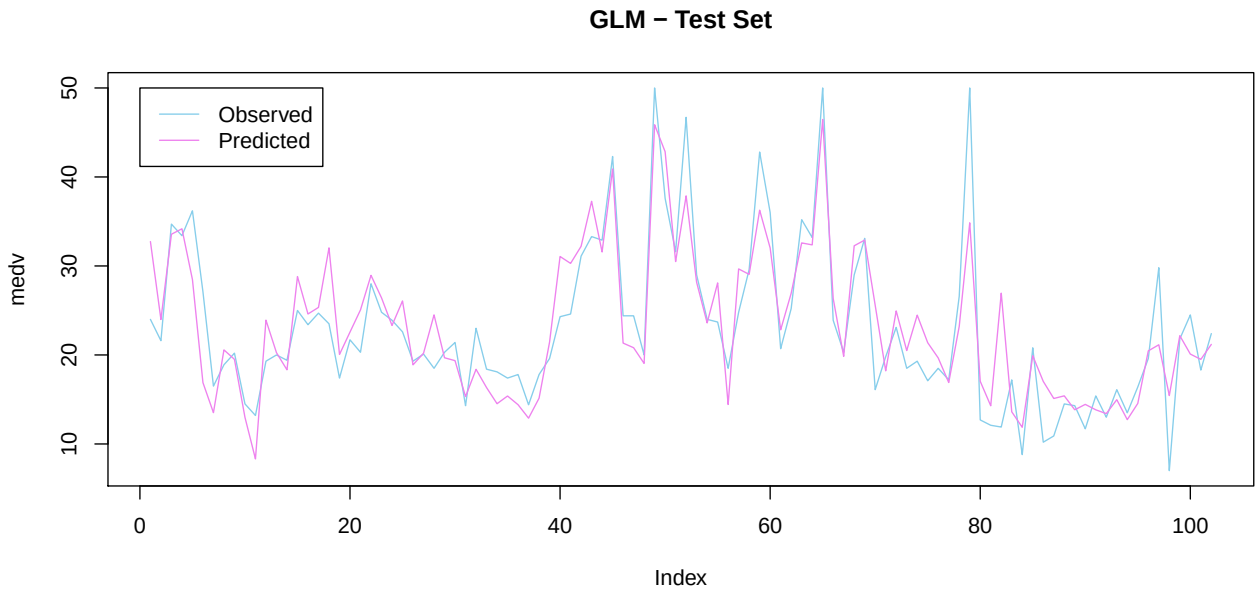
Now I move to the evaluation of the GLM on the test data.

Obtaining predictions:

```
glm.preds<-predict(finalmodel.glm, data.test)
```

A visual comparison of out-of-sample predictions with the observed values:

```
plot(data.test$medv, main="GLM - Test Set", ylab="medv",type="l", col= "skyblue")
lines(glm.preds,col="violet")
legend(0, 50, c("Observed","Predicted"), lty=c(1,1), col=c("skyblue","violet"))
```



The mean squared error is finally calculated and will be used as metric to compare the GLM performances with those of Random Forests and Gradient Boosting Machines:

```
GLM.MSE<-mean((data.test$medv-glm.preds)^2)
cat("GLM out of sample MSE:",GLM.MSE)
```

```
## GLM out of sample MSE: 18.35538
```

3.2 Random Forest

Model Training

First I remove the variables created for the GLM from the data and recode some variables according to algorithms needs.

```
data.train<-data.train[,!names(data.train)%in%c("lstat2","rm2","crim2")]
data.test<-data.test[,!names(data.test)%in%c("lstat2","rm2","crim2")]

data.train$chas<-as.numeric(data.train$chas)
data.train$rad<-as.numeric(data.train$rad)
data.test$chas<-as.numeric(data.test$chas)
data.test$rad<-as.numeric(data.test$rad)
```

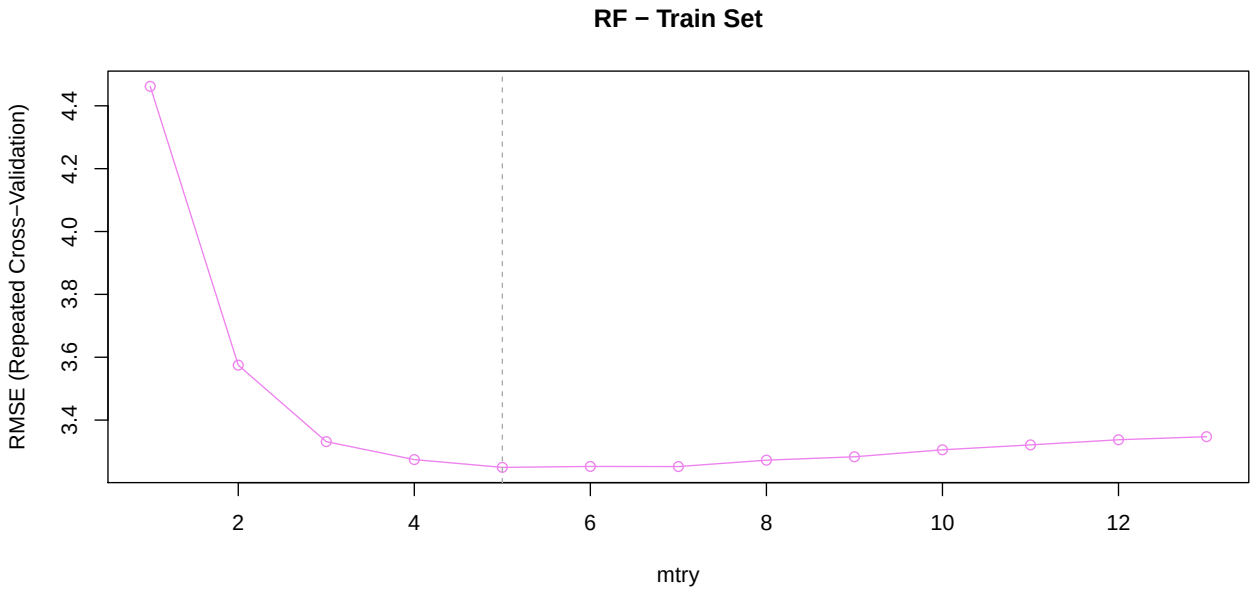
It's time to set the parameters for the grid search and the validation procedure.

```
rf.ctrl <-trainControl(method="repeatedcv",
                       number=10,
                       repeats=10,
                       verboseIter=TRUE,
                       savePredictions = TRUE)

rf.myGrid<-expand.grid(mtry=seq(1,13,by=1))
```

Now I train the model:

```
set.seed(1)
rf.modelspace = train(medv ~ ., data = data.train,
                      method = "rf",
                      trControl = rf.ctrl,
                      tuneGrid = rf.myGrid,
                      ntree=500,
                      metric="RMSE")

finalmodel.rf<-rf.modelspace$finalModel
```

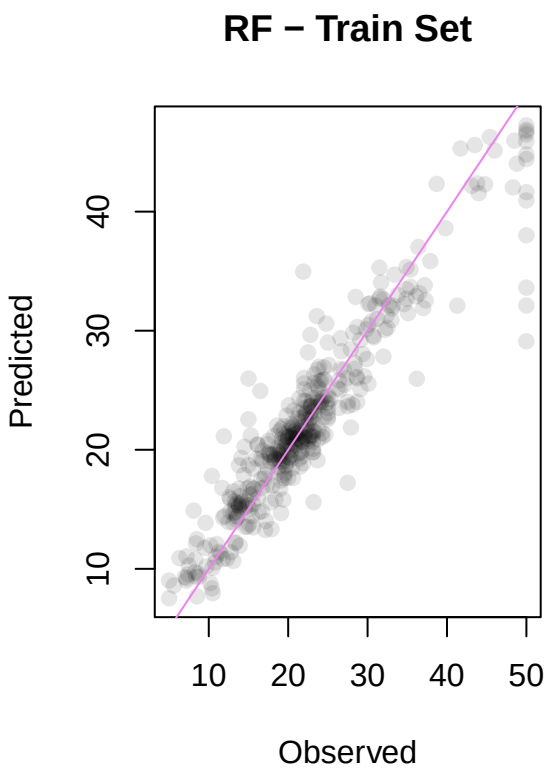
The figure below shows that the RMSE evaluated via 10-fold cross-validation is minimized when the *mtry* parameter (number of predictors sampled for each tree) is equal to 5. From the graph it is also clear how overfitting startsafter $mtry = 7$. The final model is a RF with 500 trees and 5 predictors sampled for each tree.

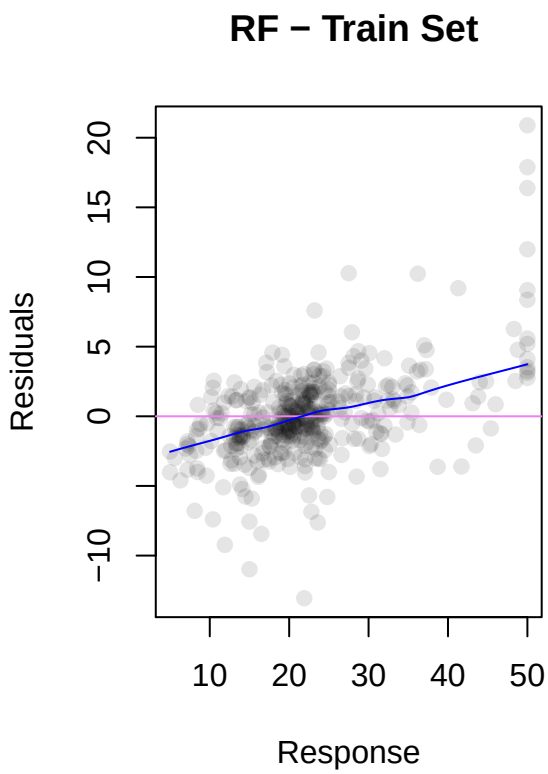


Model Checking

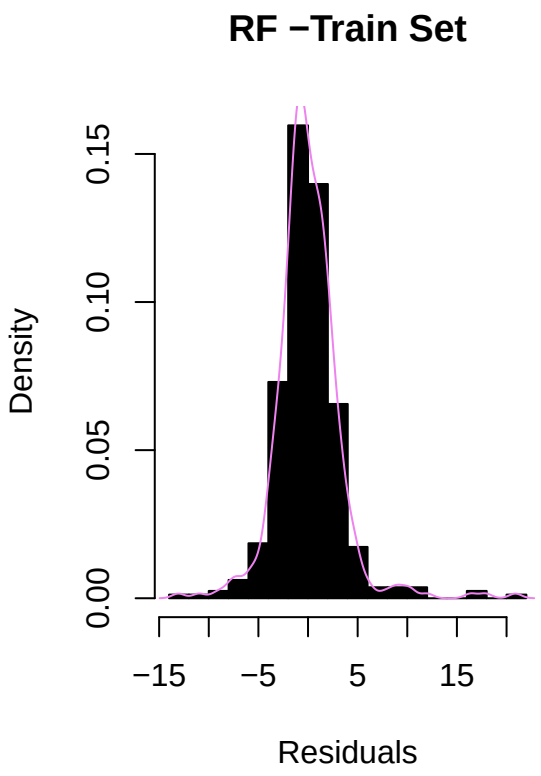
As can be seen from the next two figures, the RF residuals' behavior is similar to those observed in the case of GLM. Underprediction of some maximum *medv* values is still an issue.

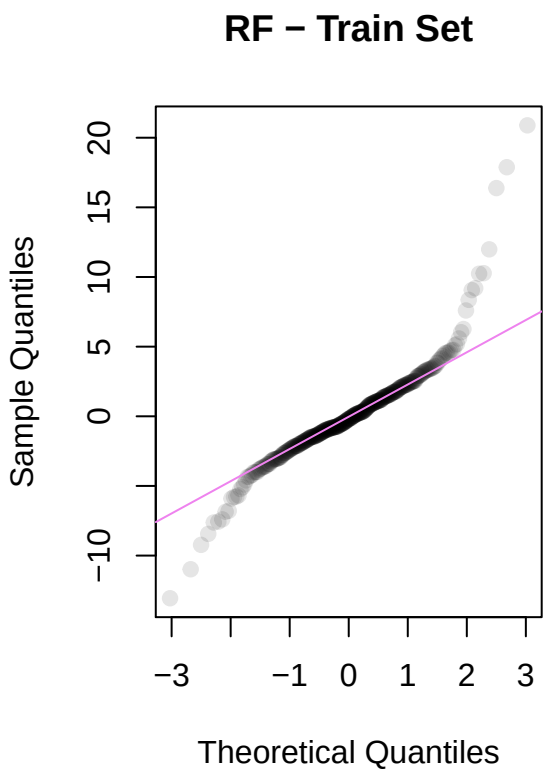
```
par(mfrow=c(1,2))

plot(data.train$medv,finalmodel.rf$predicted, main="RF - Train Set",
     ylab="Predicted",xlab="Observed", pch=19, col=rgb(0,0,0,0.1))
abline(0,1, col="violet")

plot(data.train$medv,data.train$medv-finalmodel.rf$predicted, main=" RF - Train Set",
     ylab="Residuals",xlab="Response", pch=19, col=rgb(0,0,0,0.1))
abline(h=0, col="violet")
lines(lowess(data.train$medv,data.train$medv-finalmodel.rf$predicted),col="blue")
```






```
par(mfrow=c(1,2))

hist(data.train$medv-finalmodel.rf$predicted,main="RF -Train Set",xlab="Residuals",
     breaks=20,col="black",probability=T)
lines(density(data.train$medv-finalmodel.rf$predicted),col="violet")

qqnorm(data.train$medv-finalmodel.rf$predicted, main="RF - Train Set",
       pch=19, col=rgb(0,0,0,0.1))
qqline(data.train$medv-finalmodel.rf$predicted,col="violet")
```





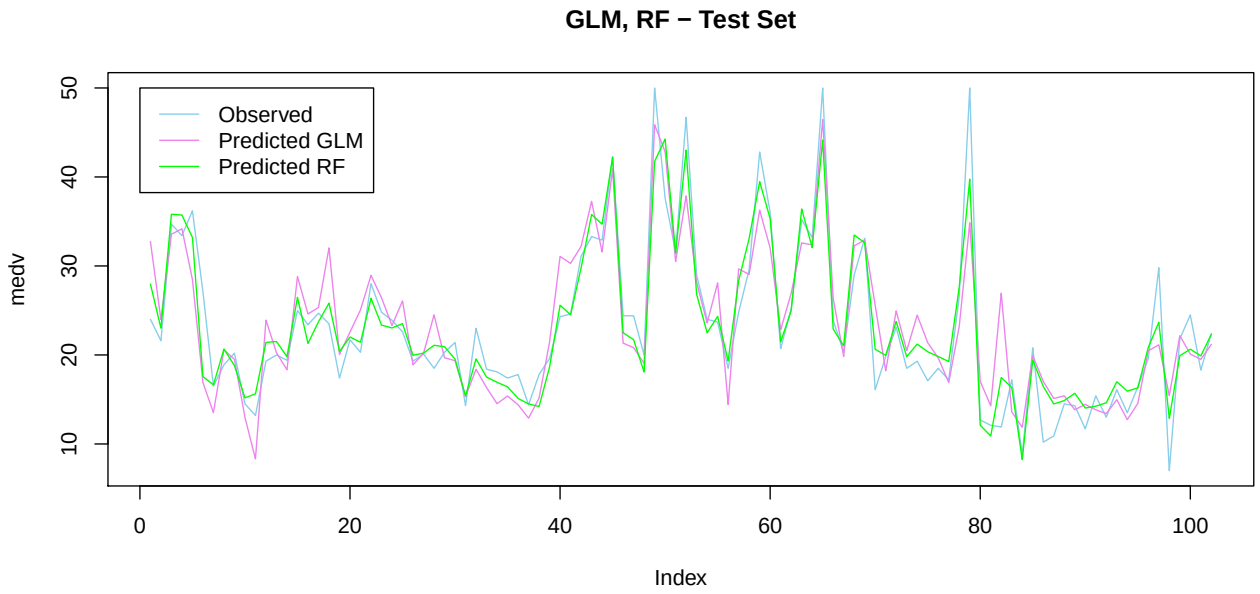
Model Predictive Performances

Now I move to a quick examination or RF out-of-sample predictive performances. First, I generate the prediction for the test-set observations.

```
rf.preds<-predict(finalmodel.rf, data.test)
```

From a a visual comparison of the out-of-sample predictions, it seems that the RF is modelling the underlying signal more closely then the GLM:

```
plot(data.test$medv, main="GLM, RF - Test Set", ylab="medv",type="l", col= "skyblue")
lines(glm.preds,col="violet")
lines(rf.preds,col="green")
legend(0, 50, c("Observed","Predicted GLM", "Predicted RF"), lty=c(1,1,1),
       col=c("skyblue","violet", "green"))
```



To summarize the RF out-of-sample performance, the mean squared error is finally calculated:

```
RF.MSE<-mean((data.test$medv-rf.preds)^2)
cat("GLM out of sample MSE: ",GLM.MSE,"\n",
    "RF out of sample MSE: ",RF.MSE, sep="")
```

```
## GLM out of sample MSE: 18.35538
## RF out of sample MSE: 8.109215
```

3.2 Gradient Boosting Machine

Model Training

The following chunk of code sets the parameters for the grid search and the validation procedure for the GBM. Again, I use 10-fold cross-validation repeated 10 times in order to select the relevant model's parameters.

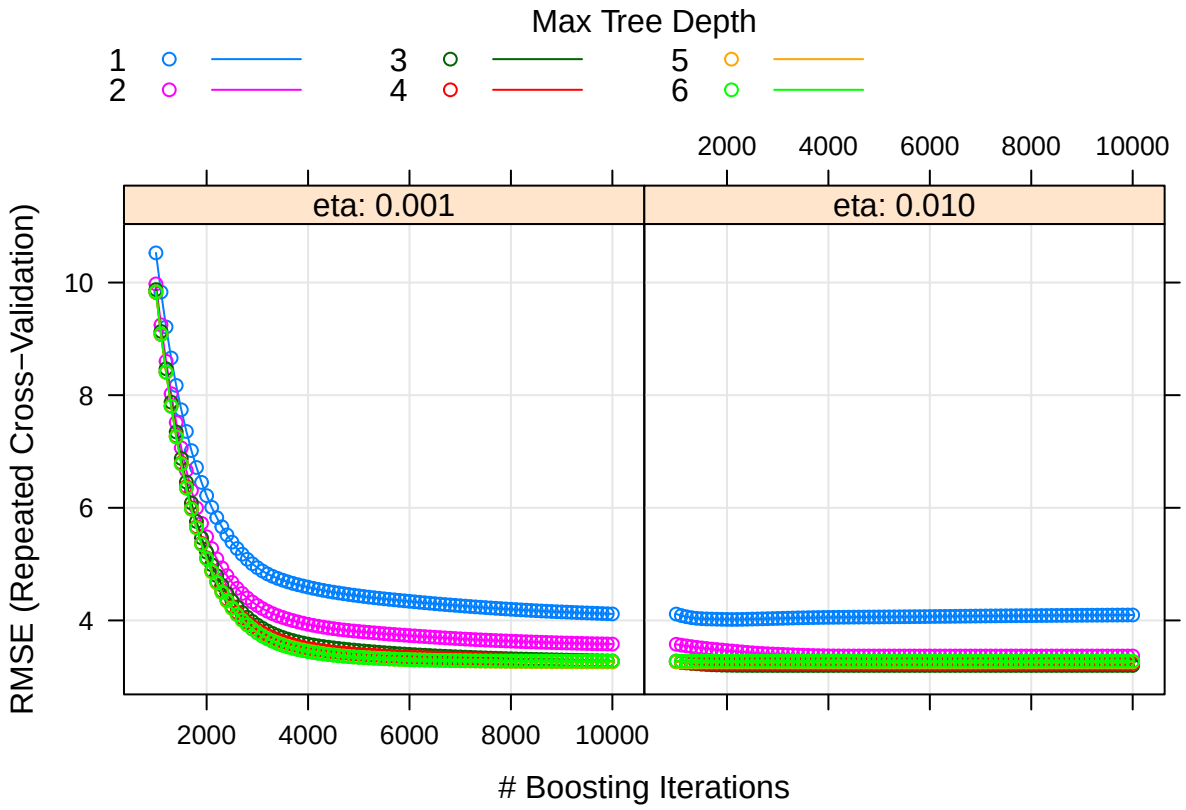
```
gbm.ctrl<- trainControl(
  method = "repeatedcv",
  number= 10,
  repeats = 10,
  verboseIter = TRUE,
  returnData = FALSE,
  returnResamp = "all",
  allowParallel = TRUE
)

gbm.myGrid<-expand.grid(
  nrounds = seq(1000, 10000, 100),
  eta = c(0.01, 0.001),
  max_depth = seq(1:6),
  gamma = 1,
  colsample_bytree=1,
  min_child_weight = 1
)
```

Train the GBM algorithm:

```
set.seed(1)
gbm.modelspace = train(medv ~ ., data = data.train,
                       method = "xgbTree",
                       trControl = gbm.ctrl,
                       tuneGrid = gbm.myGrid,
                       metric="RMSE")
```

```
## Loading required package: plyr
```

The final model has a shrinkage parameter of 0.01 and ensembles 3200 sequentially grown third-order regression trees.

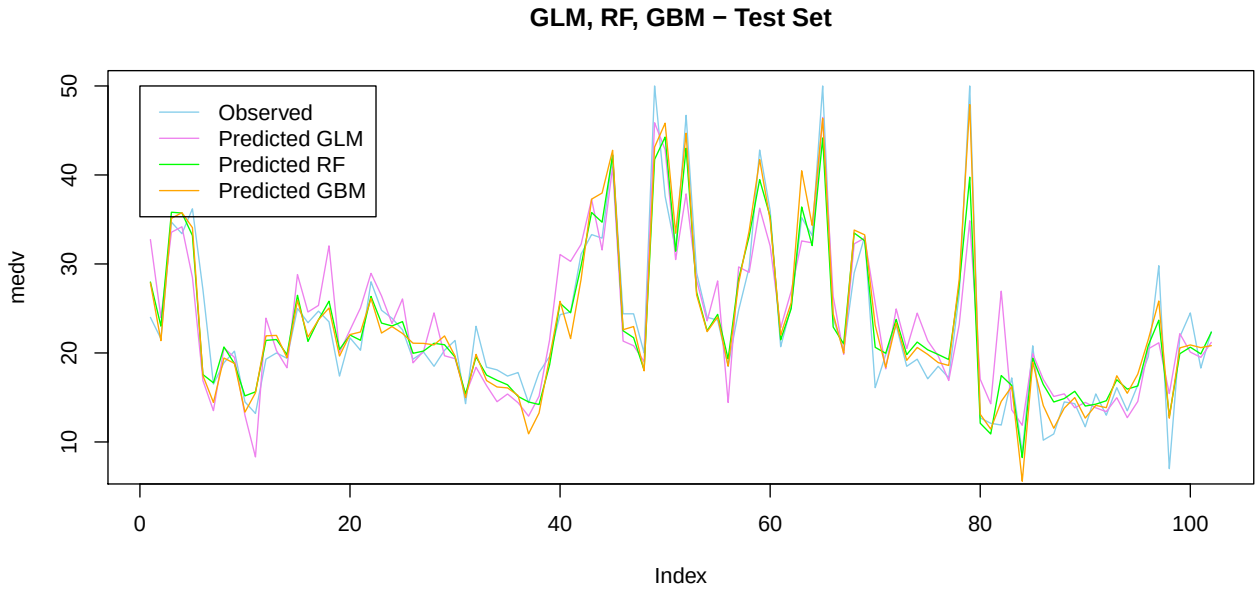


Model predictive performances

```
gbm.preds<-predict(gbm.modelspace, data.test)
```

A visual comparison:

```
plot(data.test$medv, main="GLM, RF, GBM - Test Set",ylab="medv",type="l", col= "skyblue")
lines(glm.preds,col="violet")
lines(rf.preds,col="green")
lines(gbm.preds,col="orange")
legend(0, 50, c("Observed","Predicted GLM", "Predicted RF", "Predicted GBM"),
       lty=c(1,1,1), col=c("skyblue","violet", "green", "orange"))
```



The mean squared error is finally used as metric for the performance assessment:

```
GBM.MSE<-mean((data.test$medv-gbm.preds)^2)
cat("GLM out of sample MSE: ",GLM.MSE,"\n",
    "RF out of sample MSE: ",RF.MSE,"\n",
    "GBM out of sample MSE: ",GBM.MSE, sep = "")
```

```
## GLM out of sample MSE: 18.35538
## RF out of sample MSE: 8.109215
## GBM out of sample MSE: 7.194892
```

Conclusion

In this report I modelled the Boston Housing Data and compared the predictive performances of Generalized Linear Models (GLM), Random Forests (RF), Gradient Boosting Machines (GBM). The analysis of residuals shows that the data have been succesfully modelled under the GLM framework. However, because of my lack of substantive knowledge on the housing market, I did not model potentially important aspects of the underlying process (e.g. important interactions among predictors). Even though I only had 13 predictors, the aforementioned limitation is often coupled with another important aspect: high data-complexity (especially large p) makes predictive modelling under the GLM framework a difficult task. I estimated the discrepancy between the response values in the test set and the out-of-sample predictions using the MSE. The GLM out-of-sample predictions corresponded to a MSE of ~18.35. After GLM I moved to the application of RF and GBM, two of the most used algorithms in the Machine Learning predictive community. During RF and GBM training I tamed the tradeoff between model complexity and the risk of overfitting the data using 10-fold cross-validation (repeated 10 times). Subsequently I produced out-of-sample predictions on the same test set previously used to assess the GLM predictive performance. RF and GBM produced a test MSE of ~8.11 and ~7.19 thus establishing the following order among competing models: $GBM > RF > GLM$.