HST.953x Workshop 2.09: Ensemble Methods Exercises

H. David Shea

16 Jul 2021

Contents

(Note: Updated and modified from "R for Statistical Learning".)

Description

This section contains examples of ensemble - bagging and boosting - methods.

Dataset

We will be usingthe `Boston` dataset from the `MASS` package for these examples. The dataset contains housing value data for the suburbs of Boston.

- **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** - nitrogen 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 mean of 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 (percent).
- **medv** - median value of owner-occupied homes in $1000s.

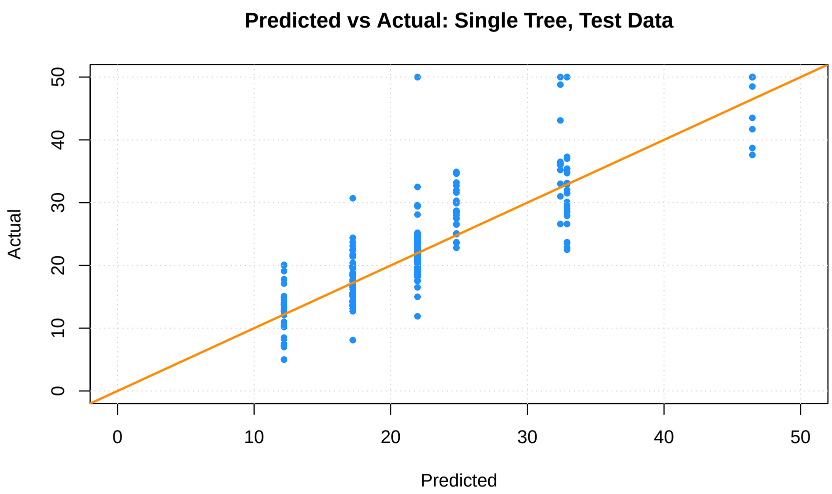
```
base_dir <- here::here("")

calc_rmse = function(actual, predicted) {
  sqrt(mean((actual - predicted) ^ 2))
}

str(Boston)
#> 'data.frame':    506 obs. of  14 variables:
#>  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
#>  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
#>  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
#>  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
#>  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
#>  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
#>  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
#>  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
#>  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
#>  $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
#>  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
#>  $ black  : num  397 397 393 395 397 ...
#>  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
#>  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

Test-Train Split the Data

```
set.seed(18)
boston_idx <- sample(1:nrow(Boston), nrow(Boston) / 2)
boston_trn <- Boston[boston_idx,]
boston_tst <- Boston[-boston_idx,]
```

Initial Tree Model of Median Value versus all other Predictors

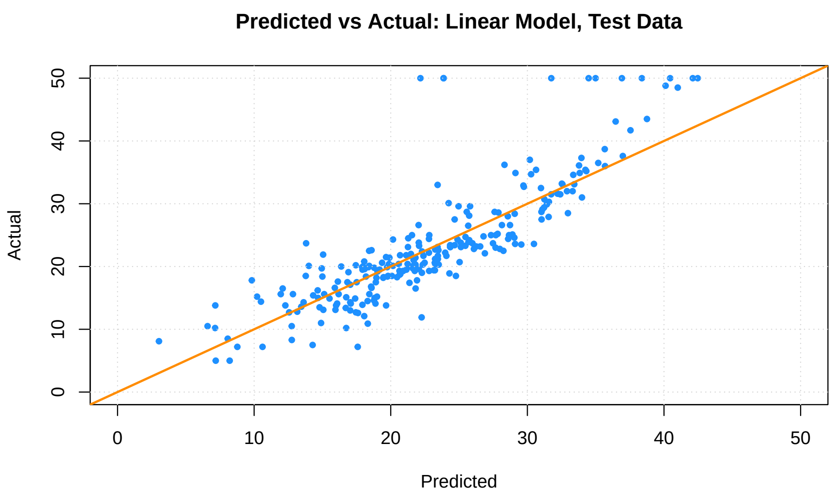
```
boston_tree <- rpart(medv ~ ., data = boston_trn)

boston_tree_tst_pred <- predict(boston_tree, newdata = boston_tst)
plot(boston_tree_tst_pred, boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Single Tree, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```




```r
# RMSE
(tree_tst_rmse <- calc_rmse(boston_tree_tst_pred, boston_tst$medv))
#> [1] 5.051138
```

Initial Linear Model of Median Value versus all other Predictors

```r
boston_lm <- lm(medv ~ ., data = boston_trn)

boston_lm_tst_pred <- predict(boston_lm, newdata = boston_tst)
plot(boston_lm_tst_pred, boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Linear Model, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```



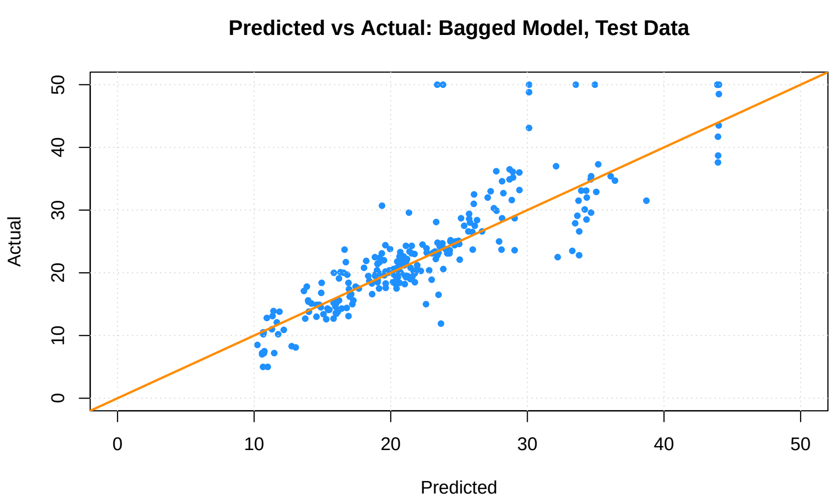
```r
# RMSE
(lm_tst_rmse <- calc_rmse(boston_lm_tst_pred, boston_tst$medv))
#> [1] 5.016083
```

Bagging

(The reference book used `randomForest` but this is not yet available for M1 machines. I am trying the `cforest` function from the `party` package as a replacement.)

Bagging is actually a special case of a random forest where `mtry` is equal to the number of predictors - 13 in this case.

```r
boston_bag <- cforest(medv ~ ., data = boston_trn, controls = cforest_unbiased(mtry = 13, ntree = 500))

boston_bag
#>
#>   Random Forest using Conditional Inference Trees
#>
#> Number of trees:  500
#>
#> Response:  medv
#> Inputs:  crim, zn, indus, chas, nox, rm, age, dis, rad, tax, ptratio, black, lstat
#> Number of observations:  253

boston_bag_tst_pred <- predict(boston_bag, newdata = boston_tst)
plot(boston_bag_tst_pred,boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Bagged Model, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```



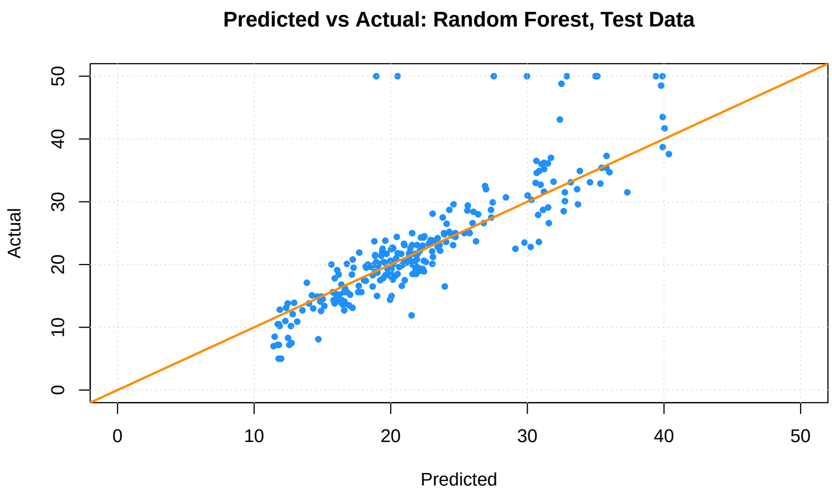
```r
(bag_tst_rmse <- calc_rmse(boston_bag_tst_pred, boston_tst$medv))
#> [1] 4.710894
```

Here we see two interesting results. First, the predicted versus actual plot no longer has a small number of predicted values. Second, our test error has dropped dramatically - note, not as dramatically as the original example did using `randomForest`.

Random Forest

We now try a random forest. For regression, the suggestion for `mtry` is number of predictors divided by 3 - 4 in this case.

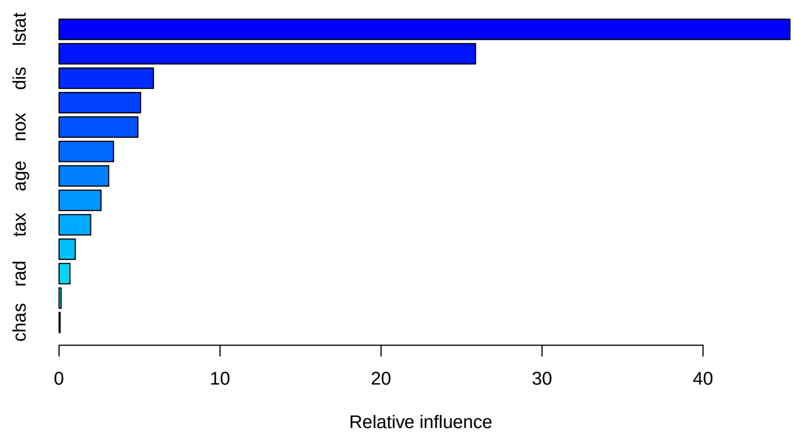
```r
boston_forest <- cforest(medv ~ ., data = boston_trn, controls = cforest_unbiased(mtry = 4, ntree = 500

boston_forest
#>
#>   Random Forest using Conditional Inference Trees
#>
#> Number of trees:  500
#>
#> Response:  medv
#> Inputs:  crim, zn, indus, chas, nox, rm, age, dis, rad, tax, ptratio, black, lstat
#> Number of observations:  253

boston_forest_tst_pred <- predict(boston_forest, newdata = boston_tst)
plot(boston_forest_tst_pred, boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Random Forest, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```



```r
(forest_tst_rmse <- calc_rmse(boston_forest_tst_pred, boston_tst$medv))
#> [1] 5.042991
```

Boosting

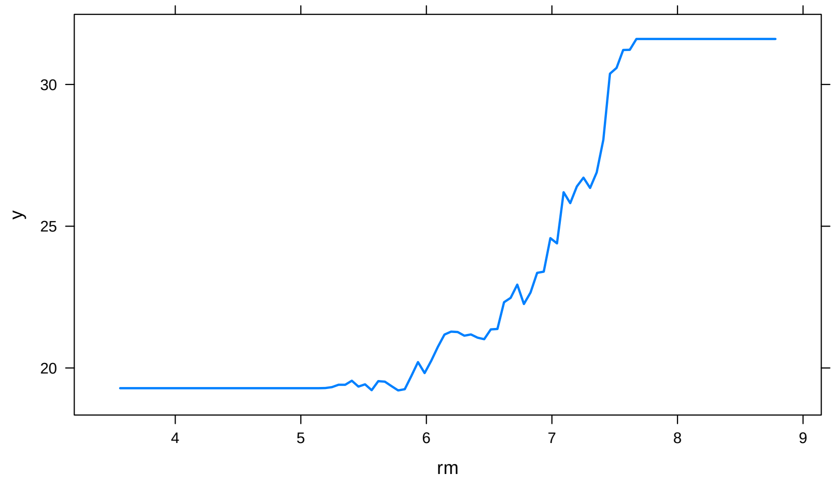
```r
booston_boost <- gbm(medv ~ ., data = boston_trn, distribution = "gaussian",
                     n.trees = 5000, interaction.depth = 4, shrinkage = 0.01)

booston_boost
#> gbm(formula = medv ~ ., distribution = "gaussian", data = boston_trn,
#>     n.trees = 5000, interaction.depth = 4, shrinkage = 0.01)
#> A gradient boosted model with gaussian loss function.
#> 5000 iterations were performed.
#> There were 13 predictors of which 13 had non-zero influence.

tibble::as_tibble(summary(booston_boost))
```



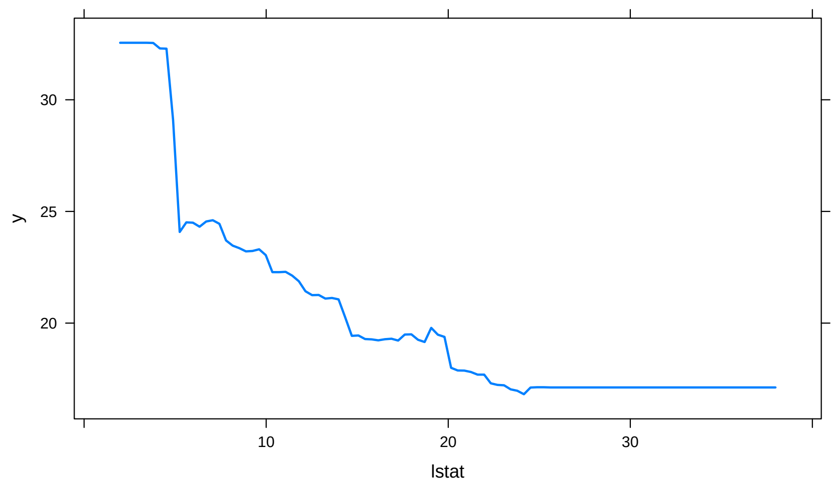

```
#> # A tibble: 13 x 2
#>    var     rel.inf
#>    <chr>     <dbl>
#>  1 lstat   45.4
#>  2 rm      25.9
#>  3 dis      5.86
#>  4 crim     5.06
#>  5 nox      4.90
#>  6 black    3.38
#>  7 age      3.08
#>  8 ptratio  2.61
#>  9 tax      1.96
#> 10 indus    1.01
#> 11 rad      0.676
#> 12 zn       0.127
```

```
#> 13 chas      0.0659
```

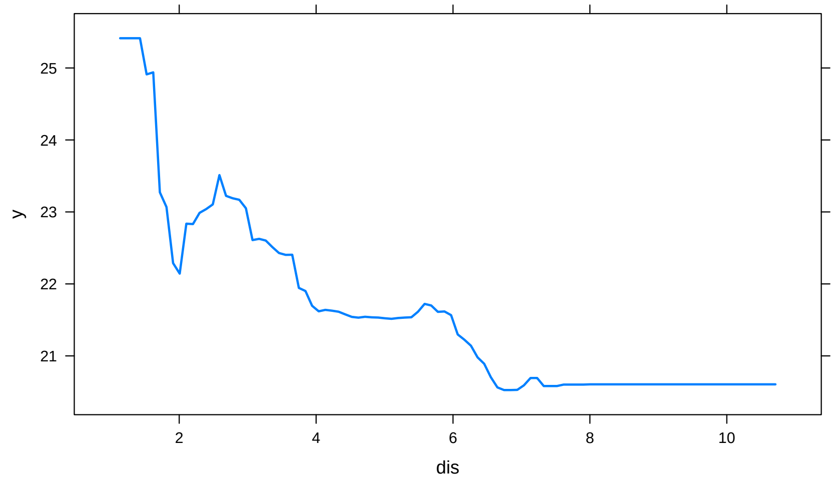
```r
par(mfrow = c(1, 3))
plot(booston_boost, i = "rm", col = "dodgerblue", lwd = 2)
```




```r
plot(booston_boost, i = "lstat", col = "dodgerblue", lwd = 2)
```

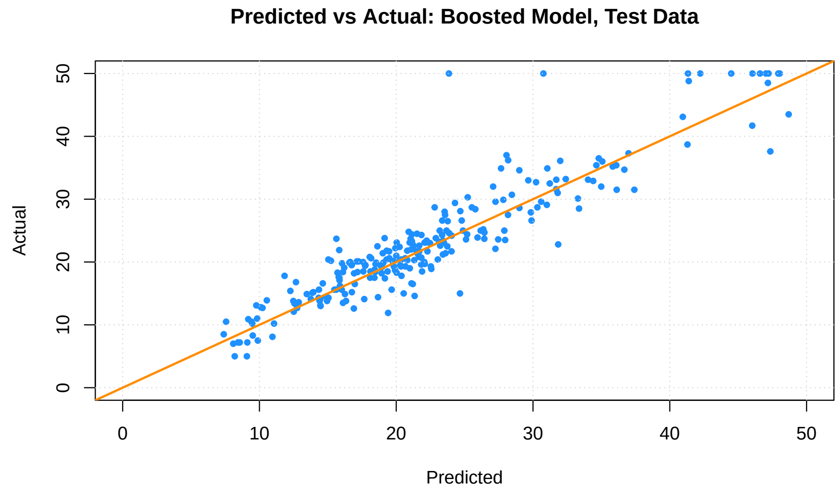



```r
plot(booston_boost, i = "dis", col = "dodgerblue", lwd = 2)
```




```
boston_boost_tst_pred = predict(booston_boost, newdata = boston_tst, n.trees = 5000)
(boost_tst_rmse <- calc_rmse(boston_boost_tst_pred, boston_tst$medv))
#> [1] 3.643578
```

```
plot(boston_boost_tst_pred, boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Boosted Model, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```

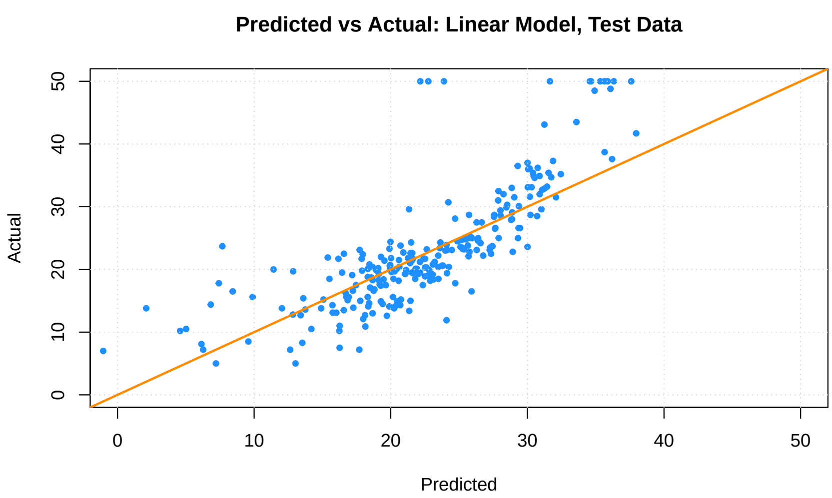


Results

Model	TestError
Single Tree	5.051138
Linear Model	5.016083
Bagging	4.710894
Random Forest	5.042992
Boosting	3.643578

Linear Model of Median Value versus two most significant predictors from boosting (`lstat` and `rm`)

```
boston_lm2 <- lm(medv ~ lstat + rm, data = boston_trn)

boston_lm2_tst_pred <- predict(boston_lm2, newdata = boston_tst)
plot(boston_lm2_tst_pred, boston_tst$medv,
     xlim = c(0,50), ylim = c(0,50),
     xlab = "Predicted", ylab = "Actual",
     main = "Predicted vs Actual: Linear Model, Test Data",
     col = "dodgerblue", pch = 20)
grid()
abline(0, 1, col = "darkorange", lwd = 2)
```




```
# RMSE
(lm2_tst_rmse <- calc_rmse(boston_lm_tst_pred, boston_tst$medv))
#> [1] 5.016083
```