STAT406- Methods of Statistical Learning Lecture 10

Matias Salibian-Barrera

UBC - Sep / Dec 2019

Curse of dimensionality

- Suppose we have $n = 100$ observations uniformly distributed on the interval $[0, 1]$.
- How many do we expect to find in $[0.25, 0.75]$?

$$0.25 \leq X_i \leq 0.75$$

$$|X_i - 0.5| \leq 0.25$$

Curse of dimensionality

- Suppose we have $n = 100$ observations uniformly distributed on the square $[0, 1] \times [0, 1]$.
- How many do we expect to find in the square $[0.25, 0.75] \times [0.25, 0.75]$?

Curse of dimensionality

- Suppose we have $n = 100$ observations uniformly distributed on the hypercube $[0, 1]^{10}$.
- How many do we expect to find in the hypercube $[0.25, 0.75]^{10}$?

Curse of dimensionality

- How many observations uniformly distributed on the hypercube $[0, 1]^{20}$ are needed to expect to find at least 50 observations in the hypercube $[0.25, 0.75]^{20}$?
- Ans:

Curse of dimensionality

- Suppose we have $n = 10,000$ observations uniformly distributed on the hypercube $[0,1]^{20}$
- How large should a be so that we can expect to find at least 50 observations in the hypercube $[0.5 - a, 0.5 + a]^{20}$?

What can we do?

- How can we build flexible predictors when there are many covariates available?
- Approximate the regression function by a piecewise constant function
- Use an iterative algorithm to build the piecewise function
- Suboptimal, but feasible

Regression trees

- Consider data $(Y_i, \mathbf{X}_i)$, $i = 1, \ldots, n$ with $\mathbf{X}_i \in \mathbb{R}^p$
- Find regions $R_1, R_2, \ldots, R_K$ that minimize

$$\sum_{j=1}^{K} \sum_{i \in R_j} \left(Y_i - \hat{\mu}_j \right)^2$$

where $\hat{\mu}_j$ is the average of the Y_i's for which $\mathbf{X}_i \in R_j$

Regression trees

- A simpler search
- Find a feature X_j and a threshold a such that

$$\sum_{i \in R_L} (Y_i - \hat{\mu}_L)^2 + \sum_{i \in R_R} (Y_i - \hat{\mu}_R)^2$$

is minimized, where

$$R_L = \{X_j < a\} \quad R_R = \{X_j \geq a\}$$

Regression trees

- Recursively split the regions R_L and R_R
- Stopping criteria?
- Regions have few observations
- The gain in RSS is below a threshold

Regression trees

- It is relatively easy to find the optimal splits
- Trees are easy to explain and visualize
- In some cases trees are interpretable

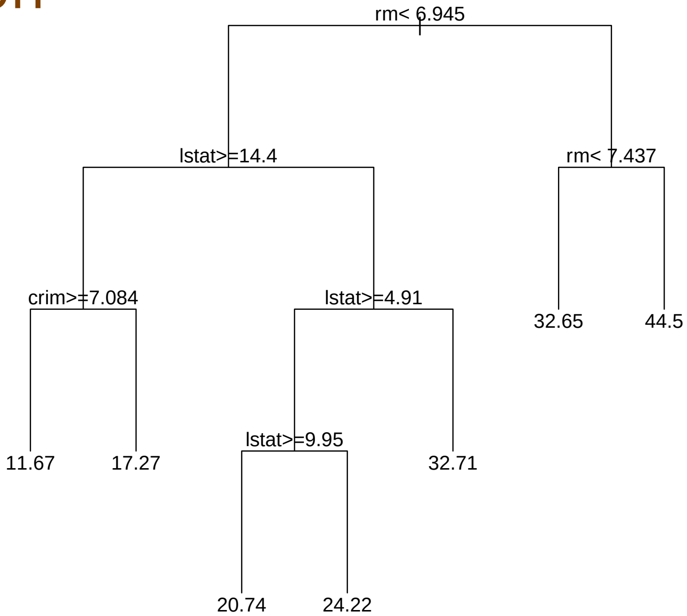
Regression trees - Example

- Consider the Boston data set
- $n = 506$, $p = 14$
- Create a training and test set ($n = 380$ and $n = 126$)
- Build a regression tree

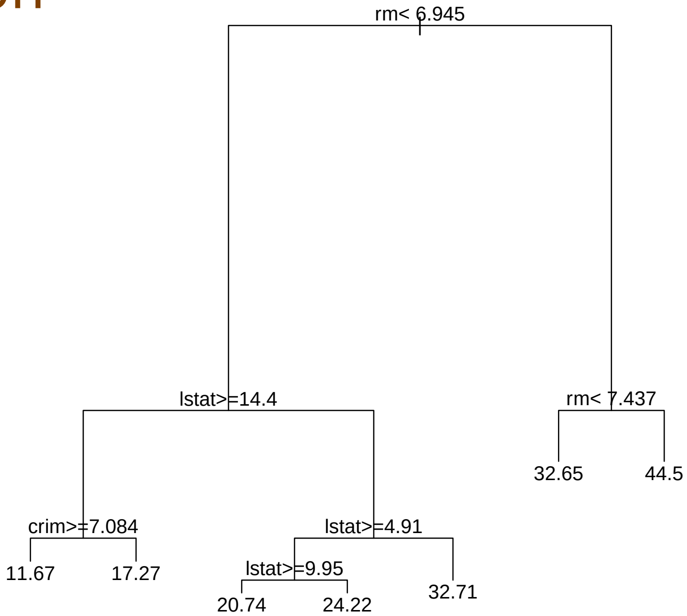
Regression trees - Example

```
data(Boston, package='MASS')

set.seed(123456)
n <- nrow(Boston)
ii <- sample(n, floor(n/4))
dat.te <- Boston[ ii, ]
dat.tr <- Boston[ -ii, ]

bos.t <- rpart(medv ~ ., data=dat.tr,
                method='anova')
plot(bos.t, uniform=FALSE)
text(bos.t, pretty=TRUE)
```


Boston


rm< 6.945
lstat>=14.4
rm< 7.437
crim>=7.084
lstat>=4.91
32.65
44.5
lstat>=9.95
11.67
17.27
32.71
20.74
24.22

Boston


rm< 6.945
lstat>=14.4
rm< 7.437
32.65
44.5
crim>=7.084
lstat>=4.91
11.67
17.27
lstat>=9.95
32.71
20.74
24.22

Boston Example

Compare prediction errors with those of a standard linear regression model

```
> # predictions on the test set
> pr.t <- predict(bos.t, newdata=dat.te,
    type='vector')
> mean((dat.te$medv - pr.t)^2)
[1] 24.43552
>
> # full linear model
> bos.lm <- lm(medv ~ ., data=dat.tr)
> pr.lm <- predict(bos.lm, newdata=dat.te)
> mean((dat.te$medv - pr.lm)^2)
[1] 26.60311
```

Boston Example

Use stepwise to get a better linear model

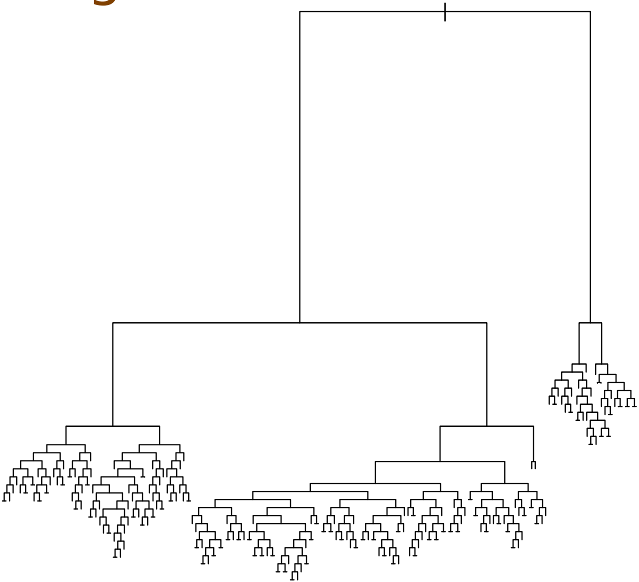
```
> # try to make it better
> null <- lm(medv ~ 1, data=dat.tr)
> full <- lm(medv ~ ., data=dat.tr)
> bos.aic <- stepAIC(null,
    scope=list(lower=null, upper=full),
    trace=FALSE)
> pr.aic <- predict(bos.aic,
    newdata=dat.te)
> with(dat.te, mean( (medv - pr.aic)^2 )
[1] 25.93452
```

Boston Example

Use LASSO

```
> set.seed(123)
> bos.la <- cv.glmnet(x=x.tr, y=y.tr,
     alpha=1)
> x.te <- as.matrix(dat.te[,-14])
> pr.la <- predict(bos.la,
     s='lambda.1se', newx=x.te)
> with(dat.te, mean((medv - pr.la)^2))
[1] 29.20216
```

Overfitting...



Boston Example

Not surprisingly, when we overfit...

```
> pr.to <- predict(bos.to,
      newdata=dat.te,
      type='vector')
> with(dat.te, mean((medv - pr.to)^2))
[1] 36.51097
```