Class Notes

Statistical Computing & Machine Learning

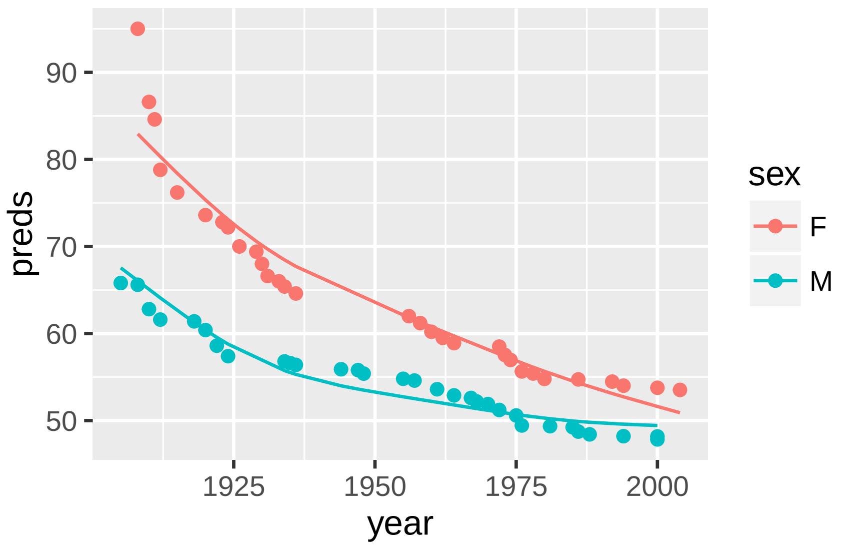
Class 20

Review

Smoothing splines

GAMs — general additive models. Give nonlinear terms for any variables of interest.

```
require(mosaicData)
require(ggplot2)
SwimRecords$post_war <- SwimRecords$year > 1945 # for later
mod <- gam(time ~ sex * s(year, 3), data = SwimRecords)
SwimRecords$preds <- predict(mod)
ggplot(SwimRecords, aes(x = year, y = preds, color = sex)) + geom_line() + geom_point(aes(y = time))
```



Change the order of the smothing spline. Try `ns()` and `bs()`. Add in the `post_war` term.

Trees and recursive partitioning

We're working on a straightforward framework for regression and classification modeling: tree-based models.

A **branch point** (fork? bifurcation?) divides the data under consideration into two branches based on a single explanatory variable. The division is chosen, by exhaustion, to decrease the **deviance** of the response variable. For regression, the model is a simple **mean**[1] and the deviance is equivalent to the **sum of squared residuals**. For

[1] which is itself a maximum-likelihood estimator

classification, the model is a vector of observed probabilities and the deviance is the **log-likelihood** of the response variable given those probabilities.

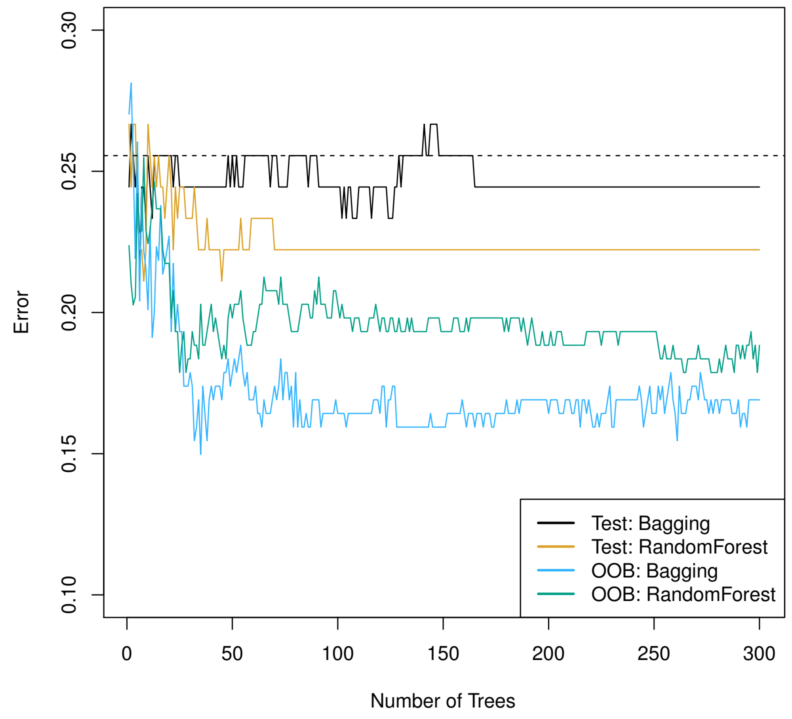


Figure 8.8 from ISL

Branches are followed by additional branch points or by a **single leaf**. A leaf contains a single case[2] from the original data (which, by necessity, cannot be further divided into cases). The process of branching is applied **recursively** until there is a leaf for every case in the original data. I call this a **pure** tree; the log likelihood is as large as possible: 0.

[2] or by multiple cases with the same value for the response variable

Deviance (which amounts to sum of square residuals in a regression setting), is a value that results from summing over every case. The data cases and model values leading into each branch point produce a given deviance. Similarly, each of the branches stemming from the branch point has a deviance. The sum of the branches' deviances is always less than the deviance of the input to the branch point. That **reduction in deviance** can be ascribed to the variable used in the branch point. By summing over all the branch points using a particular variable, the total reduction in deviance due to that variable can be calculated. This can be a useful measure of the **importance** of each variable in contributing to the "explanation" of the response variable. It's analogous to the sum of squares reported in ANOVA for each model term. But, unlike ANOVA, there is no fixed order of which term comes first. (This is good in some ways and bad in others.)

Splitting Criteria for Classification Trees

See p. 312 of the textbook

Note that the Gini index is proportional to the negative log likelihood. Just multiply the Gini index by the number of cases involved to get the negative log likelihood. I prefer this because it is an extensive quantity and so we can consider the whole as the sum of the parts. Example from physics: temperature is intensive, energy is extensive.

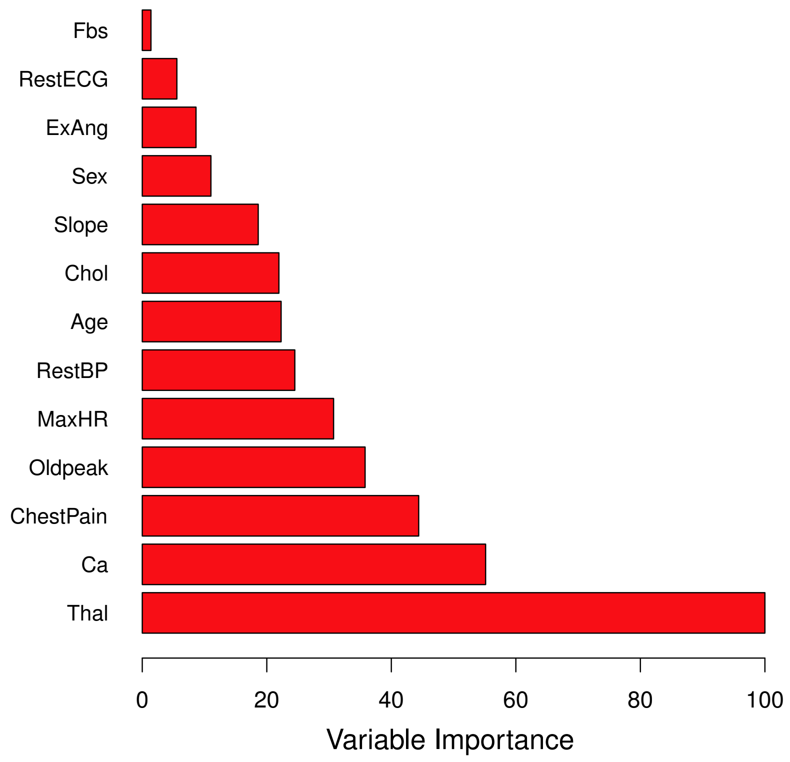


Figure 8.9 from ISL

Of course, the pure tree is likely to be an overfit. The out-of-sample prediction error is almost certain to be larger than the in-sample error. There are several ways to extend the process to produce better out-of-sample performance.

1. **Prune** the tree. The controlling parameter is the number of leaves in the pruned tree.

2. **Average** over bootstrap replications. **Don't prune**.
3. **Shrink** the tree to a very simple structure (e.g. 2-4 leaves) and fit successive trees to the residual. **Heavy pruning** for each tree.

```r
library(ISLR)
library(tree)
Heart <- na.omit(read.csv("Heart.csv"))
Heart$X <- NULL
mod1 <- tree(ChestPain ~ ., data = Heart)
in_sample <- predict(mod1, data = Heart, type= "class", split=TRUE)
table(Heart$ChestPain, in_sample)
```

```
##               in_sample
##                asymptomatic nonanginal
##   asymptomatic          125         10
##   nonanginal             28         44
##   nontypical              5         12
##   typical                 5          4
##               in_sample
##                nontypical typical
##   asymptomatic          3       4
##   nonanginal            8       3
##   nontypical           29       3
##   typical               1      13
```

Pruning

```r
prune1 <- prune.misclass(mod1, best=8)
in_sample <- predict(prune1, data = Heart, type= "class", split=TRUE)
table(Heart$ChestPain, in_sample)
```

```
##               in_sample
##                asymptomatic nonanginal
##   asymptomatic          125         14
##   nonanginal             30         47
##   nontypical             12         28
##   typical                 8          6
##               in_sample
##                nontypical typical
##   asymptomatic          0       3
##   nonanginal            0       6
##   nontypical            7       2
##   typical               0       9
```

Averaging

Each tree produces sharp division points. The precise division point has high variance. So bootstrap to produce a cohort of trees and average them.

```
library(randomForest)

## randomForest 4.6-10

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine

mod2 <- randomForest(ChestPain ~ ., data = Heart, importance = TRUE)
```

In **random forests**, we also "bootstrap" the available variables for each branching point. This produces trees that are uncorrelated with one another, thereby giving additional opportunities to create variance and average over it.

```
mod3 <- randomForest(ChestPain ~ ., data = Heart, mtry=5, importance = TRUE)
```

Shrinking ("Boosting")

```
library(gbm)

## Loading required package: survival

## Loading required package: parallel

## Loaded gbm 2.1.1

Heart$pain <- ifelse(Heart$ChestPain == "asymptomatic", 0, 1)
mod4 <- gbm(pain ~ ., data=Heart, distribution="bernoulli", n.trees=5000, interaction.depth = 4)
```

Slow down the learning process to avoid the pitfalls of greedy optimization.

This importance is analogous to the mean square

Similarity to **stepwise selection** in linear regression.