

CLASSIFICATION VIA TREES

-STATISTICAL MACHINE LEARNING-

Lecturer: Darren Homrighausen, PhD

WHAT IS A (DECISION) TREE?

- Trees involve **stratifying** or **segmenting** the predictor space into a number of simple regions.
- Trees are simple and useful for interpretation.
- Basic trees are not great at prediction.
- More modern methods that use trees are much better.

VISUALIZATION

Phylogenetic Tree of Kingdom Animalia

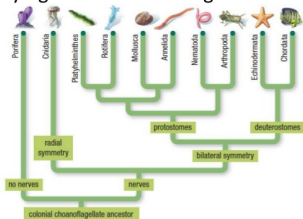
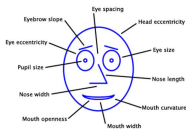
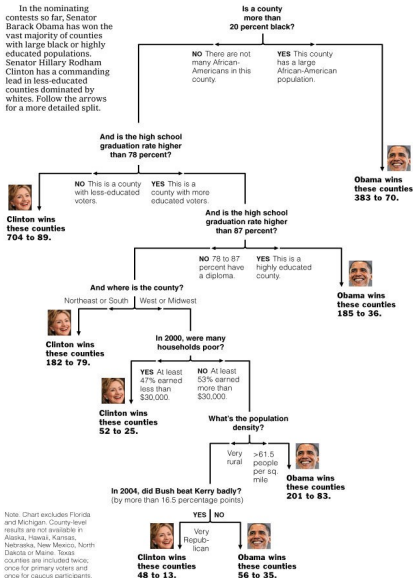


Figure 5 This phylogenetic tree of the Animal Kingdom is based on recent evidence from genetics and molecular biology. The tree shows 10 of the 17 animal phyla.

chernoff face



In the nominating contests so far, Senator Barack Obama has won the vast majority of counties with large black or highly educated populations. Senator Hillary Rodham Clinton has a commanding lead in less-educated counties dominated by whites. Follow the arrows for a more detailed split.



Note: Chart excludes Florida and Michigan. County-level results are not available in Alaska, Hawaii, Kansas, Nebraska, New Mexico, North Dakota or Maine. Texas counties are included twice; once for primary voters and once for caucus participants.

A MOTIVATING EXAMPLE

Suppose we are interested in predicting whether or not the economy will be in a **recession**

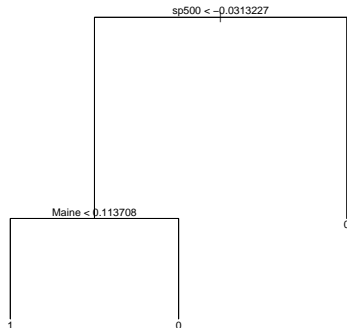
We have quarterly measurements of

- State level economic growth
(Larger number is better)
- Federal level variables such as GDP, interest rates, employment, S&P 500, ...

Here, we will code the supervisor as

$$Y = \begin{cases} 1 & \text{if recession} \\ 0 & \text{if growth} \end{cases}$$

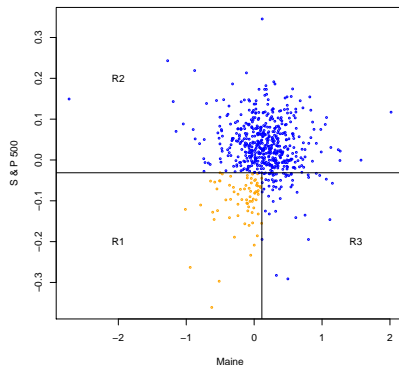
DENDROGRAM VIEW



TERMINOLOGY

- We call each split or end point a **node**. Each terminal node is referred to as a **leaf**.
 - ▶ This tree has 2 interior nodes and 3 terminal nodes.
- The interior nodes lead to **branches**.
 - ▶ This graph has two main branches (the S&P 500 split).

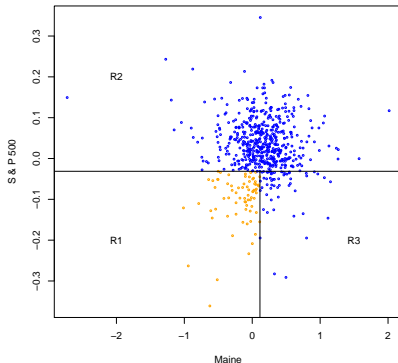
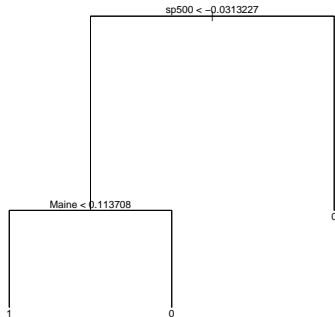
PARTITIONING VIEW



NOTES

- We classify all observations in a region the same.
- The three regions R1, R2, and R3 are the leaves of the tree.

TREE



We can interpret this as

- S&P 500 is the most important variable.
- If S&P 500 is large enough, then we predict no recession.
- If S&P 500 is small enough, then we need to know the change in the employment level of Maine.

HOW DO WE BUILD A TREE?

1. Divide the predictor space into M non-overlapping regions R_1, \dots, R_M
(this is done via greedy, binary splitting)
2. Every observation that falls into a given region R_m is given the same prediction
 - ▶ **REGRESSION:** The average of the supervisors for a region
 - ▶ **CLASSIFICATION:** Determined by majority (or plurality) vote in that region

Important:

- Trees can only make rectangular regions that are **aligned** with the coordinate axis.
- The fit is **greedy**, which means that after a split is made, all further decisions are conditional on that split.
- The tree stops splitting when there are too few observations in a terminal node

Regression trees

IMPLICIT MODEL

For a given partition R_1, \dots, R_M , the model for the supervisor is

$$f(X) = \sum_{m=1}^M c_m \mathbf{1}(X \in R_m)$$

We need to estimate both (R_m) and (c_m)

Generally, searching over all possible regions is infeasible

(This would involve sifting through all $M \leq n$ and all configurations for R_m)

So we use a **greedy** approach instead

REGRESSION TREES

Define the two half-planes

$$r_1(j, s) = \{X | X_j \leq s\} \quad \text{and} \quad r_2(j, s) = \{X | X_j > s\}$$

For squared error loss, we solve

$$\min_{j,s} \left[\min_{c_1} \sum_{X_i \in r_1(j,s)} (Y_i - c_1)^2 + \min_{c_2} \sum_{X_i \in r_2(j,s)} (Y_i - c_2)^2 \right]$$

This generates, for $n_k = \sum_{i=1}^n \mathbf{1}(X_i \in r_k)$,

$$\hat{c}_k = n_k^{-1} \sum_{i: X_i \in r_k} Y_i$$

The next splits will be conditional on the minimizing \hat{s}

Classification trees

CLASSIFICATION TREES

For a given region R_m and class g , define training proportions

$$\begin{aligned}\hat{p}_{mg}(X) &= \mathbf{1}(X \in R_m) n_m^{-1} \sum_{i: X_i \in R_m} \mathbf{1}(Y_i = g) \\ &= \begin{cases} \frac{1}{n_m} \sum_{i: X_i \in R_m} \mathbf{1}(Y_i = g) & \text{if } X \text{ is in } R_m \\ 0 & \text{if } X \text{ is not in } R_m \end{cases}\end{aligned}$$

Our classification is

$$\hat{g}(X) = \arg \max_g \hat{p}_{mg}(X)$$

This presumes a given partition (R_m)

To estimate the partition we need a **loss function**

HOW DO WE MEASURE QUALITY OF FIT?

There are many possibilities:

CLASSIFICATION ERROR RATE: $ER = 1 - \max_g(\hat{p}_{mg})$

GINI INDEX: $GI = \sum_g \hat{p}_{mg}(1 - \hat{p}_{mg})$

CROSS-ENTROPY: $CE = -\sum_g \hat{p}_{mg} \log(\hat{p}_{mg})$

(Cross-entropy is also known as deviance)

We build a classifier by **growing** a tree that **greedily** minimizes one of these criteria

We would like the tree to have **pure** nodes, in the sense that

$$\max_g \hat{p}_{mg} \approx 1$$

HOW DO WE MEASURE QUALITY OF FIT?

The m^{th} node is split by minimizing ER , GI , or CE over all

- features
- split points of those feature

such that the loss function is minimized over the 2 new regions

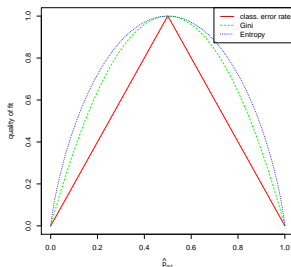
What do these loss functions look like?

HOW DO WE MEASURE QUALITY OF FIT?

EXAMPLE: Suppose $G = 2$. Then $\hat{p} = \hat{p}_{m1} = 1 - \hat{p}_{m2}$

The m^{th} node is made by minimizing E , G , or D over all

- Features
- split points of that feature



Generally, **GINI INDEX** or **CROSS-ENTROPY** is preferred

(They penalize values of \hat{p} far from 0 or 1 more severely)

HOW DO WE MEASURE QUALITY OF FIT?

Additionally, **GINI INDEX** or **CROSS-ENTROPY** are preferred as..

- They are differentiable everywhere
- They are amenable to treating qualitative features with a large number of levels in a computationally efficient way
(There are $2^{L-1} - 1$ possibilities for L levels)
(See ESL 9.2.4)
- The Gini index can be interpreted in suggestive ways
(See ESL 9.2.3)

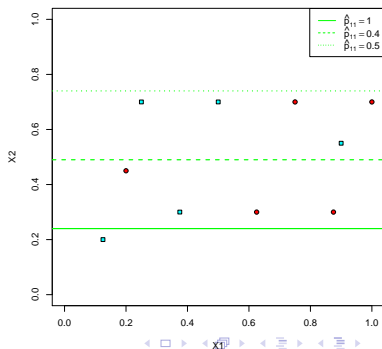
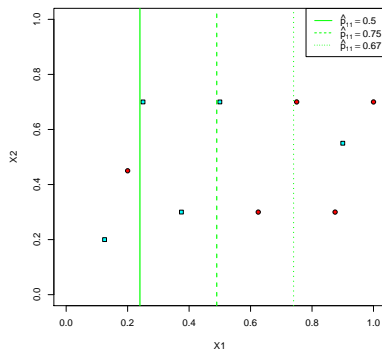
HOW DO WE MEASURE QUALITY OF FIT?

EXAMPLE: Suppose $G = 2$ and we want to make the first split

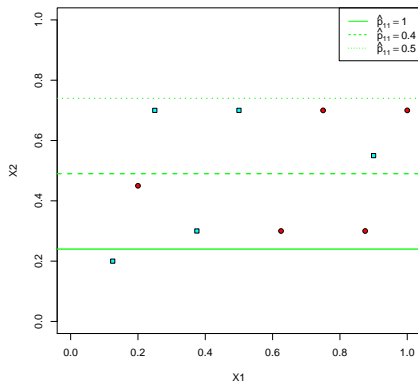
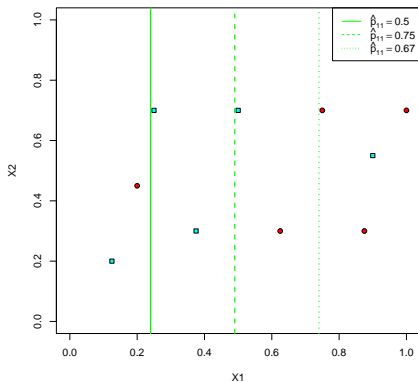
$$\text{Then } \hat{p}_{11} = 1 - \hat{p}_{12}$$

(Define the 'left' or 'bottom' region as R_1)

Let's look at some possible splits:

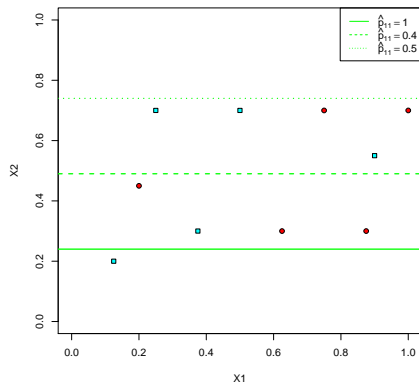
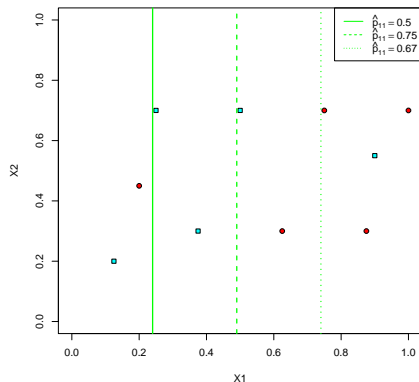


HOW DO WE MEASURE QUALITY OF FIT?



Where would we split?

HOW DO WE MEASURE QUALITY OF FIT?



Where would we split if we required ≥ 2 observations in a node for G_I loss?

THERE'S A PROBLEM

Following this procedure **overfits!**

- The process described so far will fit overly complex trees, leading to poor predictive performance.
(In fact, G_I can be interpreted as the variance of Bernoullis)
- Overfit trees mean they have too many leaves.
- To stretch the analogy further, trees with too many leaves must be **pruned**.

PRUNING THE TREE

- Cross-validation can be used to directly prune the tree, but it is far too expensive (computationally) to use in practice (combinatorial complexity)
- Instead, we use **weakest link pruning**

$$\sum_{m=1}^{|T|} \sum_{i \in R_m} \mathbf{1}(Y_i \neq \hat{Y}_{R_m}) + \lambda |T|$$

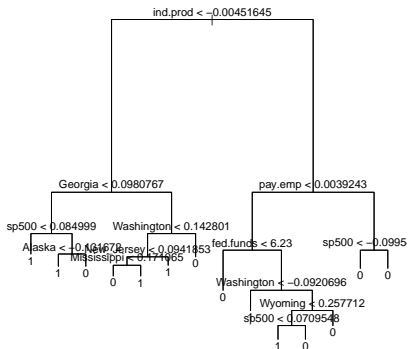
where $|T|$ is the number of terminal nodes.

Essentially, we are trading **training fit** (first term) with **model complexity** (second term)

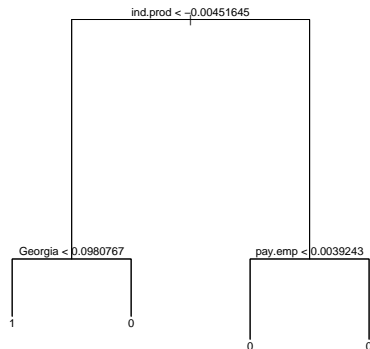
(compare to lasso)

- Now, cross-validation can be used to pick λ .

RESULTS OF TREES ON RECESSION DATA

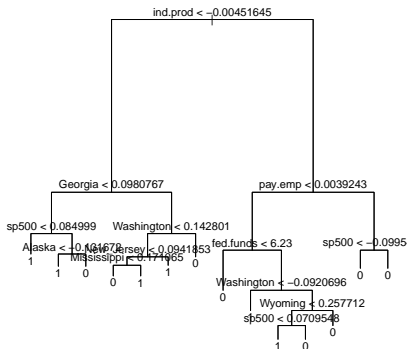


Unpruned tree

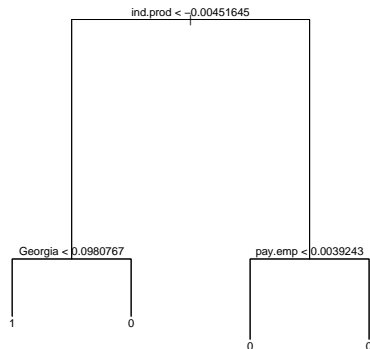


Pruned Tree

RESULTS OF TREES ON RECESSION DATA



Unpruned tree

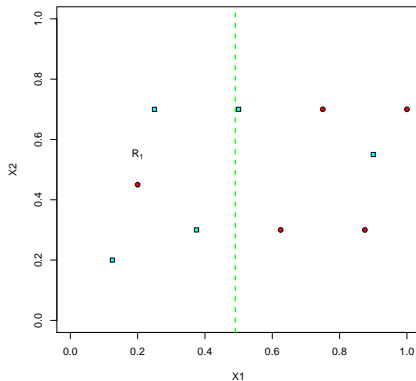


Pruned Tree

The pruned tree is a **subset** of the unpruned tree (**nested**)

There are splits that result the same prediction. **WHY?**

SPLITS WITH SAME PREDICTION



Suppose we split at vertical, dashed line. Then $\hat{p}_{11} = 0.75$.

What happens if we were to now split R_1 at $X_2 = 0.5$?

TREES IN R

Create a basic, unpruned tree:

```
require(tree)
out.tree = tree(Y~.,data=X,split='gini')
plot(out.tree)
text(out.tree)
```

(There is also the **rpart** package as well)

TREES IN R

Prune the tree via **cross-validation**

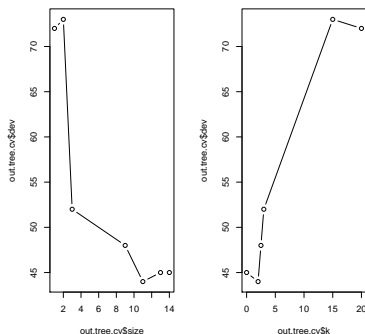
```
out.tree.orig = tree(Y~.,data=X)
out.tree.cv   = cv.tree(out.tree.orig,FUN=prune.misclass)
> names(out.tree.cv)
[1] "size"    "dev"     "k"       "method"
```

TREES IN R

Prune the tree via **cross-validation**

```
plot(out.tree.cv$size,out.tree.cv$dev,type="b")
```

```
plot(out.tree.cv$k,out.tree.cv$dev,type="b")
```



NOTE:

k corresponds to λ in weakest-link pruning.

dev means missclassifications in **cv.tree**

TREES IN R

Prune the tree via **cross-validation**

```
best.size = out.tree.cv$size[which.min(out.tree.cv$dev)]  
> best.size  
[1] 11  
out.tree = prune.misclass(out.tree.orig,best=best.size)  
class.tree = predict(out.tree,X_0,type='class')
```

AN INTRODUCTORY EXAMPLE

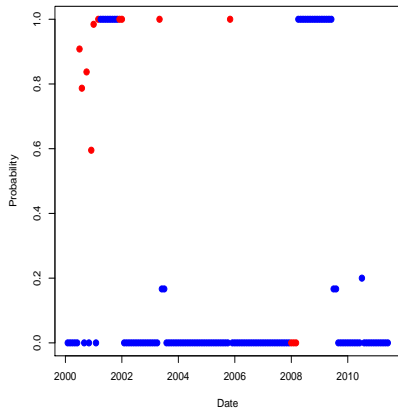
EXAMPLE: Use macroeconomic data to predict recessions

We will use data from 1960 through 1999 as **training data**

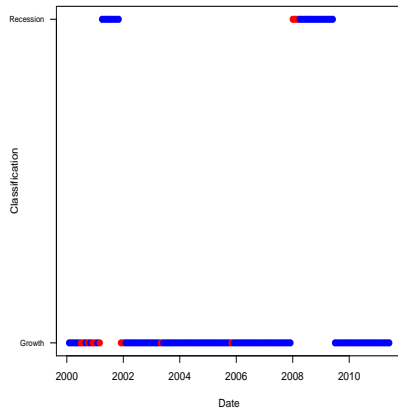
We will use data from 2000 through 2011 as **testing data**

(In the following plots: **correct** vs. **incorrect** classifications)

RESULTS OF TREES ON RECESSION DATA



Posterior probability of prediction



Predictions

ADVANTAGES AND DISADVANTAGES OF TREES

- + Trees are very easy to explain (much easier than even linear regression).
- + Some people believe that decision trees mirror the human decision-making process.
- + Trees can easily be displayed graphically no matter the dimension of the data.
- + Trees can easily handle qualitative features without the need to create dummy variables.
- Trees aren't very good at prediction.

To fix this last one, we can try to grow many trees and average their performance.