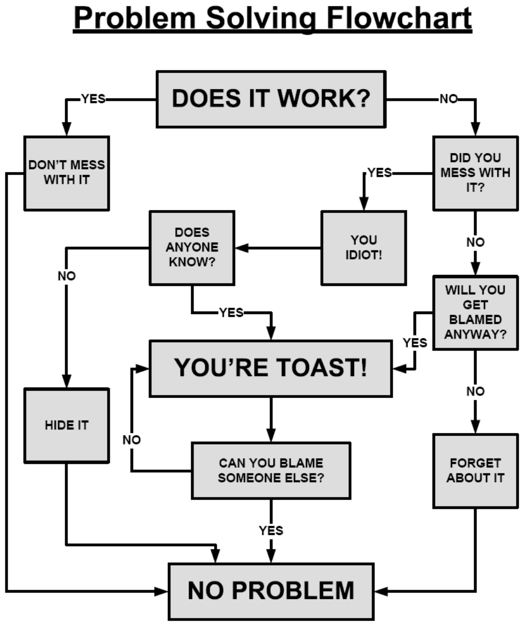
Data, Environment and Society: Lecture 22: Classification and regression trees

Instructor: Duncan Callaway
GSI: Seigi Karasaki

November 6, 2018

Today's objectives

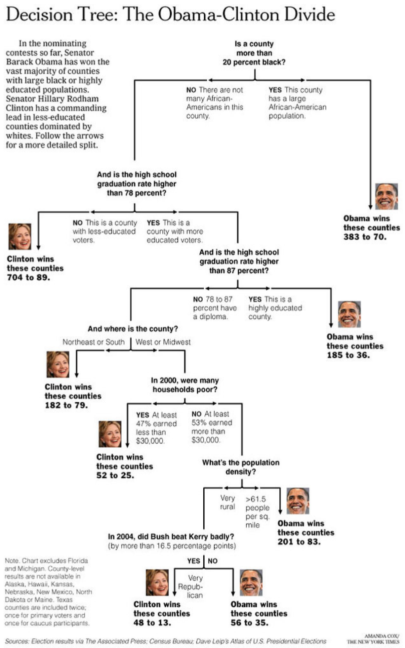


Today's objectives

- Introduction to regression trees
  - Terminology
  - How they are built

Announcements

Regression trees


Decision Tree: The Obama-Clinton Divide

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.

Is a county more than 20 percent black?
NO There are not many African-Americans in this county.
YES This county has a large African-American population.
Obama wins these counties 383 to 70.
And is the high school graduation rate higher than 78 percent?
NO This is a county with less-educated voters.
YES This is a county with more educated voters.
Clinton wins these counties 704 to 89.
And is the high school graduation rate higher than 87 percent?
NO 78 to 87 percent have a diploma.
YES This is a highly educated county.
Obama wins these counties 185 to 36.
And where is the county?
Northeast or South
West or Midwest
Clinton wins these counties 182 to 79.
In 2000, were many households poor?
YES At least 47% earned less than $30,000.
NO At least 53% earned more than $30,000.
Clinton wins these counties 52 to 25.
What's the population density?
Very rural
>61.5 people per sq. mile
Obama wins these counties 201 to 83.
In 2004, did Bush beat Kerry badly? (by more than 16.5 percentage points)
YES
NO
Very Repub-lican
Clinton wins these counties 48 to 13.
Obama wins these counties 56 to 35.

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.

Sources: Election results via The Associated Press; Census Bureau; Dave Leip's Atlas of U.S. Presidential Elections

AMANDA COX/ THE NEW YORK TIMES

Basic idea for regression trees

All we are doing is “splitting” the observations into regions and averaging the dependent variable within each region.

Basic idea for regression trees

All we are doing is “splitting” the observations into regions and averaging the dependent variable within each region.

Doing predictions with the model just involves locating a set of predictors in a region, then setting the response variable equal to the average from the training data in that region.

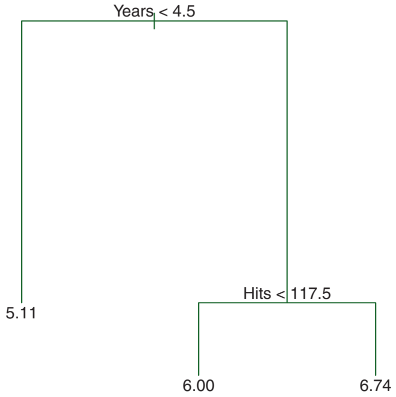
Basic idea for regression trees

All we are doing is “splitting” the observations into regions and averaging the dependent variable within each region.

Doing predictions with the model just involves locating a set of predictors in a region, then setting the response variable equal to the average from the training data in that region.

Big decision in regression trees: *What are the regions we should use?*

Example, from the textbook



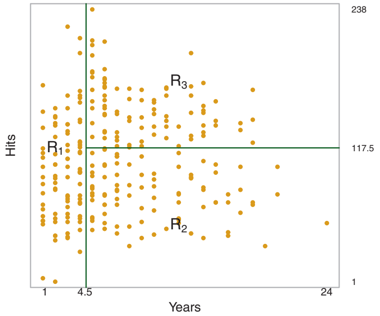
"Hitters" data from ISLR.

263 major league players stats.

Here, this tree is "spliting" on two variables – years in league and number of hits

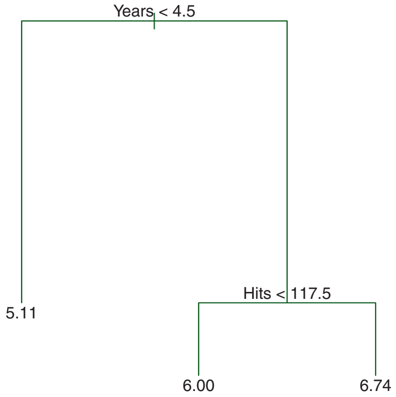
The numbers at the ends are the average (log-transformed) average salaries for players

Example, from the textbook, ctd



$$R_1 = \{X \mid \text{years} < 4.5\}$$
$$R_2 = \{X \mid \text{years} \geq 4.5, \text{hits} < 117.5\}$$
$$R_3 = \{X \mid \text{years} \geq 4.5, \text{hits} \geq 117.5\}$$

Terminology

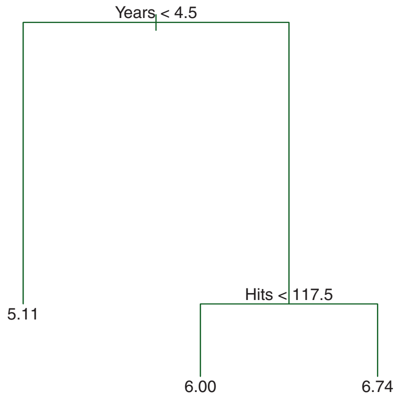


Each region R_i is a *terminal node*, or *leaf*

Each numeric value at which a split happens is an *internal node*

Segments connecting nodes (terminal or internal) are...

Terminology

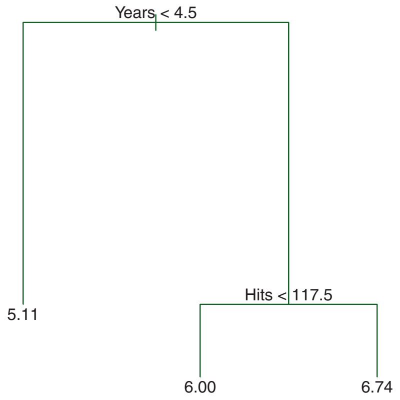


Each region R_i is a *terminal node*, or *leaf*

Each numeric value at which a split happens is an *internal node*

Segments connecting nodes (terminal or internal) are...*branches*

Terminology



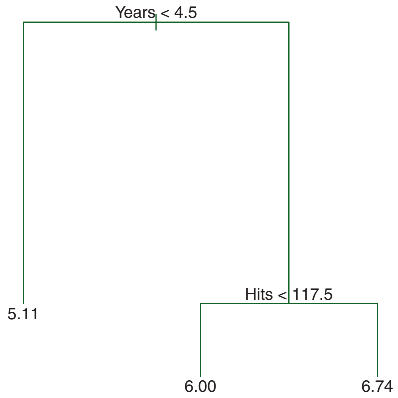
Each region R_i is a *terminal node*, or *leaf*

Each numeric value at which a split happens is an *internal node*

Segments connecting nodes (terminal or internal) are...*branches*

The numbers at the end of the branches are...

Terminology



Each region R_i is a *terminal node*, or *leaf*

Each numeric value at which a split happens is an *internal node*

Segments connecting nodes (terminal or internal) are... *branches*

The numbers at the end of the branches are... *leaves*

Regression trees – basic approach

1. Divide the *predictor* space into non-overlapping regions
2. Within each region, the prediction is just the average of the training data.

Regression trees – basic approach

1. Divide the *predictor* space into non-overlapping regions
2. Within each region, the prediction is just the average of the training data.

Two Basic Questions:

1. Where should I put the internal nodes?
2. How many regions should there be?

The answers are, as it turns out, really simple.

Where to put the internal nodes?

First, for simplicity, the nodes are structured to make rectangles in the predictor space.

Which region and split location to choose first?

Let

- j index predictor variables
- s denote the location of the split within the region

Then all splits can be described as:

$$R_1(j,s) = \{X|X_j < s\} \text{ and } R_2(j,s) = \{X|X_j \geq s\}$$

Then we partition any region by choosing j and s as follows:

$$\{j,s\} = \arg\min_{j \in J, s \in X_j} \sum_{i: x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

where $\hat{y}_{R_1}$ is the mean of all response variables in region 1.

It would be tedious to identify j and s by hand, but it's actually very quick computationally.

Ok, we've split one predictor in two. Now what?

After you've chosen the first partition, then do splits separately within each of the two identified regions.

Choose the best split from among all possible splits of those two regions (you'll only wind up splitting one additional region). **Now we'll have three regions.**

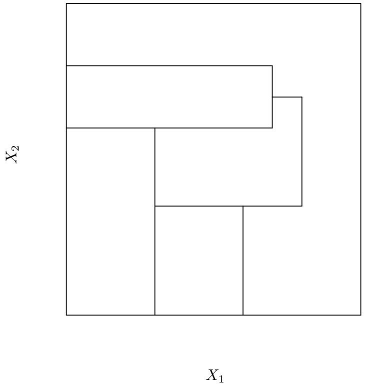
Choose the best split from among all possible splits of those three, again just splitting one additional region. **Now we'll have four regions.**

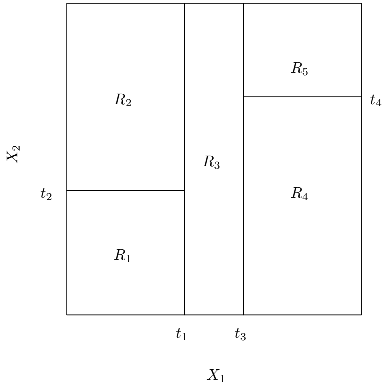
Repeat this process until you reach a stopping criterion – typically a maximum number of observations in each region.

Call the resulting tree T_0.

We call this approach "greedy" because when we do the first partition we're not thinking ahead to future partitions to evaluate it.

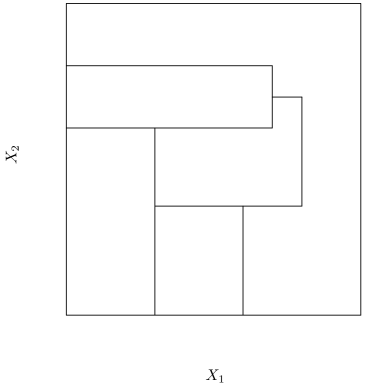
One of these doesn't belong...

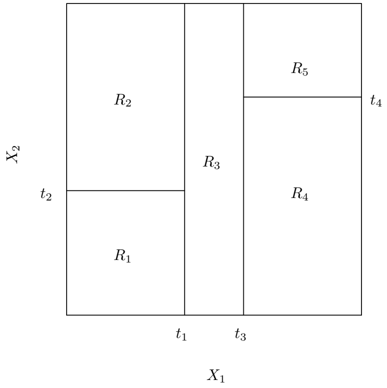




Q: Which picture results from successively splitting the regions into values greater or less than predictor values?

One of these doesn't belong...

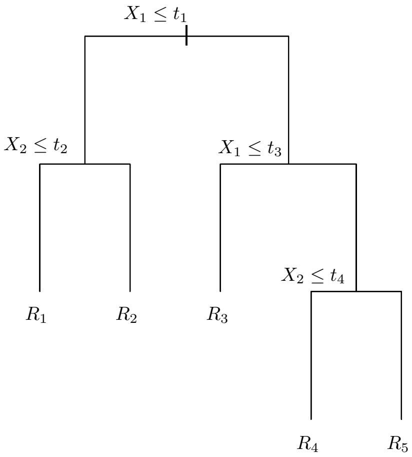


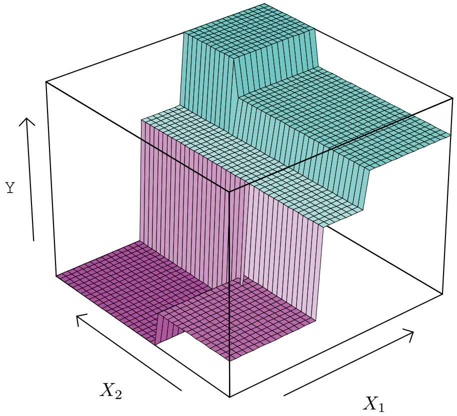


Q: Which picture results from successively splitting the regions into values greater or less than predictor values?

A: The right one. The left one is not possible with simple splitting.

A five region example... with two dimensional predictor space





What do we call it?

The process of splitting regions over and over is called...

“recursive binary splitting”

You can also call it a **“top-down greedy’**’ approach.

Because it’s “greedy” we can’t be sure that the splits we’re getting are the best possible splits.

Why binary?

In other words, why not multiway splits?

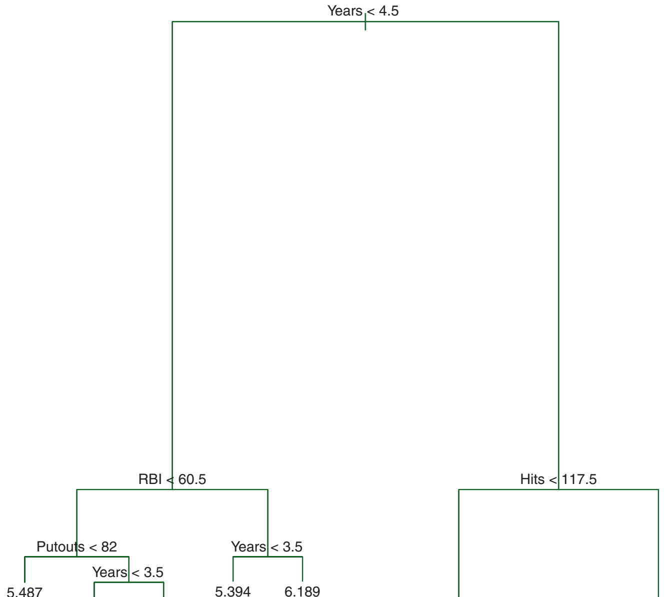
Why binary?

In other words, why not multiway splits?

In general multiway splits fragment the data too quickly, leaving insufficient data at the next level down

Since we do the binary splitting recursively, we get the same flexibility as a multiway split, since a region can be split a second time later.

Example T_0



Years < 4.5
RBI < 60.5
Hits < 117.5
Putouts < 82
Years < 3.5
Years < 3.5
5.487
5.394
6.189

Important note

Recursive binary splitting, at least as presented here, does not consider standard errors in the splitting process.

An alternative (improved?) splitting rule would only consider splits where the standard errors of the means in the candidate regions are not overlapping.

Something similar is accomplished by stopping the splits when the leaves get small (the standard approach), but why not take the formality of a standard error test?

When will we test?

All the steps above involve model *building*. We have yet to evaluate different models against one another. First let's build the candidate models, then we can evaluate.

Step 1: "cost complexity pruning"

We'll test models that are **subtrees** of T_0. (trees that are the same as T_0 except they are missing some internal nodes and branches).

We identify subtrees using **cost-complexity pruning** a.k.a. weakest link pruning:

- To get the first subtree, evaluate model performance for all subtrees with one leaf removed from T_0. Choose the best one, call it T_1.
    - R^2 works for measuring performance
    - ...but not for categorical variables, stay tuned!
- Then evaluate performance for all models with one leaf removed from T_1. Choose the best, call it T_2. And so on.

Step 1: "cost complexity pruning"

We'll test models that are **subtrees** of T_0. (trees that are the same as T_0 except they are missing some internal nodes and branches).

We identify subtrees using **cost-complexity pruning** a.k.a. weakest link pruning:

- To get the first subtree, evaluate model performance for all subtrees with one leaf removed from T_0. Choose the best one, call it T_1.
  - R^2 works for measuring performance
  - ...but not for categorical variables, stay tuned!
- Then evaluate performance for all models with one leaf removed from T_1. Choose the best, call it T_2. And so on.

Smart researchers have shown that, given a tree, cost complexity pruning identifies the best subtrees.

That is, this "greedy" approach is an optimal *pruning* strategy. (But recursive binary splitting is not always optimal for growth.)

Step 2: Tune up your α

Take your set of subtrees, T_0 through T_{N-2}. Call $|T|$ the number of terminal nodes in the tree.

For a given α, *one* of the T_i will minimize :

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

- $\alpha = 0$ will choose T_0, the biggest tree.
- As α grows you'll choose successively smaller trees.

Fill in the blank: As α increases, bias goes __ and variance goes _____

Step 2: Tune up your α

Take your set of subtrees, T_0 through T_{N-2}. Call $|T|$ the number of terminal nodes in the tree.

For a given α, *one* of the T_i will minimize :

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|$$

- $\alpha = 0$ will choose T_0, the biggest tree.
- As α grows you'll choose successively smaller trees.

Fill in the blank: As α increases, bias goes **up** and variance goes **down**

Bigger α means fewer leaves, which means more bias but less variance.

Though it seems unnecessary to define α (why not just evaluate all subtrees?), we'll see it's useful for cross validation.

The (cross validation) process

Basic steps:

a. Grow a large tree via recursive binary splitting. "Large" means each leaf has some pre-specified maximum number of observations (e.g. 5)

b. Then "prune" the tree to get a sequence of subtrees. Choose one that minimizes $\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$ for each of a range of values of α.

The (cross validation) process

Basic steps:

a. Grow a large tree via recursive binary splitting. "Large" means each leaf has some pre-specified maximum number of observations (e.g. 5)

b. Then "prune" the tree to get a sequence of subtrees. Choose one that minimizes $\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$ for each of a range of values of α.

The cross validation process:

1. Split your data into K folds.
2. Withhold each fold and Repeat **a.** and **b.**
3. *For each value of* α, evaluate the error on the withheld fold, then average the error across all folds.
4. Choose the α that gives the lowest cross validated error,
5. Build your final model with the chosen α with *all the data.*

Why use α?

Why didn't we just evaluate cross validated error for each tree size?

That is, is α just overly complicating things?

Why use α?

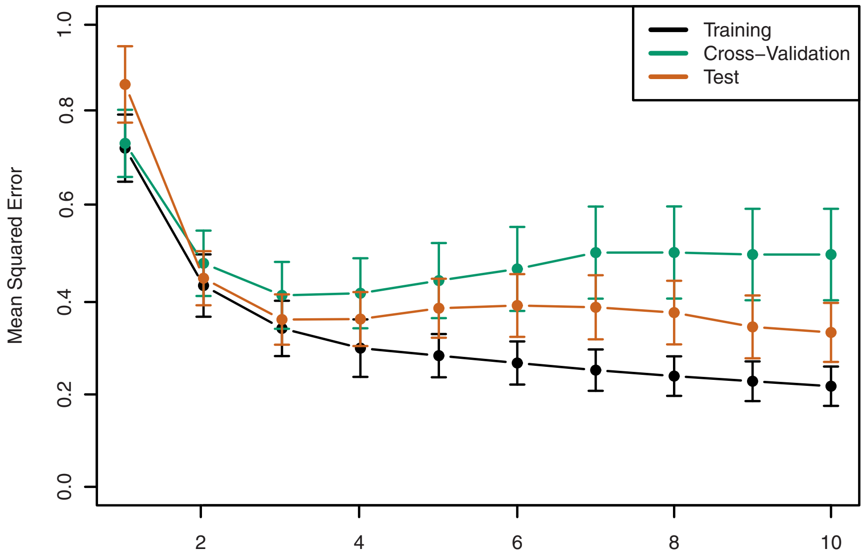
Why didn't we just evaluate cross validated error for each tree size?

That is, is α just overly complicating things?

Ans: Sometimes it might. But it may be that across different folds we'd choose different subtrees. α does a better job of tuning the bias-variance tradeoff.

But: out of convenience the book *displays* results in terms of tree size rather than α. Argh!

Results on Hitters data



Questions

Are regression trees supervised or unsupervised learning?

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Do regression trees utilize linear regression?

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Do regression trees utilize linear regression? Nope.

What do you think their advantages are (vs LASSO or nonlinear models...)?

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Do regression trees utilize linear regression? Nope.

What do you think their advantages are (vs LASSO or nonlinear models...)?

- Easy to explain and non-experts can understand the results.
- They're more like human decision-making. Doctors like them.
- They easily handle qualitative predictors – no need for dummies.

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Do regression trees utilize linear regression? Nope.

What do you think their advantages are (vs LASSO or nonlinear models...)?

- Easy to explain and non-experts can understand the results.
- They're more like human decision-making. Doctors like them.
- They easily handle qualitative predictors – no need for dummies.

How about some disadvantages?

Questions

Are regression trees supervised or unsupervised learning? Supervised!

Do regression trees utilize linear regression? Nope.

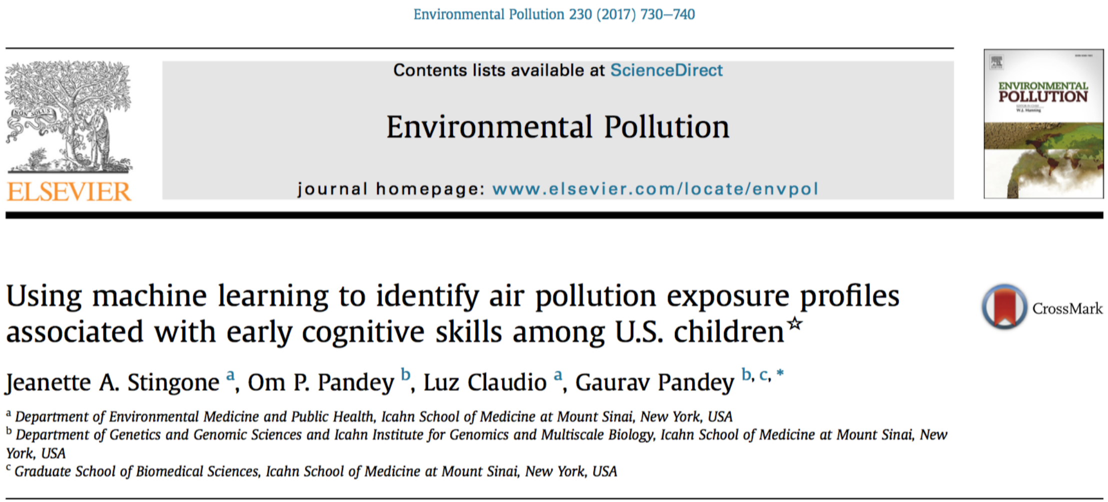
What do you think their advantages are (vs LASSO or nonlinear models...)?

- Easy to explain and non-experts can understand the results.
- They're more like human decision-making. Doctors like them.
- They easily handle qualitative predictors – no need for dummies.

How about some disadvantages?

- As described, they don't usually provide the same predictive power that the other tools we've studied can.
- They can be pretty sensitive to small changes in the data.
- Recursive binary splitting may generate a suboptimal tree (like forward / backward model selection). Things later in the chapter address this.

Example: Test scores and pollution



Environmental Pollution 230 (2017) 730–740

ELSEVIER

Contents lists available at ScienceDirect

Environmental Pollution

journal homepage: www.elsevier.com/locate/envpol

ENVIRONMENTAL POLLUTION

Using machine learning to identify air pollution exposure profiles associated with early cognitive skills among U.S. children☆

CrossMark

Jeanette A. Stingone [a], Om P. Pandey [b], Luz Claudio [a], Gaurav Pandey [b, c, *]

[a] Department of Environmental Medicine and Public Health, Icahn School of Medicine at Mount Sinai, New York, USA
[b] Department of Genetics and Genomic Sciences and Icahn Institute for Genomics and Multiscale Biology, Icahn School of Medicine at Mount Sinai, New York, USA
[c] Graduate School of Biomedical Sciences, Icahn School of Medicine at Mount Sinai, New York, USA

Does pollution change cognitive ability?

Stingone et al point out that few studies have looked at the effects of multiple pollutants at once
Key data:

- Kindergarten math scores from National Center of Education Statistics Early Childhood Longitudinal Study
- Census tract estimates of 104 toxic pollutants from U.S. Environmental Protection Agency's National Air Toxics Assessment (NATA)
- Other confounders including mother age, marital status, hhld income, etc. (Used in second stage *after* tree building.)

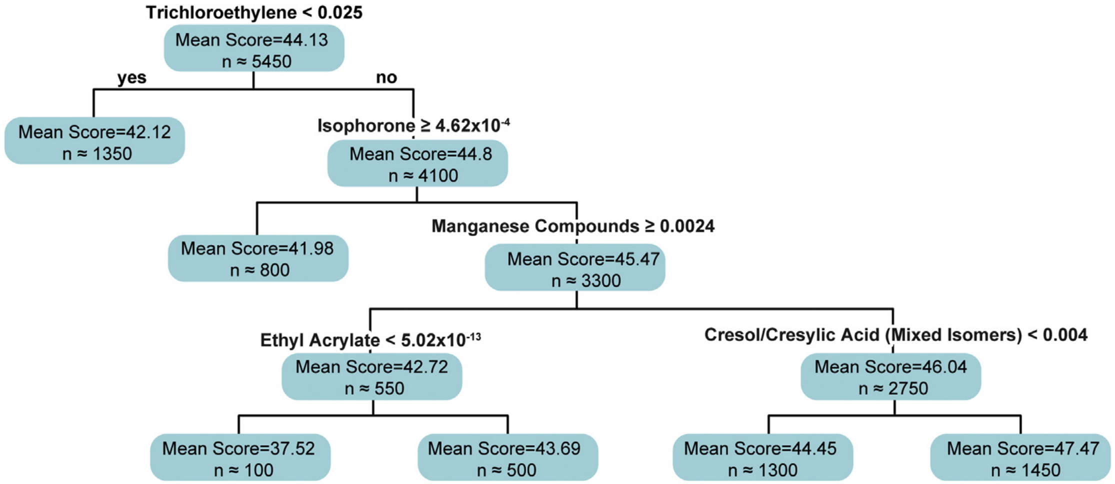
Stingone *et al* two step approach

1. Build trees for test score outcome based on pollutant exposure (what we'll focus on here)
2. Run basic multiple linear regression *within* each leaf to identify the effect of pollutants on test scores. (We won't cover this part.)

Why trees? Stingone *et al's* justification

- Easy interpretability in terms of understandable trees and/or rules,
- Ability to identify non-linear relationships between the features (exposures) and the outcome (math scores),
- Possibility of identifying interactions among the features (exposures),
- Making no/minimal assumptions about data distributions,
- Tolerance to missing values and outliers in the data,

Example result



Constructed with 10-fold cross-validation. Also used additional random partitioning – stay tuned.

A trick that Stingone *et al* used

They note that Trees are:

- Prone to overfitting the (training) data,
- Sensitive to small perturbations in the data and/or model/algorithm parameters

Their approach to manage this is to build a lot of different trees using random partitions of the data.

Their approach is a little unconventional (for reasons they don't provide).

Instead, on Thursday we'll talk about formal strategies to deal with this sensitivity – boosting, bagging and random forests.

Classification trees

What’s a classification tree?

As you might imagine, it’s just like a regression tree, but we use it to predict a categorical or qualitative variable.

Rather than setting the prediction equal to the mean, the prediction is:

What's a classification tree?

As you might imagine, it's just like a regression tree, but we use it to predict a categorical or qualitative variable.

Rather than setting the prediction equal to the mean, the prediction is:

the most commonly occurring class within the partition.

However we still use recursive binary splitting and cost-complexity pruning

Though the *criteria* for splitting and pruning will have to change

What's the error?

The typical error, RSS $= \sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ won't work.

Alternatives? Let's start by defining

$$p_{mk} = \text{ fraction of observations belonging to class } k \text{ in region } m.$$

Then a simple measure is:

Classification error rate $=$ how many training observations don't fall into the assigned class.

Within-region this is simply:

$$E_m = 1 - \max_k(\hat{p}_{mk})$$

The trouble with Classification Error

Suppose you have a two-class problem with 400 observations in each class. Consider two possible splits (S1 and S2):

S1: $R_1 : (100, 300)$, $R_2 : (300, 100) \Rightarrow$ weighted error $E_{S1} = 0.25$

S2: $R_1 : (200, 400)$, $R_2 : (200, 0) \Rightarrow$ weighted error $E_{S2} = 0.25$

Can you make a case for one of these being preferable to the other?

The trouble with Classification Error

Suppose you have a two-class problem with 400 observations in each class. Consider two possible splits (S1 and S2):

S1: $R_1 : (100, 300)$, $R_2 : (300, 100) \Rightarrow$ weighted error $E_{S1} = 0.25$

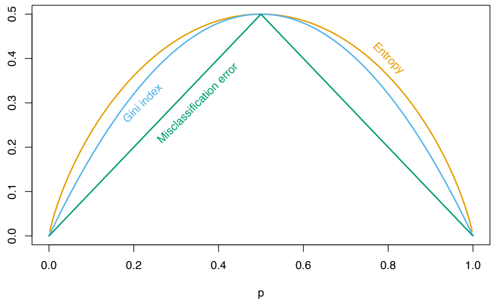
S2: $R_1 : (200, 400)$, $R_2 : (200, 0) \Rightarrow$ weighted error $E_{S2} = 0.25$

Can you make a case for one of these being preferable to the other?

S2 has a “pure” split, meaning there are *no* errors in one of the splits. You won’t need to split this region any further.

Alternative errors

Remember, p_{mk} = fraction of observations in class k in region m.



(Measures for two-class classification; p is the proportion in class 2. Cross-entropy has been scaled to pass through (0.5, 0.5).)

$$E_m = 1 - \max_k(\hat{p}_{mk})$$

$$G_m = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \quad \text{“Gini”}$$

$$D_m = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \quad \text{“Entropy”}$$

G and D have two advantages:

1. Differentiable everywhere – good for optimization
2. Score better for “pure” splits

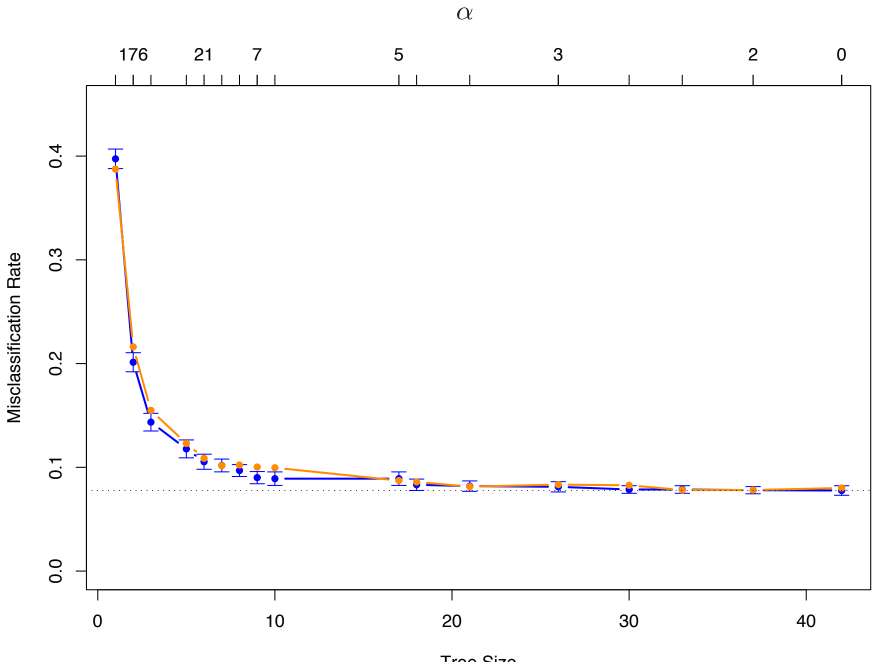
Which error rate to use?

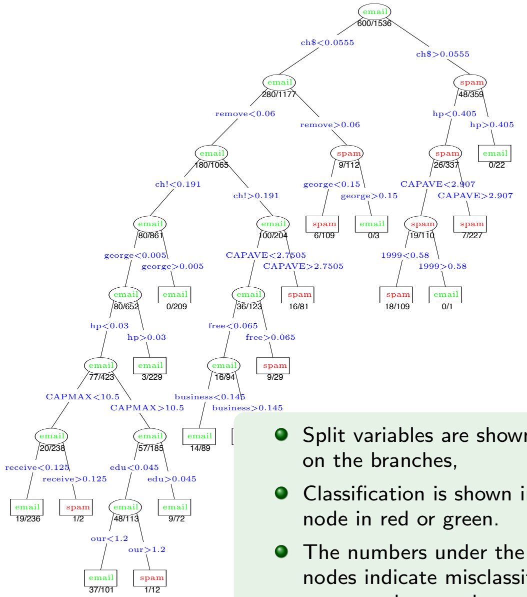
- Since they are more "sensitive" to pure splits, it's better to use either Gini or cross-entropy when *growing* the tree.
- Any of the three measures can be used for cost-complexity pruning. Common practice is to use the misclassification rate.
  - That's because prediction is usually the final goal, and misclassification measures ability to do that.

A spam example

Email data set, donated by George Forman from HP. 4601 messages.

- 48 quantitative predictors: the percentage of words in the email that match a given word. Examples include business, address, internet, free, and george. (These could be customized for individual users.)
- 6 quantitative predictors: the percentage of characters in the email that match a given character. The characters are ch;, ch(, ch[, ch!, ch$, and ch#.
- The average length of uninterrupted sequences of capital letters: CAPAVE.
- The length of the longest uninterrupted sequence of capital letters: CAPMAX.
- The sum of the length of uninterrupted sequences of capital letters: CAPTOT.

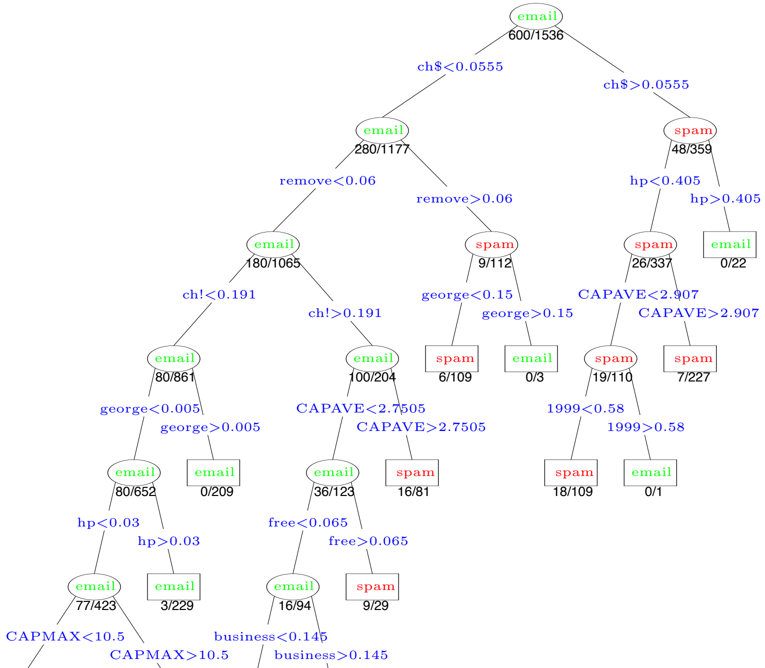

α
176
21
7
5
3
2
0
Misclassification Rate
0.0
0.1
0.2
0.3
0.4
0
10
20
30
40
Tree Size



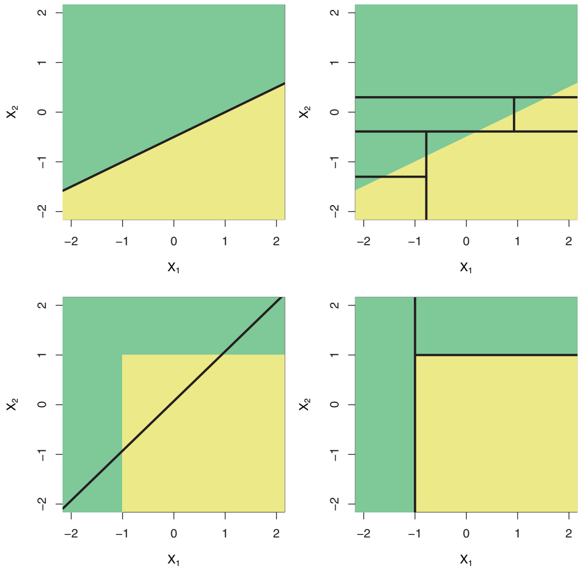
- Split variables are shown in blue on the branches,
- Classification is shown in every node in red or green.
- The numbers under the terminal nodes indicate misclassification rates on the test data.

Figure from ESLII

A zoom in


email
600/1536
ch$<0.0555
ch$>0.0555
email
280/1177
spam
48/359
remove<0.06
remove>0.06
hp<0.405
hp>0.405
email
180/1065
spam
9/112
spam
26/337
email
0/22
ch!<0.191
ch!>0.191
george<0.15
george>0.15
CAPAVE<2.907
CAPAVE>2.907
email
80/861
email
100/204
spam
6/109
email
0/3
spam
19/110
spam
7/227
george<0.005
george>0.005
CAPAVE<2.7505
CAPAVE>2.7505
1999<0.58
1999>0.58
email
80/652
email
0/209
email
36/123
spam
16/81
spam
18/109
email
0/1
hp<0.03
hp>0.03
free<0.065
free>0.065
email
77/423
email
3/229
email
16/94
spam
9/29
CAPMAX<10.5
CAPMAX>10.5
business<0.145
business>0.145

When are trees better than linear models?



Reminder: advantages and disadvantages

Advantages

- Easy to explain and non-experts can understand the results.
- They're more like human decision-making. Doctors like them.
- They easily handle qualitative predictors – no need for dummies.

Disadvantages

- As described, they don't usually provide the same predictive power that the other tools we've studied can.
- They can be pretty sensitive to small changes in the data.
- Recursive binary splitting may generate a suboptimal tree (like forward / backward model selection). Things later in the chapter address this.