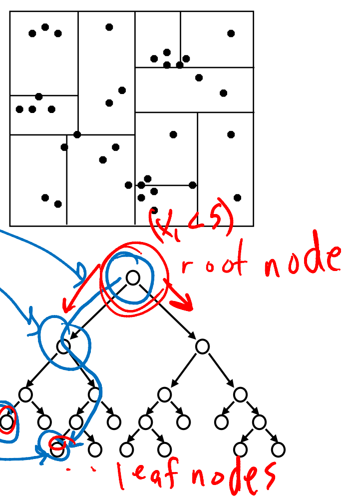
Module 4: Coping with Multiple Predictors

Regression Trees

STAT/BIOSTAT 527, University of Washington
Emily Fox
May 15th, 2014

Recursive Binary Partitions

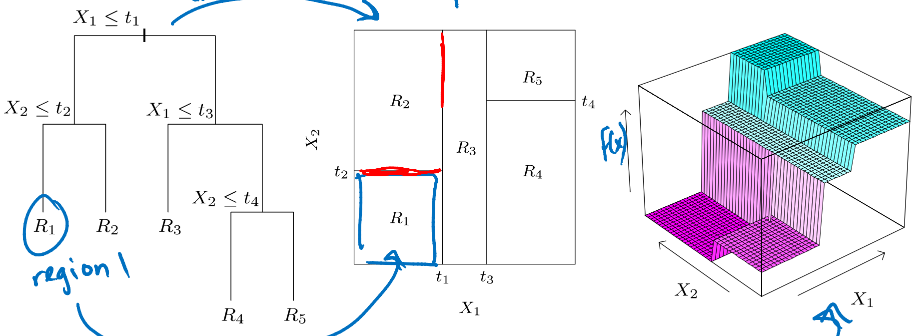
- To simplify the process and interpretability, consider ***recursive binary partitions***
- Described via a rooted tree
  - Every node of the tree corresponds to split decision
  - Leaves contain a subset of the data that satisfy the conditions
    - all conditions on path from root to leaf
    - think of pinball falling to leaf



Figures from Andrew Moore kd-tree tutorial

Resulting Model

tree maps to this partition



region 1

- Model the response as constant within each region

$$f(x) = \sum_{m=1}^{M} \beta_m I(x \in R_m)$$

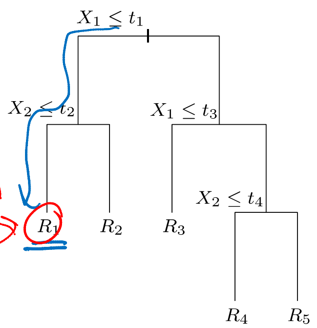
Figures from Hastie, Tibshirani, Friedman book

Basis Expansion Interpretation

- Equivalent to a basis expansion

$$f(x) = \sum_{m=1}^{M} \beta_m h_m(x)$$

indicators on regions (R_m)



- In this example:

$$h_1(x_1, x_2) = I(x_1 \leq t_1)I(x_2 \leq t_2)$$
$$h_2(x_1, x_2) = I(x_1 \leq t_1)I(x_2 > t_2)$$
$$h_3(x_1, x_2) = I(x_1 > t_1)I(x_1 \leq t_3)$$
$$h_4(x_1, x_2) = I(x_1 > t_1)I(x_1 > t_3)I(x_2 \leq t_4)$$
$$h_5(x_1, x_2) = I(x_1 > t_1)I(x_1 > t_3)I(x_2 > t_4)$$

reduced tensor product spline w/ step fcn basis

Choosing a Split Decision

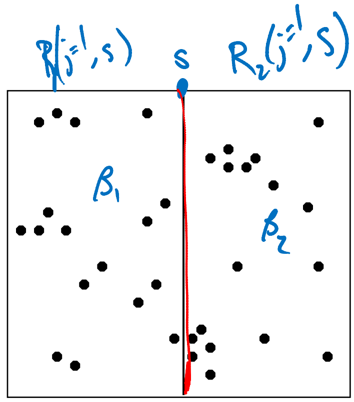
- Starting with all of the data, consider splitting on variable j at point s
- Define

$$R_1(j,s) = \{x \mid x_j \le s\}$$

$$R_2(j,s) = \{x \mid x_j > s\}$$

- Our objective is

$$\min_{j,s}\left[\min_{\beta_1}\sum_{x_i\in R_1(j,s)}(y_i-\beta_1)^2 + \min_{\beta_2}\sum_{x_i\in R_2(j,s)}(y_i-\beta_2)^2\right]$$



- For any (*j*, *s*), the inner minimization is solved by

$$\hat{\beta}_k = \text{avg}(y_i \mid x_i \in R_k(j,s)) \quad k=1,2$$

Cost-Complexity Pruning

- Searching over all subtrees and selecting using AIC or CV is not possible since there is an exponentially large set of subtrees

→ look at penalized RSS instead

- Define a subtree $T \subset T_0$ (full tree) to be any tree obtained by pruning T_0

prune = collapse an internal node

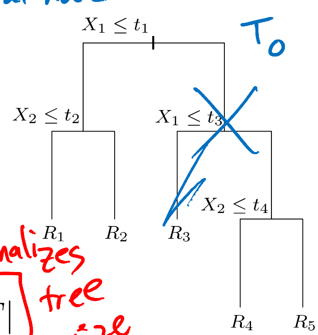
and $|T| =$ # of leaf nodes

$$n_m = |\{x_i \in R_m\}|$$

$$\hat{\beta}_m = \frac{1}{n_m}\sum_{x_i\in R_m} y_i$$

region-specific RSS ↓

$$Q_m(T) = \frac{1}{n_m}\sum_{x_i\in R_m}(y_i-\hat{\beta}_m)^2$$



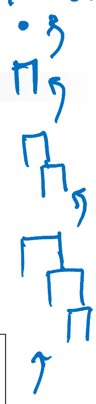
- We examine a complexity criterion

$$C_\lambda(T) = \underbrace{\sum_{m=1}^{|T|} n_m Q_m(T)}_{\text{RSS}} + \boxed{\lambda|T|}$$

penalizes tree size

Cost-Complexity Pruning

Sequence:

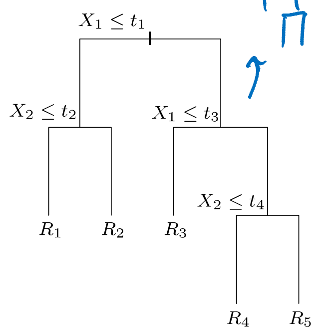


compute for λ and all trees in sequence

$$C_\lambda(T) = \sum_{m=1}^{|T|} n_m Q_m(T) + \lambda |T|$$

- Can find using *weakest link pruning*
  - Successively collapse the internal node that produces smallest increase in RSS

    $\sum_m n_m Q_m(T)$
  - Continue until at single-node (root) tree
  - Produces a finite sequence of subtrees, which must contain T_λ
  - See Breiman et al. (1984) or Ripley (1996)



- Choose λ via 5- or 10-fold CV $\rightarrow \hat{\lambda}$
- Final tree: $T_{\hat{\lambda}}$

Issues

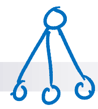
- Unordered categorical predictors
  - With unordered categorical predictors with *q* (levels) possible values, there are 2^{q-1}-1 possible choices of partition points to consider for each variable
  - Prohibitive for large *q*
  - Can deal with this for binary *y*…will come back to this in "classification"

- Missing predictor values…how to cope?
  - Can discard
  - Can fill in, e.g., with mean of other variables
  - With trees, there are better approaches

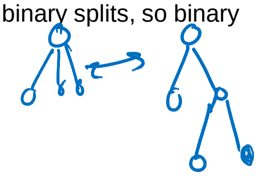
    -- Categorical predictors: make new category "missing"

    -- Split on observed data. For every split, create an ordered list of "surrogate" splits (predictor/value) that create similar divides of the data. When examining observation with a missing predictor, when splitting on that dimension, use top-most surrogate that is available instead

Issues



- Binary splits
  - Could split into more regions at every node
  - However, this more rapidly fragments the data leaving insufficient data and subsequent levels
  - Multiway splits can be achieved via a sequence of binary splits, so binary splits are generally preferred



- Instability
  - Can exhibit high variance
  - Small changes in the data → big changes in the tree
  - Errors in the top split propagates all the way down
  - ***Bagging*** averages many trees to reduce variance
- Inference
  - Hard…need to account for stepwise search algorithm

Issues

- Lack of smoothness
  - Fits piecewise constant models…unlikely to believe this structure
  - ***MARS*** address this issue (can view as modification to CART)

↖ later this lecture

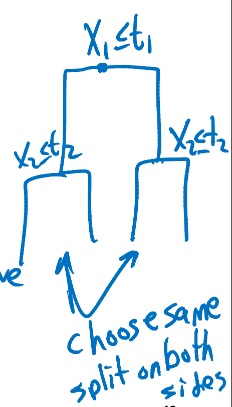
- Difficulty in capturing additive structure
  - Imagine true structure is

$$y = \beta_1 I(x_1 < t_1) + \beta_2 I(x_2 < t_2) + \epsilon$$

  - No encouragement to find this structure

- hard w/o sufficient data

- this is just w/ 2 additive effects. Harder to happen or notice w/ more.



Multiple Adaptive Regression Splines

- MARS is an adaptive procedure for regression
  - Well-suited to high-dimensional covariate spaces
- Can be viewed as:
  - Generalization of step-wise linear regression
  - Modification of CART

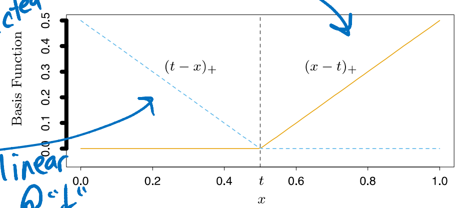
focus on this first

- Consider a basis expansion in terms of piecewise linear basis functions (linear splines)

CART
$I(x-t>0)$
$I(x-t<0)$

$(x-t)_+$
$(t-x)_+$
"reflected pairs"

piecewise linear w/ knot @ "t"



From Hastie, Tibshirani, Friedman book

Multiple Adaptive Regression Splines

$X \in \mathbb{R}^d$

- Take knots at all observed x_{ij}

$$\mathcal{C} = \{(x_j - t)_+, (t - x_j)_+\}, \quad t \in \{x_{1j}, \ldots, x_{nj}\}, \; j = 1, \ldots, d$$

  - If all locations are unique, then $2*n*d$ basis functions (2nd)
  - Treat each basis function as a function on x, just varying with x_j ($\in \mathbb{R}^d$)

$$\underset{\in \mathbb{R}^d}{h_m(x)} = (x_j - t)_+$$

- The resulting model has the form

$$f(x) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(x)$$

LBE

$h_m \in \mathcal{C}$ or products of fn's in $\mathcal{C}$

- Built in a forward stepwise manner in terms of this basis

MARS Forward Stepwise

- Given a set of h_m, estimation of β_m proceeds as with any linear basis expansion (i.e., minimizing the RSS)
- How do we choose the set of h_m?

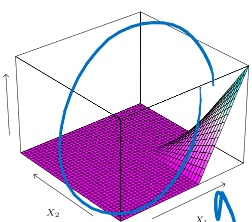
basis fn's

1. Start with $h_0(x) = 1$ and M=0
2. Consider product of all h_m in current model with reflected pairs in C
   -- Add terms of the form
   $$\hat{\beta}_{M+1}h_\ell(x)(x_j - t)_+ + \hat{\beta}_{M+2}h_\ell(x)(t - x_j)_+ \quad , h_l \in \text{"Model"}$$
   $\hat\beta_{M+1}, \hat\beta_{M+2}$ are est. using LS + all other terms in Model.
   -- Select the one that decreases the training error most
3. Increment M and repeat $M = M + 2$
4. Stop when preset M is hit
5. Typically end with a large (overfit) model, so backward delete
   -- Remove term with smallest increase in RSS
   -- Choose model based on generalized CV

MARS Forward Stepwise Example

general terms:
$$\hat{\beta}_{M+1}h_\ell(x)(x_j - t)_+ + \hat{\beta}_{M+2}h_\ell(x)(t - x_j)_+$$

- At the first stage, add term of form
  $$\beta_1(x_j - t)_+ + \beta_2(t - x_j)_+$$
  $h_0(x) = 1$
  "x_{ij}"
  with the optimal pair being
  $\hat\beta_1 (x_2 - x_{72})_+ + \hat\beta_2 (x_{72} - x_2)_+$



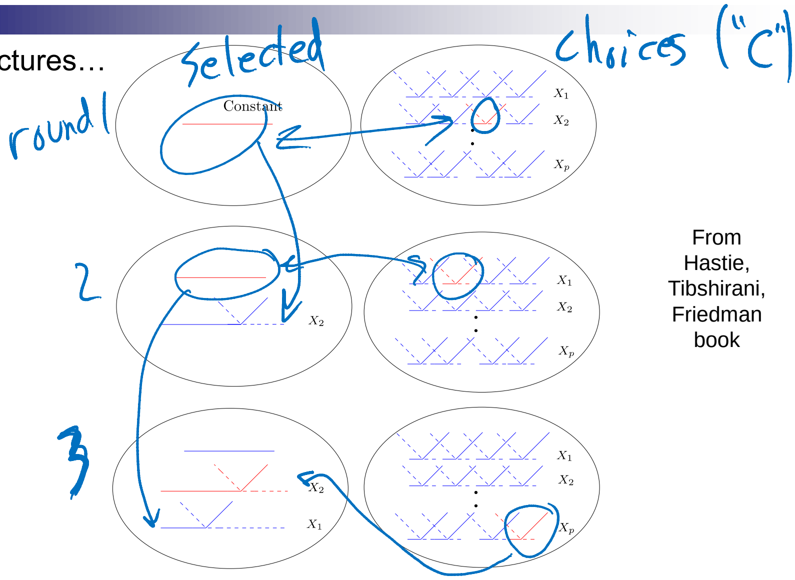
- Add pair to the model and then consider including a pair like
  $$\beta_3 h_m(x)(x_j - t)_+ + \beta_4 h_m(x)(t - x_j)_+$$
  with choices for h_m being:
  $h_0(x) = 1$
  $h_1(x) = (x_2 - x_{72})_+$
  $h_2(x) = (x_{72} - x_2)_+$

the term $(x_1 - x_{51})_+ (x_{72} - x_2)_+$ is considered

Figure from Hastie, Tibshirani, Friedman book

MARS Forward Stepwise

- In pictures…



From Hastie, Tibshirani, Friedman book

Why MARS?

- Why these piecewise linear basis functions?
  - Ability to operate locally
    - When multiplied, non-zero only over small part of the input space
    - Resulting regression surface has local components and only where needed (spend parameters carefully in high dims)
  - Computations with linear basis are very efficient
    - Naively, we consider fitting n reflected pairs for each input x_j → $O(n^2)$ operations
    - Can exploit simple form of piecewise linear function (just like CART)
    - Fit function with rightmost knot. As knot moves, basis functions differ by 0 over the left and by a constant over the right → Can try every knot in $O(n)$

Why MARS?

- Why forward stagewise?
  - Hierarchical in that multiway products are built from terms already in model (e.g., 4-way product exists only if 3-way already existed)
  - Higher order interactions tend to only exist if some of the lower order interactions exist as well
  - Avoids search over exponentially large space (i.e. all subsets)

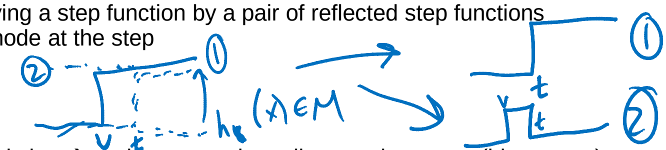
NO NO $(x_1 - x_{71})_+ \; (x_{21} - x_1)_+$ etc.

- Notes:
  - Each input can appear at most once in a product…Prevents formation of higher-order powers of an input
  - Can place limit on order of interaction. That is, one can allow pairwise products, but not 3-way or higher.
  - Limit of 1 → additive model

R package: "earth"

Connecting MARS and CART

- MARS and CART have lots of similarities
- Take MARS procedure and make following modifications:
  - Replace piecewise linear with step functions $I(x-t>0), I(x-t\le 0)$
  - When a model term h_m is involved in a multiplication by a candidate term, replace it by the interaction and is not available for further interaction in "C"
- Then, MARS forward procedure = CART tree-growing algorithm
  - Multiplying a step function by a pair of reflected step functions = split node at the step



  - 2nd restriction → node may not be split more than once (binary tree)
- MARS doesn't force tree structure → can capture additive effects

What you need to know

- Regression trees provide an adaptive regression method
- Fit constants (or simple models) to each region of a partition
- Relies on estimating a binary tree partition
  - Sequence of decisions of variables to split on and where
  - Grown in a greedy, forward-wise manner
  - Pruned subsequently
- Implicitly performs variable selection
- MARS is a modification to CART allowing linear fits

Readings

- Wakefield – 12.7
- Hastie, Tibshirani, Friedman – 9.2.1-9.2.2, 9.2.4, 9.4
- Wasserman – 5.12

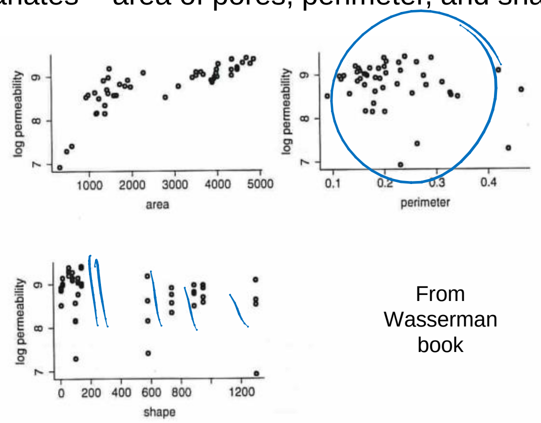
Module 4: Coping with Multiple Predictors

A Short Case Study

STAT/BIOSTAT 527, University of Washington
Emily Fox
May 15th, 2014

Rock Data

- 48 rock samples from a petroleum reservoir
- Response = permeability
- Covariates = area of pores, perimeter, and shape

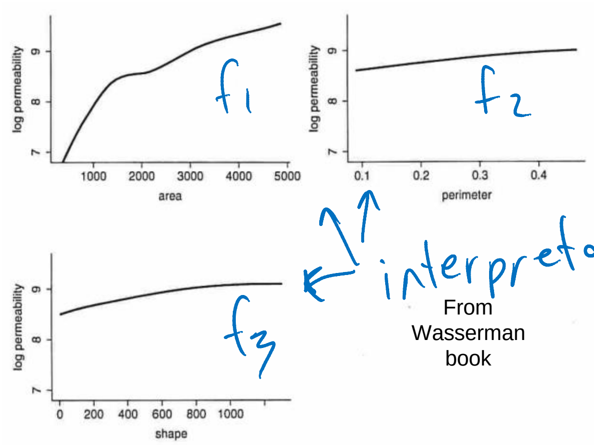


From Wasserman book

Generalized Additive Model

- Fit a GAM:

$$\text{permeability} = f_1(\text{area}) + f_2(\text{perimeter}) + f_3(\text{shape}) + \epsilon$$



interpretable!!

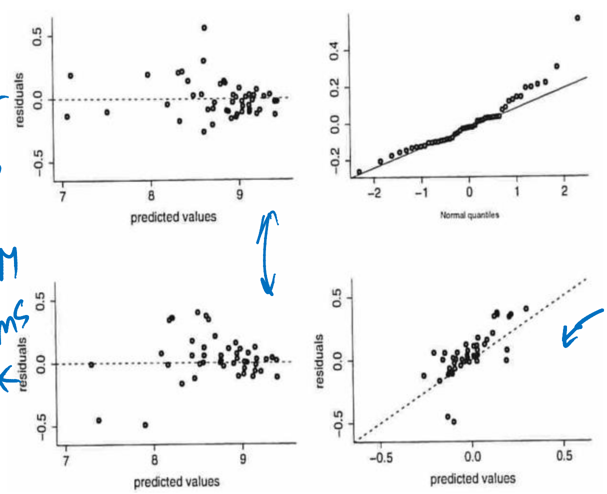
From Wasserman book

GAM vs. Local Linear Fits

- Comparison to a 3-dimensional local linear fit

similar fits

→ GAM seems ok



From Wasserman book

residuals of GAM vs. local linear

Projection Pursuit

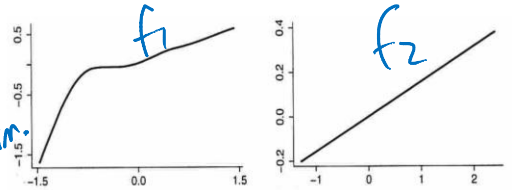
$$f(x_1, \ldots, x_d) = \alpha + \sum_{m=1}^{M} f_m(w_m^T x)$$

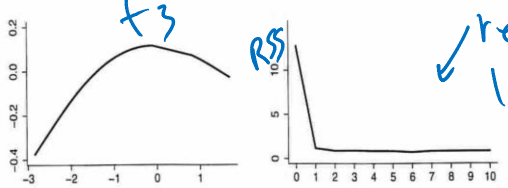
V_m

- Applying projection pursuit with M = 3 yields

$w_1 = (.99, .07, .08)^T, \; w_2 = (.43, .35, .83)^T, \; w_3 = (.74, -.28, -.61)^T$

$V_1 = 0.99$ area $+ 0.07$ perim. $+ 0.08$ shape



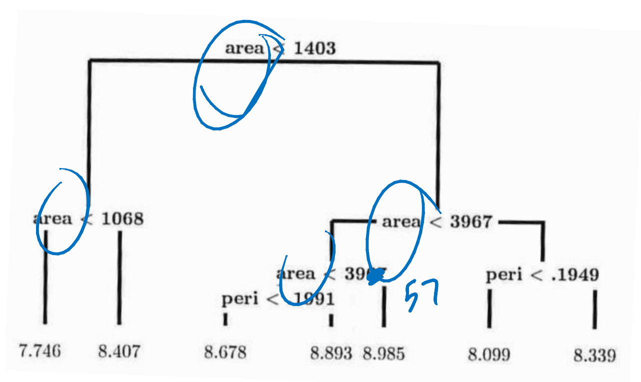


From Wasserman book

really only need 1 or 2 terms ... able to capture interactions!

Regression Trees

- Fit a regression tree to the rock data
- Note that the variable "shape" does not appear in the tree



From Wasserman book

Module 5: Classification

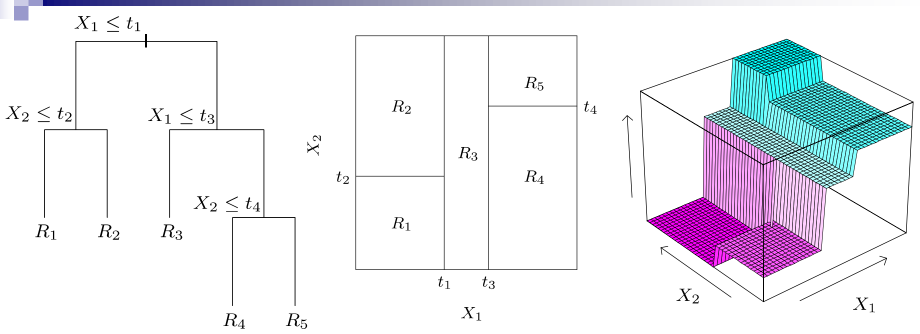
A First Look at Classification: CART

STAT/BIOSTAT 527, University of Washington

Emily Fox

May 15th, 2014

Regression Trees



- So far, we have assumed continuous responses *y* and looked at regression tree models:

$$f(x) = \sum_{m=1}^{M} \beta_m I(x \in R_m)$$

Figures from Hastie, Tibshirani, Friedman book

Classification Trees

- What if our response *y* is ***categorical*** and our goal is classification?

$y \in \{\text{'email'}, \text{'spam'}\} \to \{0,1\}$

$y \in \{G_1, \ldots, G_K\}$

- Can we still use these tree structures? YES!!!
- Recall our ***node impurity*** measure

$$Q_m(T) = \frac{1}{n_m} \sum_{x_i \in R_m} (y_i - \hat{\beta}_m)^2 \quad \text{(RSS)}$$

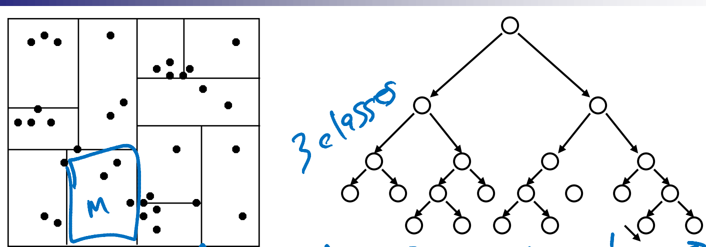
  - Used this for growing the tree

$$\min_{j,s} \left[\sum_{x_i \in R_1(j,s)} (y_i - \hat{\beta}_1)^2 + \sum_{x_i \in R_2(j,s)} (y_i - \hat{\beta}_2)^2 \right]$$

  - As well as pruning $C_\lambda(T) = \sum_{m=1}^{|T|} n_m Q_m(T) + \lambda |T|$

- Clearly, squared-error is not the right metric for classification

Classification Trees



$\hat{p}_{m1} = 0 \quad \hat{p}_{m2} = \frac{2}{3} \; \ldots \; \hat{p}_{m3} = \frac{1}{3} \to$ class 2

- First, what is our decision rule at each leaf?
  - Estimate probability of each class given data at leaf node:

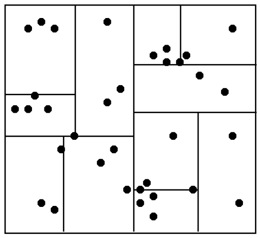
$$\hat{p}_{mk} = \frac{1}{n_m} \sum_{x_i \in R_m} \mathbb{I}(y_i = k)$$

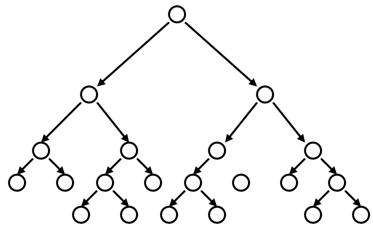
  - Majority vote:

$$k(m) = \arg\max_k \hat{p}_{mk}$$

Figures from Andrew Moore kd-tree tutorial

Classification Trees





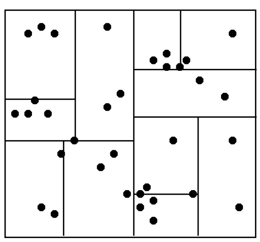
$Q_m(T)$

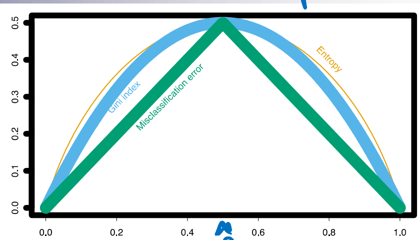
- How do we measure ***node impurity*** for this fit/decision rule?
  - Misclassification error: $\frac{1}{n_m}\sum_{i \in R_m} \mathbb{I}(y_i \neq k(m)) = 1-\hat{p}_{mk(m)}$
  - Gini index: $\sum_{k \neq k'} \hat{p}_{mk}\hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1-\hat{p}_{mk})$
  - Cross-entropy or deviance: $-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$

Figures from Andrew Moore kd-tree tutorial

Classification Trees

max at $\hat{p}=0.5$





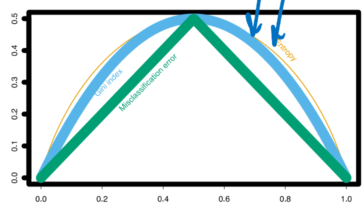
$\hat{p}$

From Hastie, Tibshirani, Friedman book

- How do we measure ***node impurity*** for this fit/decision rule?
  - Misclassification error (K=2): $1-\max(\hat{p}, 1-\hat{p})$, $\hat{p}$ = prop. in class 2
  - Gini index (K=2): $2\hat{p}(1-\hat{p})$
  - Cross-entropy or deviance (K=2): $-\hat{p}\log\hat{p}-(1-\hat{p})\log(1-\hat{p})$

Notes on Impurity Measures

diff.



From Hastie, Tibshirani, Friedman book

- Impurity measures
  - Misclassification error: $1 - \hat{p}_{mk(m)}$
  - Gini index: $\sum_{k=1}^{K} \hat{p}_{mk}(1-\hat{p}_{mk})$
  - Cross-entropy or deviance: $-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$
- Comments:
  - Differentiability — Gini + cross-entropy
  - Sensitivity to changes in node probabilities

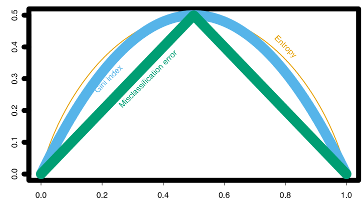
$(400^{(1)}, 400^{(2)}) \rightarrow (100^{(1)}, 300^{(2)}) + (300^{(1)}, 100^{(2)})$ ⟩ misclass. rate = 0.25

$\rightarrow (200, 400) + (200, 0)$ ← pure node, want this

(Gini + entropy are lower there)

  - Often use Gini or cross-entropy for growing tree, and misclass. for pruning

Notes on Impurity Measures



From Hastie, Tibshirani, Friedman book

- Impurity measures
  - Misclassification error: $1 - \hat{p}_{mk(m)}$
  - Gini index: $\sum_{k=1}^{K} \hat{p}_{mk}(1-\hat{p}_{mk})$
  - Cross-entropy or deviance: $-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$
- Other interpretations of Gini index:
  - Instead of majority vote, classify observations to class *k* with prob. $\hat{p}_{mk}$

$\text{Error} = \sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'}$ P(classify to k')

prop. of class k'

  - Code each observation as 1 for class *k* and 0 otherwise
    - Variance: 1 against all $\hat{p}_{mk}(1-\hat{p}_{mk})$
    - Summing over *k* gives the Gini index

Classification Tree Issues

- Unordered categorical predictors
  - With unordered categorical predictors with q possible values, there are $2^{q-1}-1$ possible choices of partition points to consider for each variable
  - For binary (0-1) outcomes, can order predictor classes according to proportion falling in outcome class 1 and then treat as ordered predictor
    - Gives optimal split in terms of cross-entropy or Gini index
  - Also holds for quantitative outcomes and square-error loss…order predictors by increasing mean of the outcome
  - No results for multi-category outcomes

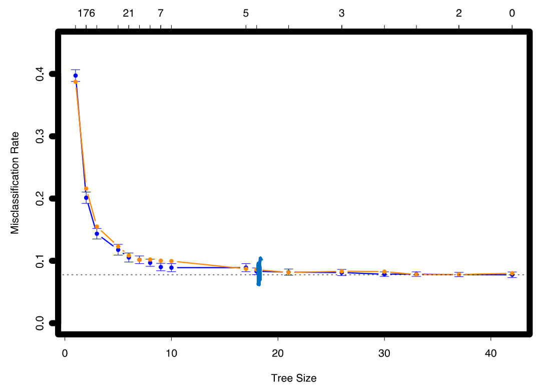
- Loss matrix
  - In some cases, certain misclassifications are worse than others
    predicting no disease when disease
  - Introduce ***loss matrix*** …more on this soon
  - See Tibshirani, Hastie and Friedman for how to incorporate into CART

Classification Tree Spam Example

- Example: *predicting spam*
- Data from UCI repository (GAMs)
- Response variable: *email* or *spam* (0 / 1)
- 57 predictors:
  - 48 quantitative – percentage of words in email that match a give word such as “business”, “address”, “internet”,…
  - 6 quantitative – percentage of characters in the email that match a given character (; , [! $ #)
  - The average length of uninterrupted 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

Classification Tree Spam Example

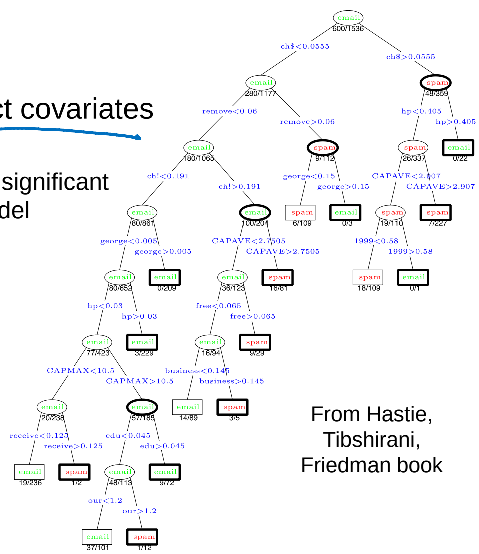
- Used cross-entropy to grow tree and misclassification to prune
- 10-fold CV to choose tree size
  - CV indexed by λ
  - Sizes refer to $|T_\lambda|$
  - Error rate flattens out around a tree of size 17



From Hastie, Tibshirani, Friedman book

Classification Tree Spam Example

- Resulting tree of size 17
- Note that there are 13 distinct covariates split on by the tree
  - 11 of these overlap with the 16 significant predictors from the additive model previously explored

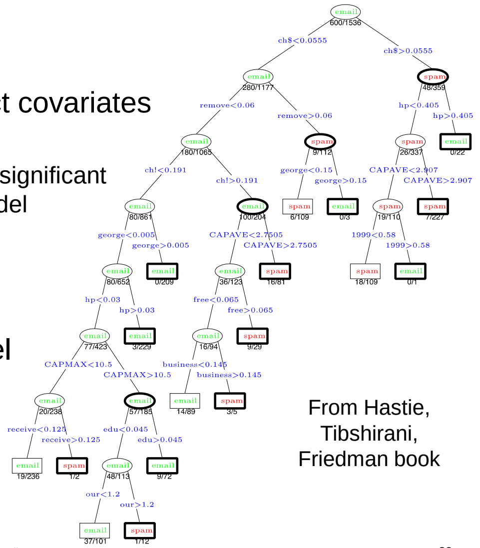


From Hastie, Tibshirani, Friedman book

Classification Tree Spam Example

- Resulting tree of size 17
- Note that there are 13 distinct covariates split on by the tree
  - 11 of these overlap with the 16 significant predictors from the additive model previously explored
- Overall error rate (9.3%) is higher than for additive model

	Predicted	
True	email	spam
email	57.3%	4.0%
spam	5.3%	33.4%



From Hastie, Tibshirani, Friedman book

What you need to know

- Classification trees are a straightforward modification to the regression tree setup
- Just need new definition of node impurity for growing and pruning tree
- Decision at the leaves is a simple majority-vote rule

Readings

- Wakefield – 10.3.2, 10.4.2, 12.8.4
- Hastie, Tibshirani, Friedman – 9.2.3, 9.2.5, 2.4