Module 4: Coping with Multiple Predictors

Multidimensional Kernel Methods

STAT/BIOSTAT 527, University of Washington
Emily Fox
May 16th, 2013

Kernel Density Estimation

- Kernel methods are often used for density estimation (actually, classical origin)
- Assume random sample $X_1, \dots, X_n \overset{iid}{\sim} P$
- Choice #1: empirical estimate? $\hat{P} = \frac{1}{n} \sum \delta_{x_i}$

- Choice #2: as before, maybe we should use an estimator

$\hat{P}(x_0) = \frac{\#x_i \in \text{Nbhd}(x_0)}{n\lambda}$ ← width of nbhd

- Choice #3: again, consider kernel weightings instead

$\hat{P}(x_0) = \frac{1}{n\lambda} \sum K_\lambda(x_0, x_i)$ Parzen est.

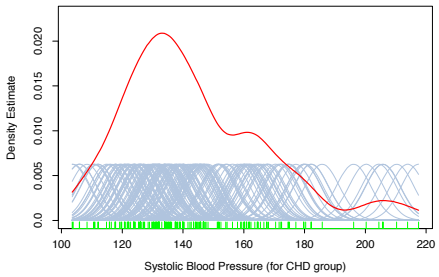
Kernel Density Estimation

- Popular choice = Gaussian kernel → ***Gaussian KDE***

$$\hat{p} = \frac{1}{n}\sum_{i=1}^{n} \phi_\lambda(x - x_i) \qquad \phi_\lambda$$

$$= (\hat{p} * \phi_\lambda)(x)$$

↑ empirical dist.



From Hastie, Tibshirani, Friedman book

Multivariate KDE

- In 1d $$\hat{p}(x_0) = \frac{1}{n\lambda}\sum_{i=1}^{n} K_\lambda(x_0, x_i)$$
- In R^d, assuming a product kernel,

$$\hat{p}(x_0) = \frac{1}{n\lambda_1 \cdots \lambda_d}\sum_{i=1}^{n}\left\{\prod_{j=1}^{d} K_{\lambda_j}(x_{0j}, x_{ij})\right\}$$

- Typical choice = Gaussian RBF

Multivariate KDE

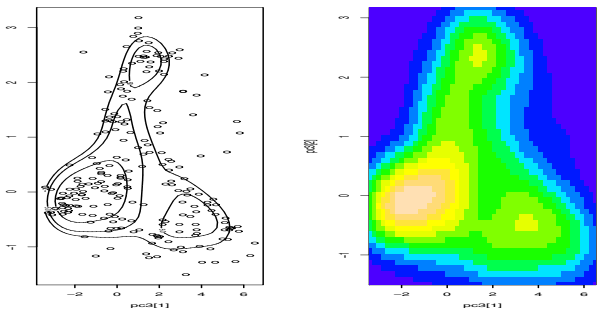
$$\hat{p}(x_0) = \frac{1}{n\lambda_1 \cdots \lambda_d} \sum_{i=1}^{n} \left\{ \prod_{j=1}^{d} K_{\lambda_j}(x_{0j}, x_{ij}) \right\}$$

- Risk grows as $O(n^{-4/(4+d)})$
- Example: To ensure relative MSE < 0.1 at 0 when the density is a multivariate norm and optimal bandwidth is chosen

- Always report confidence bands, which get wide with *d*

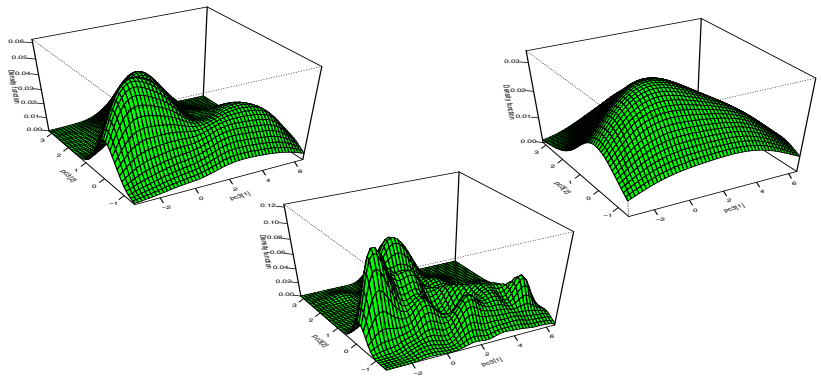
Multivariate KDE Example

- Data on 6 characteristics of aircraft (Bowman and Azzalini 1998)
- Examine first 2 principle components of the data
- Perform KDE with independent kernels



Multivariate KDE Example

- Data on 6 characteristics of aircraft (Bowman and Azzalini 1998)
- Examine first 2 principle components of the data
- Perform KDE with independent kernels



Module 4: Coping with Multiple Predictors

Regression Trees

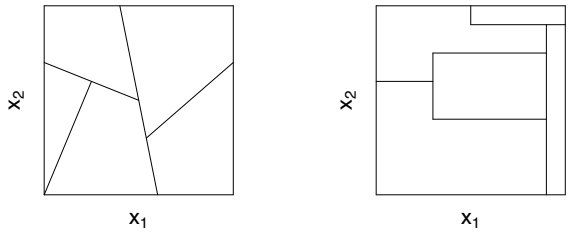
STAT/BIOSTAT 527, University of Washington

Emily Fox

May 16th, 2013

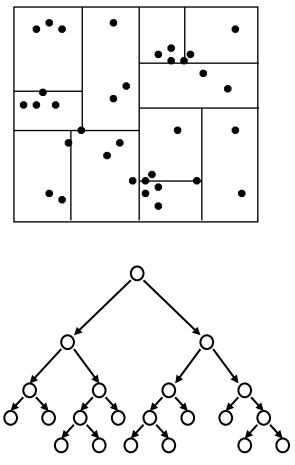
Regression Trees Overview

- An alternative adaptive regression technique
  - Conceptually simple
  - Powerful
- Partition the covariate space into regions and then fit a simple model in each (e.g., constant)
- How to partition?

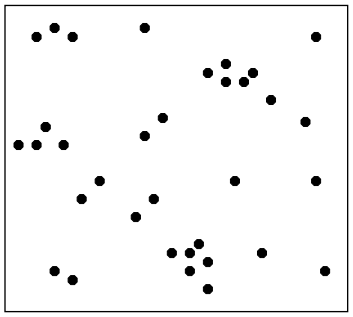


Recursive Binary Partitions

- 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



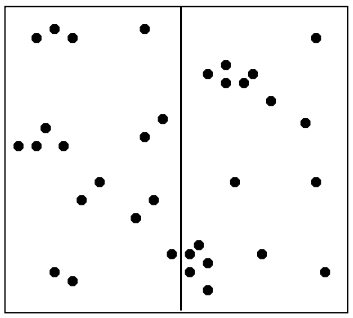
Recursive Binary Partitions

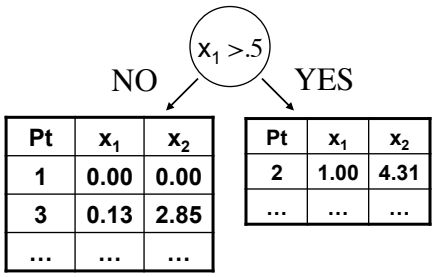


Pt	x_1	x_2
1	0.00	0.00
2	1.00	4.31
3	0.13	2.85
…	…	…

- Start with a list of *d*-dimensional points.

Recursive Binary Partitions



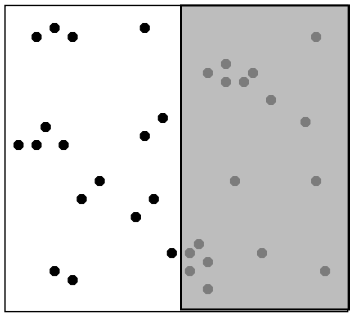


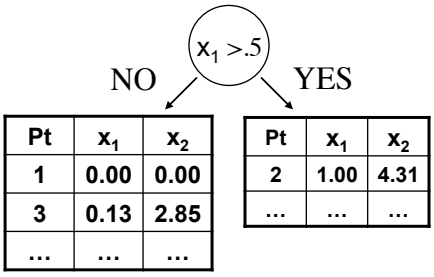
Pt	x_1	x_2
1	0.00	0.00
3	0.13	2.85
…	…	…

Pt	x_1	x_2
2	1.00	4.31
…	…	…

- Split the points into 2 groups by:
  - Choosing dimension d_j and value t_j (methods to be discussed…)
  - Separating the points into $x_{id_j} > t_j$ and $x_{id_j} <= t_j$.

Recursive Binary Partitions



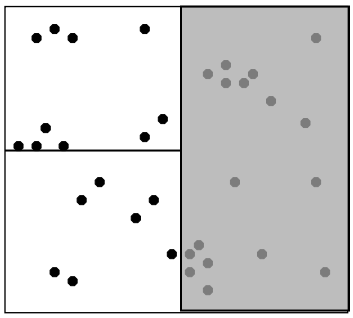


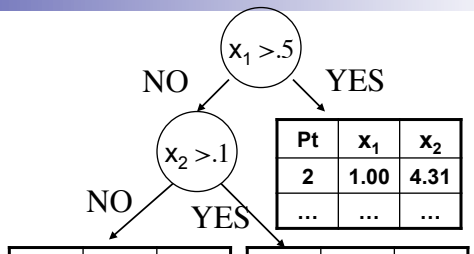
Pt	x_1	x_2
1	0.00	0.00
3	0.13	2.85
…	…	…

Pt	x_1	x_2
2	1.00	4.31
…	…	…

- Consider each group separately and possibly split again (along same/different dimension).
  - Stopping criterion to be discussed…

Recursive Binary Partitions





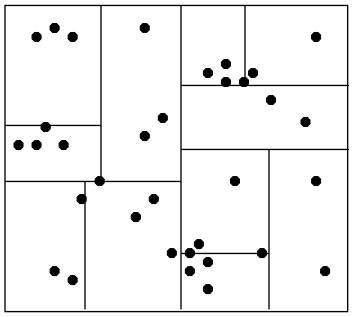
Pt	x_1	x_2
2	1.00	4.31
…	…	…

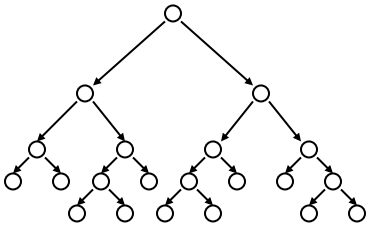
Pt	x_1	x_2
3	0.13	2.85
…	…	…

Pt	x_1	x_2
1	0.00	0.00
…	…	…

- Consider each group separately and possibly split again (along same/different dimension).
  - Stopping criterion to be discussed…

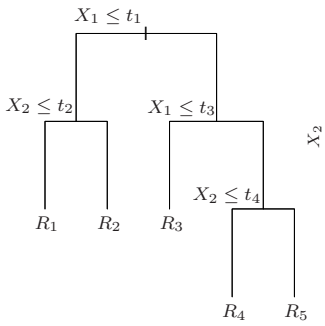
Recursive Binary Partitions

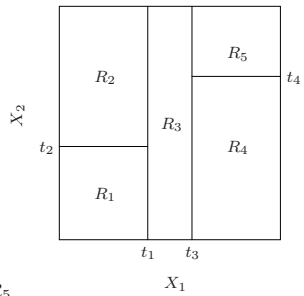


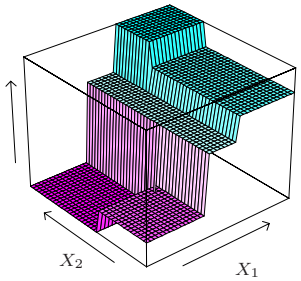


- Continue splitting points in each set
  - creates a binary tree structure
- Each leaf node contains a list of points

Resulting Model







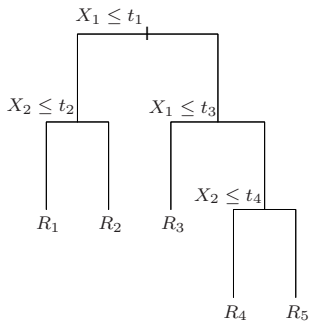
- Model the response as constant within each region

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

Basis Expansion Interpretation

- Equivalent to a basis expansion

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



- In this example:

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

Questions on Building the Tree

- Which variable should we split on?
- What threshold value should we consider?
- When should we stop the process?

Building the Tree

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

- Assume the partition ($R_1, \ldots R_M$) is given
- If criterion is to minimize RSS, then

$$\hat{\beta}_m =$$

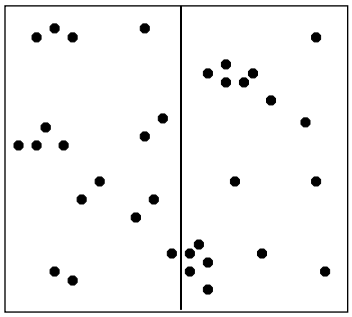
- How do we find the partition ($R_1, \ldots R_M$) ?
  - Finding the optimal tree that minimizes RSS is generally computationally infeasible
  - Consider a greedy algorithm instead

Choosing a Split Decision

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

$$R_1(j, s) = \{x \mid x_j \leq s\}$$
$$R_2(j, s) = \{x \mid x_j > s\}$$

- Our objective is



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

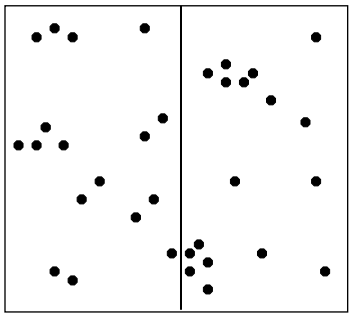
Choosing a Split Decision

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

$$\hat{\beta}_1 = \text{avg}(y_i \mid x_i \in R_1(j,s))$$

$$\hat{\beta}_2 = \text{avg}(y_i \mid x_i \in R_2(j,s))$$

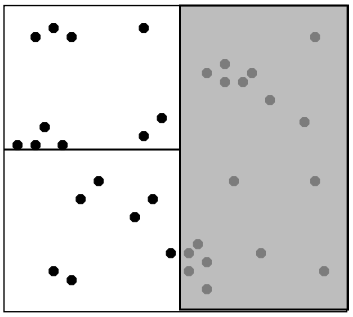
- For each splitting variable *j*, finding the optimal *s* can be done efficiently
  - Why?



- Max of *d(n-1)* partitions to consider
- So, determining (*j,s*) is feasible

Choosing a Split Decision

- Conditioning on the best split just found, we recurse on each of the two regions
- Repeat on all resulting regions
- When do we stop recursing?



How Large of a Tree?

- Large tree, like partitioning until each node has one observation →
- Small tree →
- Tree size is a tuning parameter that governs model complexity
  - Optimal tree size should be chosen adaptively from the data
- Stopping criterion
  - Stop when decrease in RSS due to a split falls below some threshold
  - Stop when a minimum node size (e.g., 5) is reached. Go back and prune.

Cost-Complexity Pruning

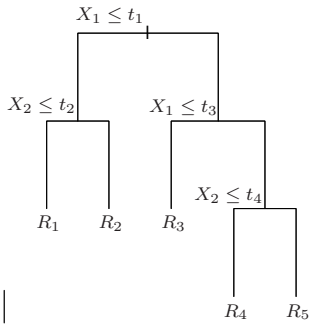
- Searching over all subtrees and selecting using AIC or CV is not possible since there is an exponentially large set of subtrees →
- Define a subtree $T \subset T_0$ to be any tree obtained by pruning T_0

and $|T| =$

$n_m =$

$\hat{\beta}_m =$

$Q_m(T) =$



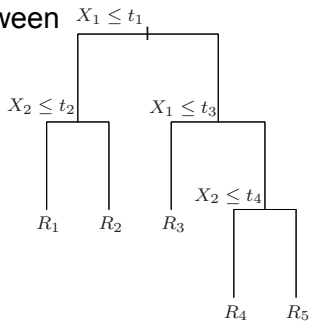
- We examine a complexity criterion

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

Cost-Complexity Pruning

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

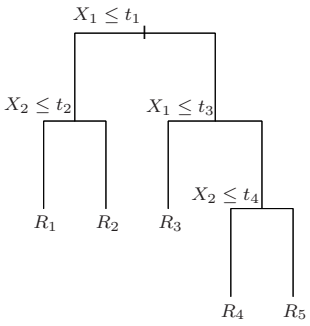
- For a given λ, want to find $T_\lambda \subset T_0$ to minimize $C_\lambda(T)$
- Tuning parameter λ governs tradeoff between tree size and goodness of fit to the data
  - Large λ →
  - λ = 0 →
- For each λ, can show that there is a unique smallest subtree T_λ



Cost-Complexity Pruning

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

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



Comments on Regression Trees

- Partition is not specified apriori, so regression trees provide a ***locally adaptive*** technique
- Effectively performs variable selection by discovering the relevant interaction terms
  - Implicit in the process
- In the construction, we are assuming that
  - Error terms are uncorrelated
  - Constant variance

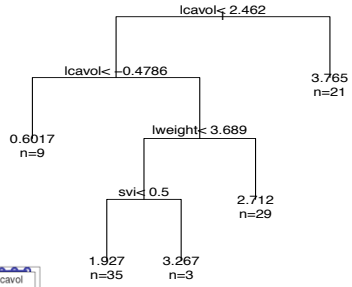
Example: Prostate Cancer

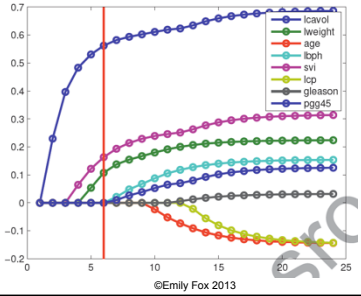
- Fit binary regression tree to log PSA with splits based on eight covariates
- Grow tree with condition of at least 3 observation per leaf
- Results in a tree with 27 splits
- Run weakest-link pruning for each candidate λ, with λ chosen according to CV

Example: Prostate Cancer

- Compare results to LASSO
  - lcavol most "important"
  - Then lweight and svi

$$h_1(\boldsymbol{x}) = I(\text{lcavol} < -0.4786)$$
$$h_2(\boldsymbol{x}) = I(\text{lcavol} < -0.4786) \times I(\text{lweight} < 3.689) \times I(\text{svi} < 0.5)$$
$$h_3(\boldsymbol{x}) = I(\text{lcavol} < -0.4786) \times I(\text{lweight} < 3.689) \times I(\text{svi} > 0.5)$$
$$h_4(\boldsymbol{x}) = I(\text{lcavol} < -0.4786) \times I(\text{lweight} \geq 3.689)$$
$$h_5(\boldsymbol{x}) = I(\text{lcavol} \geq 2.462).$$





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
  - 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

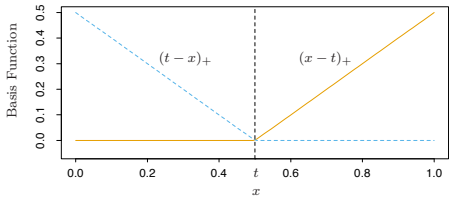
- Lack of smoothness
  - Fits piecewise constant models…unlikely to believe this structure
  - ***MARS*** address this issue (can view as modification to CART)
- Difficulty in capturing additive structure
  - Imagine true structure is

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

  - No encouragement to find this structure

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



Multiple Adaptive Regression Splines

- Take knots at all observed x_{ij}

$$\mathcal{C} = \{(x_j - t)_+, (t - x_j)_+\}$$

  - If all locations are unique, then *2*n*d* basis functions
  - Treat each basis function as a function on *x*, just varying with x_j

$$h_m(x) =$$

- The resulting model has the form

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

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

MARS Forward Stepwise

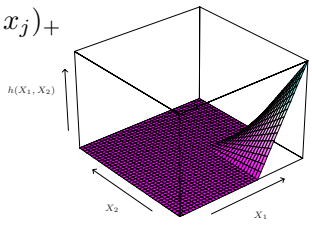
- Given a set of h_m, estimation of β_m proceeds as with any linear basis expansion (i.e., minimizing the RSS)
- How do we choose the set of h_m?

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
   $$\hat{\beta}_{M+1} h_\ell(x)(x_j - t)_+ + \hat{\beta}_{M+2} h_\ell(x)(t - x_j)_+$$
   -- Select the one that decreases the training error most
3. Increment M and repeat
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

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

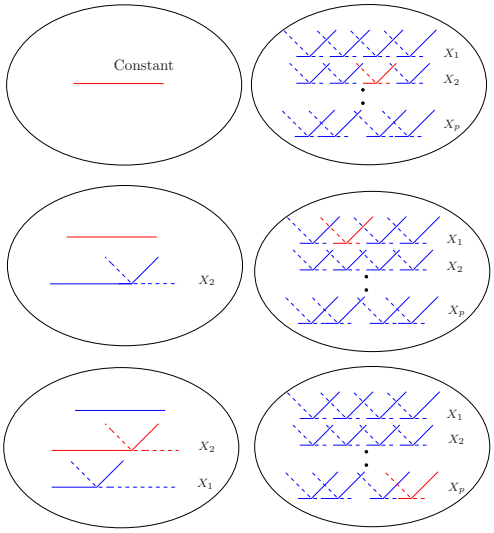
- At the first stage, add term of form
  $$\beta_1(x_j - t)_+ + \beta_2(t - x_j)_+$$
  with the optimal pair being



- 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:

MARS Forward Stepwise

- In pictures…



Why MARS?

- Why these piecewise linear basis functions?
  - Ability to operate locally
    - When multiplied, non-zero only over a 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
    - Fit function with rightmost knot. As knot moves, the basis functions differ by 0 over the left and by a constant over the right → Can try every knot in $O(n)$
- 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

Why MARS?

- 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

Connecting MARS and CART

- MARS and CART have lots of similarities
- Take MARS procedure and make following modifications:
  - Replace piecewise linear with step functions
  - 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
- 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

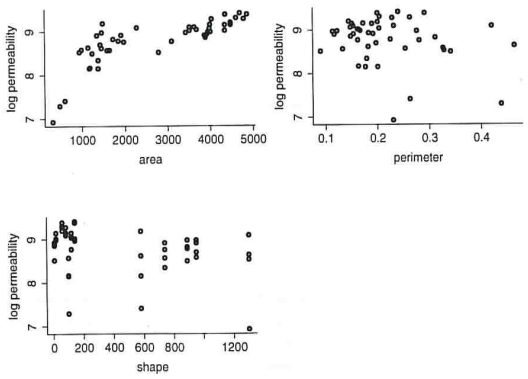
STAT/BIOSTAT 527, University of Washington

Emily Fox

May 16th, 2013

Rock Data

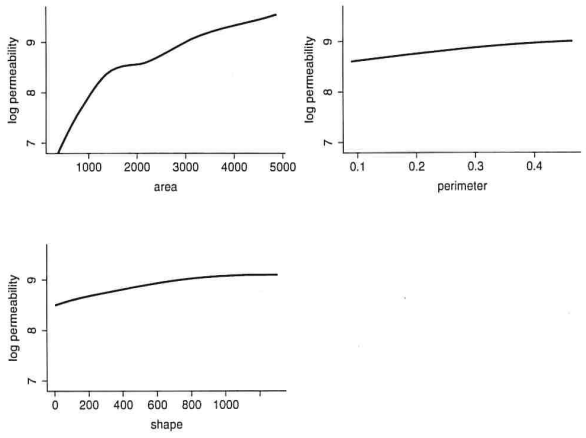
- 48 rock samples from a petroleum reservoir
- Response = permeability
- Covariates = area of pores, perimeter, and shape



Generalized Additive Model

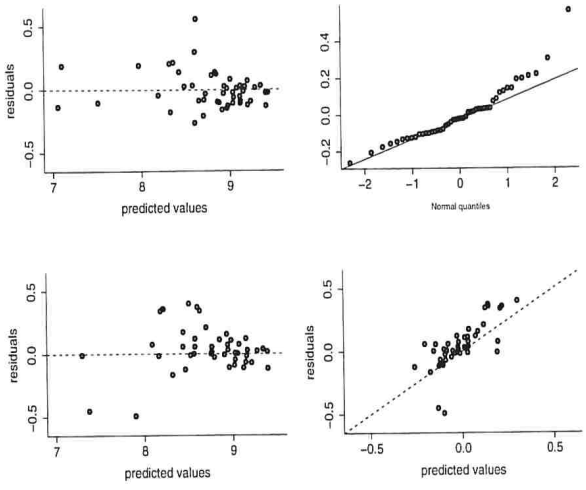
- Fit a GAM:

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



GAM vs. Local Linear Fits

- Comparison to a 3-dimensional local linear fit

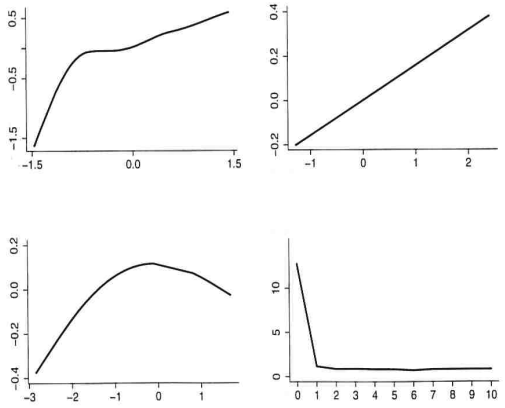


Projection Pursuit

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

- Applying projection pursuit with *M* = 3 yields

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



Regression Trees

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