## 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.

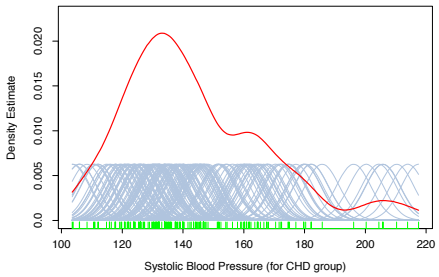
# Kernel Density Estimation

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

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

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

↑ 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, $x \in \mathbb{R}^d$

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

lots of params to choose

- Typical choice = Gaussian RBF → Gaussian KDE

$e^{-\frac{\|x_0 - x\|}{\lambda}}$

# 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 $O(n^{-4/(4+d)})$ ★ increase very rapidly w/ d
- Example: To ensure relative MSE < 0.1 at 0 when the density is a multivariate norm and optimal bandwidth is chosen

| dim | sample size |
|---|---|
| 1 | 4 |
| 2 | 19 |
| 3 | 67 |
| ⋮ | ⋮ |
| 7 | 10,700 |
| ⋮ | ⋮ |
| 10 | 842,000 |

!!!

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

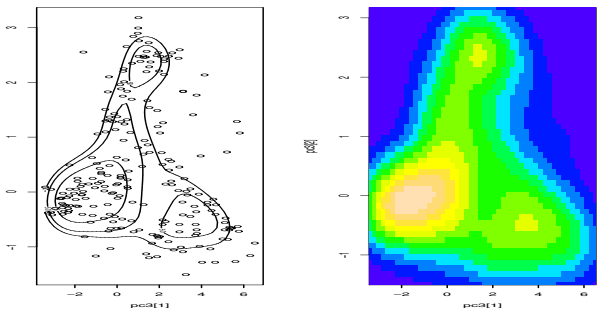
reflects difficulty of problem





# Multivariate KDE Example

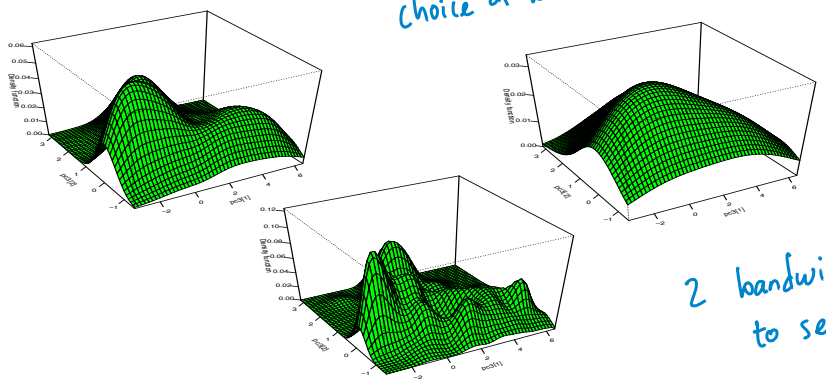
- Data on 6 characteristics of aircraft (Bowman and Azzalini 1998)
- Examine first 2 principle components of the data
- Perform KDE with independent kernels



# 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

choice of bandwidth matters a lot



2 bandwidths to select




**Module 4: Coping with Multiple Predictors**

# Regression Trees

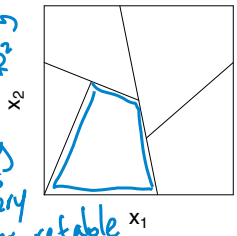
STAT/BIOSTAT 527, University of Washington

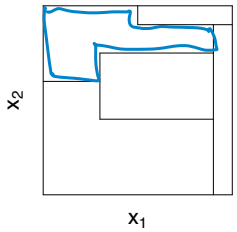
Emily Fox

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?



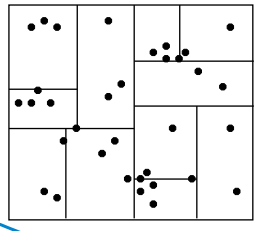



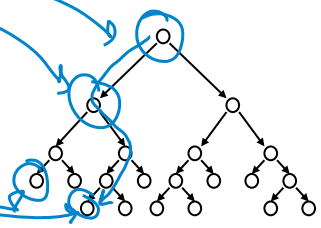




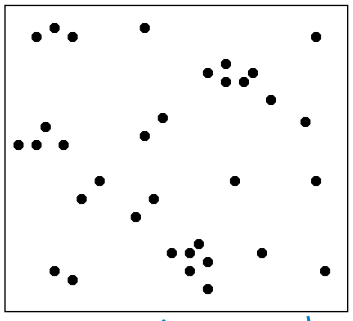

# Recursive Binary Partitions

- 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



2d example

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

$(x_{11}, x_{12})$

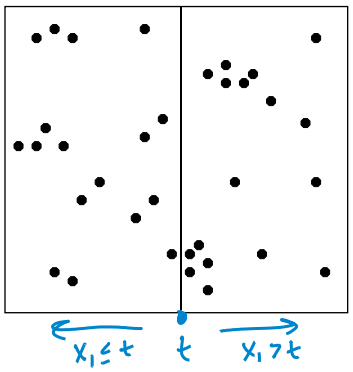
$(x_1, y_1)$

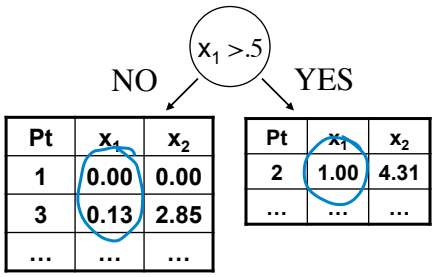
$(x_2, y_2)$

- 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 |
| … | … | … |

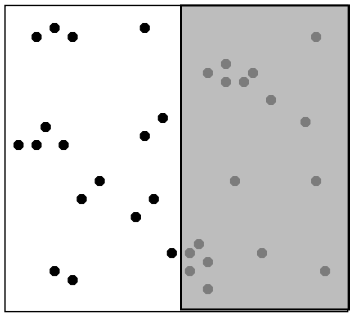
| Pt | $x_1$ | $x_2$ |
|---|---|---|
| 2 | 1.00 | 4.31 |
| … | … | … |

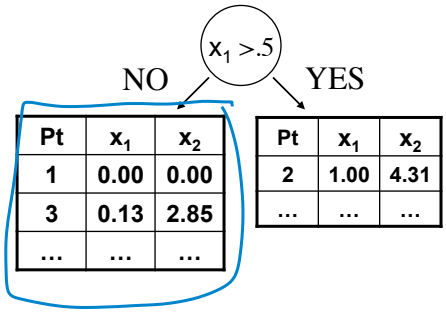
Here: $d_j = 1$ $(x_1)$

$t_j = 0.5$

- 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





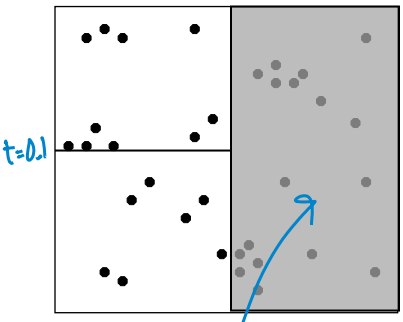

| Pt | $x_1$ | $x_2$ |
|---|---|---|
| 1 | 0.00 | 0.00 |
| 3 | 0.13 | 2.85 |
| … | … | … |

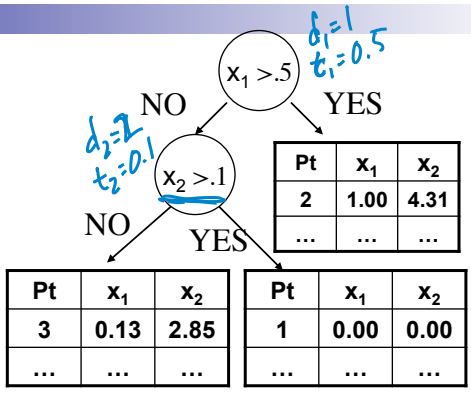
| Pt | $x_1$ | $x_2$ |
|---|---|---|
| 2 | 1.00 | 4.31 |
| … | … | … |

- 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 |
| … | … | … |

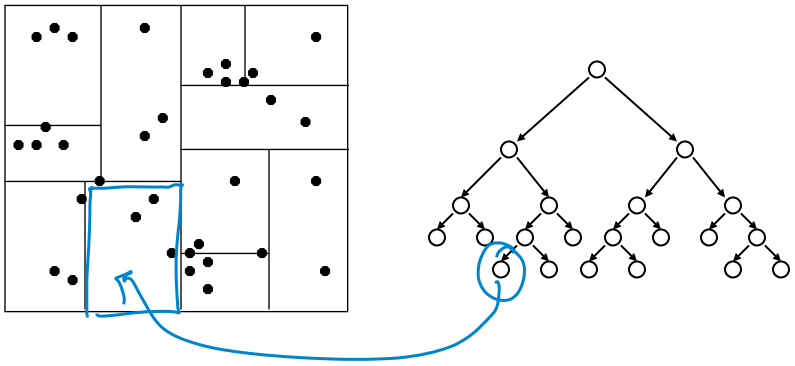
| Pt | $x_1$ | $x_2$ |
|---|---|---|
| 3 | 0.13 | 2.85 |
| … | … | … |

| 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…

recurse in this region as well

# Recursive Binary Partitions

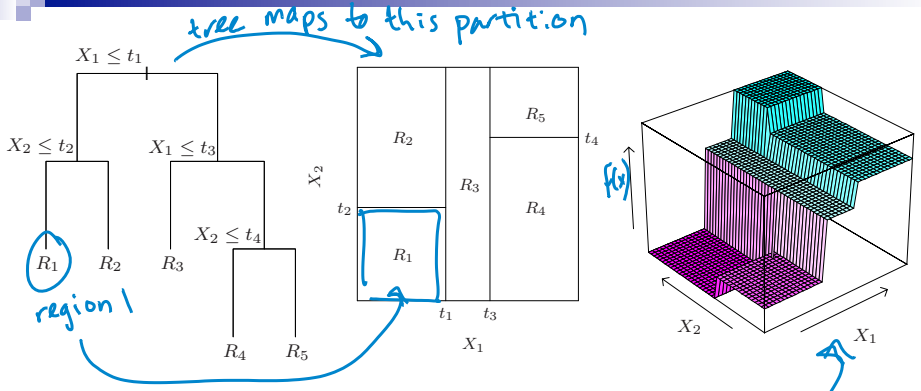


- Continue splitting points in each set
  - creates a binary tree structure
- Each leaf node contains a list of points

satisfying all conditions down the tree to that point



# Resulting Model




- Model the response as constant within each region

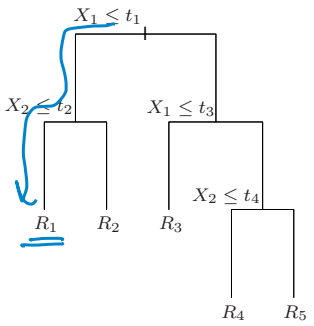
$$f(x) = \sum_{m=1}^{M} \beta_m I(x \in R_m)$$

# Basis Expansion Interpretation

- Equivalent to a basis expansion

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

indicators on regions




- 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




# Questions on Building the Tree

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

could run until 1 obs. at each leaf, but overfit

# 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

$$\hat{\beta}_m = \text{avg}(y_i \mid x_i \in R_m)$$

- 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



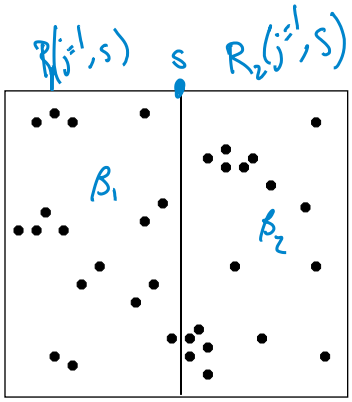
# Choosing a Split Decision

- 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

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




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

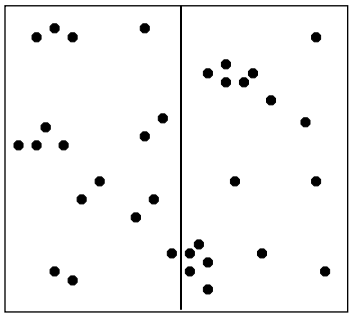
$$\hat{\beta}_k = \text{avg}(y_i \mid x_i \in R_k(j,s)) \quad k = 1,2$$

# 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?

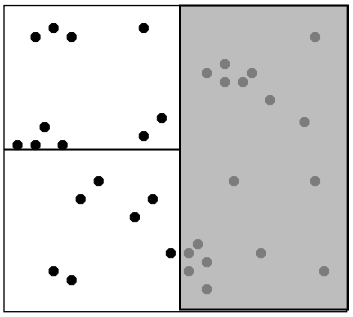
- start at one end
- obj. only changes when s passes an obs.
- update to $\hat{\beta}_1, \hat{\beta}_2$ is O(1) ... 1 obs. diff.

- 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 → overfit (var)
- Small tree → miss key features (bias)
- 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
    Shortsighted. Splits later could be very good.
  - Stop when a minimum node size (e.g., 5) is reached. Go back and prune.

easy how?




# 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 $T \subset T_0$ to be any tree obtained by pruning $T_0$
  prune = collapse an internal node

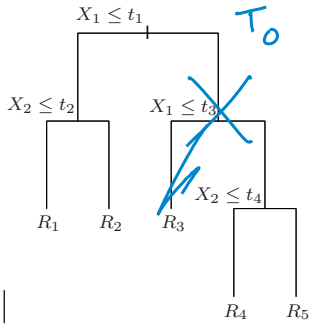
  and $|T| =$ # of leaf nodes

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

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

  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

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

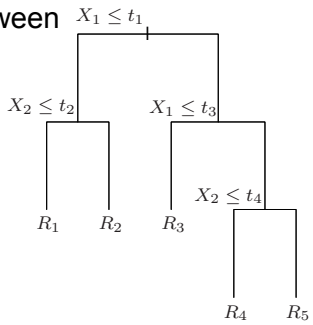
# Cost-Complexity Pruning

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

penalty

RSS over regions

- 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 → small trees
  - λ = 0 → $T_0$ full tree
- For each λ, can show that there is a unique smallest subtree $T_\lambda$






# Cost-Complexity Pruning

Sequence:

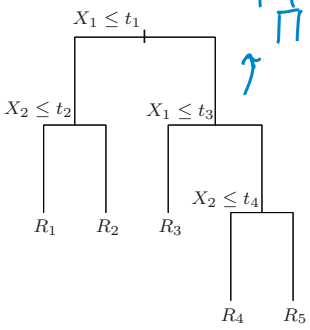
compute for $\lambda$ and all trees in sequence

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

- 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 $T_\lambda$
  - See Breiman et al. (1984) or Ripley (1996)
- Choose λ via 5- or 10-fold CV → $\hat{\lambda}$
- Final tree: $T_{\hat{\lambda}}$



# 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

recall reduced tensor product...

- In the construction, we are assuming that
  - Error terms are uncorrelated
  - Constant variance

→ RSS is the right metric



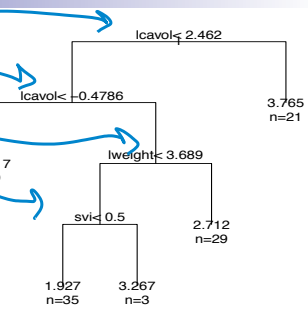

# Example: Prostate Cancer

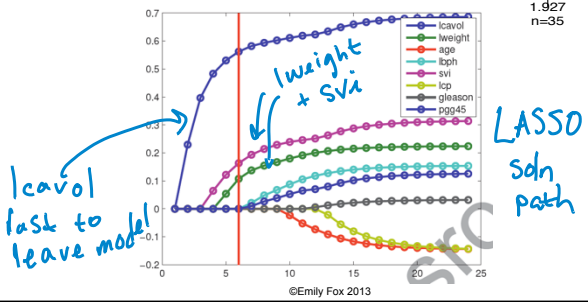
- Fit binary regression tree to log PSA with splits based on eight covariates
- Grow tree with condition of at least 3 observation per leaf
- Results in a tree with 27 splits
- Run weakest-link pruning for each candidate $\lambda$, with $\lambda$ chosen according to CV

# 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

# Readings

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