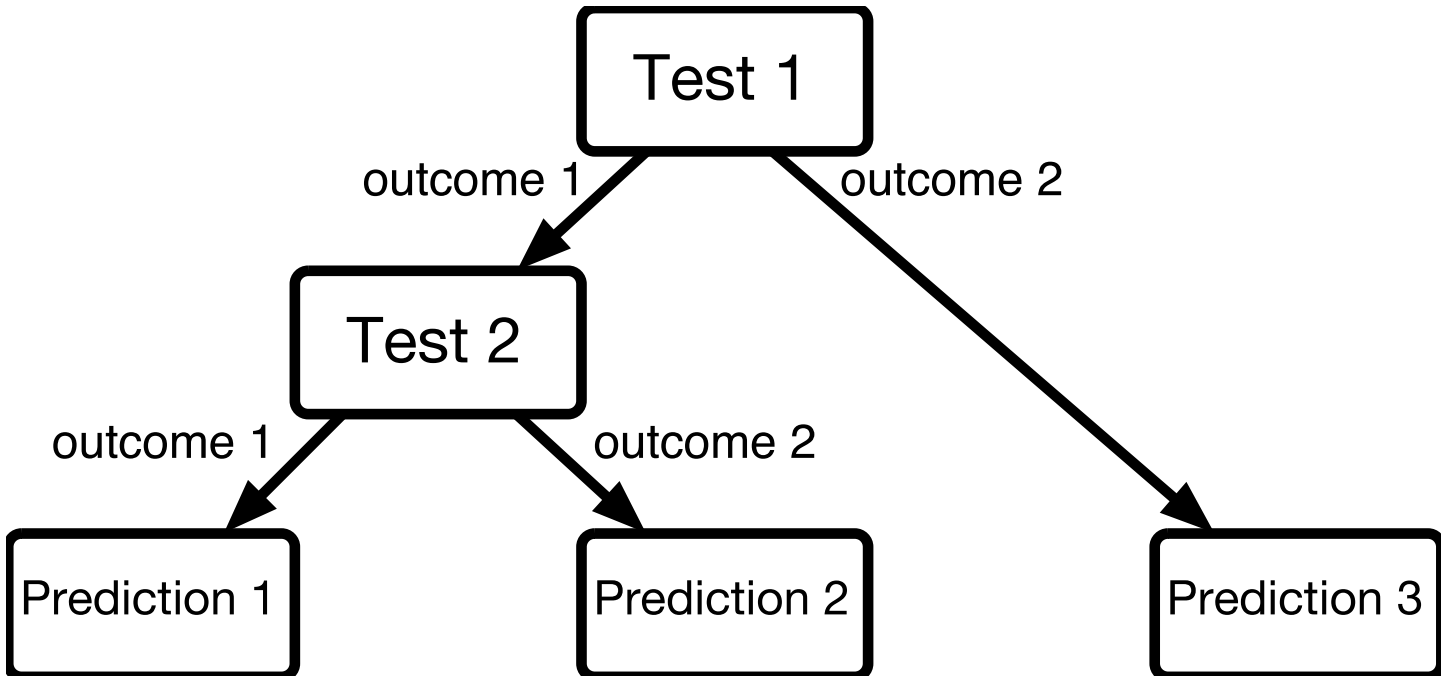
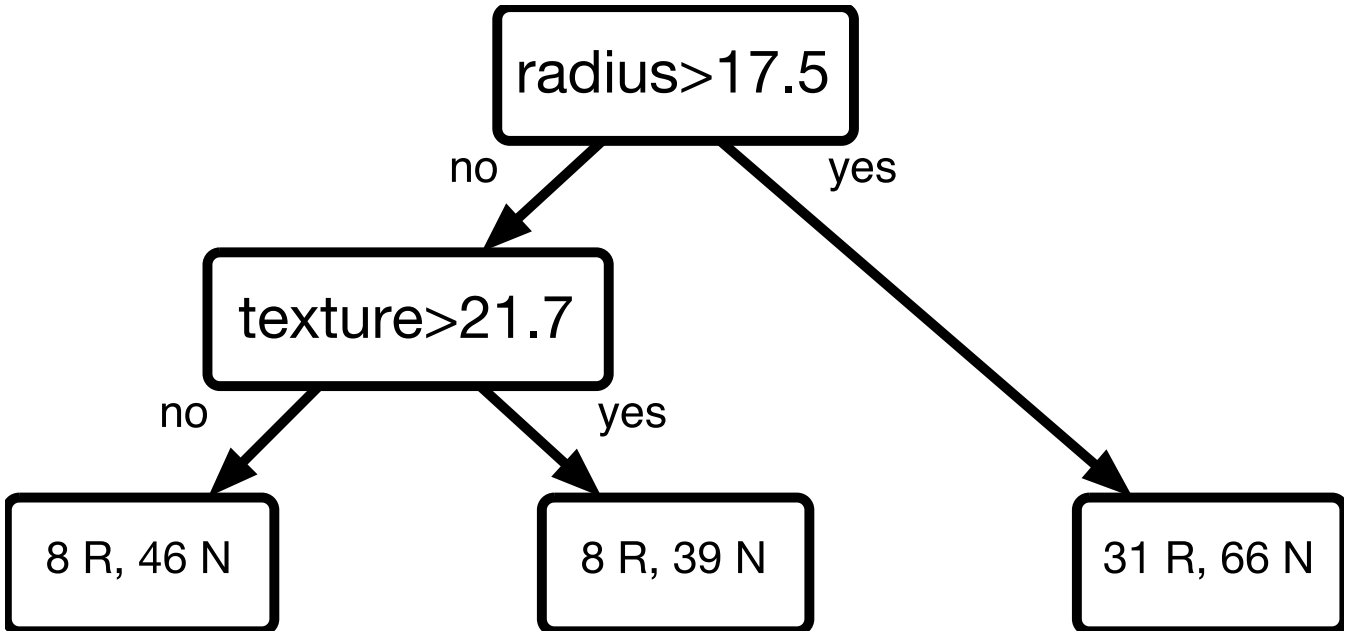
Decision and Regression Trees

What is a decision/regression tree?



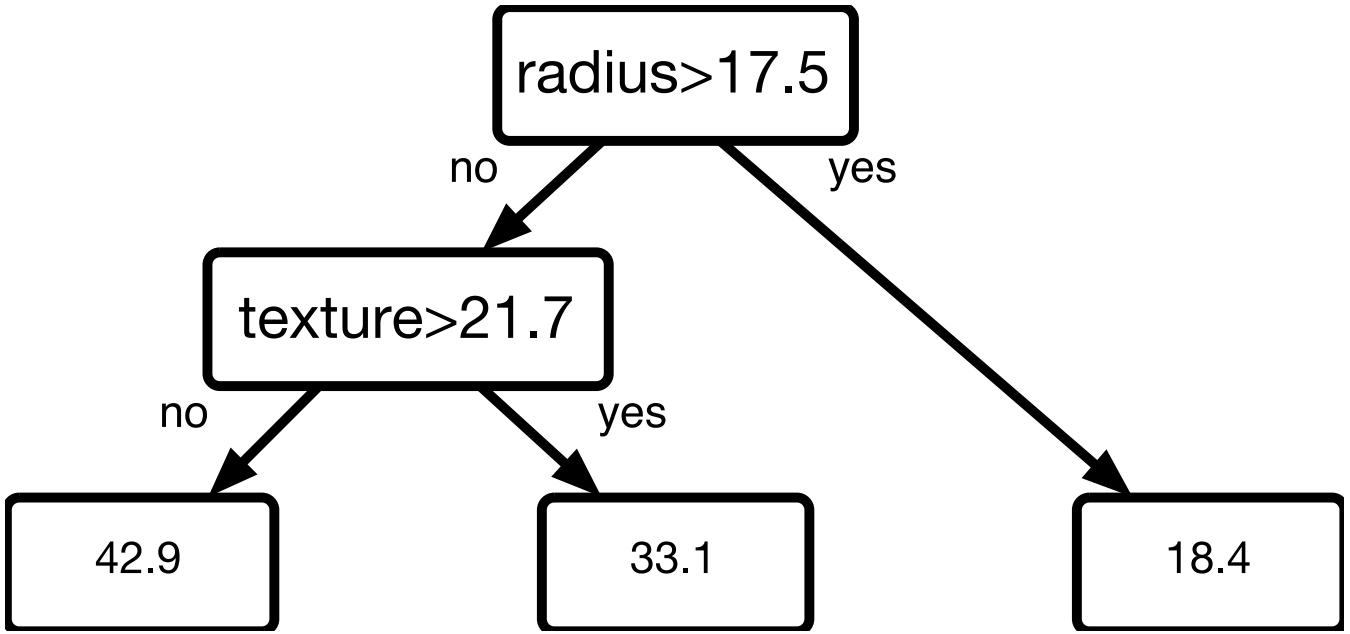
- Internal nodes perform discrete-outcome “tests” based on input features.
- Leaf nodes are predictions.

Example 1: A *decision tree* for predicting cancer recurrence



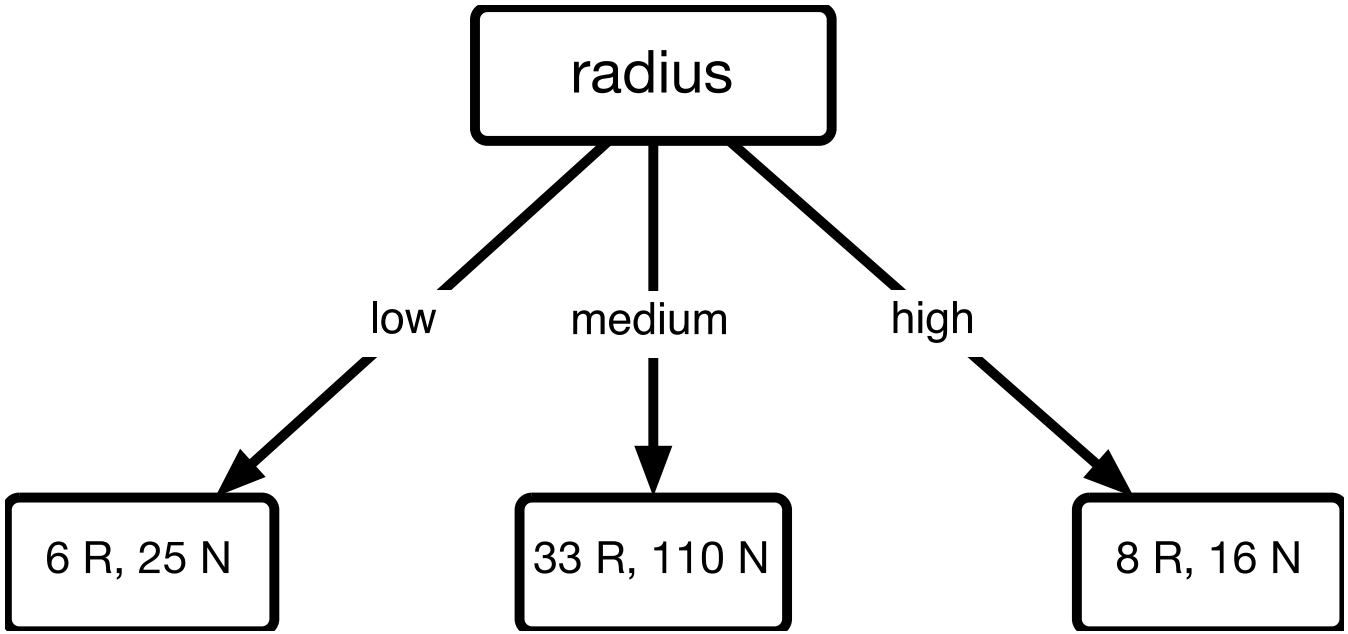
- Each training example $(\mathbf{x}_i, \mathbf{y}_i)$ falls in precisely one leaf.
- For a new input, $\mathbf{x}$, one can predict majority (N/R) or probabilities of N,R for corresponding leaf.

Example 2: A *regression tree* for predicting time-to-recurrence



- The prediction for a leaf can be the mean (shown here), mean and variance, a linear regression fit. . .

Example 3: Predicting time-to-recurrence with discrete features.



What are tests?

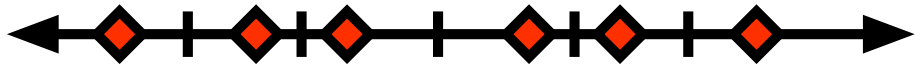
- For a discrete attribute j, one can branch on
  - all possible values, corresponding to a test $\mathbf{x}(j) =$?
  - inclusion in a subset, $\mathbf{x}(j) \in A$?
- For a real-valued attribute j or attributes, one can branch on
  - comparison to a threshold, $\mathbf{x}(j) > c$?
  - output of a perceptron, $\mathbf{x} \cdot \mathbf{w} + w_0 > c$?

Finding a good tree

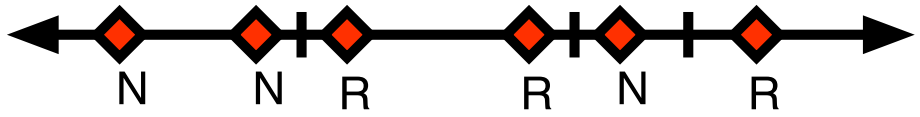
- One *could* exhaustively enumerate all trees and tests (if finite in number), and use a validation set to estimate which is best, but. . .
  - There are *many* possible trees.
  - We'd probably overfit the data anyway.
- Usually, decision/regression trees are constructed in two phases:
  - An incremental top-down procedure "grows" a tree, until the training data is completely fit.
  - The tree is "pruned" back to avoid overfitting.

Picking a test for the root (1)

- Suppose the data is $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}$, and there are a finite number of possible tests τ_k we can pick for the root of the tree.
- For discrete-valued features, there can only be a finite number of tests.
- For real-valued features, the values $\mathbf{x}_i(j)$ determine possible choices for c in comparison tests $\mathbf{x}(j) > c$.
  - We can restrict attention to mid-points between adjacent $\mathbf{x}_i(j)$.

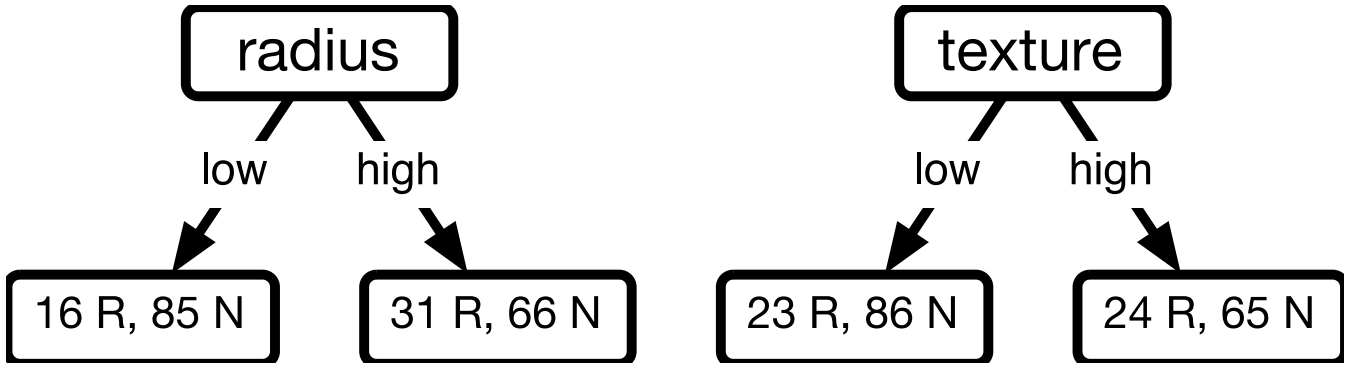


  - If $\mathbf{y}$ is discrete-valued, we can restrict attention to mid-points between adjacent pairs at which the output changes.



Picking a test for the root (2)

- Intuition: if the $\mathbf{y}_i$ are all the same, or nearly all the same, then prediction is easy.
- Heuristic: the best test moves us towards constant $\mathbf{y}_i$, conditional on the test.



- How to measure (in)homogeneity?

Measuring (in)homogeneity

- Suppose a test τ splits the data $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}$ into k groups, $D_1, \ldots, D_k$.
- If $\mathbf{y}_i$ is real, one measure of the goodness of that split is the variance of $\mathbf{y}_i$ in the resulting groups:

$$\sum_{j=1}^{k} \text{var}(\mathbf{y}_i \in D_j), \text{ or}$$

$$\sum_{j=1}^{k} \frac{|D_j|}{|D|} \text{var}(\mathbf{y}_i \in D_j),$$

either of which should be minimized.

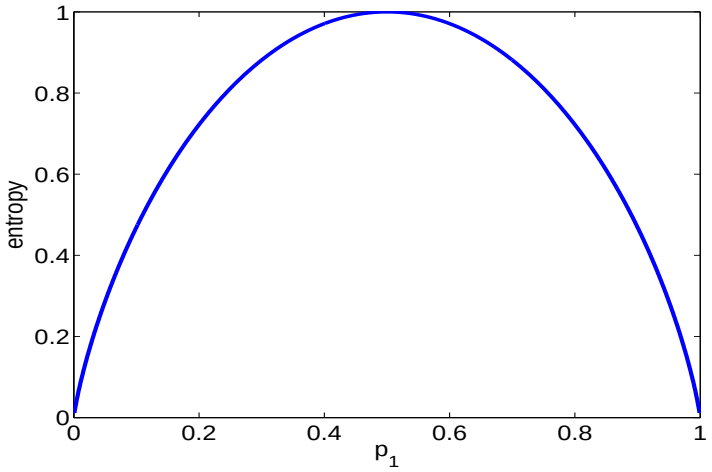
Measuring (in)homoeneity (2)

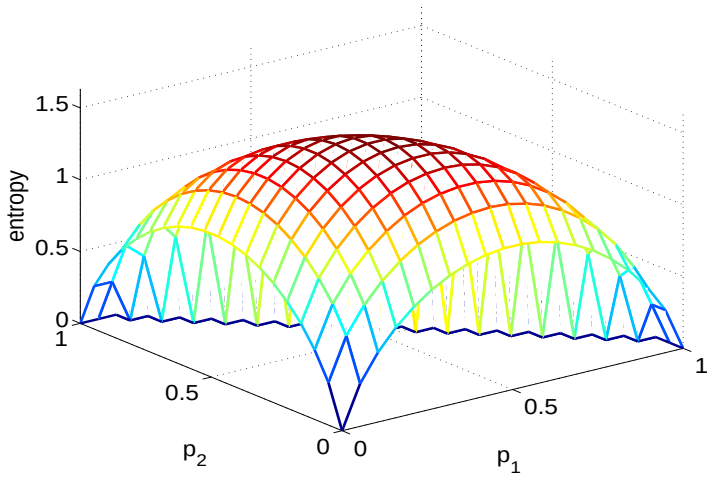
- If $\mathbf{y}_i$ is discrete-valued, then empirical entropy can be used to measure the goodness of a set of splits $D_1, \ldots, D_k$ created by a test.

to be continued...

Entropy

- Entropy captures the uncertainty in the outcome of a random variable.
- With k possible outcomes, the entropy is $-\sum_{l=1}^{k} p_l \log_2 p_l$.
- Entropy is maximal when all $p_l = \frac{1}{k}$, and equals $\log_2 k$.
- Entropy is minimal when any $p_l = 1$, and equals zero.





(Empirical) entropy

- Let $\{\mathbf{y}_i\}_{i=1}^m$ be a sequence of discrete values.
- Let p_l be the empirical frequency of value l,

$$p_l = \frac{|\{\mathbf{y}_i = l\}|}{m}$$

- Then the (empirical) entropy of the $\mathbf{y}_i$ is $-\sum_l p_l \log_2 p_l$.

example sequence	entropy
0 0 0 0 1 1 1 1 1	0.991
1 0 1 0 1 0 1 0 1	0.991
0 0 0 1 1 1 1 1 1	0.918
0 1 1 1 1 1 1 1 1	0.503
0 0 1 1 1 2 2 2 2	1.531

Measuring (in)homoeneity (2) continued

- If $\mathbf{y}_i$ is discrete-valued, then empirical entropy can be used to measure the goodness of a set of splits $D_1, \ldots, D_k$ created by a test.

- One measure is the expected empirical entropy

$$\sum_j \frac{|D_j|}{|D|} \text{entropy}(\mathbf{y}_i \in D_j)$$

  which is low for a good split.

- For historical reasons, *information gain* is often used

$$\text{entropy}(\mathbf{y}_i \in D) - \sum_j \frac{|D_j|}{|D|} \text{entropy}(\mathbf{y}_i \in D_j)$$

  which is large for a good split.

Incremental tree construction and pruning

- We assume a finite number of possible tests τ_j, data $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}$, and purity measure P.
- Construct a tree which exactly fits the data :
  - If $\mathbf{y}_i = \mathbf{y}$ for some $\mathbf{y}$ and all i, stop.
  - Otherwise, evaluate each possible test τ_j according to P.
  - Split the data into subsets $D_1, \ldots, D_K$ according to τ_j.
  - Recursively build a tree on each subset.
- Prune the tree by repeating:
  - Estimate the generalization performance of the tree using a validation set.
  - Replace any subtree with a single node, as long as doing so improves estaimted generalization performance.

Some pros and cons of trees (1)

- Modeling power:
  - The overall decision boundary / prediction is nonlinear in the inputs.
  - Typically, tests are on single variables, and constitute axis-parallel cuts.
- Comprehensibility:
  - Small trees are easy to understand.
  - Trees can be converted into lists of conjunctive rules.
  - Variables occurring in test *may* be the more relevant ones.
  - However, the structure of the tree and variables appearing in it can be quite sensitive to the data.

Some pros and cons of trees (2)

- Efficiency:
  - Finding a tree by incremental growth and pruning can be done very rapidly.
  - Though trees are not necessarily optimal.