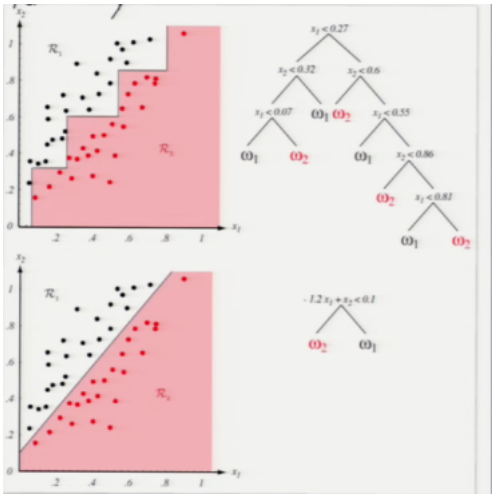
03/28/2016

Decision Trees (continued)

Multivariate Splits

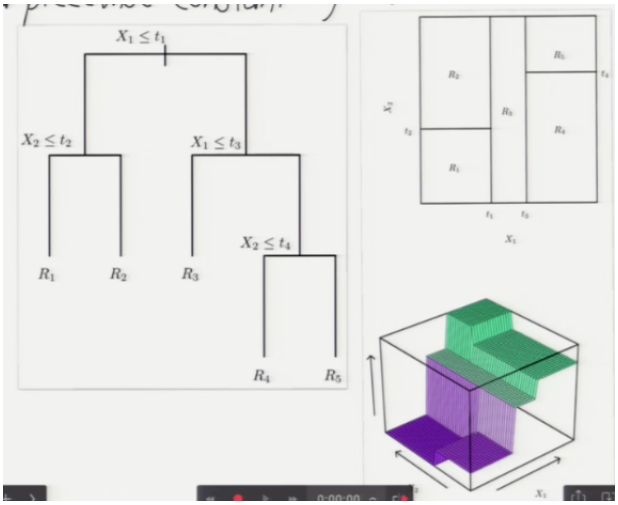
- Split on multiple features at a time.
- Find non-axis-aligned splits with other classification algorithms or by generating them randomly.



- Need to look at more than one feature.
- May gain better classifier at the cost of worse interpretability.
- Can limit number of features per split:
  - Forward stepwise selection.
  - Lasso.

Decision Tree Regression

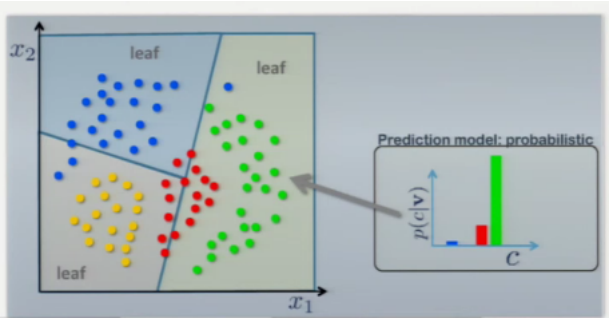
- Creates a piecewise constant regression function.



- S is the set of sample indices that had trickled down to this node of the tree. The more different the y values are the greater the cost should be.
- Cost $J(S) = \sum_{i \in S}(y_i - \bar{y})^2$, where $\bar{y}$ is the mean y_i for sample subset S.
- If all samples in this node agree on the y then the cost is zero.
- When we try to split a node, we look at all the different ways we could split a node and we choose whichever minimized the weighted average of the cost functions of the two children.
- If you have several points in a leaf average the y values and assign that value.

Stopping early

- Why?
  - Limit tree depth (for speed).
  - Limit tree size (big data sets).
  - Complete tree may overfit.
  - Given noise or overlapping distribution, purity of leaves is counterproductive; better to estimate posterior probabilities.



- How? Select stopping condition(s):
  - Next split doesn't reduce entropy/error enough (dangerous; pruning better).
  - Most of node's points (e.g. $> 95\%$) have same class.
  - Node contains few sample points (e.g. < 10).
  - Node covers tiny volume.
  - Depth too great.
  - Use (cross)-validation to compare.
- Leaves with multiple points return:
  - a majority vote or class posterior probabilities (classification)
  - an average (regression)

Pruning

- Grow tree too large; greedily remove each split whose removal improves cross-validation performance. More reliable than stopping early.

Ensemble Learning

- Decision trees are fast, simple, interpretable, invariant under scale/translation, robust to irrelevant features.
- But not the best at prediction. High variance.
- We can take average of output of:
  - Different learning algorithms
  - Same learning algorithm on many training sets.
  - Bagging: same learning algorithm on many random subsamples of one training set.
  - Random forests: Randomized decision trees on random subsamples.
- Regression algorithms: take median or mean output.
- Classification algorithms: take majority vote or average posterior probabilities.
- Use learners with low bias (e.g. deep decision trees).
- High variance and some overfitting is okay. Averaging reduces the variance!
- Averaging sometimes reduces the bias and increases flexibility; e.g averaging linear classifiers $\rightarrow$ nonlinear decision boundaries.
- Hyper-parameters settings usually different than 1 learner.
- Number of trees is another hyper-parameter.

Bagging = Bootstrap Aggregating (Leo Breinman, 1994)

- Given n-point training sample, generate random subsample of size n', by sampling with replacement. Same points chosen multiple times; some not chosen.
- If $n' = n, \sim 63.2\%$ are chosen.
- Build learner. Points chosen j times have greater weight:
  - Decision trees: j-time point has j x weight in entropy.
  - SVMs: j-time point incurs j x penalty to violate margin.
  - Regression: j-time point incurs j x loss.
- Repeat until T learners. Metalearner takes test point, feeds it into all T learners, returns average/majority output.

Random Forests

- Random sampling isn't random enough!
- Idea: at each split, take random sample of m features (out of d). Choose best split from m features. Different random samples for each split. $m \sim \sqrt{d}$ good for classification; $m \sim \frac{d}{3}$ for regression.
- Smaller $m \rightarrow$ more randomness, less tree correlation, more bias.
- Sometimes test error reduction up to 100's or even 1000's of trees. Advantage much more accurate tree but less interpretability.
- Variation: Generate m random multivariate splits (oblique lines, quadrics); choose best split.