Decision Trees & Nearest Neighbor Classifiers

CS 510

Lecture #20

April 26th, 2013

Colorado State University

Programming Assignment #4

- Due two weeks from today
  - Any questions?
  - How is it going?

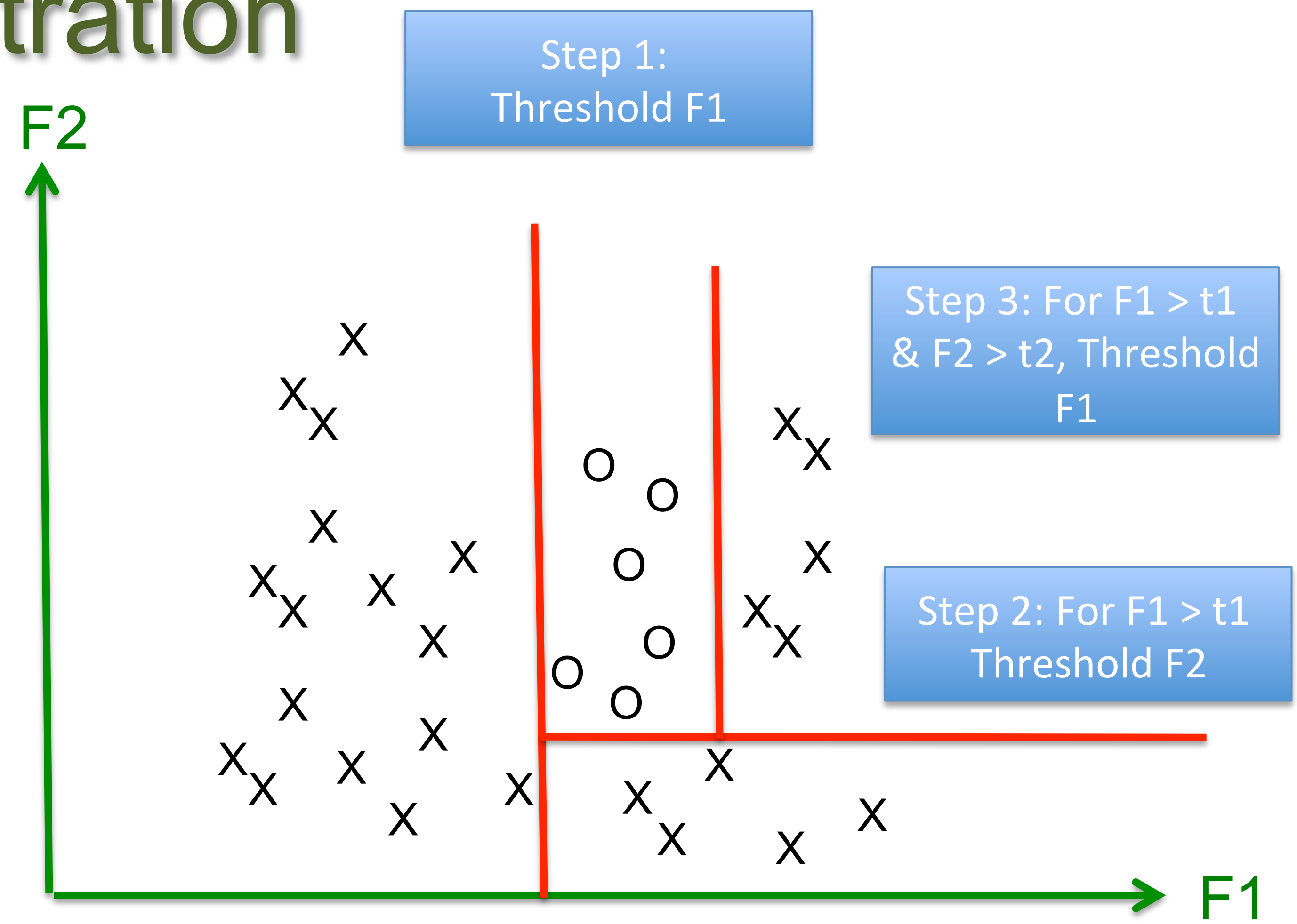
Where are we?

- Learning about classifiers from a user's perspective
- SVMs & Backpropagation Networks
  – Highly effective
  – Hard to analyze
- Bayesian Networks
  – Combine statistical and semantic knowledge
  – Harder to use / slower to use
- Today: Decision trees & Lookup tables

Decision Trees : basic idea

- Work in feature space (no kernels)
- Simple linear separators
- Approximate complex functions by recursive division

Illustration



Decision Trees

- Are all (or most) of your samples from a single class? If so, done.
- Pick a feature and threshold to divide your data into two sets
- Recurse on both sets

Question: how do you pick which feature to threshold, and what the threshold should be?

Entropy

- A quick detour into Shannon's entropy…
  - Let X be a r.v. with values $\{x_1,\ldots x_m\}$
  - Then the entropy H of X is defined as :

$$H(X) = E(I(X)) = E\left[-\ln(P(X))\right]$$

  - More relevantly, for a finite sample :

$$H(X) = \sum_i P(x_i) I(x_i) = -\sum_i P(x_i) \log_b P(x_i)$$

Colorado State University

Entropy Intuitions

- If $P(x_i) = 1$:
  - for all j, $P(x_j) = 0$
  - so $H(X) = 0$
- Or, if $P(x_i) = P(x_j)$ for all i & j :
  - $P(x_i) = 1/m$
  - so $H(X) = -m\ \Sigma(1/m)\log_m(1/m) = 1$
- In general, entropy is maximized by uniform distributions and minimized by impulses
- Entropy can be intuited as the amount of information gained by sampling the random value

Information Gain

- The information gain as a result of dividing a set T into a & b is :

$$H(T) - H(T \mid a, b)$$

- Or, for finite samples,

$$\sum P(x_1)\log\big(P(x_i)\big) - \sum_a P(x_i \mid a)P(x_i \mid a) - \sum_b P(x_i \mid b)P(x_i \mid b)$$

Decision Trees & Information Gain

- Intuitively, information gain measures how much more you know about the samples as a result of dividing them
- Choose the feature and threshold that maximizes the information gain
- How? Try them all…

Overfitting

- What if a positive training sample is surrounded by negative ones?
  - Following the previous algorithm will result in a small positive zone around the positive training sample
  - But what if the example is an error? (or fluke?)
- Overfitting : performance can be 100% on the training samples, but poor on novel test data

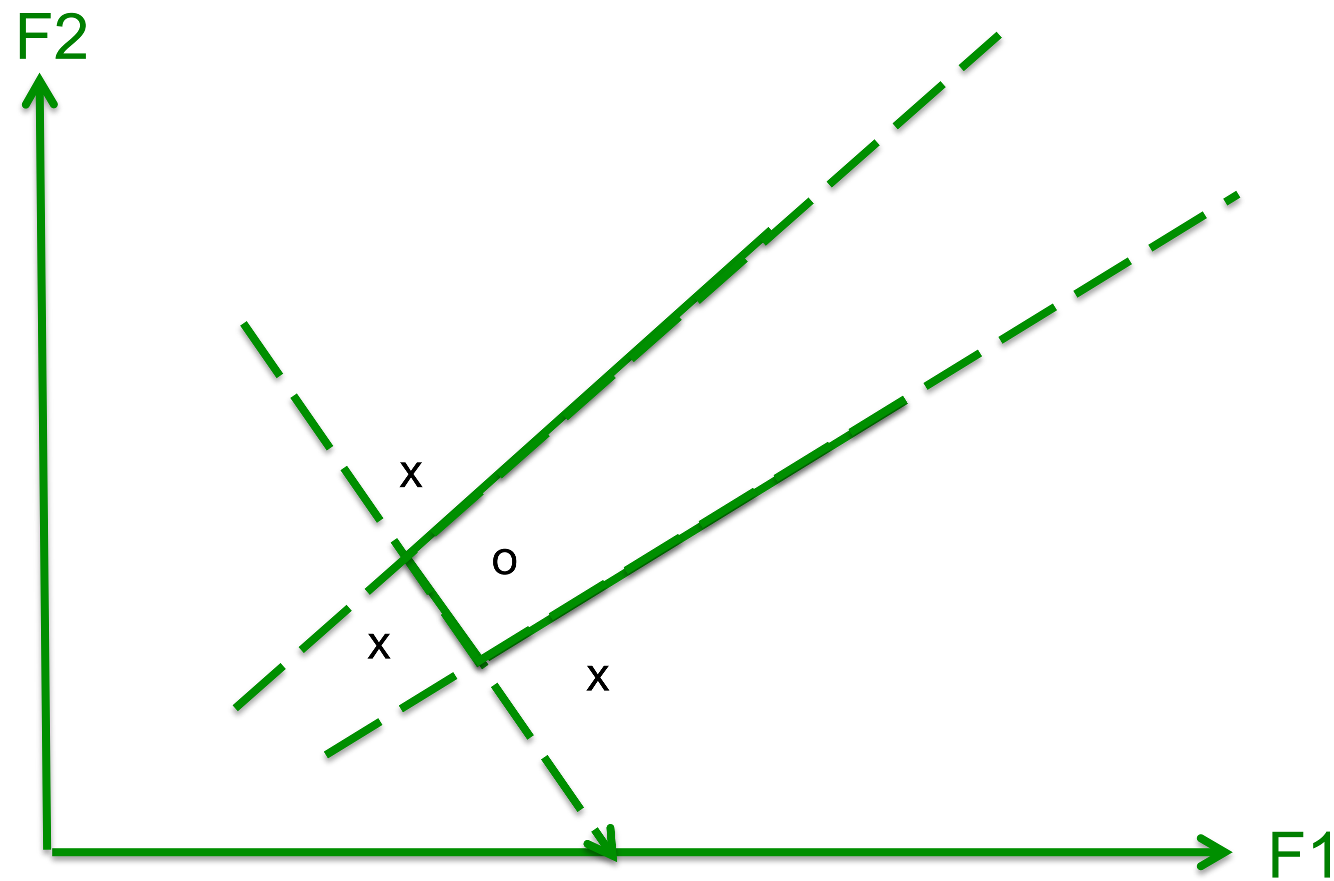
Training/Validation/Test

- To test generalization, the test data must be distinct from the training data
- But without using the test data, how do you know when to stop splitting the data?
- Answer:
  – Split the training data into training + validation
  – On every recursion
    - Find the split that improves the training data the most
    - Stop if the split does not improve validation performance

The Simplest Classifier : Lookup Tables

- SVMs, Nets, and Decision Trees carve up feature space based on samples
- Why not just memorize the training samples?
- Given a test sample, measure the distance to every training sample, and take the closest
- This is called a *nearest neighbor* classifier

Geometric Interpretation of Nearest Neighbor Classifiers



K-nearest neighbors

- Problem: Nearest neighbor classifiers overfit
- Solution: find the K nearest neighbors, let them vote for the best label
  - Set K to be odd (to avoid ties)
- Problem: Computing the distance to every training sample is expensive
- Solution: Approximate Nearest Neighbor trees find the (approx) nearest neighbor in log(n) comparisons