CSCE 496/896 Lecture 9: Instance-Based Learning

Stephen D. Scott
(Adapted from Tom Mitchell's slides)

November 8, 1999

Outline

- k-Nearest Neighbor
- Locally weighted regression
- Radial basis functions
- Case-based reasoning
- Lazy and eager learning

Nearest Neighbor

Key idea: just store all training examples $\langle x_i, f(x_i)\rangle$

Need some distance measure between instances (e.g. Euclidean distance, Hamming distance)

Nearest neighbor:

- Given query instance x_q, first locate nearest training example x_n, then estimate $\hat{f}(x_q) = f(x_n)$

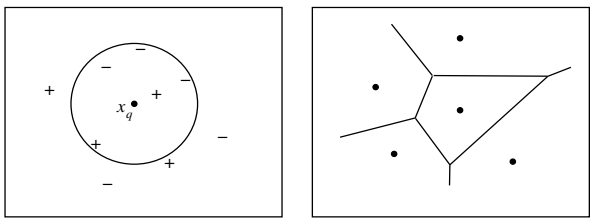
k-Nearest neighbor:

- Given x_q, take vote among its k nearest nbrs (if discrete-valued target function)
  - Let k not be divisible by number of possible labels
- Take mean of f values of k nearest nbrs if real-valued

$$\hat{f}(x_q) = \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

Voronoi Diagram

Decision surface for 1-NN



When To Consider Nearest Neighbor

- Instances map to points in $\Re^n$
- Less than 20 attributes per instance
  - To avoid curse of dimensionality, where many irrelevant attributes causes distance to be large, but distance is small if only relevant attributes used
  - Also, large number of attributes increases classification complexity
- Lots of training data

Advantages:

- Training is very fast
- Learn complex target functions
- Don't lose information

Disadvantages:

- Slow at query time (active research area: fast indexing and accessing algorithms)
- Easily fooled by irrelevant attributes

Nearest Neighbor's Behavior in the Limit

Consider $p(x)$ defines probability that instance x will be labeled 1 (positive) versus 0 (negative).

Nearest neighbor:

- As number of training examples $\rightarrow \infty$, approaches Gibbs Algorithm

  Recall Gibbs has at most twice the expected error of Bayes optimal

k-Nearest neighbor:

- As number of training examples $\rightarrow \infty$ and k gets large, approaches Bayes optimal (best possible with given hyp. space and prior information)

  Bayes optimal: if $p(x) > .5$ then predict 1, else 0

Distance-Weighted k-NN

Might want weight nearer neighbors more heavily:

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

where

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

and $d(x_q, x_i)$ is distance between x_q and x_i

Note now it makes sense to use *all* training examples instead of just k (Shepard's method)

Increased time to classify instances

Curse of Dimensionality

Imagine instances described by 20 attributes, but only 2 are relevant to target function

Curse of dimensionality: nearest nbr is easily misled by high-dimensional X

One approach:

- Stretch jth axis by weight z_j, where $z_1, \ldots, z_n$ chosen to minimize prediction error
- Use cross-validation to automatically choose weights $z_1, \ldots, z_n$
- Note setting z_j to zero eliminates this dimension altogether

see [Moore and Lee, 1994]

Locally Weighted Regression

Note k-NN forms local approximation to f for each query point x_q

Why not form an explicit approximation $\hat{f}(x)$ for region surrounding x_q?

- Fit linear, quadratic, etc. function to k nearest neighbors
- Produces "piecewise approximation" to f
- Do this for each new query point x_q

Several choices of error to minimize:

- Squared error over k nearest neighbors

$$E_1(x_q) \equiv \frac{1}{2} \sum_{x \in \ k \ nearest \ nbrs \ of \ x_q} \left(f(x) - \hat{f}(x)\right)^2$$

- Distance-weighted squared error over all nbrs

$$E_2(x_q) \equiv \frac{1}{2} \sum_{x \in D} \left(f(x) - \hat{f}(x)\right)^2 \ K(d(x_q, x))$$

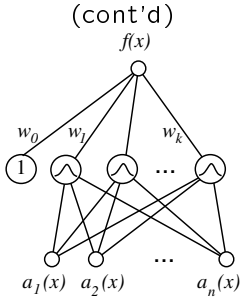
(K is decreasing in its argument)

- Combine E_1 and E_2

Radial Basis Function (RBF) Networks

- Global approximation to target function, in terms of linear combination of local approximations
- Used, e.g., for image classification
- A different kind of neural network
- Closely related to distance-weighted regression, but "eager" instead of "lazy"

RBF Networks (cont'd)



where $a_i(x)$ are the attributes describing instance x, and

$$f(x) = w_0 + \sum_{u=1}^{k} w_u K_u(d(x_u, x))$$

(Note no weights from input to hidden layer)

One common choice for $K_u(d(x_u, x))$ is

$$K_u(d(x_u, x)) = \exp\left(-\frac{1}{2\sigma_u^2} \ d^2(x_u, x)\right),$$

i.e. Gaussian with mean at x_u and variance σ_u^2, all features independent

[note bug on p. 239]

Training Radial Basis Function Networks

1. Choose number of kernel functions (hidden units)
   - If = number training exs, can fit training data exactly by placing one center per ex
   - Using fewer ⇒ more efficient, less chance of overfitting
2. Choose center (= mean for Gaussian) x_u of kernel function $K_u(d(x_u, x))$
   - Use all training instances if enough kernels avail.
   - Use subset of training instances
   - Scatter uniformly throughout instance space
   - Can cluster data and assign one per cluster (helps answer step 1 also)
   - Can use EM to find means of mixture of Gaussians
   - Can also use e.g. EM to find σ_u's (for Gaussian)
3. Hold kernels fixed and train weights to fit linear function (output layer), e.g. GD or EG

Case-Based Reasoning and CADET

Can apply instance-based learning even when $X \neq \Re^n$

Need different "distance" metric

Case-Based Reasoning is instance-based learning where instances have symbolic logic descriptions

```
((user-complaint error53-on-shutdown)
 (cpu-model PowerPC) (operating-system Windows)
 (memory 48meg)
 (installed-apps Excel Netscape VirusScan)
 (disk 1gig)
 (likely-cause ???))
```

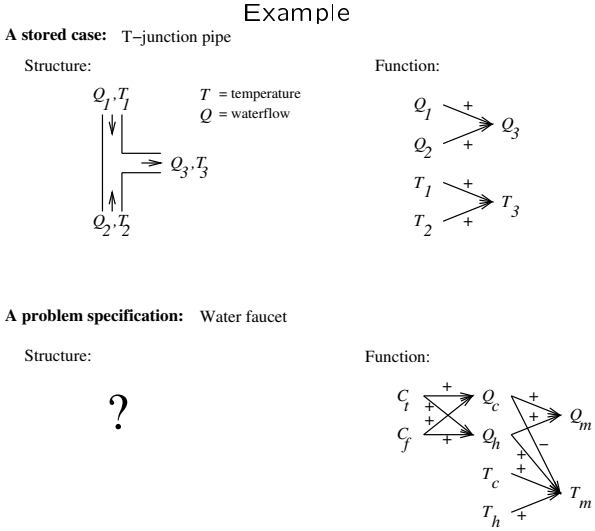
CADET: 75 stored examples of mechanical devices, e.g. water faucets

- Training ex: ⟨qualitative function, mech. structure⟩
- New query: desired function
- Target value: mechanical structure for this function

Distance metric: match qualitative function descriptions

Case-Based Reasoning in CADET

Example



E.g. distance measure = size of largest isomorphic subgraph

Case-Based Reasoning in CADET
(cont'd)

- Instances represented by rich structural (symbolic) descriptions, vs. points in $\Re^n$ for k-NN
- Multiple cases retrieved (and combined) to form solution to new problem: Similar to k-NN, except combination procedure can rely on knowledge-based reasoning (e.g. can two components be fit together?)
- Tight coupling between case retrieval, knowledge-based reasoning, and problem solving, e.g. application of rewrite rules in function graphs and backtracking in search space

Bottom line:

- Simple matching of cases useful for tasks such as answering help-desk queries
- Area of ongoing research, including improving indexing and search methods

Lazy and Eager Learning

Lazy: Wait for query before generalizing

- k-NN, locally weighted regression, Case based reasoning

Eager: Generalize before seeing query

- Radial basis function networks, ID3, Backpropagation, Naive Bayes

Does it matter?

- Computation time for training and generalization
- Eager learner must create global approximation, lazy learner can create many local approximations
- If they use same H, lazy can represent more complex fns (e.g. consider H = linear functions) since it considers the query instance x_q before generalizing, i.e. lazy produces a new hypothesis for each new x_q