Lecture 8: Instance-based Learning

Cognitive Systems II - Machine Learning

SS 2005

Part II: Special Aspects of Concept Learning

k-nearest neighbors, locally weighted linear regression
radial basis functions, lazy vs. eager learning

Motivation

- all learning methods presented so far construct a general explicit description of the target function when examples are provided
- **Instance-based learning:**
  - examples are simply stored
  - generalizing is postponed until a new instance must be classified
  - in order to assign a target function value, its relationship to the previously stored examples is examined
  - sometimes referred to as **lazy learning**

Motivation

- **advantages:**
  - instead of estimating for the whole instance space, local approximations to the target function are possible
  - especially if target function is complex but still decomposable
- **disadvantages:**
  - classification costs are high

    efficient techniques for indexing examples are important to reduce computational effort
  - typically all attributes are considered when attempting to retrieve similar training examples

    if the concept depends only on a few attributes, the truly most similar instances may be far away

k-nearest Neighbor Learning

- most basic instance-based method
- **assumption:**
  - instances correspond to a point in a n-dimensional space $\Re^n$
  - thus, nearest neighbors are defined in terms of the standard **Euclidean Distance**

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

  where an instance x is described by $< a_1(x), a_2(x), ..., a_n(x) >$

- target function may be either discrete-valued or real-valued

k-nearest Neighbor Learning

- **discrete-valued target function:**
  - $f : \Re^n \rightarrow V$ where V is the finite set $\{v_1, v_2, ..., v_s\}$
  - the target function value is the most common value among the k nearest training examples

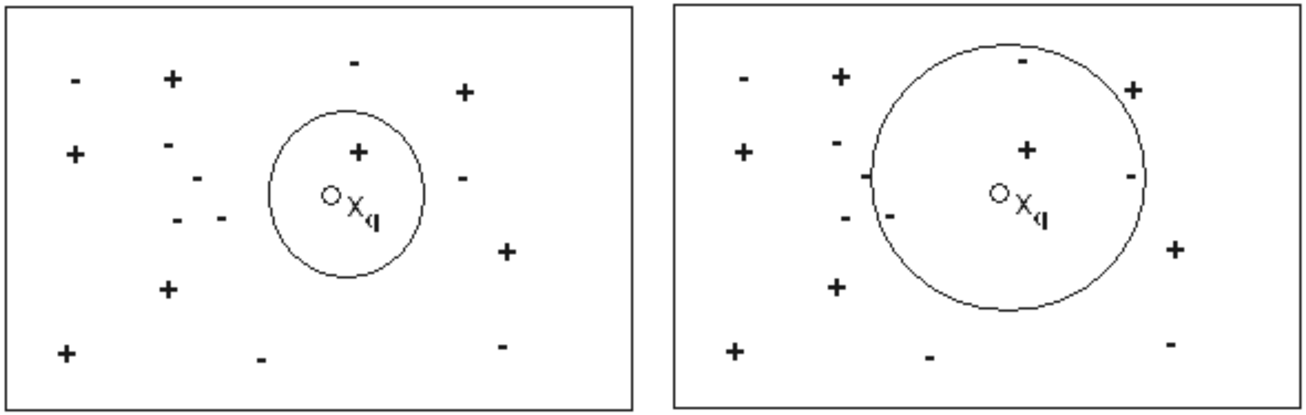
$$\hat{f}(x_q) \leftarrow \underset{v \in V}{argmax} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = (a == b)$

- **continuous-valued target function:**
  - algorithm has to calculate the mean value instead of the most common value
  - $f : \Re^n \rightarrow \Re$

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

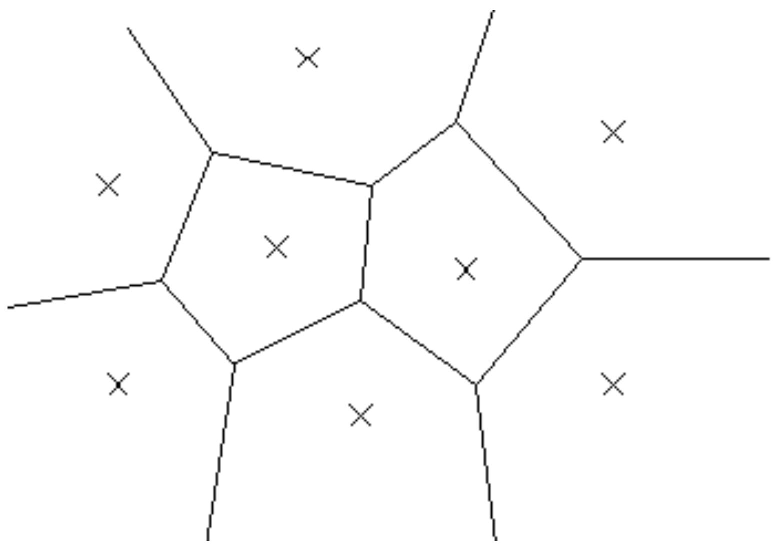
k-nearest Neighbor Learning



- e.g. instances are points in a two-dimensional space where the target function is boolean-valued
  - 1-nearest neighbor: x_q is classified positive
  - 4-nearest neighbor: x_q is classified negative

Hypothesis Space

- **no explicit** hypothesis is formed
- decision surface is a combination of convex polyhedra surrounding each of the training examples
- for each training example, the polyhedron indicates the set of possible query points x_q whose classification is completely determined by this training example (**Voronoi diagram**)



Distance-Weighted Nearest Neighbor

- contribution of each of the k nearest neighbors is weighted accorded to their distance to x_q
  - **discrete-valued target functions**

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{argmax} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

where $w_i \equiv \frac{1}{d(x_q,x_i)^2}$ and $\hat{f}(x_q) = f(x_i)$ if $x_q = x_i$

  - **continuous-valued target function:**

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

Remarks

- highly effective inductive inference method for many practical problems provided a sufficiently large set of training examples
- robust to noisy data
- weighted average smoothes out the impact of isolated noisy training examples
- **inductive bias of k-nearest neighbors**
  - assumption that the classification of x_q will be similar to the classification of other instances that are nearby in the Euclidean Distance
- **curse of dimensionality**
  - distance is based on all attributes
  - in contrast to decision trees and inductive logic programming
  - solutions to this problem
    - attributes can be weighted differently
    - eliminate least relevant attributes from instance space

Locally Weighted Regression

- a note on terminology:
  - *Regression* means approximating a real-valued target function
  - *Residual* is the error $\hat{f}(x) - f(x)$ in approximating the target function
  - *Kernel function* is the function of distance that is used to determine the weight of each training example. In other words, the kernel function is the function K such that $w_i = K(d(x_i, x_q))$
- nearest neighbor approaches can be thought of as approximating the target function at the single query point x_q
- locally weighted regression is a generalization to this approach, because it constructs an explicit approximation of f over a local region surrounding x_q

Locally Weighted Linear Regression

- target function is approximated using a **linear function**

$$\hat{f}(x) = w_0 + w_1 a_1(x) + ... + w_n a_n(x)$$

- methods like **gradient descent** can be used to calculate the coefficients $w_0, w_1, ..., w_n$ to minimize the error in fitting such linear functions

- ANNs require a global approximation to the target function
- here, just a local approximation is needed

$\Rightarrow$ the error function has to be redefined

Locally Weighted Linear Regression

- possibilities to redefine the error criterion E
  1. Minimize the squared error over just the k nearest neighbors

$$E_1(x_q) \equiv \frac{1}{2} \sum_{x \in k \text{ nearest neigbors}} (f(x) - \hat{f}(x))^2$$

  2. Minimize the squared error over the entire set D, while weighting the error of each training example by some decreasing function K of its distance from x_q

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

  3. Combine 1 and 2

$$E_3(x_q) \equiv \frac{1}{2} \sum_{x \in k \text{ nearest neighbors}} (f(x) - \hat{f}(x))^2 \cdot K(d(x_q, x))$$

Locally Weighted Linear Regression

- choice of the error criterion
  - E_2 is the most esthetically criterion, because it allows every training example to have impact on the classification of x_q
  - however, computational effort grows with the number of training examples
  - E_3 is a good approximation to E_2 with constant effort

$$\Delta w_j = \eta \sum_{x \in k \text{ nearest neighbors}} K(d(x_q, x))(f(x) - \hat{f}(x))a_j$$

- Remarks on locally weighted linear regression:
  - in most cases, constant, linear or quadratic functions are used
  - costs for fitting more complex functions are prohibitively high
  - simple approximations are good enough over a sufficiently small subregion of X

Radial Basis Functions

- closely related to distance-weighted regression and to ANNs
- learned hypotheses have the form

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

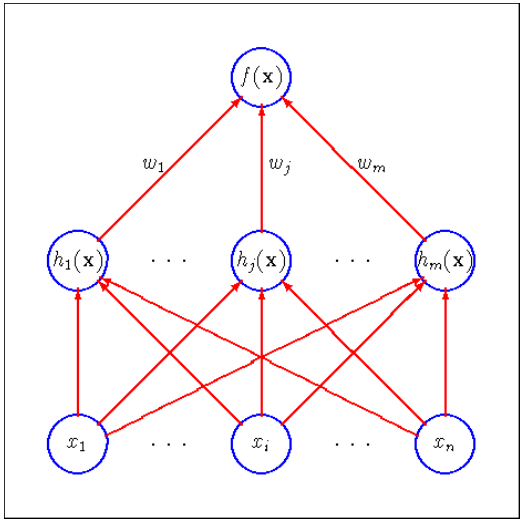
  where
  - each x_u is an instance from X and
  - $K_u(d(x_u, x))$ decreases as $d(x_u, x)$ increases and
  - k is a user-provided constant
- though $\hat{f}(x)$ is a global approximation to $f(x)$, the contribution of each of the K_u terms is localized to a region nearby the point x_u

Radial Basis Functions

- it is common to choose each function $K_u(d(x_u, x))$ to be a Gaussian function centered at x_u with some variance σ^2

$$K_u(d(x_u, x)) = e^{\frac{1}{2\sigma_u^2} d^2(x_u, x)}$$

- the function of $\hat{f}(x)$ can be viewed as describing a two-layer network where the first layer of units computes the various $K_u(d(x_u, x))$ values and the second layer a linear combination of the results



Case-based Reasoning

Remarks on Lazy and Eager Learning

- **lazy methods** defer the decision of how to generalize beyond the training data until a new query instance x_q is encountered
- **eager methods** generalize before any new query instance is encountered
- differences in computation time are obvious
- essential differences in the **inductive bias**
  - lazy methods are able to consider the query instance x_q when deciding how to generalize
  - eager methods already have commited to a global approximation of the target function before any x_q is encountered

  $\Rightarrow$ a lazy learner uses a richer H, because it uses many different local hypotheses to form a global approximation

Summary

- instance-based learning simply stores examples and postpones generalization until a new instance is encountered
- able to learn discrete- and continuous-valued conepts
- noise in the data is allowed (smoothed out by weighting distances)
- **Inductive Bias of k-nearest neighbors**: classification of an instance is similar to the classification of other instances nearby in the Euclidean Distance
- Locally Weighted Regression forms a local approximation of the target function