Lazy vs. Eager Learning

- **Lazy vs. eager learning**
  - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - Eager learning (e.g. Decision trees, SVM, NN): Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify

- Lazy: less time in training but more time in predicting

- **Accuracy**
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space
Lazy Learner: Instance-Based Methods

• Instance-based learning:
  – Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified

• Typical approaches
  – $k$-nearest neighbor approach
    • Instances represented as points in a Euclidean space.
  – Locally weighted regression
    • Constructs local approximation
The $k$-Nearest Neighbor Algorithm

- All instances correspond to points in the $n$-D space
- The nearest neighbor are defined in terms of Euclidean distance, $\text{dist}(X_1, X_2)$
- Target function could be discrete- or real- value
For discrete-valued, $k$-NN returns the most common value among the $k$ training examples nearest to $x_q$
Vonoroi diagram: the decision surface induced by 1-NN for a typical set of training examples
Discussion on the $k$-NN Algorithm

• $k$-NN for real-valued prediction for a given unknown tuple
  – Returns the mean values of the $k$ nearest neighbors

• Distance-weighted nearest neighbor algorithm
  – Weight the contribution of each of the $k$ neighbors according to their distance to the query $x_q$
    • Give greater weight to closer neighbors

• Robust to noisy data by averaging $k$-nearest neighbors

• Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
Case-Based Reasoning (CBR)

- CBR: Uses a database of problem solutions to solve new problems
- Store symbolic description (tuples or cases)—not points in a Euclidean space
- Applications: Customer-service (product-related diagnosis), legal ruling
- Methodology
  - Instances represented by rich symbolic descriptions (e.g., function graphs)
  - Search for similar cases, multiple retrieved cases may be combined
  - Tight coupling between case retrieval, knowledge-based reasoning, and problem solving
- Challenges
  - Find a good similarity metric
  - Indexing based on syntactic similarity measure, and when failure, backtracking, and adapting to additional cases