Introduction to Machine Learning

Ranga Raju Vatsavai, Ph.D.
Chancellors Faculty Excellence Associate Professor in Geospatial Analytics
Department of Computer Science, North Carolina State University (NCSU)

Feb. 25-27, 2019

Today

- Nearest Neighbor Classification

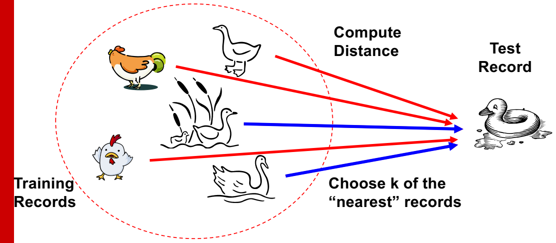
2/7/16

Types of Learners

- Eager learner
  - Designed to learn a model from instances (examples) that maps input attributes to the class label as soon as training data is available
    - Decision trees, rule-based, SVM, …
- Lazy learner
  - Opposite of eager learner. Delay the process of learning a model from training data until it is needed to classify a new sample
    - Rote-learner
      - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
    - Nearest neighbor
      - Uses k "closest" instances (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

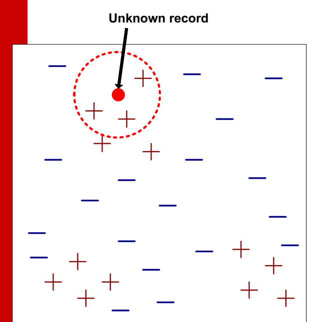
- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



Basic k-nearest neighbor algorithm

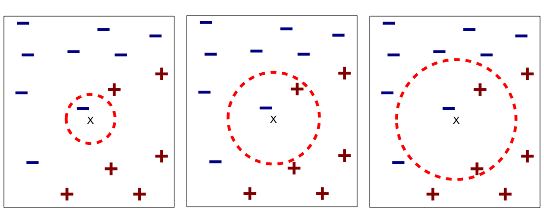
- Training method
  – Save the training examples
- Prediction
  – Find the k training examples (x_1,y_1), …, (x_k, y_k) that are closest to the test example (x, ?)
  – Assign the most frequent class among those y_i's

Nearest Neighbor Classifiers



- Requires three things
  – The set of labeled records
  – Distance Metric to compute distance between records
  – The value of *k*, the number of nearest neighbors to retrieve
- To classify an unknown record:
  – Compute distance to other training records
  – Identify *k* nearest neighbors
  – Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor

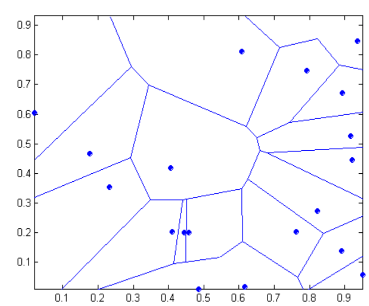


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distances to x

1 nearest neighbor

- Voronoi Diagram



Nearest Neighbor Classification

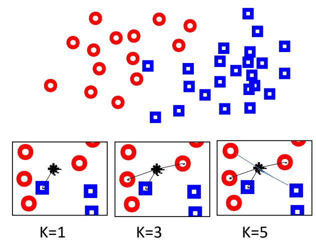
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - Take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest Neighbor Classification…

- Selection of the right similarity measure is critical:

1 1 1 1 1 1 1 1 1 1 1 0	vs	0 0 0 0 0 0 0 0 0 0 0 1
0 1 1 1 1 1 1 1 1 1 1 1		1 0 0 0 0 0 0 0 0 0 0 0

Euclidean distance = 1.4142 for both pairs

Distance Measure

- Euclidian
  - Why its not best?
    - Distances in each dimension are squared before summation places great emphasis on those features which dissimilarity is large
  - How about just absolute differences
    - Manhattan or city block or taxi-cab distance
  - How would you deemphasize single large feature difference and more influenced by numerous small ones?

Nearest neighbor Classification…

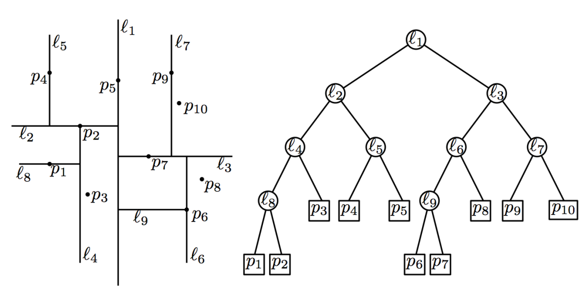
- k-NN classifiers are lazy learners since they do not build models explicitly
- Classifying unknown records are relatively expensive
- Can produce arbitrarily shaped decision boundaries
- Easy to handle variable interactions since the decisions are based on local information
- Selection of right proximity measure is essential
- Superfluous or redundant attributes can create problems
- Missing attributes are hard to handle

Improving KNN Efficiency

- Avoid having to compute distance to all objects in the training set
  - Multi-dimensional access methods (k-d trees)
  - Fast approximate similarity search
  - Locality Sensitive Hashing (LSH)
- Condensing
  - Determine a smaller set of objects that give the same performance
- Editing
  - Remove objects to improve efficiency

Improving KNN Efficiency

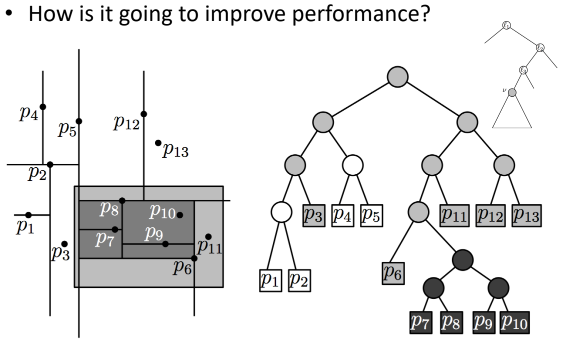
- Multi-dimensional access methods (k-d trees)



Acknowledgements: UFL Database Group

Improving KNN Efficiency

- How is it going to improve performance?



Acknowledgements: UFL Database Group

Practice

- Compare KNN with K-Means
  - Assume that the data is labeled, so we can label clusters
  - Given a new point, how each method predicts