Classifiers: kNN

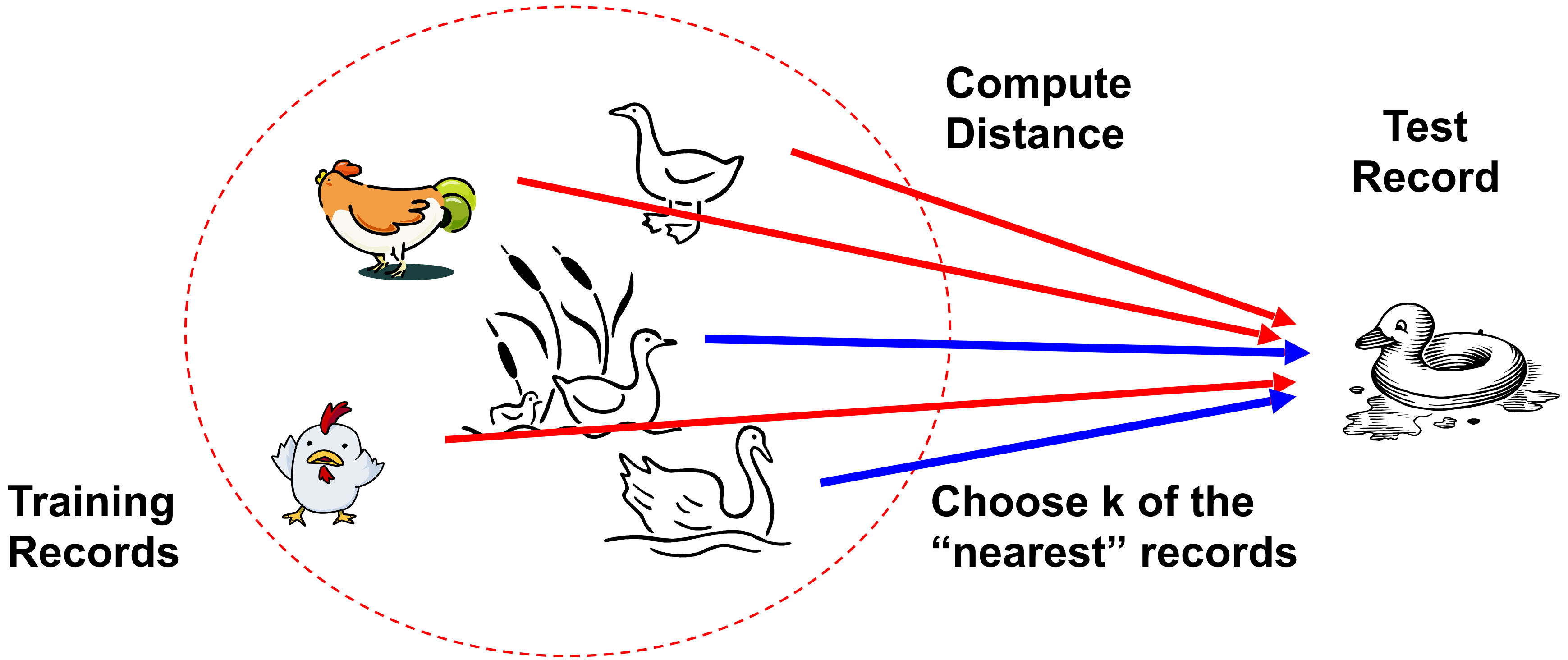
Dr. Srijith P K

Slide Credits : Dr. Vineeth N Balasubramanian

आई आई टी हैदराबाद
IIT Hyderabad

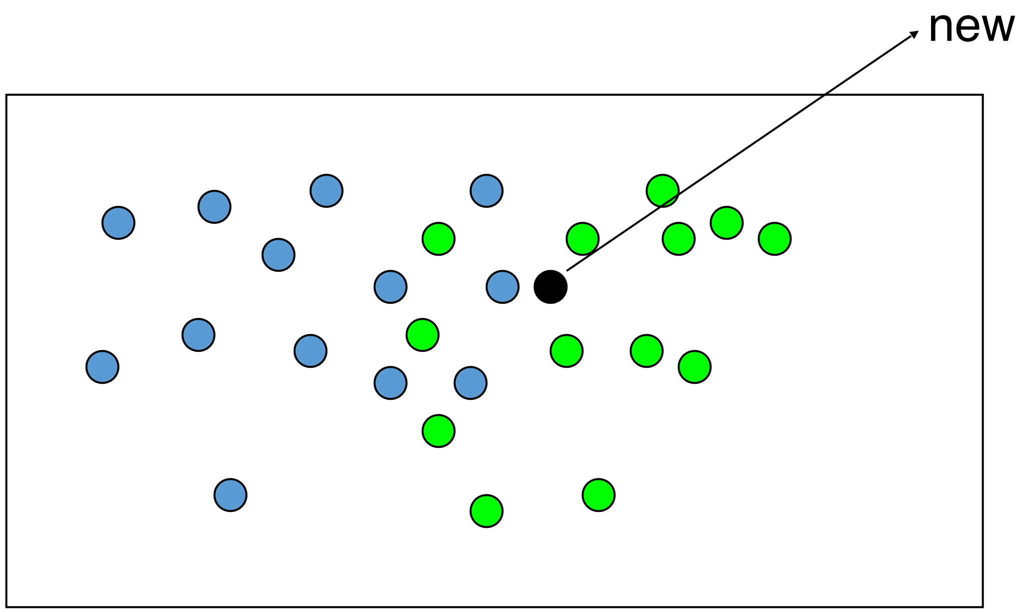
k-Nearest Neighbors

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



k-Nearest Neighbors

- Majority vote within the k nearest neighbors



K= 1: blue
K= 3: green

k-Nearest Neighbors

- An arbitrary instance is represented by $(\mathbf{a_1(x), a_2(x), a_3(x),.., a_n(x)})$
  - $\mathbf{a_i(x)}$ denotes features
- Euclidean distance between two instances
  - $\mathbf{d(x_i, x_j)=sqrt\ (sum\ for\ r=1\ to\ n\ (a_r(x_i) - a_r(x_j))^2)}$
- L_p distance
  - p=2: Euclidean distance
  - p=1: Manhattan distance
  - p =∞: Max distance
  - p= 0: Count non-zero distance
- In case of continuous-valued target function
  - Mean value of k nearest training examples

आई आई टी हैदराबाद
IIT Hyderabad

Other Distance Metrics

- Cosine Distance Metric $\rho(\vec{x}_1, \vec{x}_2) = \cos(\angle(\vec{x}_1, \vec{x}_2)) \quad = \dfrac{\vec{x}_1 \cdot \vec{x}_2}{\|\vec{x}_1\|_2 \ \|\vec{x}_2\|_2}$

- Edit Distance $x_1 =$ AAATCCCGTAA

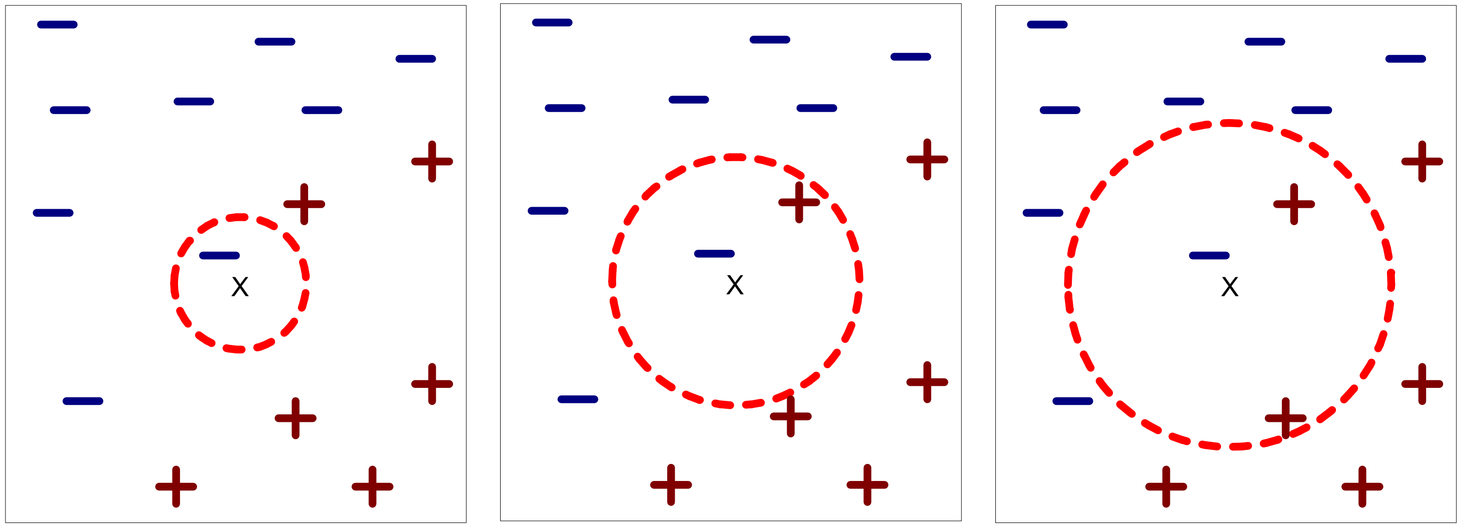
- $x_2 =$ AATCGCGTAA

Minimum number of insertions, deletions and mutations needed

$$\rho(x_1, x_2) = 2$$

k-Nearest Neighbors

- Choosing k is important
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

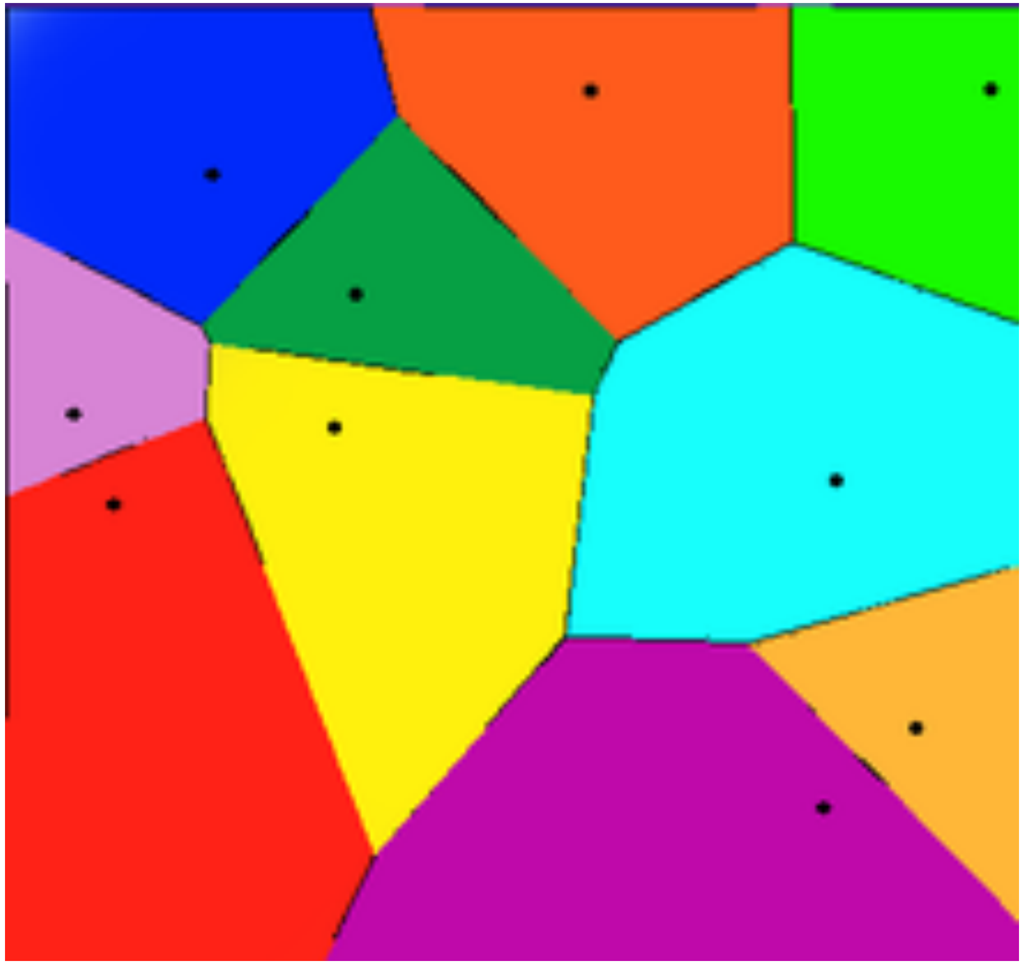


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

How to determine k

- Determined experimentally (think cross-validation!)
  - Start with k=1 and use a test set to validate the error rate of the classifier
  - Repeat with k=k+2
  - Choose the value of k for which the error rate is minimum
  - Note: k typically an odd number to avoid ties in binary classification

Voronoi Diagram



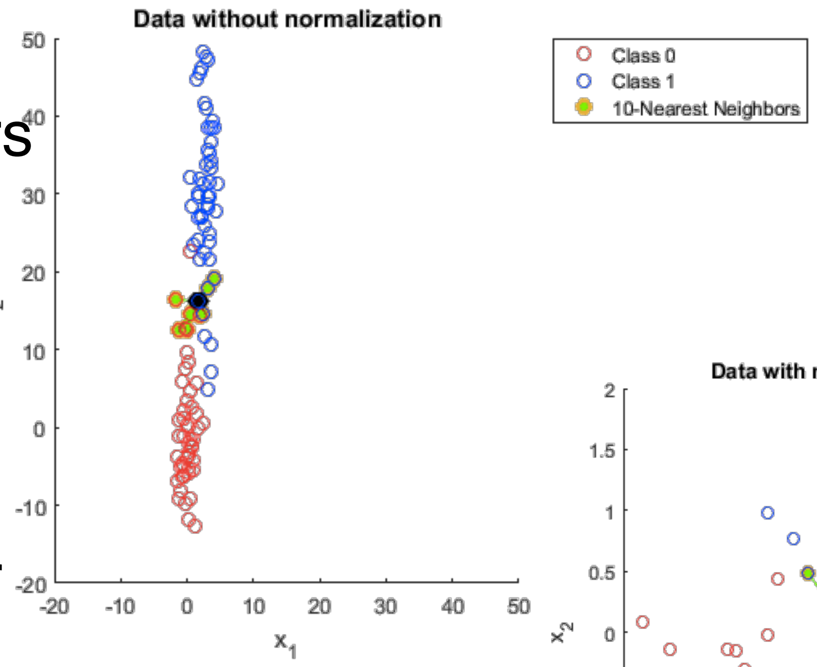
Decision surface formed by the training examples!

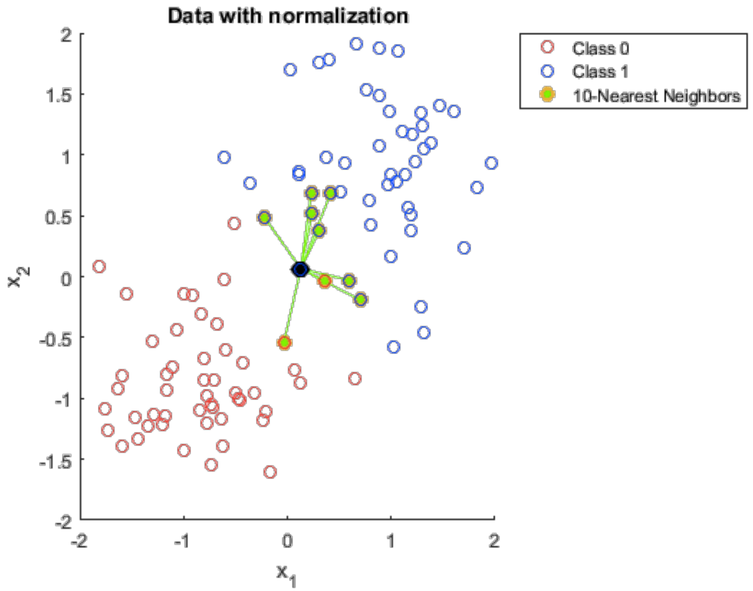
Pros and Cons

- Pros
  - Highly effective and simple method
  - Trains very fast (“Lazy” learner)
- Cons
  - Curse of dimensionality
    - In higher dimensions, all data points lie on the surface of the unit hypersphere!
  - Closeness in raw measurement space may not be good for the task
  - Storage: all training examples are saved in memory
    - A decision tree or linear classifier is much smaller
  - Slow at query time
    - Can be overcome and presorting and indexing training samples
  -

Improvements

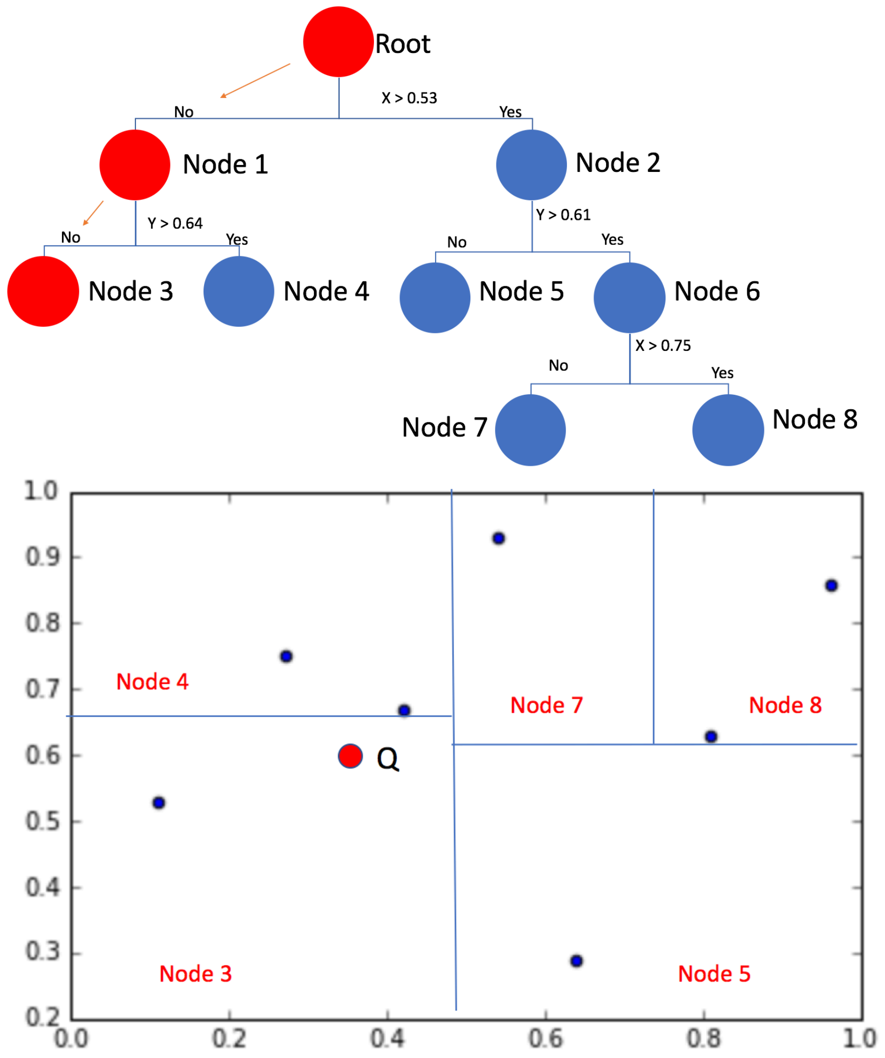
- Distance-Weighted Nearest Neighbors
  - Assign weights to the neighbors based on their ‘distance’ from the query point (E.g., weight ‘may’ be inverse square of the distances)
  - Can also learn this -> “**Metric Learning**”
- Scaling (**normalization**) attributes for fair computation of distances





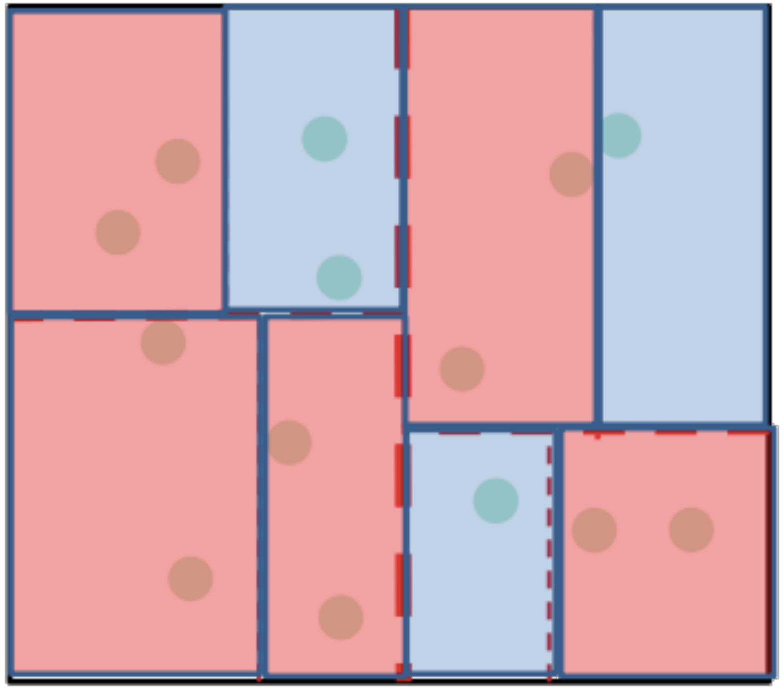
Improvements

- Finding “close” examples in a large training set quickly
  - E.g. Efficient memory indexing using kd-trees
    - In 1-dimension, can reduce complexity from O(n) to O(log n) – assuming data is sorted
  - Other methods
    - Locality-Sensitive Hashing, Clustering-based methods



Improvements

- Not storing all examples
  - We can label each cell instead and discard the training data



Readings

- Chapters 8, 9, EA Introduction to ML 2nd Edn
- Chapter 14 (Sec 14. 4) + Chapter 2 (Sec 2. 5), Bishop, PRML