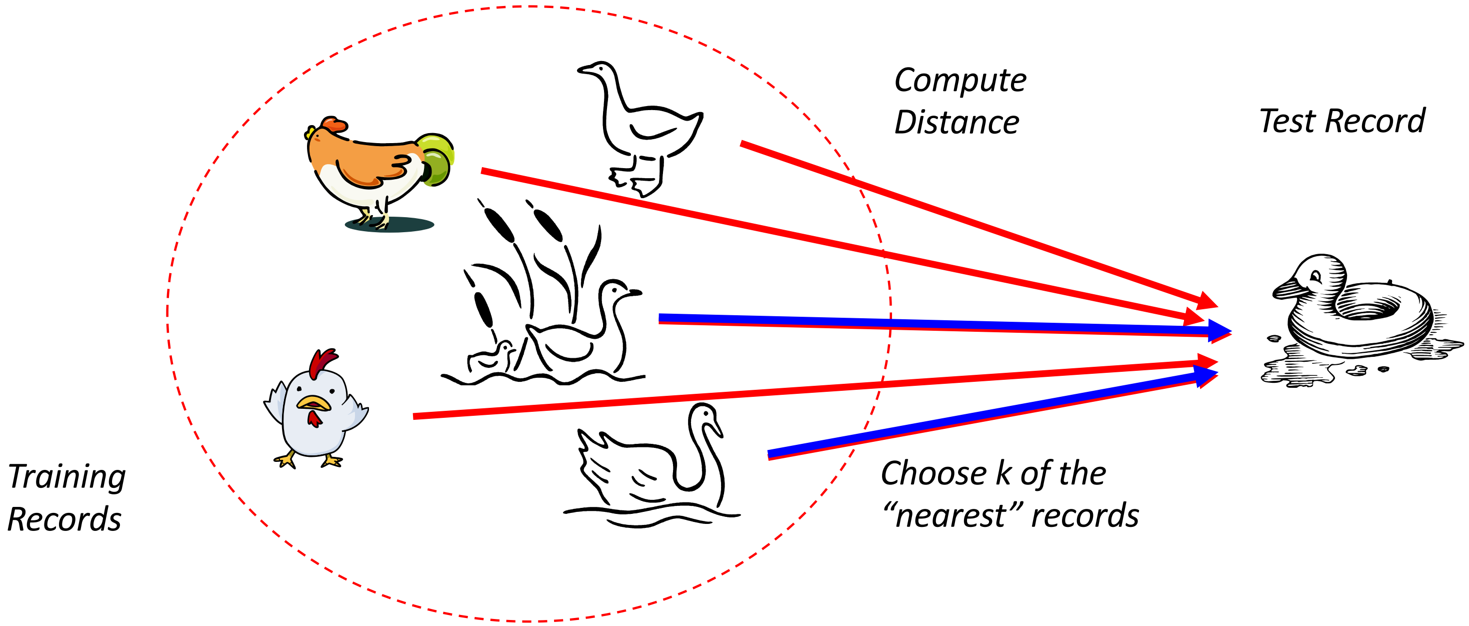
Outline

- Classification basics
- Decision tree classifiers
- Overfitting
- Lazy vs Eager Learners
- k-Nearest Neighbors (or learning from your neighbors)
- Evaluation of classifiers

Lazy learners/ Instance-based learners: k-Nearest Neighbor classifiers

- Nearest-neighbor classifiers compare a given unknown instance with training tuples that are similar to it
  - Basic idea: *If it walks like a duck, quacks like a duck, then it's probably a duck*



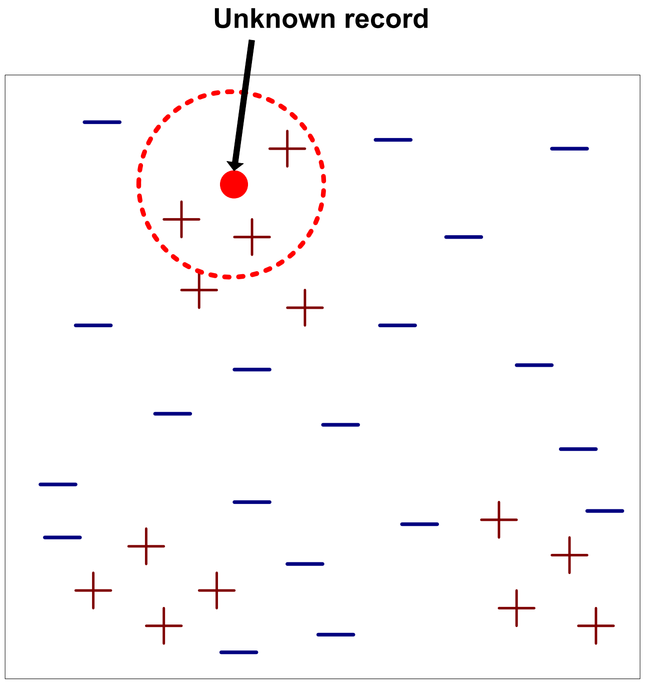
k-Nearest Neighbor classifiers

Input:

- A training set D (with known class labels)
- A distance metric to compute the distance between two instances
- The number of neighbors k

Method: Given a new unknown instance *X*

- 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)



It requires O(|D|) for each new instance

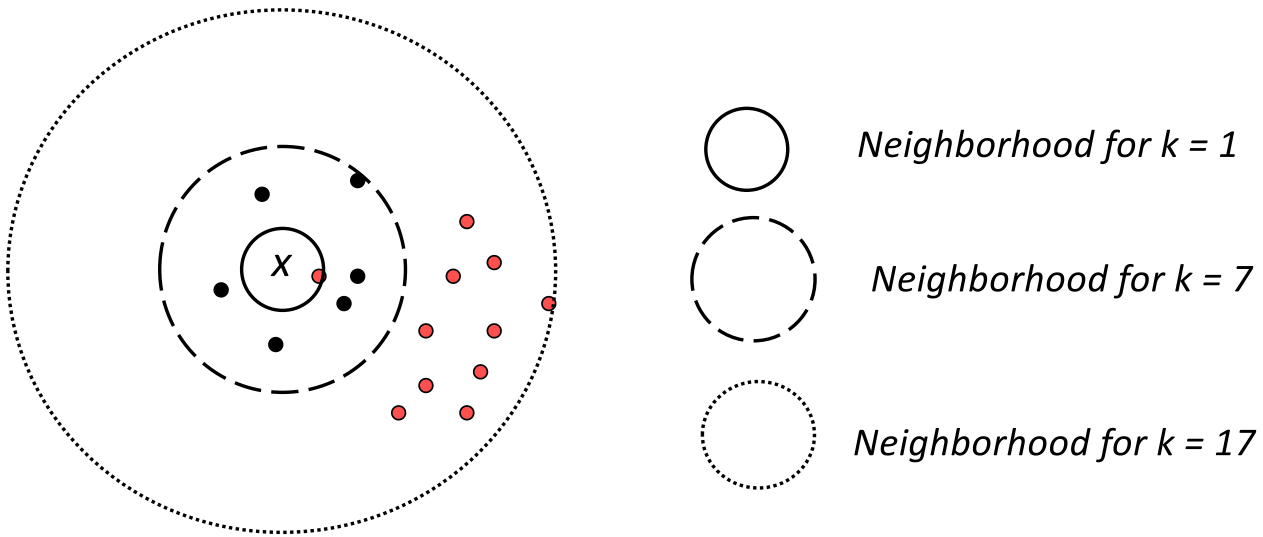
kNN algorithm

- Pseudocode:

```
Input:
  T                //training data
  K                //Number of neighbors
  t                //Input tuple to classify
Output:
  c                //Class to which t is assigned
KNN algorithm:  //Algorithm to classify tuple using KNN
begin
  N = ∅;
  //Find set of neighbors, N, for t
  for each d ∈ T do
        if |N| ≤ K
        then N = N ∪ {d};
        else if ∃ u ∈ N such that
                 sim(t,u) ≤ sim(t,d) AND sim(t,u) ≤ sim(t,u') ∀ u' ∈ N
        then N = N – {u}; N = N ∪ {d};
  //Find class for classification
  c = class to which the most u ∈ N are classified
end
```

Definition of k nearest neighbors

- too small k: high sensitivity to outliers
- too large k: many objects from other classes in the resulting neighborhood
- average k: highest classification accuracy, usually 1 << k < 10



x: unknown instance

Nearest neighbor classification

- "Closeness" is defined in terms of a distance metric
  - e.g. Euclidean distance

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

- The k-nearest neighbors are selected among the training set
- The class of the unknown instance X is determined from the neighbor list
  - If k=1, the class is that of the closest instance
  - Majority voting: take the majority vote of class labels among the neighbors
    - Each neighbor has the same impact on the classification
    - The algorithm is sensitive to the choice of k
  - Weighted voting: Weigh the vote of each neighbor according to its distance from the unknown instance
    - weight factor, $w = 1/d^2$

Nearest neighbor classification: example

Name	Gender	Height	Output1	
Kristina	F	1.6m	Short	1
Jim	M	2m	Tall	
Maggie	F	1.9m	Medium	
Martha	F	1.88m	Medium	
Stephanie	F	1.7m	Short	3
Bob	M	1.85m	Medium	
Kathy	F	1.6m	Short	2
Dave	M	1.7m	Short	4
Worth	M	2.2m	Tall	
Steven	M	2.1m	Tall	
Debbie	F	1.8m	Medium	
Todd	M	1.95m	Medium	
Kim	F	1.9m	Medium	
Amy	F	1.8m	Medium	
Wynette	F	1.75m	Medium	5
Pat	F	1.6m	?	Short

Nearest neighbor classification issues I

- Different attributes have different ranges
  - e.g., height in [1.5m-1.8m]; income in [$10K -$1M]
  - Distance measures might be dominated by one of the attributes
  - Solution: normalization

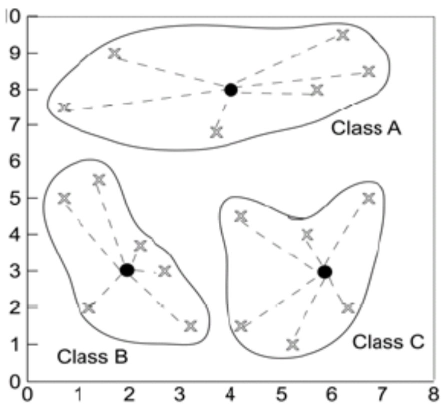
- k-NN classifiers are lazy learners
  - No model is built explicitly, like in eager learners such as decision trees
  - Classifying unknown records is relatively expensive
  - Possible solutions:
    - Use index structures to speed up the nearest neighbors computation
    - Partial distance computation based on a subset of attributes

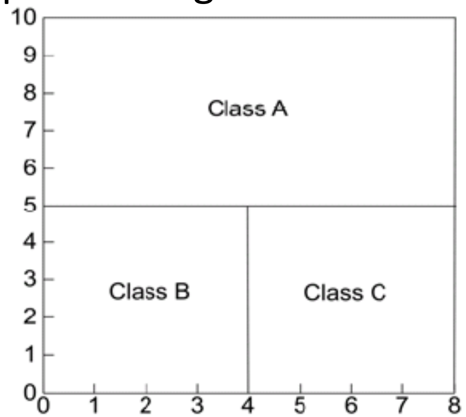
Nearest neighbor classification issues II

- The “curse of dimensionality”
  - Ratio of (D_{max_d} – D_{min_d}) to D_{min_d} converges to zero with increasing dimensionality *d*
    - D_{min_d}: distance to the nearest neighbor in the d-dimensional space
    - D_{max_d}: distance to the farthest neighbor in the d-dimensional space
  - This implies that:
    - all points tend to be almost equidistant from each other in high dimensional spaces
    - the distances between points cannot be used to differentiate between them
  - Possible solutions:
    - Dimensionality reduction (e.g., PCA)
    - Work with a subset of dimensions instead of the complete feature space

k-NN classifiers: overview

- (+-) Lazy learners: Do not require model building , but testing is more expensive
- (-) Classification is based on local information in contrast to e.g. DTs that try to find a global model that fits the entire input space: Susceptible to noise
- (+) Incremental classifiers
- (-) The choice of distance function and k is important
- (+) Nearest-neighbor classifiers can produce arbitrarily shaped decision boundaries, in contrary to e.g. decision trees that result in axis parallel hyper rectangles





Outline

- Classification basics
- Decision tree classifiers
- Overfitting
- Lazy vs Eager Learners
- k-Nearest Neighbors (or learning from your neighbors)
- Evaluation of classifiers