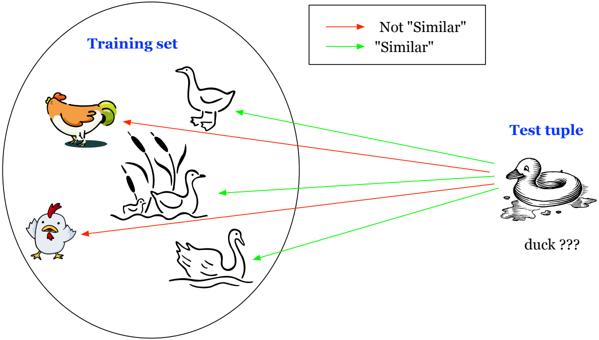
Nearest-Neighbor Classification

Basic Idea



"If it walks like a duck, quacks like a duck, and looks like a duck, then it is probably a duck!"

kNN Classification

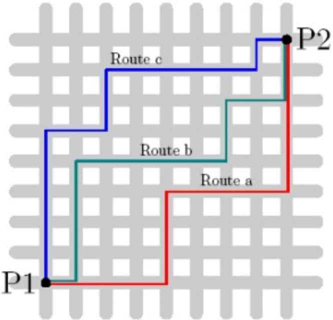
- every tuple in the training set is described by n numeric attributes
  - each tuple can be perceived as a point in a n-dimensional space
- two tuples $\mathbf{x}_1$ and $\mathbf{x}_2$ are "close" or "near" or "similar" if they have a "small" distance based on a distance metric
- the most popular metric is the Euclidean distance (ℓ_2-distance)
  - $\mathbf{x}_1 = (x_{11}, x_{12}, \ldots, x_{1n})$ and $\mathbf{x}_2 = (x_{21}, x_{22}, \ldots, x_{2n})$
  - Euclidean distance between $\mathbf{x}_1$ and $\mathbf{x}_2$ is

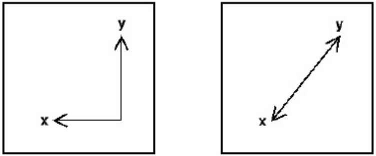
$$dist(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

Distance Metrics

Other distances can also be used

- e.g., ℓ_1-distance: (Manhattan distance) $\sum_{i=1}^{n} |x_{1i} - x_{2i}|$





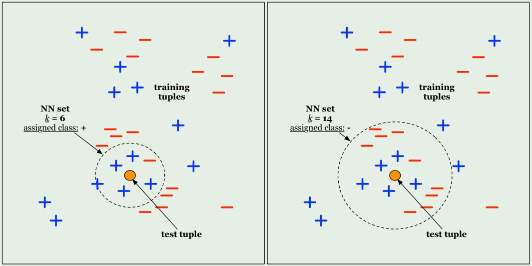
Manhattan Euclidean

k-Nearest-Neighbor (kNN) Classifier

Given an unknown tuple

- search for its k nearest training tuples ("neighbors") based on a pre-defined distance metric
- assign the unknown tuple to the majority class of its k nearest neighbor set

Example (assume Euclidean distance)



Data Preprocessing

- distance metric is affected by the ranges of the attribute values
- large ranges (e.g., *income*) may outweigh small ranges (e.g., *age*) in the distance computation
- perform data normalization on the training set, such that all values fall within range $[0, 1]$
  - e.g., transform a value $v \in [min_A, max_A]$ of an attribute A to

$$v' = \frac{v - min_A}{max_A - min_A} \quad \text{(min-max normalization)}$$

What about categorical data?

- suppose that we measure the Euclidean distance between $\mathbf{x}_1$ and $\mathbf{x}_2$
- need to calculate $x_{1i} - x_{2i}$ on a categorical attribute A_i
- assign “0” if the values are identical, and “1” otherwise

Curse of Dimensionality

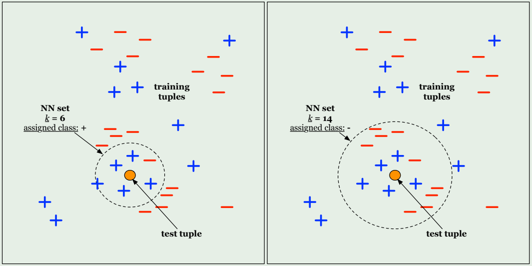
- when dimensionality increases, data becomes increasingly sparse
- density and distance between points $\rightarrow$ less meaningful

- as a result, a nearest neighbor query may become non-meaningful
- in particular, distance between neighbors could be dominated by irrelevant attributes
  - $\rightarrow$ dimension reduction
  - use weights to attribute higher importance to some attributes (so far we have implied that all attributes are assigned an equal weight)
  - e.g,. $dist(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{n} \beta_i (x_{1i} - x_{2i})^2}$

Notes

- k is a user-defined parameter which affects the accuracy of the classifier

Example (assume Euclidean distance)



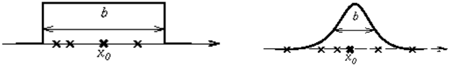
- k too small $\rightarrow$ the classifier becomes sensitive to noise
- k too large $\rightarrow$ the neighborhood may "mix" points from different classes

- can tune k by cross-validation

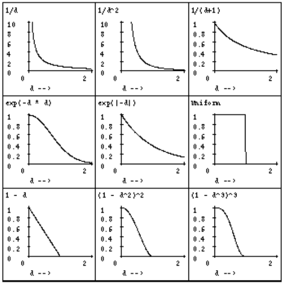
How?

Notes...

- on prediction, assign the unknown tuple to the majority class of its k nearest neighbor set



- can also take into account its distance from its neighbors
  - idea: give greater weights to closer neighbors
  - e.g., weight the contribution of each of the k neighbors according to their distance to the query x_q: $w = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_k)^2}$



- the precise choice of kernel shape usually does not matter

Lazy Learning

decision trees, naive Bayes classifiers (and many other classifiers)

- build a model as soon as the training set is available (before seeing the test examples)
- eager learning

kNN classifier

- just store all training examples
- lazy learning

	eager	lazy
different new instances	estimates based on the same function	estimates based on different functions
approximation to the target function	global	local
computation on training	a lot	little
computation on testing	little	a lot