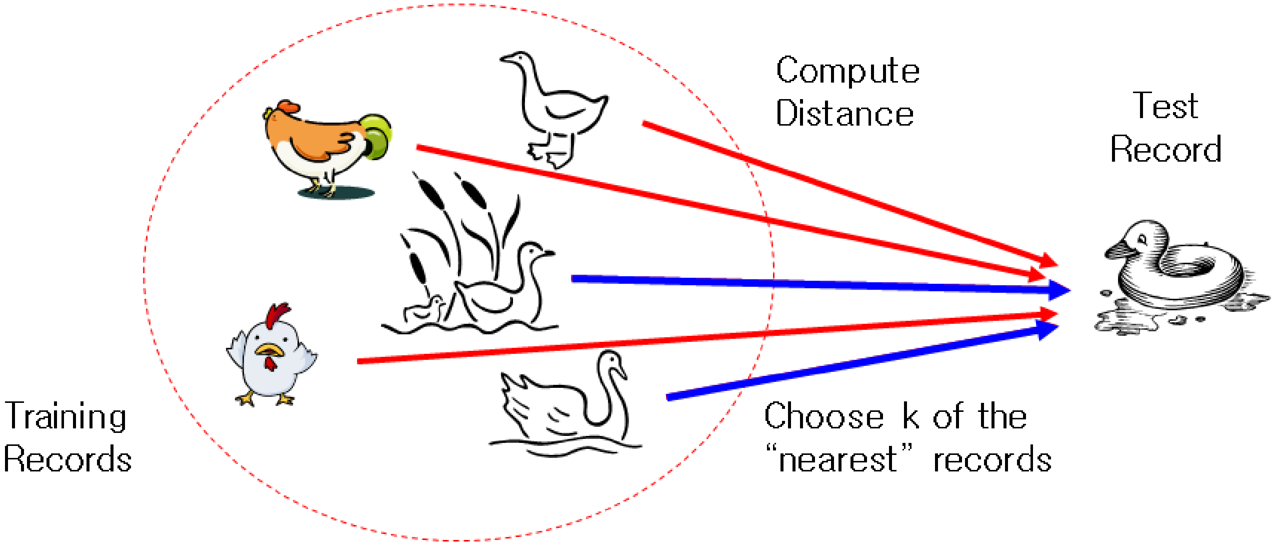
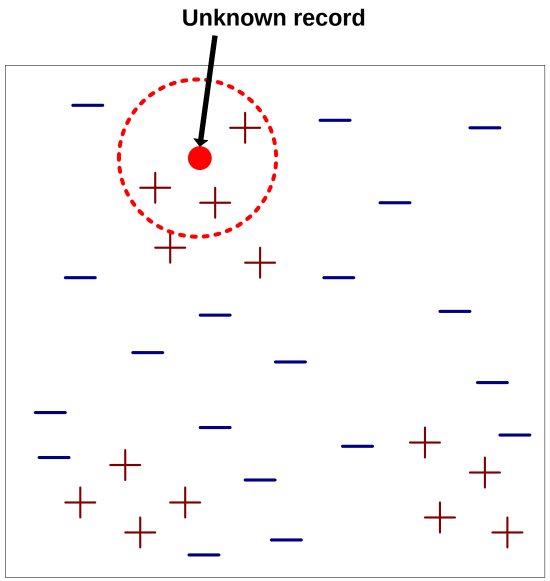
Instance-Based Learning
(K Nearest Neighbor)

Instance-Based Learning(IBL)

- Instance-Based Learning(IBL) determines unknown categories by finding the most similar cases in a database

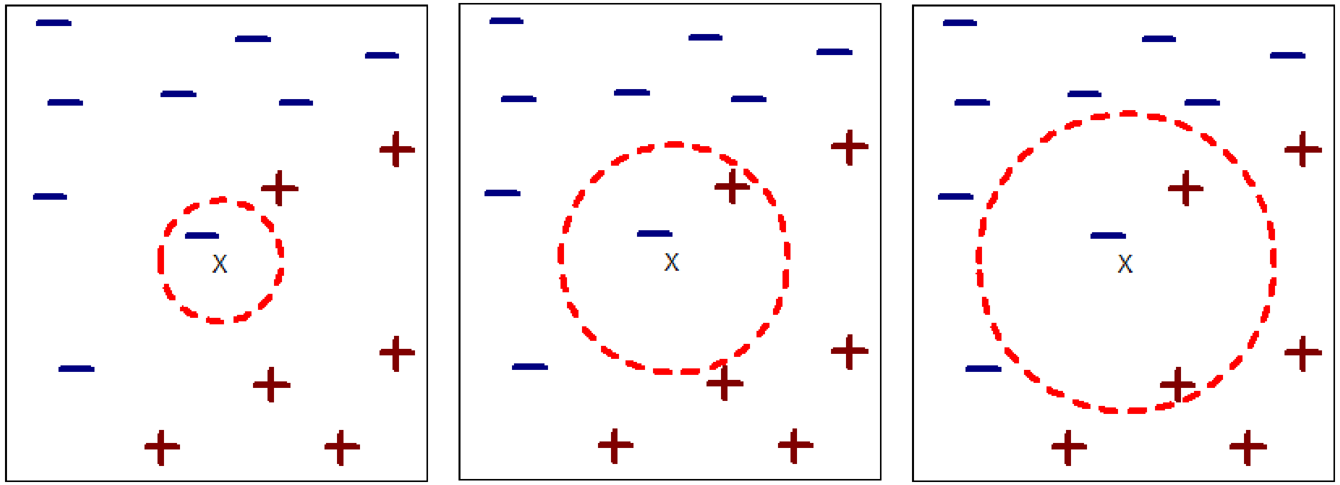
- basic idea
  - if it walks like a duck, quacks like a duck, then it's probably a duck





Instance-Based Learning

- nearest-neighbor techniques are based on the concept of similarity.
  - also known as **Instance-Based Learning(IBL)** or **k nearest neighbor(kNN)** algorithm



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

- cares about the existence of two operations
  - **distance function** and **combination function**

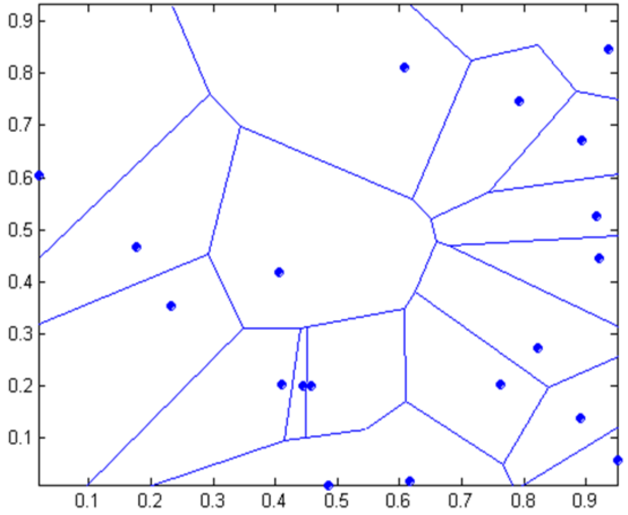
Instance-Based Learning

- IBL classifiers are **lazy learners**
  - unlike eager learners such as decision tree and NNs, it does not build models explicitly
  - classifying unknown records are relatively expensive

- strength
  - use data "as is." Doesn't care about the format of the records.
  - simplicity of algorithm
  - no training

- cons
  - store large amount of historical data
  - classifying requires is time-consuming
  - difficulty of finding optimal distance function or combination function

- Voronoi Diagram (1 nearest neighbor)



Estimating Rents in Tuxedo, New York

- goal is to estimate the cost of renting an apartment in the target town by combining data on rents in several similar towns--its nearest neighbors

- first identifies neighbors and combine information from them
  - not its geographic neighbors, rather its neighbors based on descriptive variables.
    (in this case, population and home value)

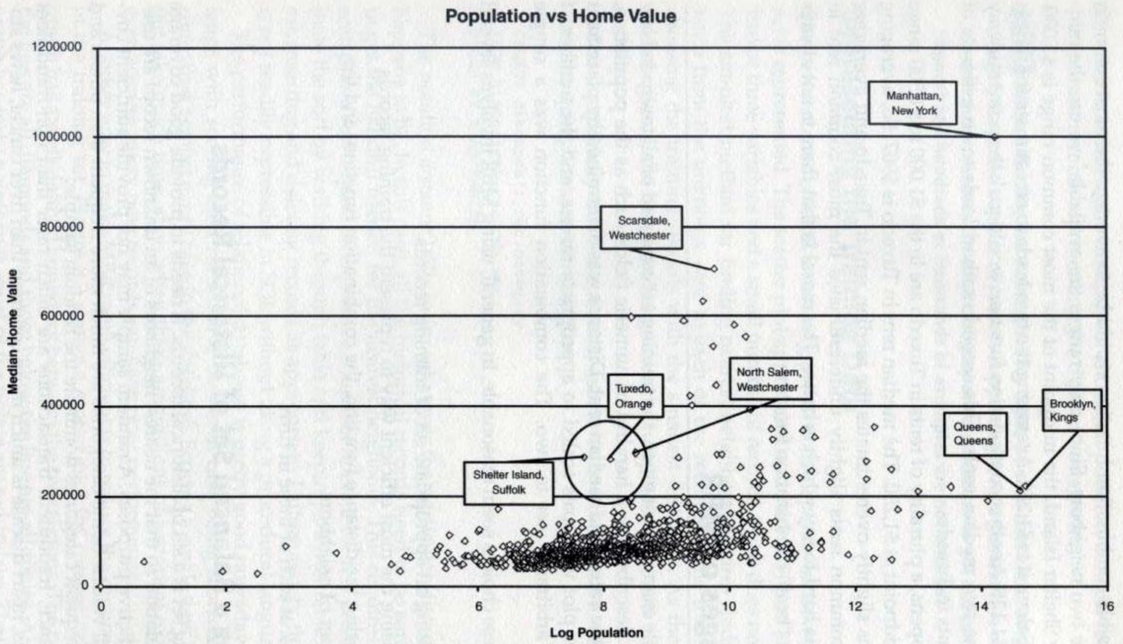


Figure 8.1 Based on 2000 census population and home value, the town of Tuxedo in Orange County has Shelter Island and North Salem as its two nearest neighbors.

"Mastering Data Mining" by Gordon S. Linoff & Michael J. A. Berry

Estimating Rents in Tuxedo, New York

- two nearest neighbors are selected

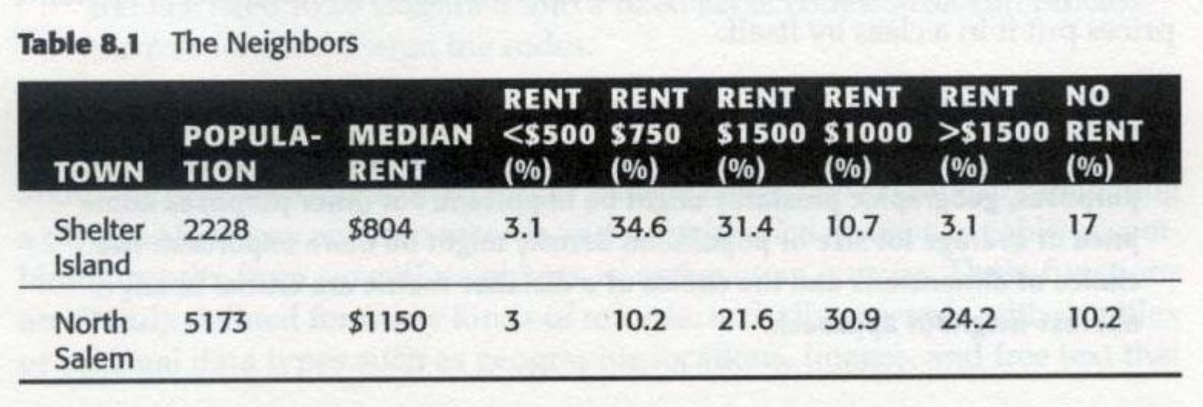


Table 8.1 The Neighbors

TOWN	POPULATION	MEDIAN RENT	RENT <$500 (%)	RENT $750 (%)	RENT $1500 (%)	RENT $1000 (%)	RENT >$1500 (%)	NO RENT (%)
Shelter Island	2228	$804	3.1	34.6	31.4	10.7	3.1	17
North Salem	5173	$1150	3	10.2	21.6	30.9	24.2	10.2

- combine information from the neighbors to infer something about the target.
  - how to combine information from the neighbors to come up with an estimate that characterizes rents in Tuxedo in the same way.
  - neighbors have quite different distributions of rents even though the median rents are similar.

Estimating Rents in Tuxedo, New York

- possible combination functions

  1) pick the point midway between the two median rents.
     - ($804 + $1,150)/ = $977
     - the actual median rent in Tuxedo is $907

  2) average the most common rents of the two neighbors
     - the midpoint of the most common range in Shelter Island is $1,000, For North Salem, it is $1,250
     - average is $1,125
     - the actual plurality rents in Tuxedo are $1,000 to $1,500 with the midpoint $1,250.

  - method 1) slightly overestimates the median rent in Tuxedo
  - method 2) slightly underestimates the most common rent in Tuxedo

Distance Function

- properties of distance function
  1) *well-defined* : always defined and non-negative. **d(A,B)>=0**
  2) *identity* : distance to itself is zero. **d(A,A)=0**
  3) *commutativity* : direction does not make a difference. **d(A,B)=d(B,A)**
  4) *triangle inequality* : visiting an intermediate point C on the way from A to B never shortens the distance. **d(A,B) >=d(A,C)+d(C,B)**

- the nearest neighbor of record B may be A, but A may have neighbor closer than B

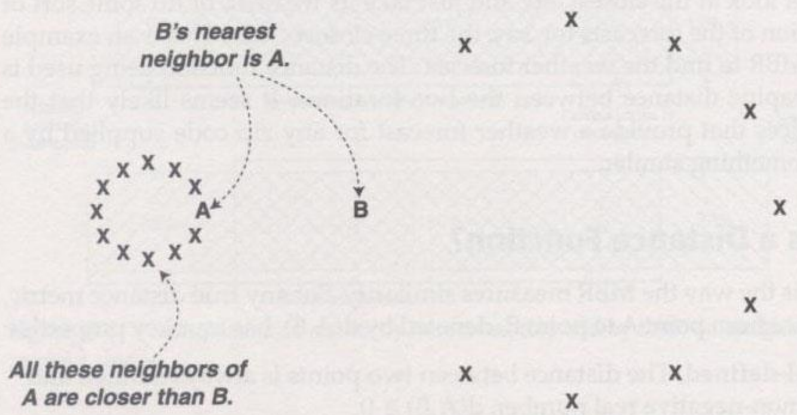

B's nearest neighbor is A.
A
B
All these neighbors of A are closer than B.

Figure 8.5 B's nearest neighbor is A, but A has many neighbors closer than B.

Building a Distance Function

- how to build a distance function for records consisting of many different attribute types:
  - one attribute at a time
- Example 1
  - two numeric attributes and one categorical attribute

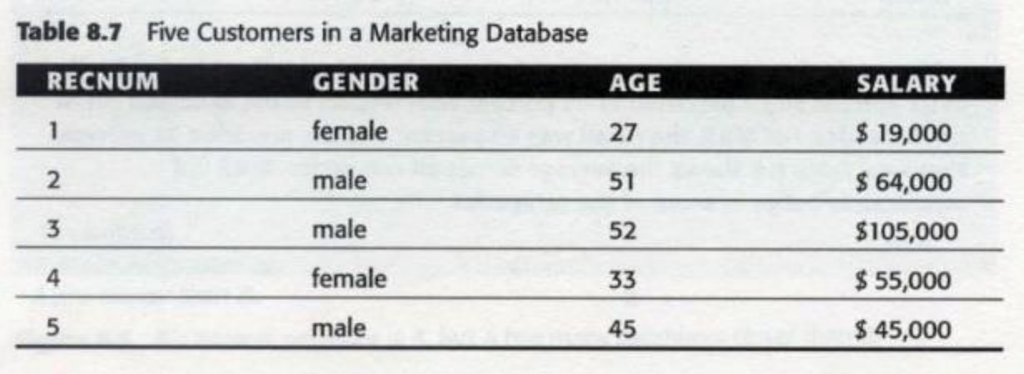


Table 8.7 Five Customers in a Marketing Database

RECNUM	GENDER	AGE	SALARY
1	female	27	$ 19,000
2	male	51	$ 64,000
3	male	52	$105,000
4	female	33	$ 55,000
5	male	45	$ 45,000

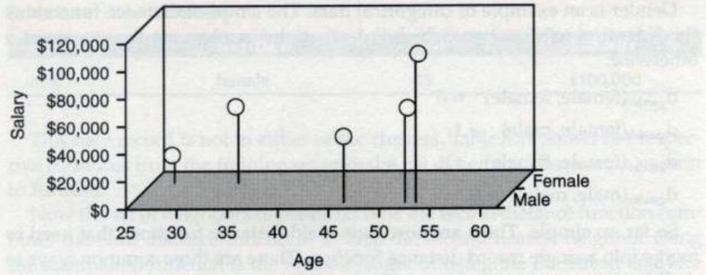


Figure 8.6 This scatter plot shows the five records from Table 8.7 in three dimensions—age, salary, and gender—and suggests that standard distance is a good metric for nearest neighbors.

- the four most common distance functions for numeric attributes
  1) absolute value of the difference : |A-B|
  2) square of the difference : (A-B)^2
  3) normalized absolute value : always between 0 and 1
     |A-B|/(maximum difference)
  4) absolute value of difference of standardized values:
     |(A-B|/(standard deviation)|

Building a Distance Function

- the distance matrix for ages
  - use normalized absolute value

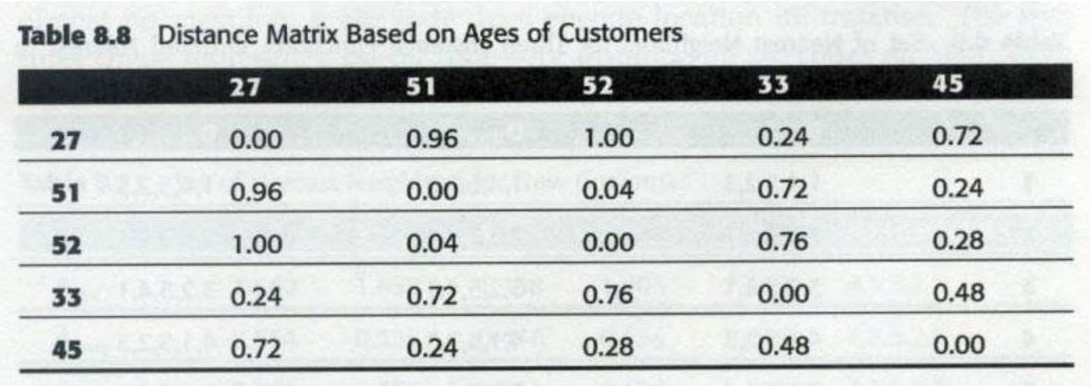


Table 8.8 Distance Matrix Based on Ages of Customers

	27	51	52	33	45
27	0.00	0.96	1.00	0.24	0.72
51	0.96	0.00	0.04	0.72	0.24
52	1.00	0.04	0.00	0.76	0.28
33	0.24	0.72	0.76	0.00	0.48
45	0.72	0.24	0.28	0.48	0.00

- for categorical attribute
  - "identical" distance function : 0 when the value is the same and 1 otherwise
  - d_{gender}(female, female) = 0; d_{gender}(female, male) = 1;
    d_{gender}(male, female) = 1; d_{gender}(male, male) = 0;

- the **attribute distance functions** merge into a single **record distance function**

  1) Manhattan distance or summation:

     $d_{sum}(A,B) = d_{gender}(A,B)+d_{age}(A,B)+d_{salary}(A,B)$

  2) normalized summation:

     $d_{norm}(A,B) = d_{sum}(A,B)/\max(d_{sum})$

  3) Euclidean distance:

     $d_{euclid}(A,B) = \mathrm{sqrt}(d_{gender}(A,B)^2 +d_{age}(A,B)^2 +d_{salary}(A,B)^2)$

  4) cosine similarity:

Building a Distance Function

- In this example, nearest neighbors are exactly the same regardless of the **record distance function**(coincidence !)

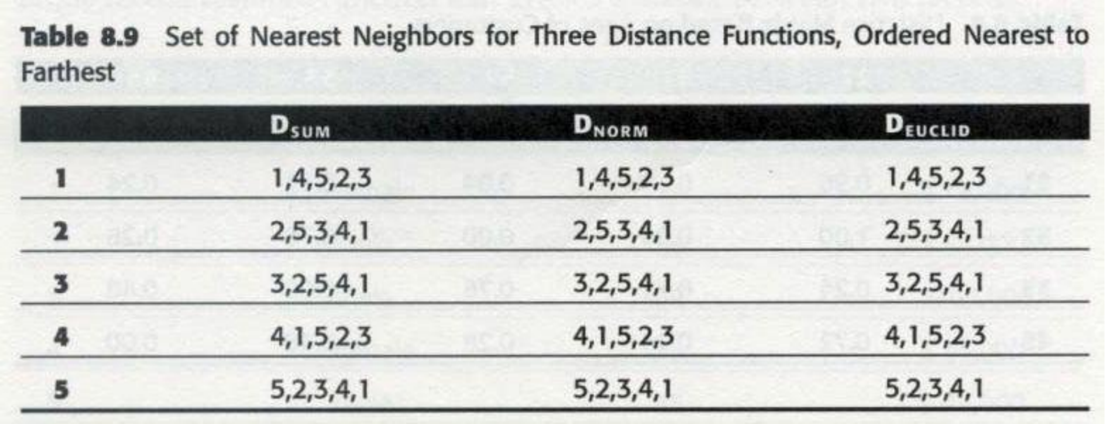


Table 8.9 Set of Nearest Neighbors for Three Distance Functions, Ordered Nearest to Farthest

	D_{SUM}	D_{NORM}	D_{EUCLID}
1	1,4,5,2,3	1,4,5,2,3	1,4,5,2,3
2	2,5,3,4,1	2,5,3,4,1	2,5,3,4,1
3	3,2,5,4,1	3,2,5,4,1	3,2,5,4,1
4	4,1,5,2,3	4,1,5,2,3	4,1,5,2,3
5	5,2,3,4,1	5,2,3,4,1	5,2,3,4,1

- consider a new record for classification

* using normalized absolute value

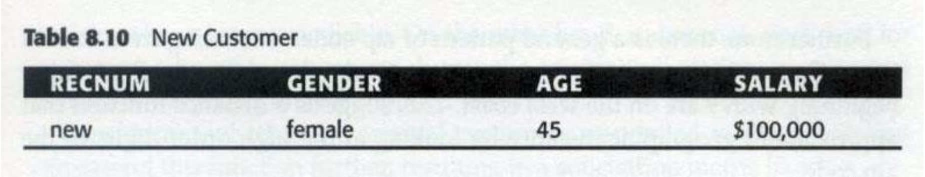


Table 8.10 New Customer

RECNUM	GENDER	AGE	SALARY
new	female	45	$100,000

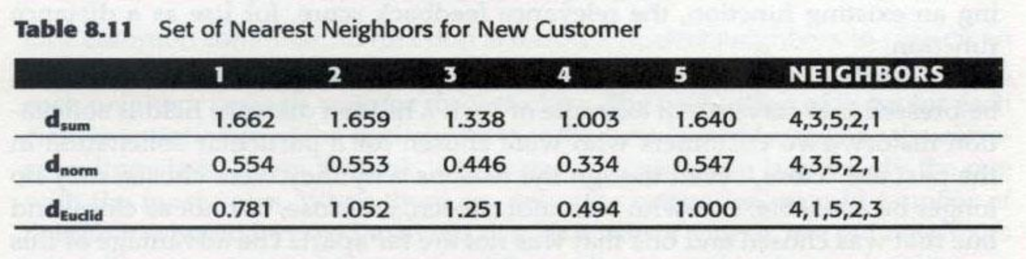


Table 8.11 Set of Nearest Neighbors for New Customer

	1	2	3	4	5	NEIGHBORS
d_{sum}	1.662	1.659	1.338	1.003	1.640	4,3,5,2,1
d_{norm}	0.554	0.553	0.446	0.334	0.547	4,3,5,2,1
d_{Euclid}	0.781	1.052	1.251	0.494	1.000	4,1,5,2,3

- you can make decision based on these neighbors
- the combination function can also incorporate weights so each attribute contributes a different amount to the record.: feature weighting
  - a way to incorporate a priori knowledge

Computing Distances

- distance of each data using **normalized absolute value** and **Manhattan distance(d_sum)**

dist(1, new) = dist(female, female) + dist(27,45) + dist(19000,100000)
= 0 + |27-45|/|52-27| + |19000-100000|/|105000-19000|
= 0.72+0.94=1.66

dist(2, new) = dist(male, female) + dist(51,45) + dist(64000,100000)
= 1 + |51-45|/25 + |64000-100000|/86000
=0.72+0.94=1.65

dist(3, new) = 1.33 (omitted)

dist(4, new) = dist(female, female) + dist(33,45) + dist(55000,100000)
= 0 + |33-45|/|52-27| + |55000-100000|/|105000-19000|
= 0.48+0.52=1.00

dist(5, new) = 1.64 (omitted)

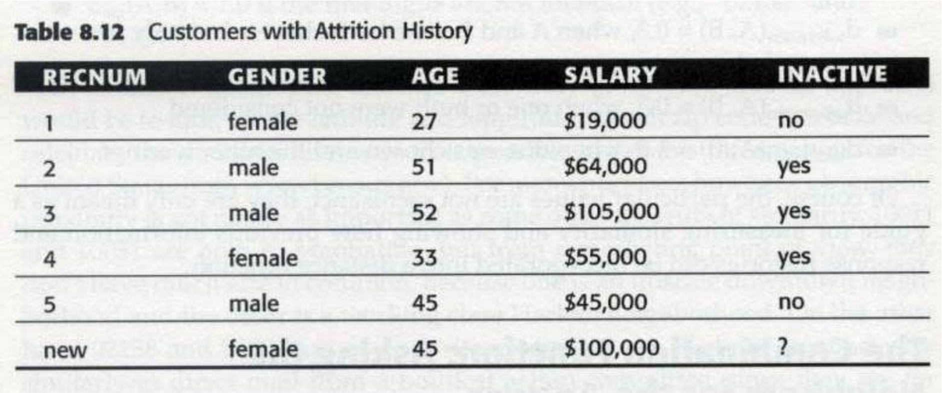


Table 8.12 Customers with Attrition History

RECNUM	GENDER	AGE	SALARY	INACTIVE
1	female	27	$19,000	no
2	male	51	$64,000	yes
3	male	52	$105,000	yes
4	female	33	$55,000	yes
5	male	45	$45,000	no
new	female	45	$100,000	?

Combination Approach

- two approaches for combining the results of nearest neighbors
  - **majority voting**
  - **weighted voting** (usually better than majority voting)

- IBL vote on the answer
  - when the task is to assign a single class, it is simply the one with the most votes.
  - *k* (# of neighbors) should be odd to avoid tie-break
  - when there are c categories, use *c+1* neighbors to ensure that at least one class has a plurity

Majority Voting

- results of record distance

dist(1,new)	dist(2,new)	dist(3,new)	dist(4,new)	dist(5,new)
1.66	1.65	1.33	1.0	1.64
n	y	y	y	n

- try to determine if the new record is active or inactive by using different values of *k* for two distance functions, d_{euclid} and d_{norm}.
  - ? means no prediction due to a tie
  - different *k* affect the classification

	Neighbors	Neighbor values	k=1	k=2	k=3	k=4	k=5
d_sum	4,3,5,2,1	y,y,n,y,n	y	y	y	y	y
d_Euclid	4,1,5,2,3	y,n,n,y,y	y	?	n	?	y

- use the percentage of neighbors in agreement to provide the level of confidence in the prediction
  - works just as well when there are more than two categories.

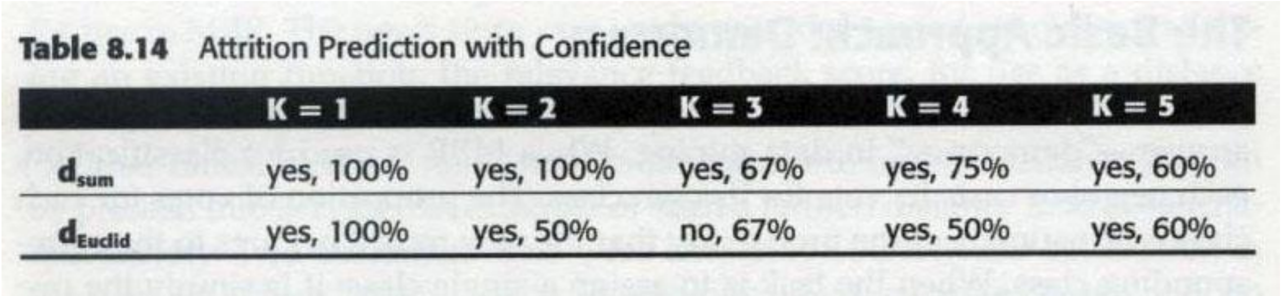


Table 8.14 Attrition Prediction with Confidence

	K = 1	K = 2	K = 3	K = 4	K = 5
d_{sum}	yes, 100%	yes, 100%	yes, 67%	yes, 75%	yes, 60%
d_{Euclid}	yes, 100%	yes, 50%	no, 67%	yes, 50%	yes, 60%

Weighted Voting

- All neighbors are not created equal--more like shareholder.
  - e.g.: the weight(importance) of neighbor is inversely proportional to the distance from the new record: **weight = 1/distance**

dist(1,new)	dist(2,new)	dist(3,new)	dist(4,new)	dist(5,new)
1/1.66=0.602	1/1.65=0.606	1/1.33=0.752	1/1.0=1	1/1.64=0.609
n	y	y	y	n

- value prediction with weighted voting

	k=1	k=2	k=3	k=4	k=5
d_sum	1 to 0	(1+0.752) to 0	(1+0.752) to 0.609	(1+0.752+0.606) to 0.609	(1+0.77+0.61) to (0.609+0.602)

- confidence level can now be calculated as the ratio of winning weights to total weights
  - confidence of prediction = (sum of winning weight)/(sum of total weight)
- confidence of d_sum prediction with k=4
  - confidence = (1+0.752+0.606)/(1+0.752+0.609+0.606+0.602) = 2.358/3.569 = 0.66
- weighted vote eliminates the ambiguous results for even numbers

More on Distance Measure

Minkowski Distance (p-norm)

- a generalization of Manhattan and Euclidean distance

$$d(x_i, x_j) = \sqrt[q]{|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \cdots + |x_{ip} - x_{jp}|^q} = \left(\sum_{k=1}^{p} |x_{ik} - x_{jk}|^q\right)^{1/q}$$

- If q = 2, d is Euclidean distance (L_2 norm)
- If q = 1, d is Manhattan distance (L_1 norm)
- If q = 0, d is Hamming distance

$$\#_p(x_{ip} \neq x_{jp})$$

- If q goes infinity, (L_{max} norm, Chebyshev distance))

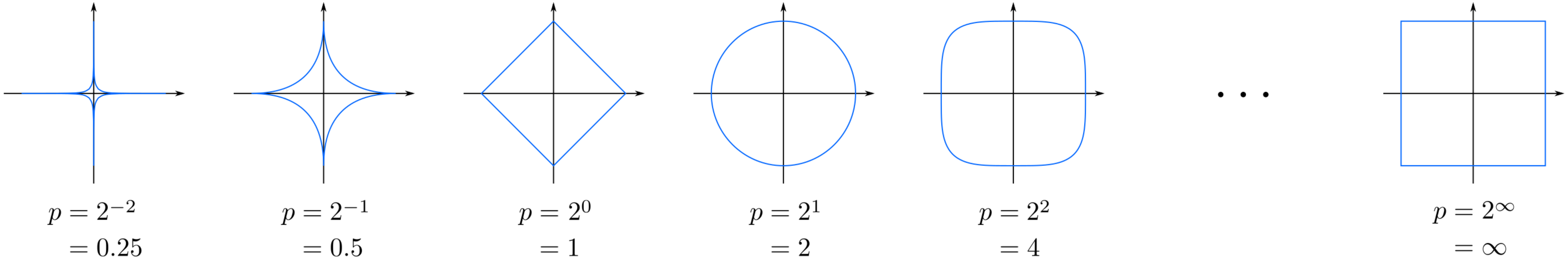
$$d(x_i, x_j) = \max_p |x_{ip} - x_{jp}|$$

- If q goes negative infinity, (L_{min} norm)

$$d(x_i, x_j) = \min_p |x_{ip} - x_{jp}|$$

More on Distance Measure

Minkowski Distance (continued)



Properties of Minkowski Distance

$d(x_i, x_i) = 0$ *(identity)*

$d(x_i, x_j) > 0$ *(well-defined)*

$d(x_i, x_j) = d(x_j, x_i)$ *(commutativity)*

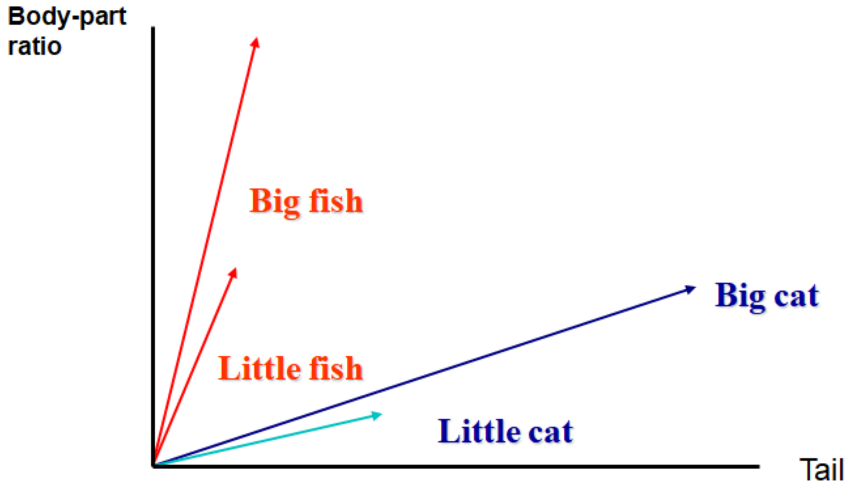
$d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j)$ *(triangle inequality)*

More on Distance Measure

Cosine Similarity

- Sometimes, the direction of the data is more important than distance itself.
- The solution is to use a different geometric interpretation of the same data.
- Think of the records as *vectors* and measure the *angle* between them.
  - a vector has both magnitude and direction. For this similarity measure, it is the direction that matters.
  - the angle between vectors provides a measure of similarity, not influenced by differences in magnitude.
  - *cosine similarity* between two vectors : a measure that calculates the cosine of the angle between them.
  - popular similarity measure used in text processing
  - cosine similarity between two vectors A & B (θ: inner angle) is given as follows:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum\limits_{i=1}^{n} A_i \times B_i}{\sqrt{\sum\limits_{i=1}^{n} (A_i)^2} \times \sqrt{\sum\limits_{i=1}^{n} (B_i)^2}}$$



How to Choose k

- large k:
  - less sensitive to noise (particularly class noise)
  - better probability estimates for discrete classes
  - larger training sets allow larger values of k
  - As $n \rightarrow \infty$, the k-NN error is no more than twice the error of the Bayes Optimal classifier
    - Bayes optimal classifier: $y^* = h_{opt}(x) = argmax_y P(y|x)$

- small k:
  - sensitive to noise
  - captures fine structure of problem space better
  - may be necessary with small training sets

- balance must be struck between large and small k

Strengths/Weakness of IBL

Strength

1) results are readily understandable
    - the list of neighbors provides an explanation of how IBL arrives at a specific result

2) applicable to arbitrary data types
    - does not depend on the underlying representation of the data
    - applicable to numeric and categorical data including images, text, and audio.

Weakness

1) Computationally Intensive during Prediction Phase
    - the performance penalty for IBL occurs during the prediction phase instead of the training phase

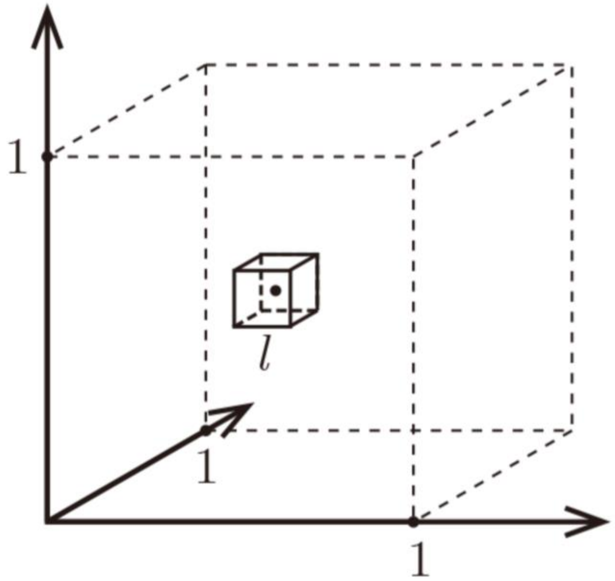
2) Large Amount of Storage for Training Set
    - the larger the training set the better the results

3) Not adequate with large dimensionality

Curse of Dimensionality

Distances Between Points

- The NN classifier makes the assumption that similar points share similar labels.
- Unfortunately, in high dimensional spaces, points that are drawn from a probability distribution, tend to never be close together.
- We can illustrate this on a simple example.
- We will draw points uniformly at random within the unit cube and we will investigate how much space the nearest neighbors of a test point inside this cube will take up.



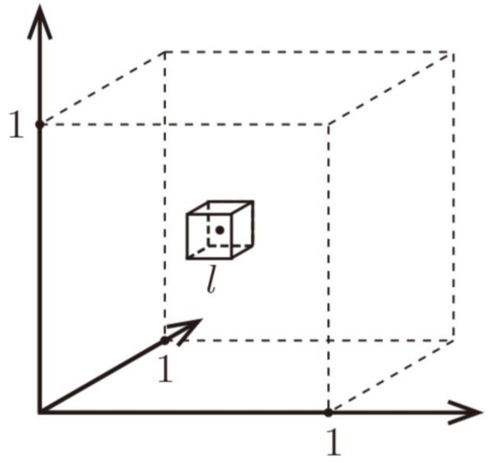
Source: Cornell CS4780 slides

Distances Between Points

- Formally, imagine the unit cube $[0,1]^d$.
- All training data is sampled *uniformly* within this cube, i.e. $x_i \in [0,1]^d \ \forall i$, and we are considering the k=10 nearest neighbors of such a test point.

- Let l be the edge length of the smallest hyper-cube that contains all nearest neighbor of a test point. Then $l^d \approx k/n$ and $l \approx \left(\frac{k}{n}\right)^{1/d}$
  (n: total number of data)

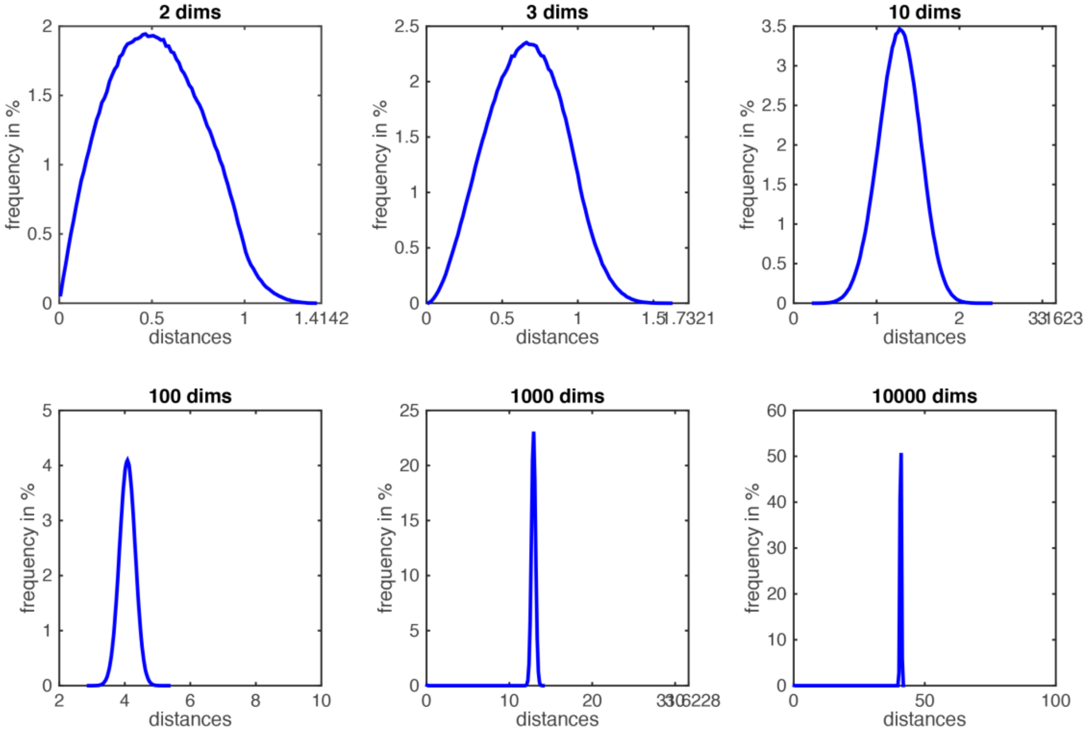
- So as d increases, almost the entire space is needed to find the 10-NN.
- This breaks down the k-NN assumptions, because the k-NN are not particularly closer (and therefore more similar) than any other data points in the training set.
- Why would the test point share the label with those k-nearest neighbors, if they are not actually similar to it?

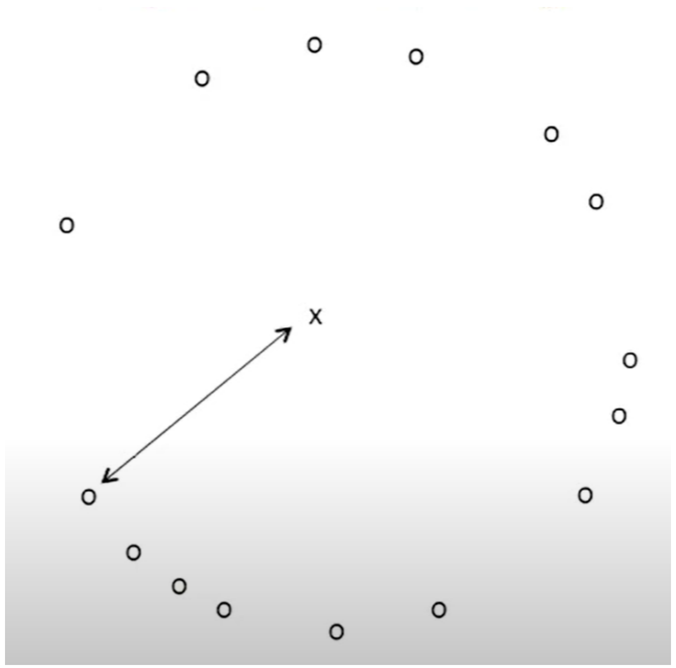


d	l
2	0.1
10	0.63
100	0.955
1000	0.9954

Distances Between Points

- The histogram plots show the distributions of all pairwise distances between randomly distributed points within d-dimensional unit squares.
- As the number of dimensions d grows, all distances concentrate within a very small range.
- In a very high dimension, each data pretty much has same distance to each other!!!



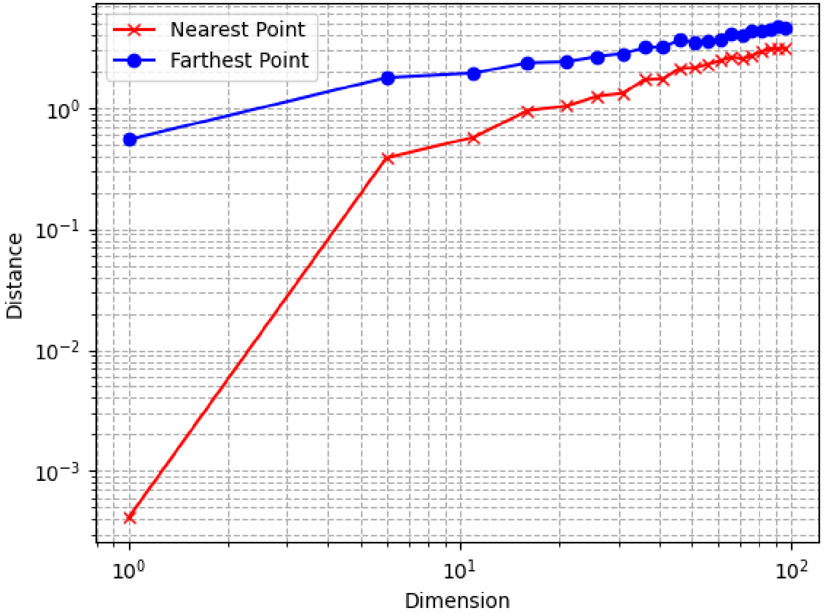


Distances Between Points

- One might think that one rescue could be to increase the number of training samples, n, until the nearest neighbors are truly close to the test point.
- How many data points would we need such that l becomes truly small?

$$l = \frac{1}{10} = 0.1 \;\Rightarrow\; n = \frac{k}{l^d} = k \cdot 10^d$$

- n grows exponentially! (For $d > 100$, n is bigger than electrons in the universe)



Data With Low Dimensional Structure

- Data may lie in low dimensional subspace or on sub-manifolds.
  - Example: natural images (digits, faces)
  - Here, the true dimensionality of the data can be much lower than its ambient space.
- The figure shows an example of a data set sampled from a 2-dimensional manifold, that is embedded within 3d.
- Human faces are a typical example of an intrinsically low dimensional data set.

- Although an image of a face may require18M pixels, a person may be able to describe this person with less than 50 attributes (e.g. male/female, blond/dark hair, ...)

- An example of a data set in 3d that is drawn from an underlying 2-dimensional manifold.
- The points are confined to the pink surface area, which is embedded in a 3-dimensional ambient space.