k-Nearest Neighbor

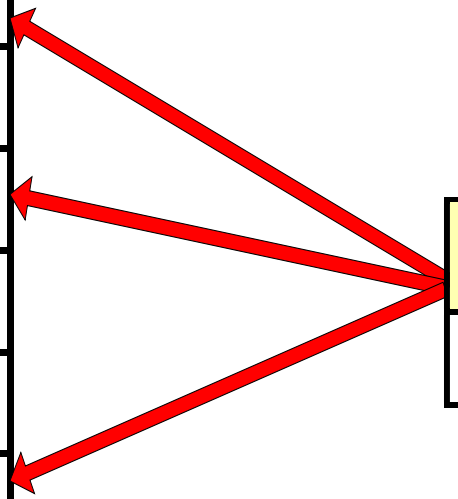
부산대학교 정보·의생명공학대학
정보컴퓨터공학부

Instance-Based Classifier

Set of Stored Cases

Atr1	……	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- Store the training records
- Use training records to predict the class label of unseen cases
- Lazy learning



Unseen Case

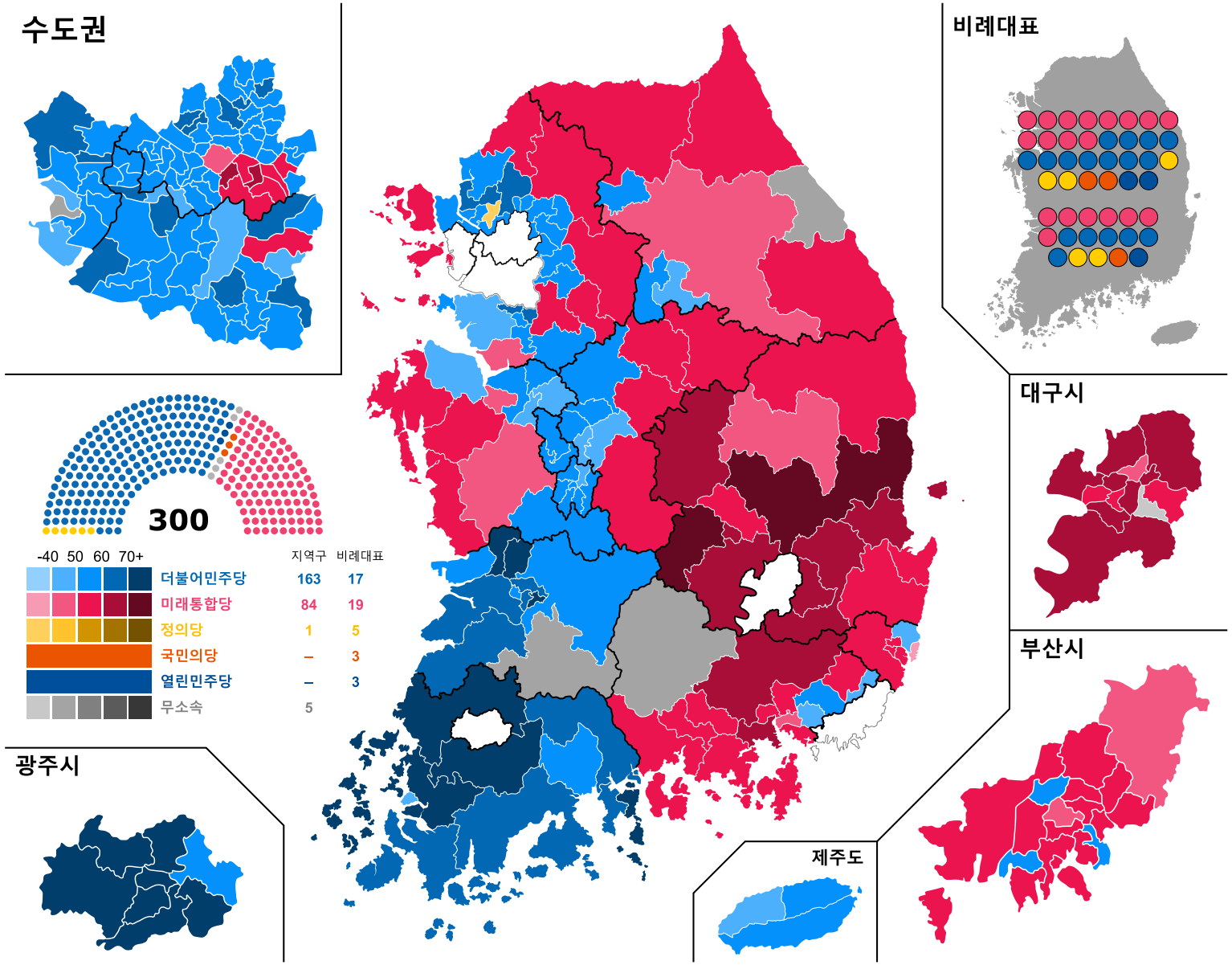
Atr1	……	AtrN

- **Lazy learning** : Simply stores training data (or only minor processing) and waits until it is given a test tuple
- **Eager learning** : Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify

Nearest neighbor

Uses k "closest" points (nearest neighbors) for performing classification

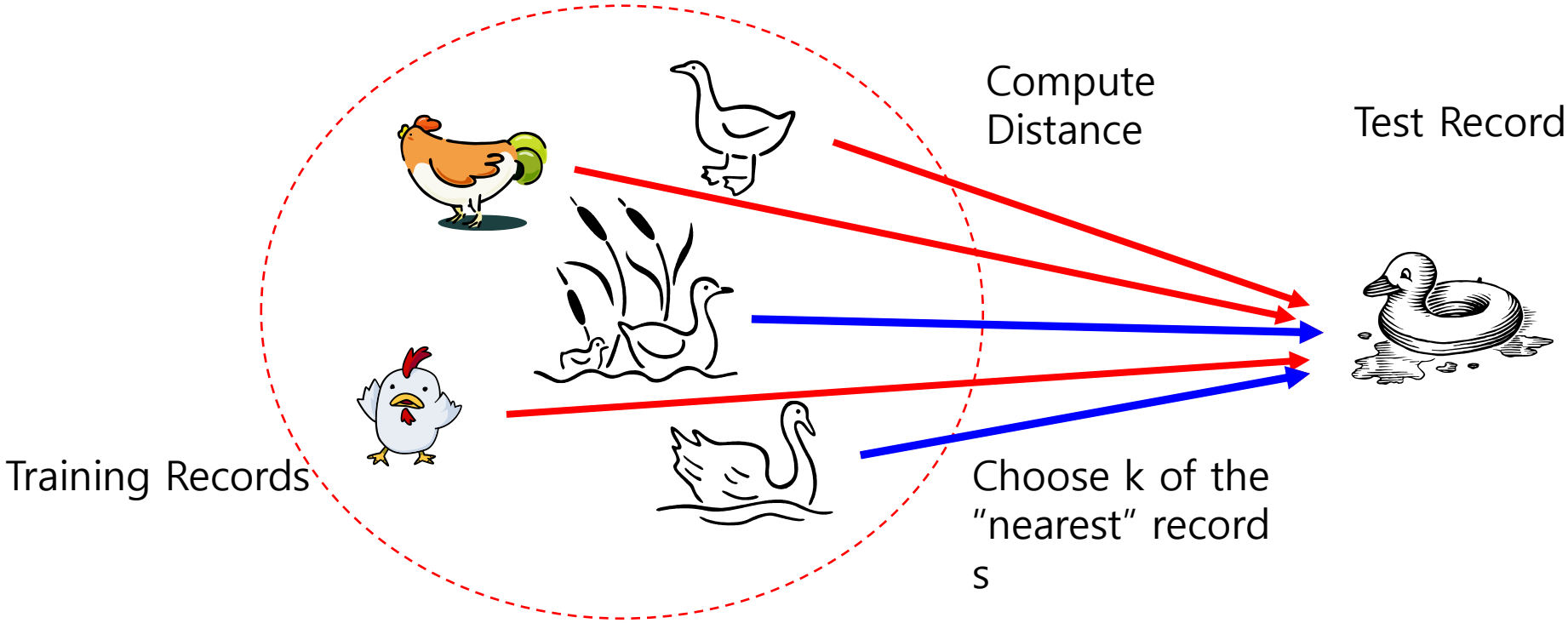
근접 이웃 분류기 (Nearest Neighbors Classifiers)



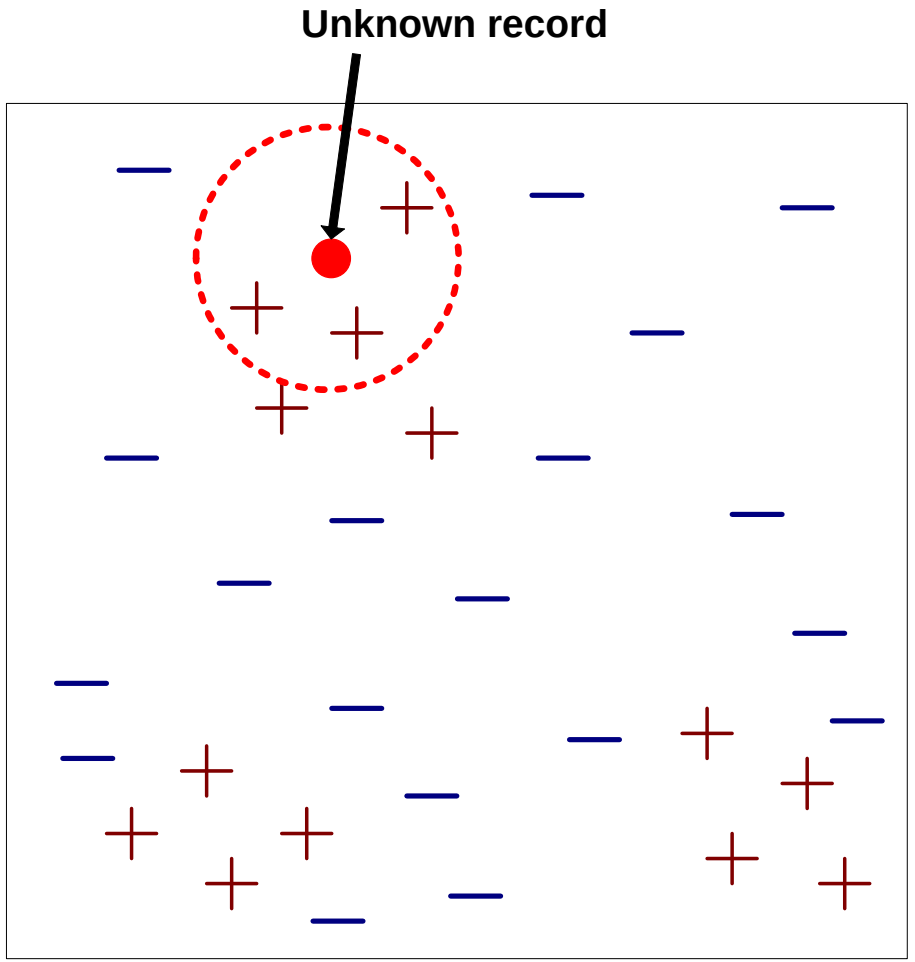
Nearest Neighbor Classifiers

❖ Basic idea:

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



Nearest-Neighbor Classifiers



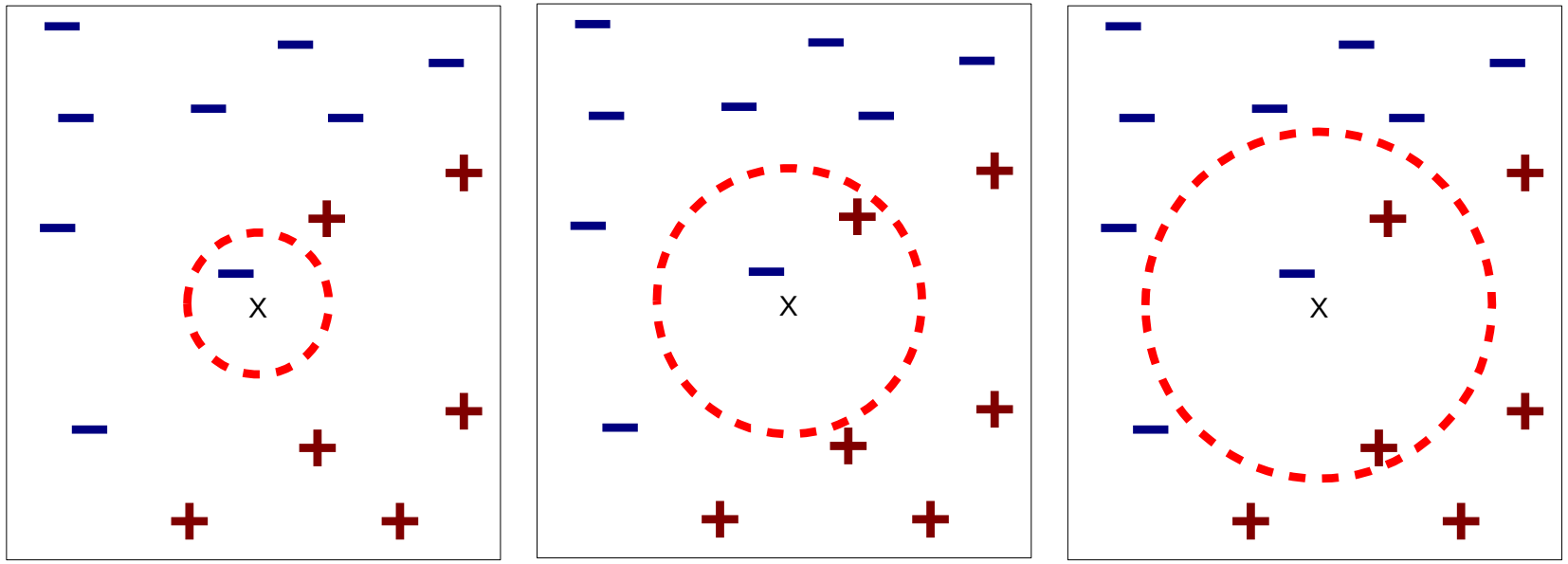
Requires the following:

- The set of stored 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

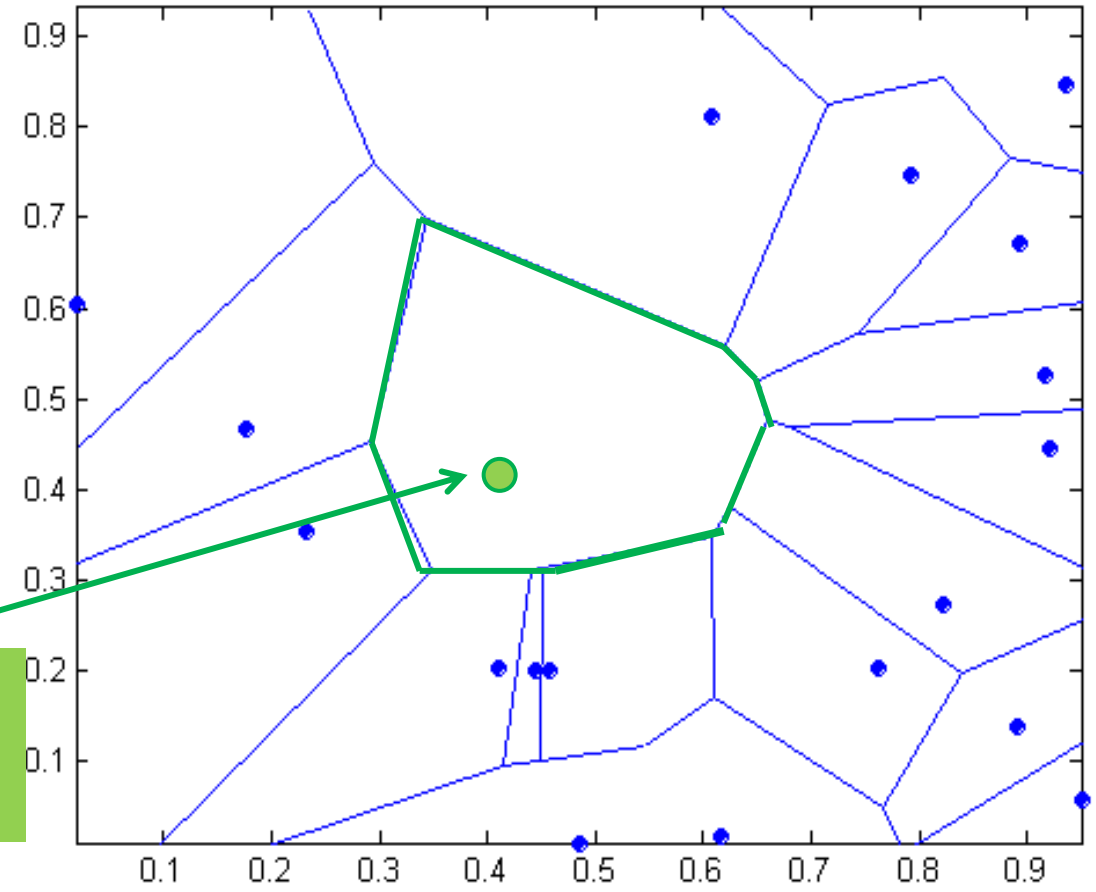


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

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

1 nearest-neighbor

Voronoi Diagram defines the classification boundary



The area takes the class of the green point

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, $w = 1/d^2$

Overview of Applying k-NN

1. Decide on your similarity or distance metric
2. Split the original labeled dataset into training and test data
3. Pick an evaluation metric (e.g., misclassification rate)
4. Run k-NN a few times, changing k and checking the evaluation measure
5. Optimize k by picking the one with the best evaluation measure
6. Apply k-NN to predict unknown labels

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

❖ **Classifier accuracy,** or recognition rate

- Percentage of test set tuples that a re correctly classified

Accuracy = (TP + TN)/All

❖ **Error rate:** *1 – accuracy*, or

Error rate = (FP + FN)/All

Scaling issue

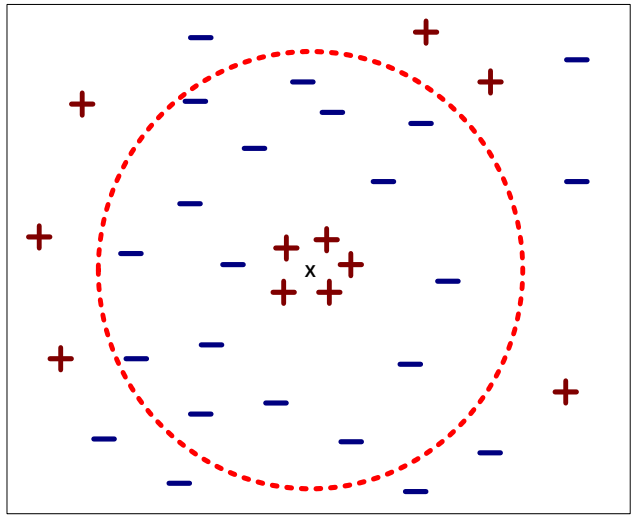
Data preprocessing is often required

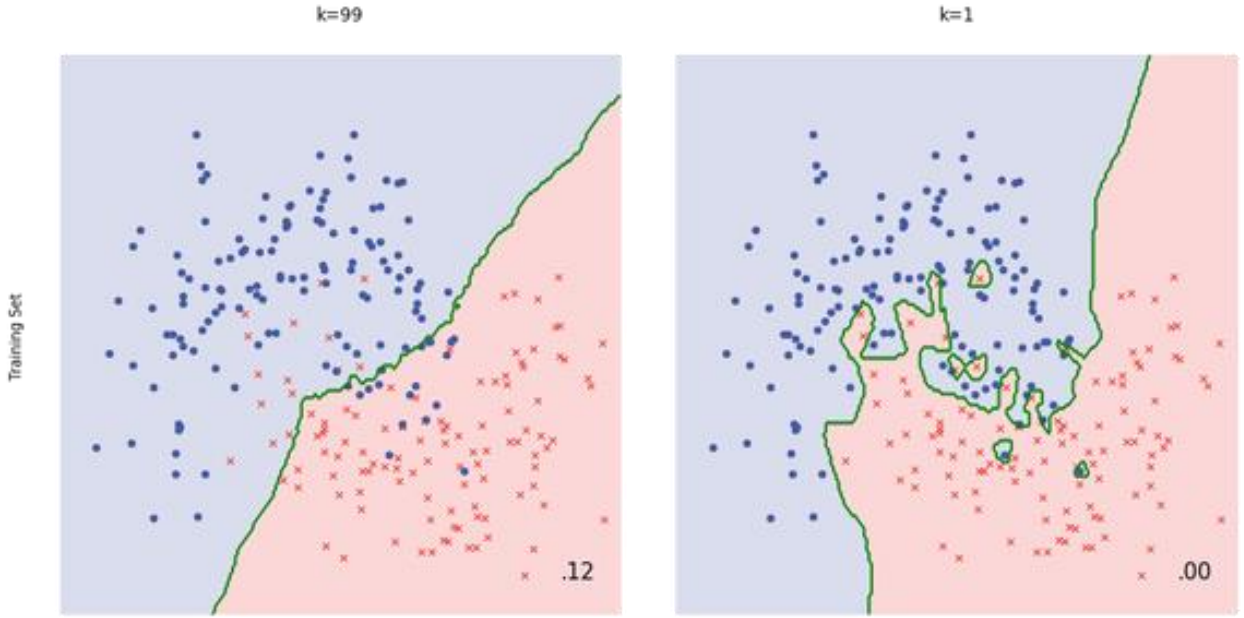
- Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
- Example:
  - height of a person may vary from 1.5m to 1.8m
  - weight of a person may vary from 90lb to 300lb
  - income of a person may vary from $10K to $1M

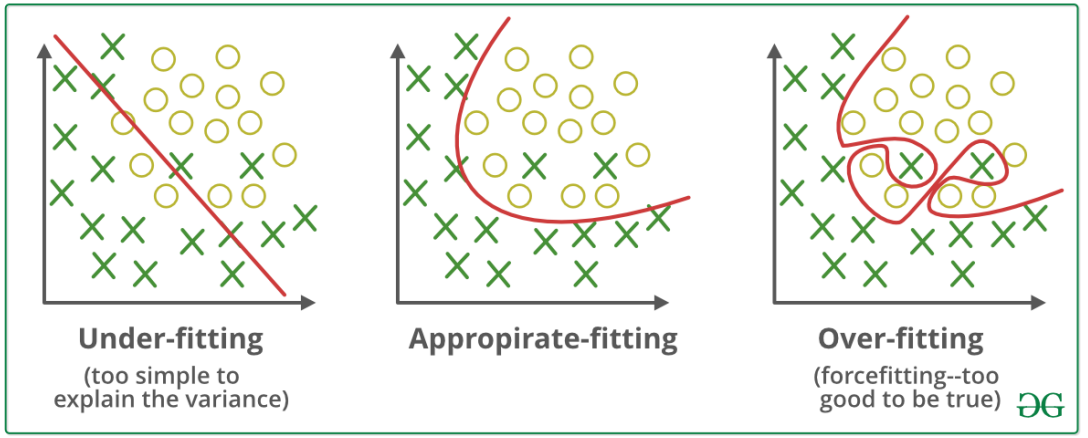
Solution?
- Time series are often standardized to have 0 means a standard deviation of 1
- Mahalanobis Distance

Choosing the value of k

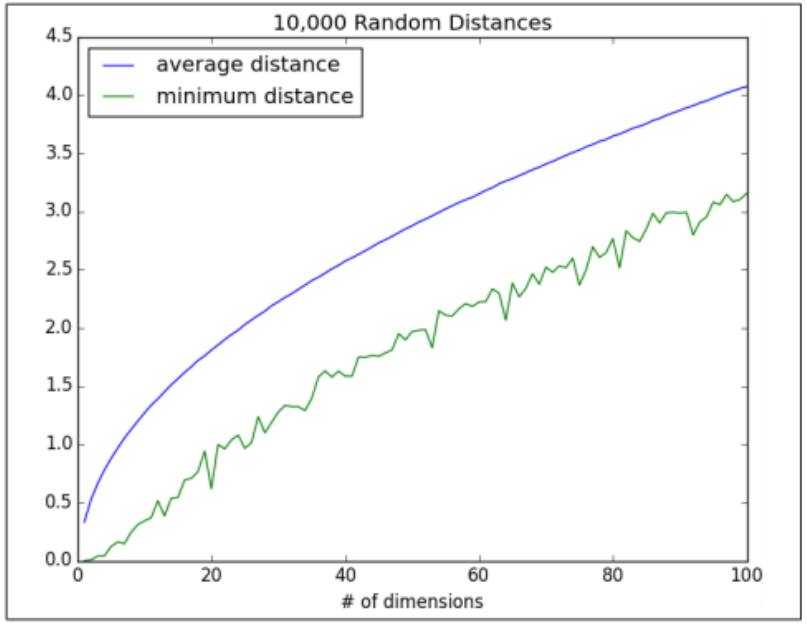
- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

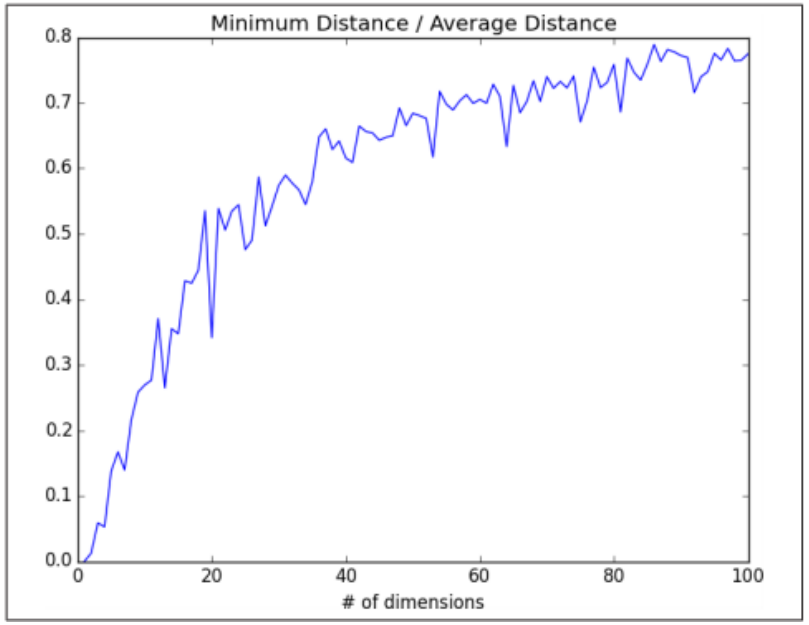






Curse of dimensionality in k-NN





- Low-dimensional datasets: the closest points tend to be much closer than average.
- High-dimensional datasets: two points are close only if they're close in every disemsion.
  - the average distance between points increases
  - the ratio between the closest distance and the average distance.
  - Accumulated noise makes two points far apart.