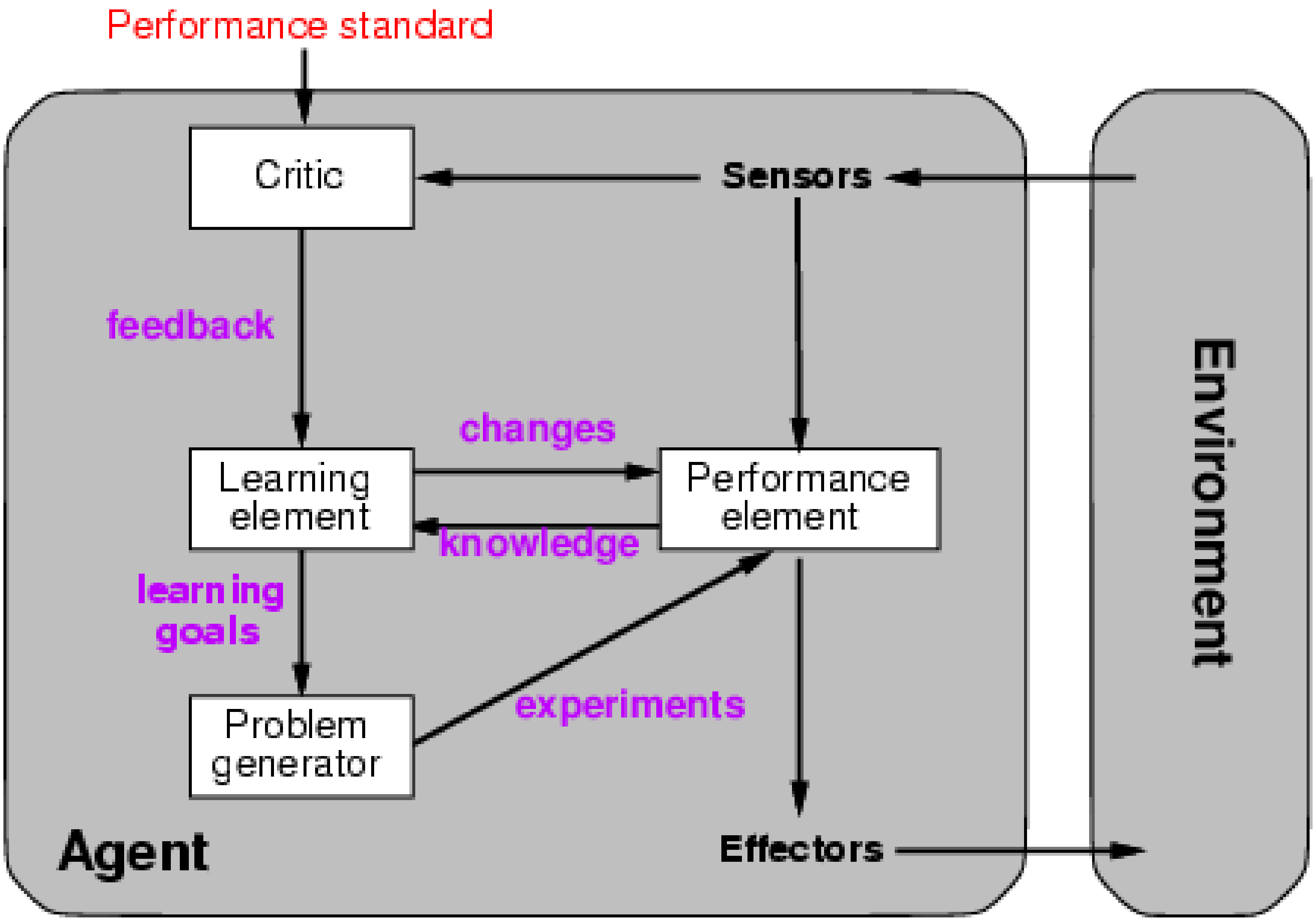
ENCS3340 - Artificial Intelligence

Learning from Observations
Part 1

Learning

- Learning is essential for unknown environments,
  - i.e., when designer lacks omniscience
- Learning is useful as a system construction method,
  - i.e., expose the agent to reality rather than trying to write it down
- Learning modifies the agent's decision mechanisms to improve performance

Learning agents



Learning element

- Design of a learning element is affected by
  - Which components of the performance element are to be learned
  - What feedback is available to learn these components
  - What representation is used for the components

- Type of feedback:
  - Supervised learning: correct answers for each example
  - Unsupervised learning: correct answers not given
  - Reinforcement learning: occasional rewards

ML: Where Used?

- Health:
  - Disease diagnosis:
  - Suicide trends
  - Extracting knowledge form report
  - Recommending stuff to patients

- Finance/Economy:
  - Predicting share prices
  - Credit approval decisions

- Law:
  - Extracting knowledge form report
  - Predicting case outcomes

ML: Where Used?

- Publishing:
  - Predict successful publications/Novels.
  - Detect Plagiarism: determining author of Docs.
  - Document Classification

- Politics:
  - Voter trends and voter influence
  - Selecting potentiall winning candidates

- Security:
  - Detecting security threats
  - Identifying potential intruders based on style

Inductive learning

- Simplest form: learn a function from examples

  f is the target function

  An example is a pair (x, $f(x)$)

  Problem: find a hypothesis *h*

  such that $h \approx f$

  given a training set of examples (table of pair (x, $f(x)$))

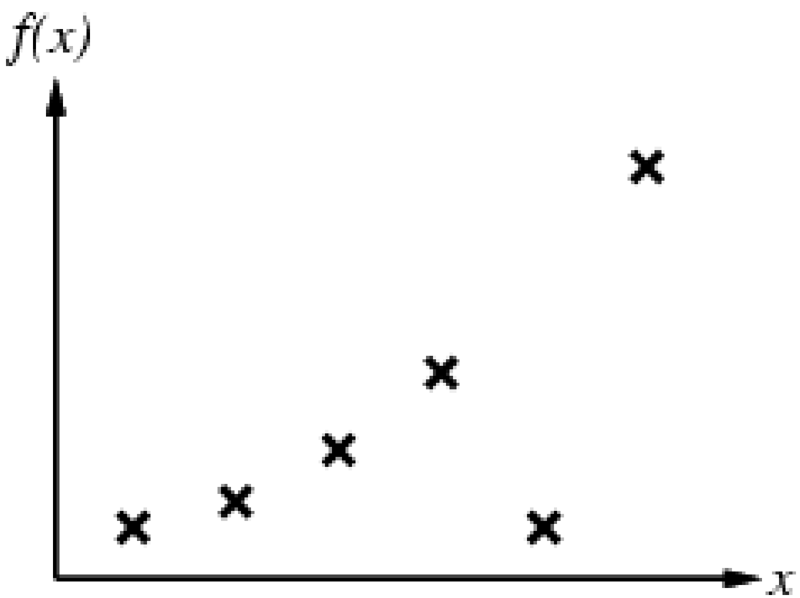
(This is a highly simplified model of real learning:

- Ignores prior knowledge
- Assumes examples are given and are consistent (not conflicting)

Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on **all** examples)

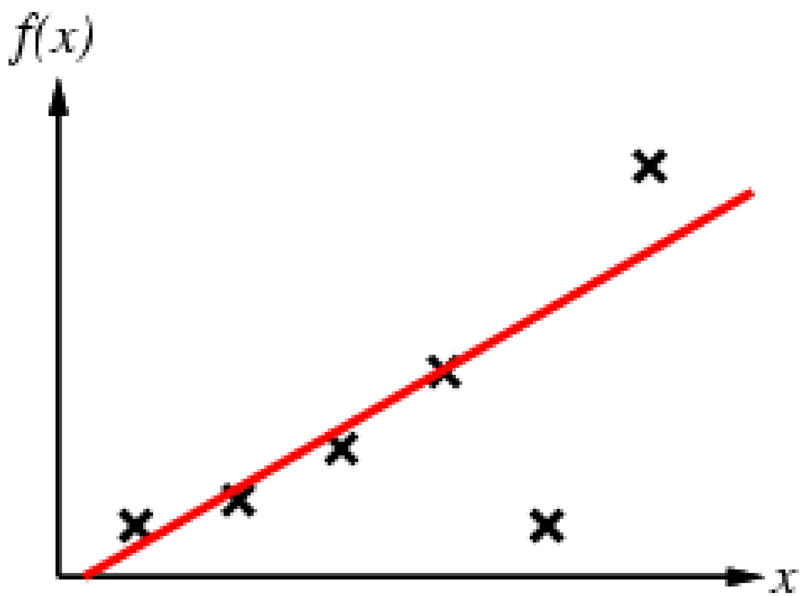
  Too strict: all most/many (error tolerance)
- E.g., curve fitting:



Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on all examples)

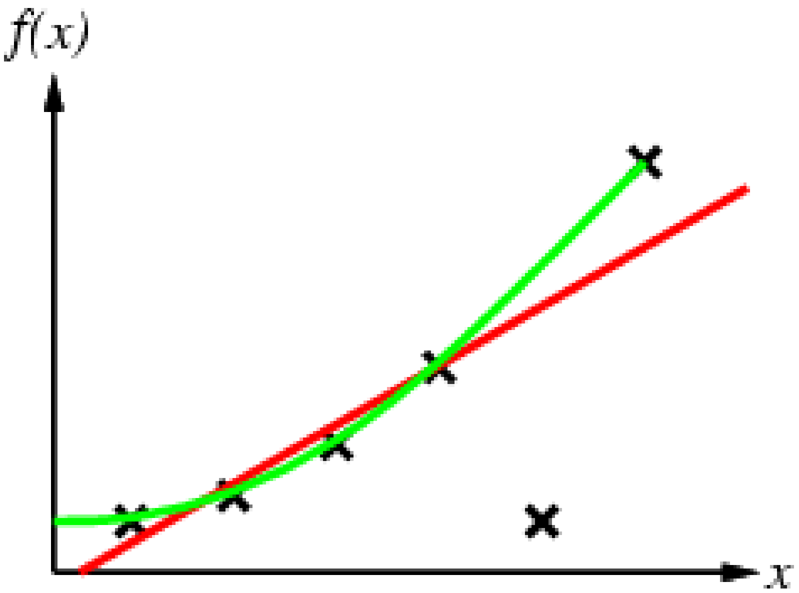
- E.g., curve fitting:



Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on all examples)

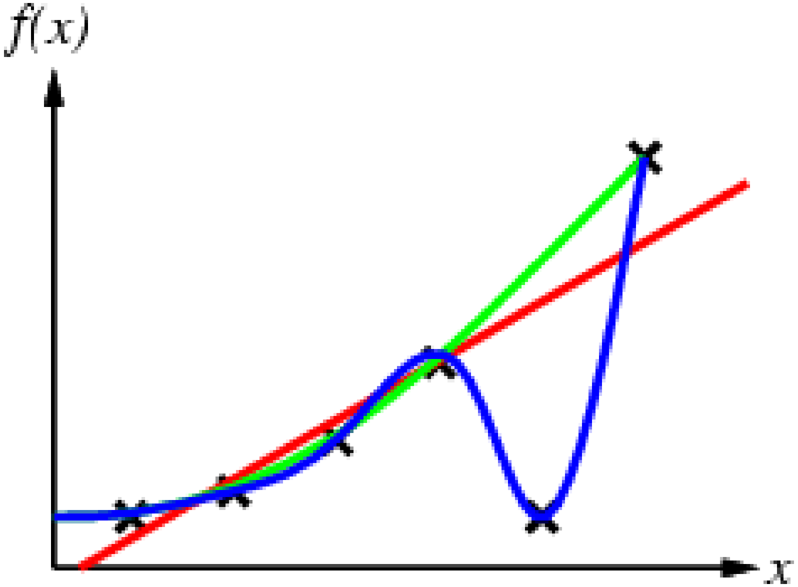
- E.g., curve fitting:



Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on all examples)

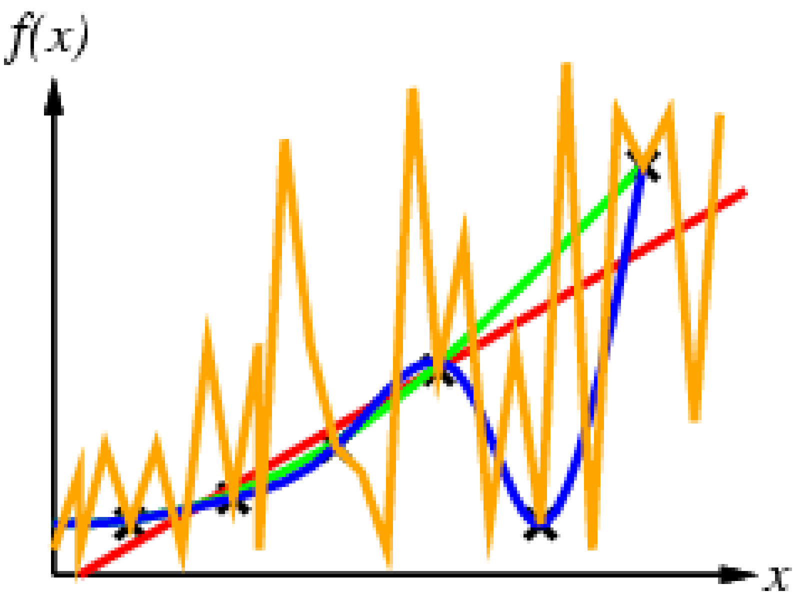
- E.g., curve fitting:



Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on all examples)

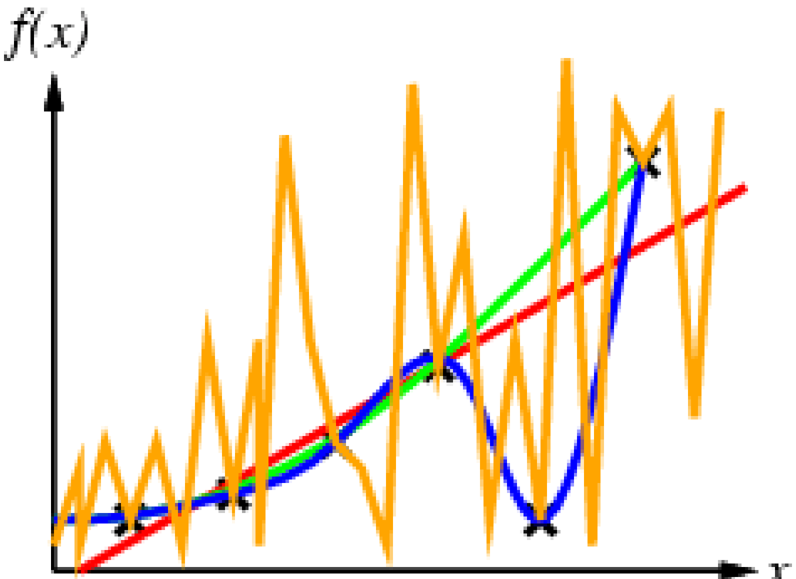
- E.g., curve fitting:



Inductive learning method

- Construct/adjust *h* to agree with *f* on training set
- (*h* is consistent if it agrees with *f* on all examples)

- E.g., curve fitting:



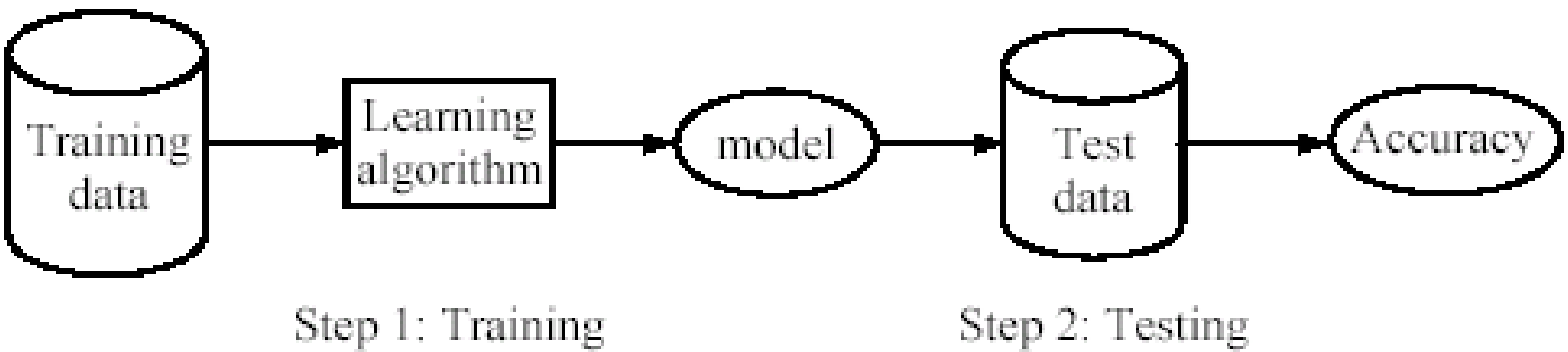
Occam's razor: prefer the simplest hypothesis consistent with data (KIS: Keep It Simple)

Supervised learning process: two steps

Learning (training): Learn a model using the training data

Testing: Test the model using unseen test data to assess the model accuracy

$$Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},$$



What do we mean by learning?

- Given
  - a **data set** *D*,
  - a task *T*, and
  - a performance measure ***M***,

  a computer system is said to **learn** from *D* to perform the task *T* if after learning the system's performance on *T* improves as measured by *M*.

- In other words, the learned model helps the system to perform *T* better as compared to no learning.

Supervised Learning

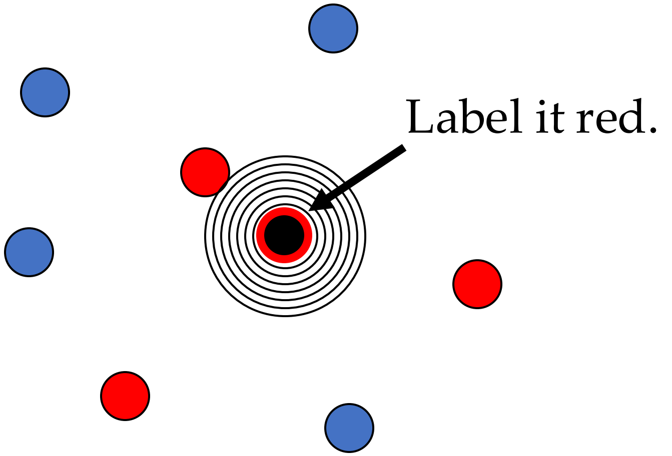
1 - K-Nearest Neighbor Classifier (KNN)

Instance-Based Learning: and KNN

- Idea:
  - Similar examples have similar label.
  - Classify new examples like similar training examples.
- Algorithm:
  - Given a new example x: predict its class y
  - Find most similar training examples
  - Classify *x* “like” these most similar examples
- Questions:
  - How to determine similarity?
  - How many similar training examples to consider?
  - How to resolve inconsistencies in training examples?

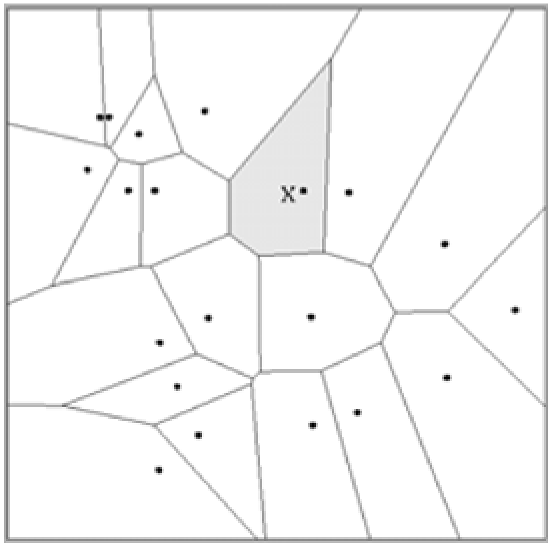
1-Nearest Neighbor

- One of the simplest of all machine learning classifiers
- Simple idea: label a new point the same as the closest known point

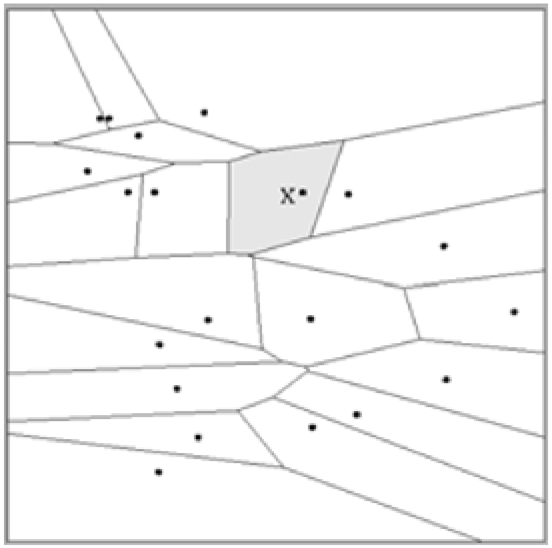


Distance Metrics

- Different metrics can change the decision surface: given points (examples) **a** and **b**



$\text{Dist}(\mathbf{a},\mathbf{b})^2 = (a_1 - b_1)^2 + (a_2 - b_2)^2$



$\text{Dist}(\mathbf{a},\mathbf{b})^2 = (a_1 - b_1)^2 + (3a_2 - 3b_2)^2$

- Standard Euclidean distance metric:
  - Two-dimensional: Dist(a,b) = $\text{sqrt}((a_1 - b_1)^2 + (a_2 - b_2)^2)$
  - Multivariate: Dist(a,b) = $\text{sqrt}(\sum (a_i - b_i)^2)$

Adapted from "Instance-Based Learning" lecture slides by Andrew Moore, CMU.

1-NN's Aspects as an Instance-Based Learner:

A distance metric

- Euclidean (as usual)
  ⇒ D(x1,x2) =number of features on which x1 and x2 differ
- Others (e.g., normal, cosine)

How many nearby neighbors to look at?

- One: 1-NN,

How to fit with the local points?

- Just predict the same output as the nearest neighbor.

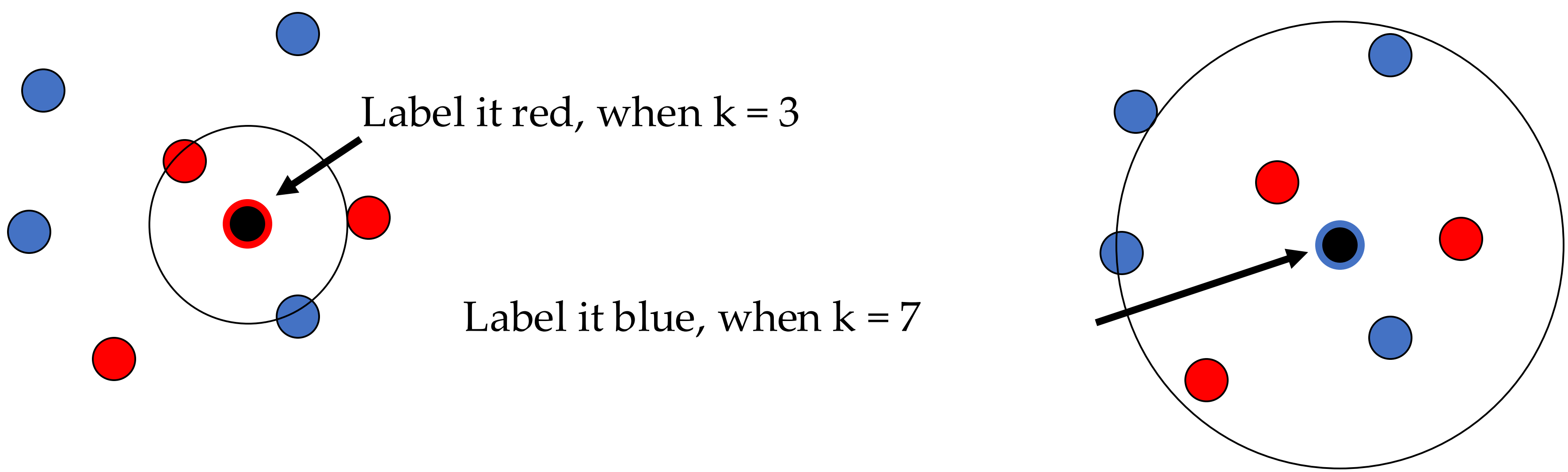
What if this only point is incorrect: Noise?

- Use more points (K), predict based on class of largest number of nearest neighbors.

Adapted from "Instance-Based Learning"
lecture slides by Andrew Moore, CMU.

k – Nearest Neighbor

- Generalizes 1-NN to smooth away noise in the labels
- A new point is now assigned the most frequent label of its *k* nearest neighbors



KNN Example

	Food (3)	Chat (2)	Fast (2)	Price (3)	Bar (2)	BigTip
1	great	yes	yes	normal	no	yes
2	great	no	yes	normal	no	yes
3	mediocre	yes	no	high	no	no
4	great	yes	yes	normal	yes	yes

Similarity metric: Number of matching attributes (k=2)

- New examples:
  - Example 1 (great, no, no, normal, no)? ⇒ Yes/No

    ⇒ most similar: number 2 (1 mismatch, 4 match) ⇒ yes

    ⇒ Second most similar example: number 1 (2 mismatch, 3 match) ⇒ yes

  - Example 2 (mediocre, yes, no, normal, no)? ⇒ Yes/No

    ⇒ Most similar: number 3 (1 mismatch, 4 match) ⇒ no

    ⇒ Second most similar example: number 1 (2 mismatch, 3 match) ⇒ yes

Selecting the Number of Neighbors

- Increase k:
  - Makes KNN less sensitive to noise

- Decrease k:
  - Allows capturing finer structure of space, sensitive to noise.

- Pick k not too large, but not too small (depends on data)

Curse-of-Dimensionality

- Prediction accuracy can quickly degrade when number of attributes grows.
  - **Irrelevant** attributes easily "swamp" information from **relevant** attributes
  - When many irrelevant attributes, similarity/distance measure becomes less reliable
- Remedy
  - Try to remove irrelevant attributes in pre-processing step
  - Weight attributes differently
  - Increase k (but not too much)

Advantages and Disadvantages of KNN

- Need distance/similarity measure and attributes that "match" target function.
- For large training sets,
  - Must make a pass through the entire dataset for each classification. This can be prohibitive for large data sets.
- Prediction accuracy can quickly degrade when number of attributes grows.