Artificial Intelligence for Medicine II

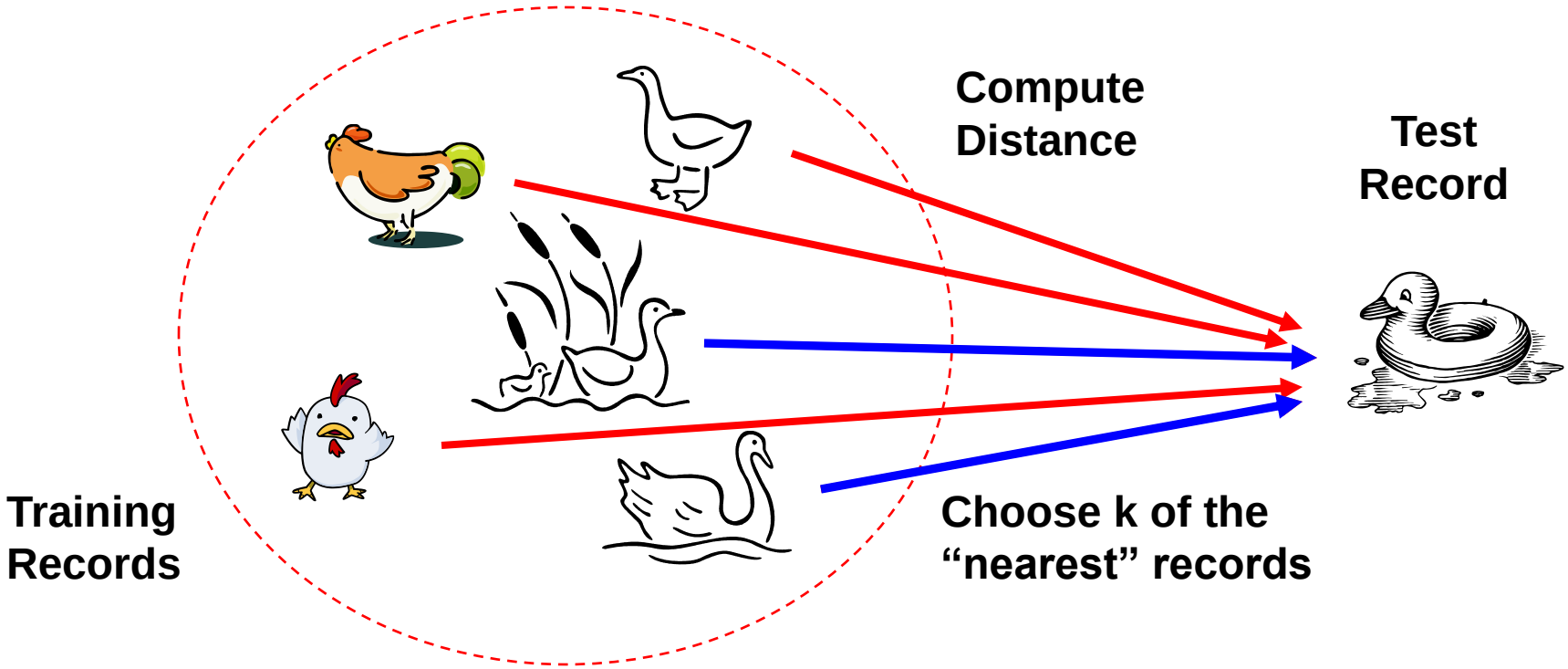
Spring 2025

Lecture 3: Supervised Learning
K-Nearest Neighbor Classifier

(Many slides adapted from Bing Liu, Han, Kamber & Pei; Tan, Steinbach, Kumar
and the web)

Nearest Neighbor Classifiers

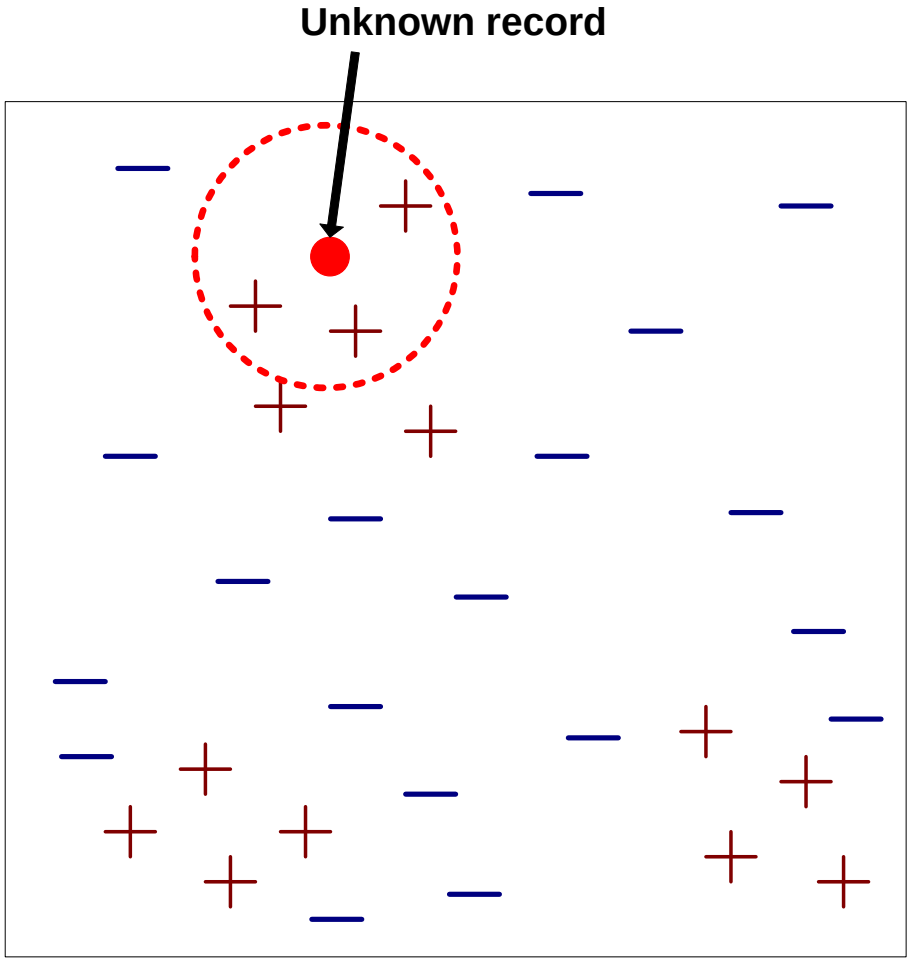
- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



K Nearest Neighbor Classification

- One classification method based on the use of distance measures is K Nearest Neighbors (KNN)
- KNN Steps:
  - **Step 1**: Using a chosen distance metric, compute the distance between the new instance and all past instances.
  - **Step 2**: Choose the *k* past instances that are closest to the new instance.
  - **Step 3**: Work out the predominant class of those *k* nearest neighbors - the predominant class is your prediction for the new instance. i.e. classification is done by *majority vote* of the *k* nearest neighbors.

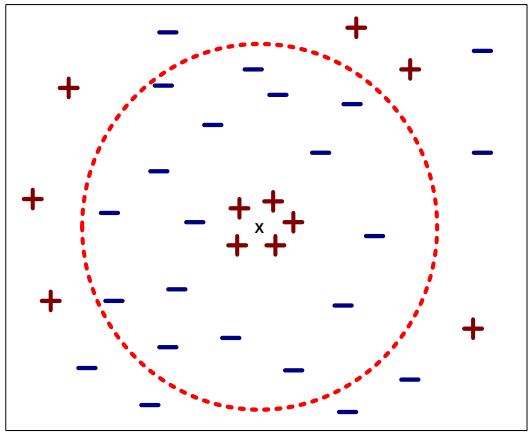
Nearest-Neighbor Classifiers

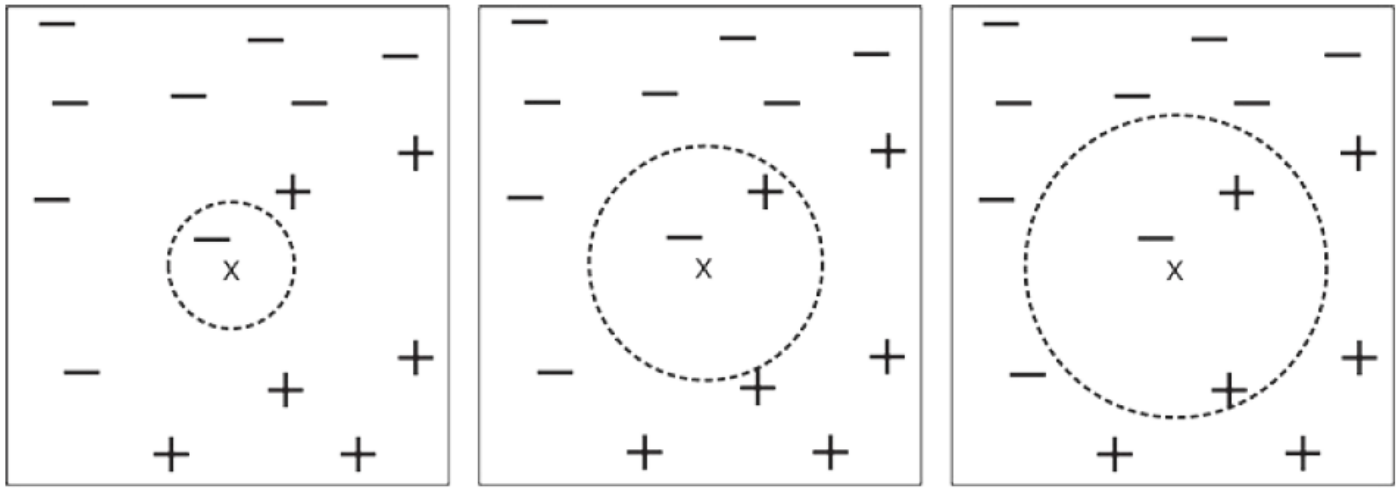


- Requires the following:
  - A set of labeled records
  - Proximity metric to compute distance/similarity between a pair of records
    - e.g., Euclidean distance
  - The value of *k*, the number of nearest neighbors to retrieve
  - A method for using class labels of K nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Nearest Neighbor Classification…

- 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





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

Distance-Based Algorithms

- Place tuples (items) in class to which they are "*closest*".
- Must determine distance/similarity metric between an tuple and a class. How?
  - Distance == dissimilarity (greater value means less similar)
- Tuples are '*similar*' if the values of their attributes are similar.
- *"If it walks like a duck, quacks like a duck, and looks like a duck, then it's probably a duck"*
- The nearest neighbors (*similar items)* are ususally defined in terms of Euclidean distance (or another distance metric)

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Example

- Suppose that you are given the data below. Using a k-nearest classifier, with *k* = *3*, classify the instance

  X = (2, 10, 20)

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

A1	A2	A3	Class
2	12	22	Good
4	14	24	Bad
3	13	21	Good
4	11	25	Bad
2	15	20	Good

Distance-weighted nearest neighbor algorithm

- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query point x_q
    - giving greater weight to closer neighbors
    - Or giving greater weight to important attributes

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} w_i (x_i - y_i)^2}$$

$$w \equiv \frac{1}{d(x_q, y_i)^2}$$

Choice of proximity measure matters

- For documents, cosine is better than correlation or Euclidean

1 1 1 1 1 1 1 1 1 1 1 0		0 0 0 0 0 0 0 0 0 0 0 1
	vs	
0 1 1 1 1 1 1 1 1 1 1 1		1 0 0 0 0 0 0 0 0 0 0 0

Euclidean distance = 1.4142 for both pairs, but the cosine similarity measure has different values for these pairs.

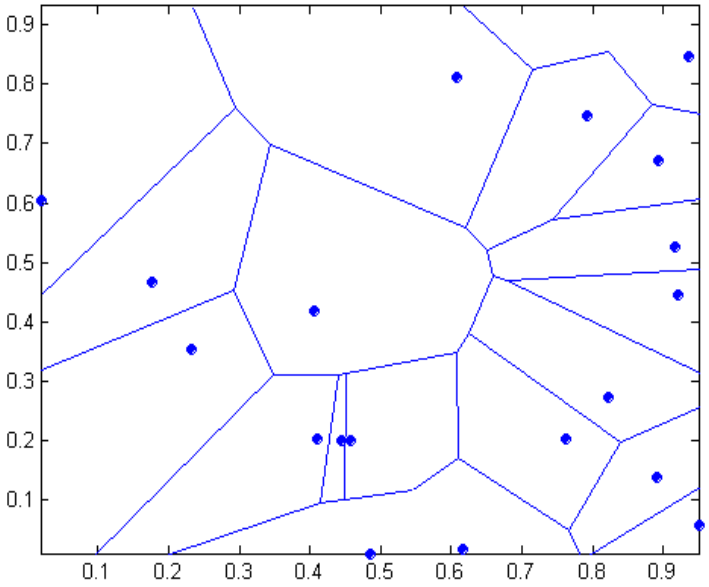
Nearest Neighbor Classification…

- **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
  - Time series are often standardized to have 0 means a standard deviation of 1

Nearest-neighbor classifiers

- Nearest neighbor classification is part of a more general technique known as instance-based learning, which does not build a global model, but rather uses the training examples to make predictions for a test instance.
- Thus, such classifiers are often said to be “model free.
- Such algorithms require a proximity measure to determine the similarity or distance between instances and a classification function that returns the predicted class of a test instance based on its proximity to other instances.
- Nearest neighbor classifiers are local classifiers
- They can produce decision boundaries of arbitrary shapes.

1-nn decision boundary is a Voronoi Diagram



Eager Learners

- In most of the methods we have learnt that classification involves
  - An *inductive step* for constructing classification models from data
  - A *deductive step* for applying the derived model to previously unseen instances
- For decision tree induction and rule based classifiers, the models are constructed immediately after the training set is provided
- Such techniques are known as *eager learners* because they intend to learn the model as soon as possible, once the training data is available

Lazy Learners (Instance-based learning)

• Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified

• Techniques that employ such strategy are known as *lazy learners*

- Typical approaches
  - *k*-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

Nearest Neighbor Classification…

- **How to handle missing values in training and test sets?**
  - Proximity computations normally require the presence of all attributes
  - Some approaches use the subset of attributes present in two instances
    - This may not produce good results since it effectively uses different proximity measures for each pair of instances
    - Thus, proximities are not comparable

K-NN Classificiers…
Handling Irrelevant and Redundant Attributes

- Irrelevant attributes add noise to the proximity measure
- Redundant attributes bias the proximity measure towards certain attributes
- the presence of irrelevant and redundant attributes can adversely affect the performance of nearest neighbor classifiers.

id	issue_d	account_id	day	int_rate	purpose	loan_amnt	loan_status
<entity id>	*<event_time>*	*<foreign key>*	*<partition key>*	*<numerical feature>*	*<categorical feature>*	*<numerical feature>*	*<categorical feature>*
string	*datetime*	*integer*	date	*double*	*string*	*double*	*boolean*
122	2022-01-01	123456	2011-01-01	5.3	debt_consolidation	$142.34	fully_paid
123	2022-04-01	324451	2011-01-02	2.3	wedding	$12.34	charged_off
124	2022-07-01	234232	2011-01-03	3.1	credit_card	$66.29	charged_off
122	2022-10-01	987890	2011-01-04	4.3	debt_consolidation	$112.33	fully_paid

Feature Types

Index Columns

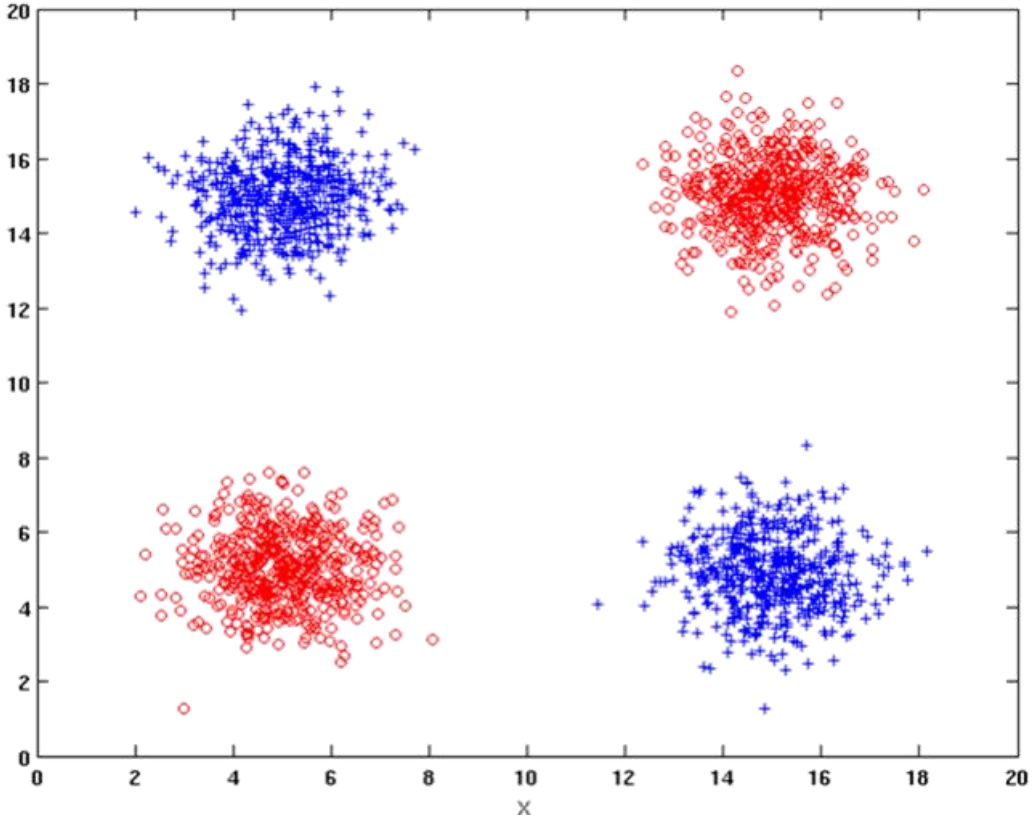
id and *issue_id* uniquely identify each row. *account_id* is a foreign key to an account table, and *day* is used to partition the table.

Feature Data
All the data inside this green box is the feature data. Each column is a feature. Each white cell is a feature value. Each row inside the green box is a feature vector. Together, in totality, it's feature data.

Label
Column used as a target for supervised learning.

K-NN Classifiers: Handling attributes that are interacting

- Nearest neighbor classifiers can handle the presence of interacting attributes, i.e., attributes that have more predictive power taken in combination then by themselves, by using appropriate proximity measures that can incorporate the effects of multiple attributes together.



Improving KNN Efficiency

- Avoid having to compute distance to all objects in the training set
  - Multi-dimensional access methods (k-d trees)
  - Fast approximate similarity search
  - Locality Sensitive Hashing (LSH)
- Condensing
  - Determine a smaller set of objects that give the same performance
- Editing
  - Remove objects to improve efficiency