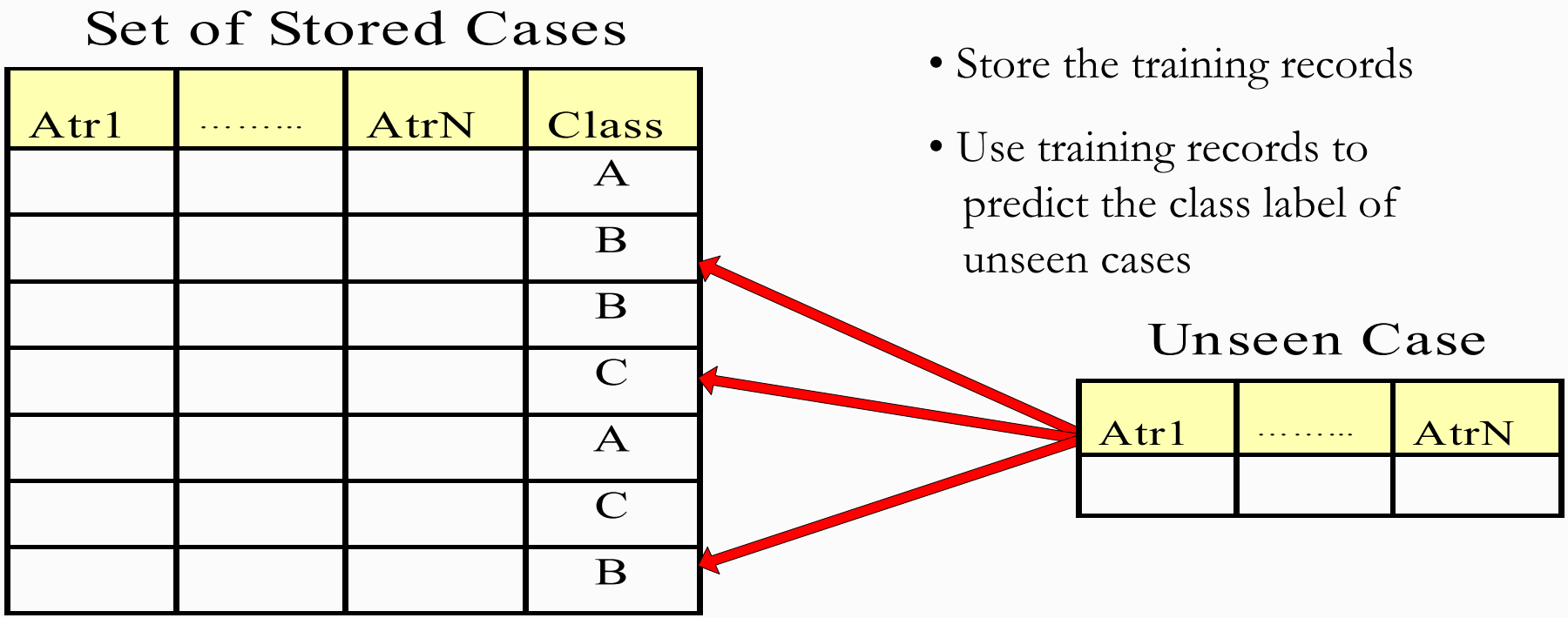
CSE4334/5334 Data Mining

9 Classification: KNN and SVM

Chengkai Li

Department of Computer Science and Engineering
University of Texas at Arlington
Fall 2018 (Slides courtesy of Pang-Ning Tan, Michael Steinbach and Vipin Kumar)

Instance-Based Classifiers



Instance Based Classifiers

Examples:

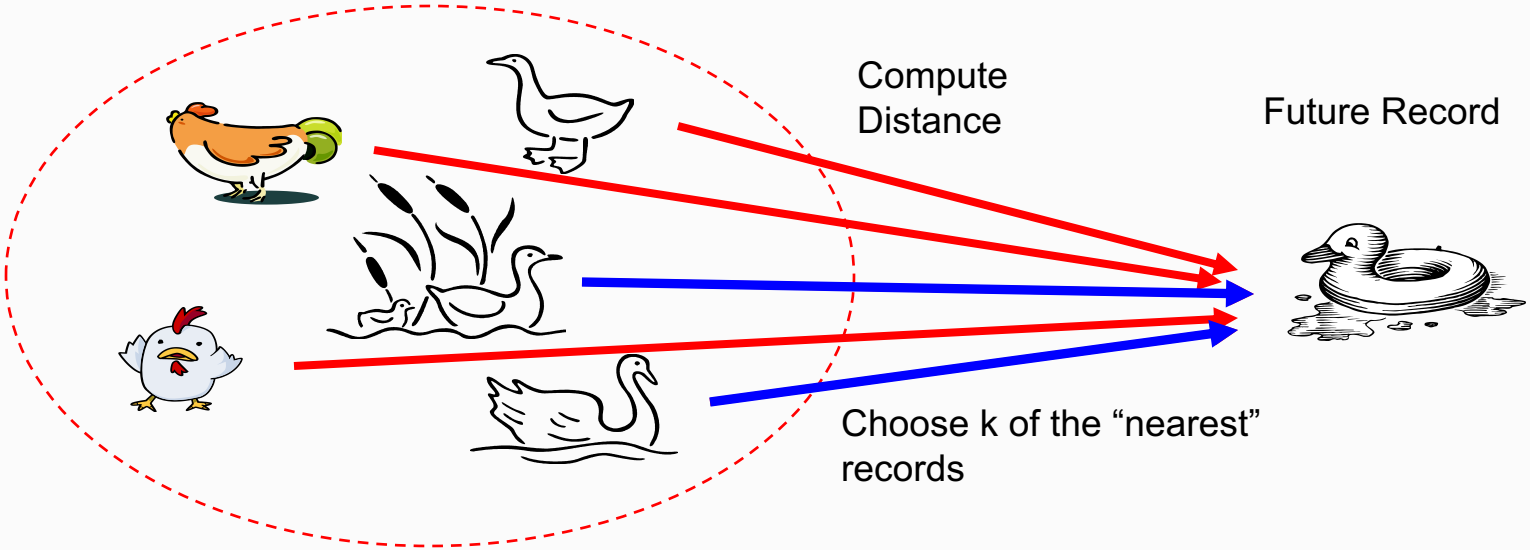
- Rote-learner
  - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly

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

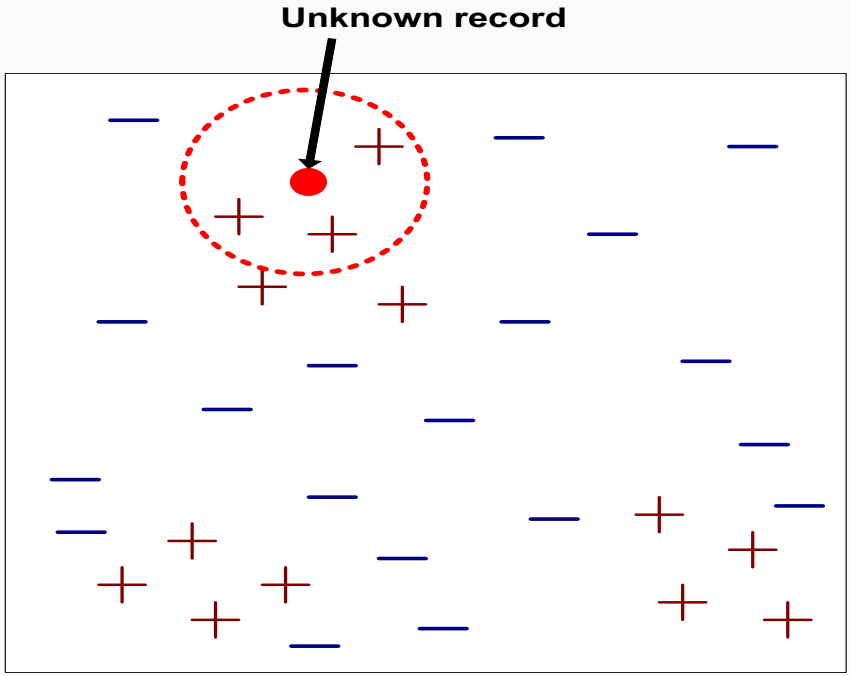
Nearest Neighbor Classifiers

Basic idea:

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



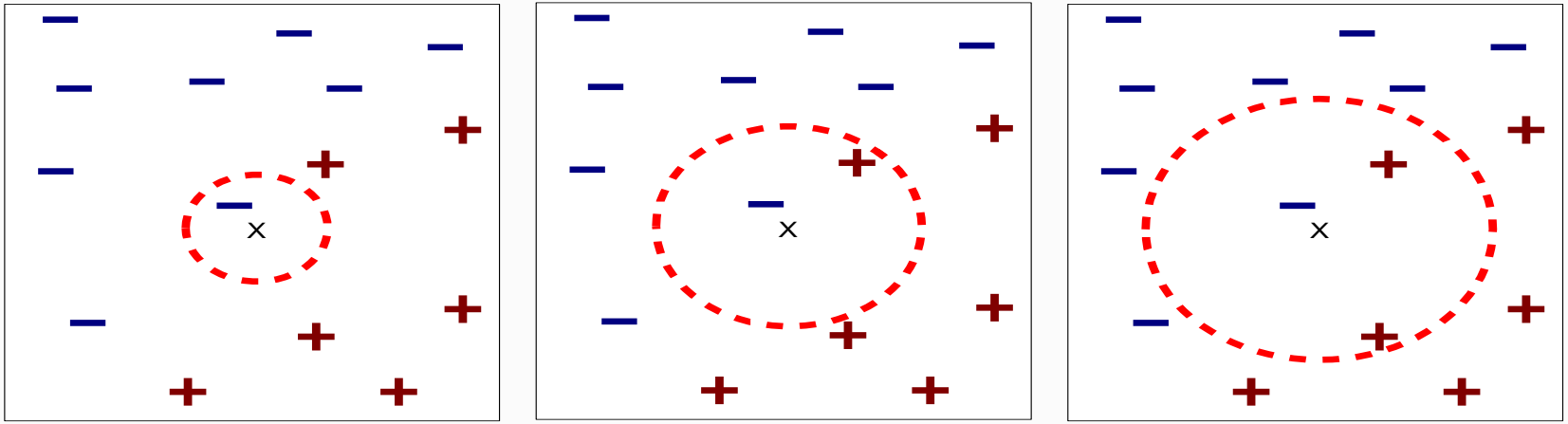
Nearest-Neighbor Classifiers



- Requires three things
  - 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

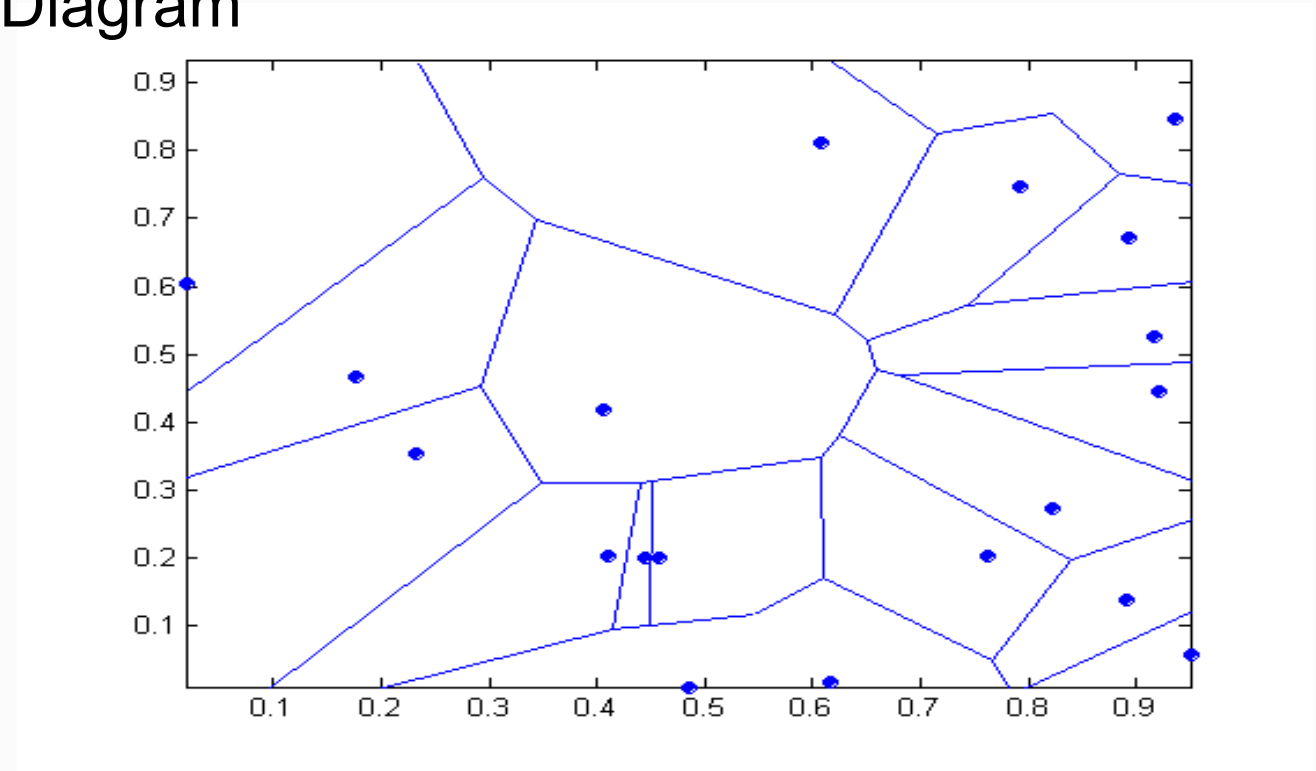


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

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

1 nearest-neighbor

Voronoi Diagram



Nearest Neighbor Classification

Compute distance between two points:

- Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

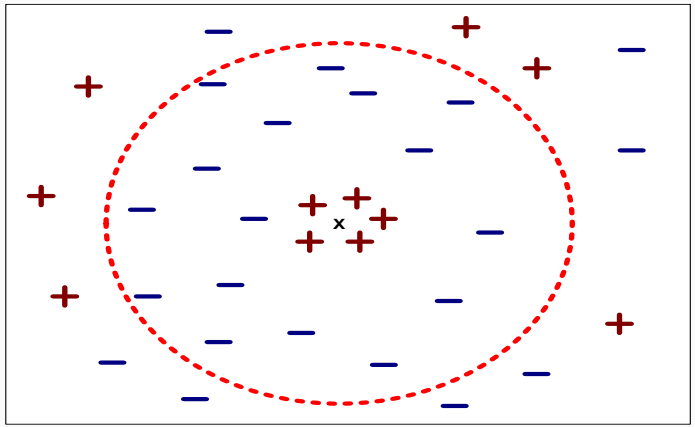
Determine the class from nearest neighbor list

- take the majority vote of class labels among the k-nearest neighbors
- Weigh the vote according to distance
  - weight factor, $w = 1/d^2$

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



Nearest Neighbor Classification…

Scaling issues

- 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

Nearest Neighbor Classification…

Problem with Euclidean measure:

- High dimensional data
  - curse of dimensionality
- Can produce counter-intuitive results

1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

d = 1.4142

vs

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

d = 1.4142

- Solution: Normalize the vectors to unit length

Nearest neighbor Classification…

k-NN classifiers are lazy learners

- It does not build models explicitly
- Unlike eager learners such as decision tree induction and rule-based systems
- Classifying unknown records are relatively expensive

Example: PEBLS

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Distance between nominal attribute values:

d(Single,Married)

= | 2/4 – 0/4 | + | 2/4 – 4/4 | = 1

d(Single,Divorced)

= | 2/4 – 1/2 | + | 2/4 – 1/2 | = 0

d(Married,Divorced)

= | 0/4 – 1/2 | + | 4/4 – 1/2 | = 1

d(Refund=Yes,Refund=No)

= | 0/3 – 3/7 | + | 3/3 – 4/7 | = 6/7

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

Class	Refund	
	Yes	No
Yes	0	3
No	3	4

$$d(V_1, V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

Example: PEBLS

Tid	Refund	Marital Status	Taxable Income	Cheat
X	Yes	Single	125K	No
Y	No	Married	100K	No

Distance between record X and record Y:

$$\Delta(X, Y) = w_X w_Y \sum_{i=1}^{d} d(X_i, Y_i)^2$$

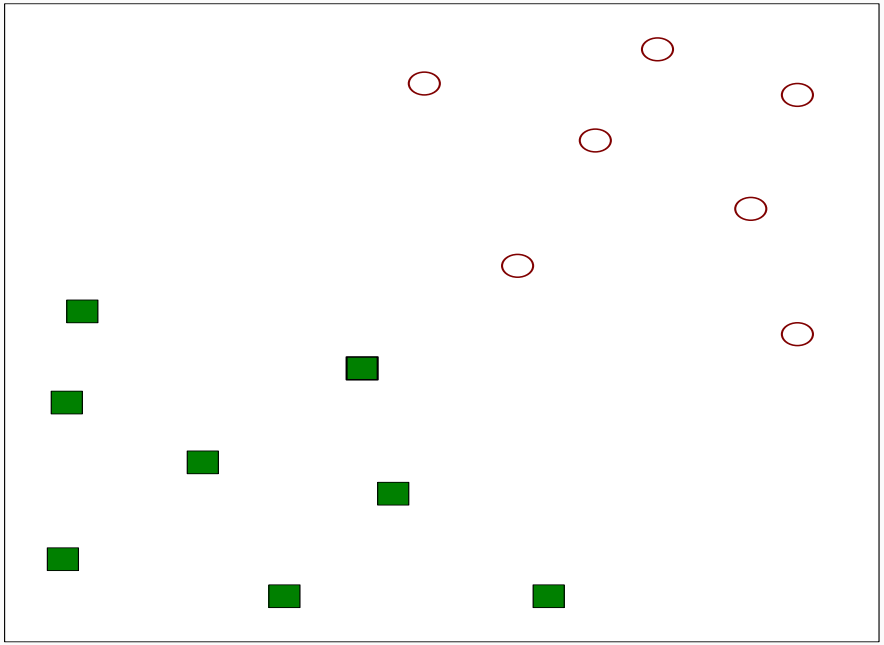
where: $$w_X = \frac{\text{Number of times X is used for prediction}}{\text{Number of times X predicts correctly}}$$

$w_X \cong 1$ if X makes accurate prediction most of the time

$w_X > 1$ if X is not reliable for making predictions

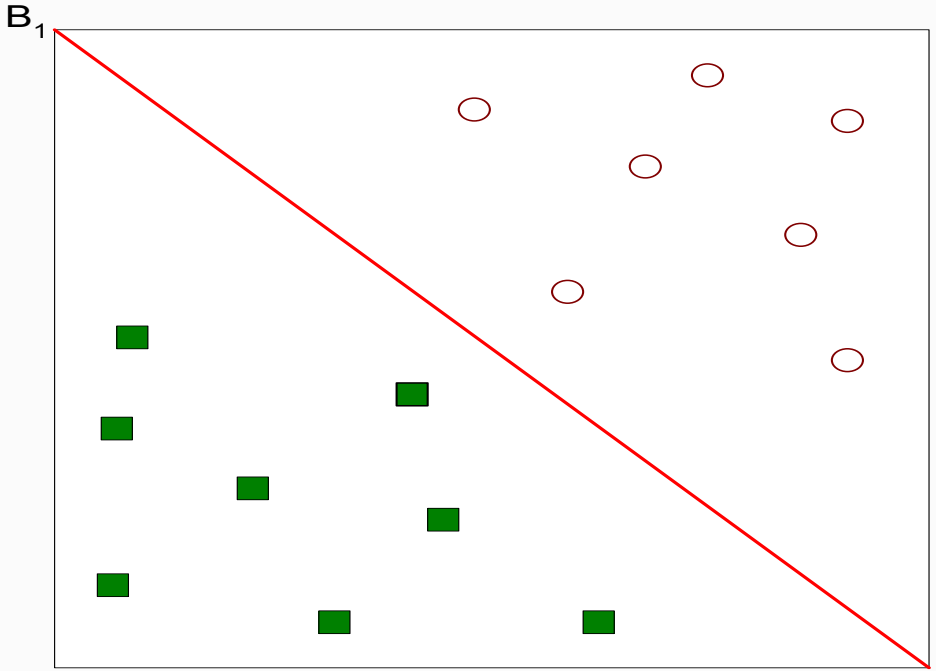
Support Vector Machines

Support Vector Machines



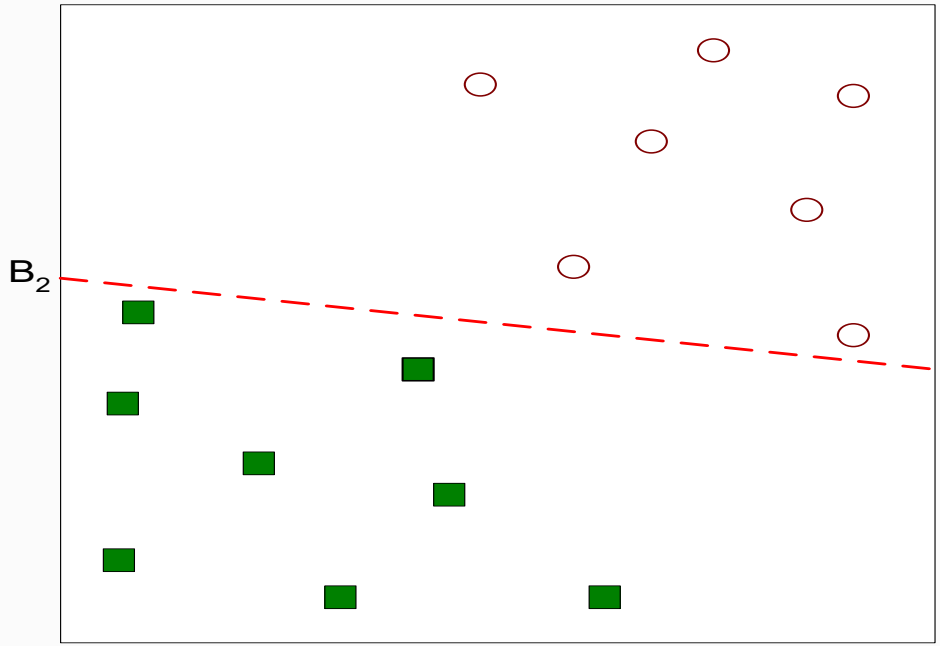
Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



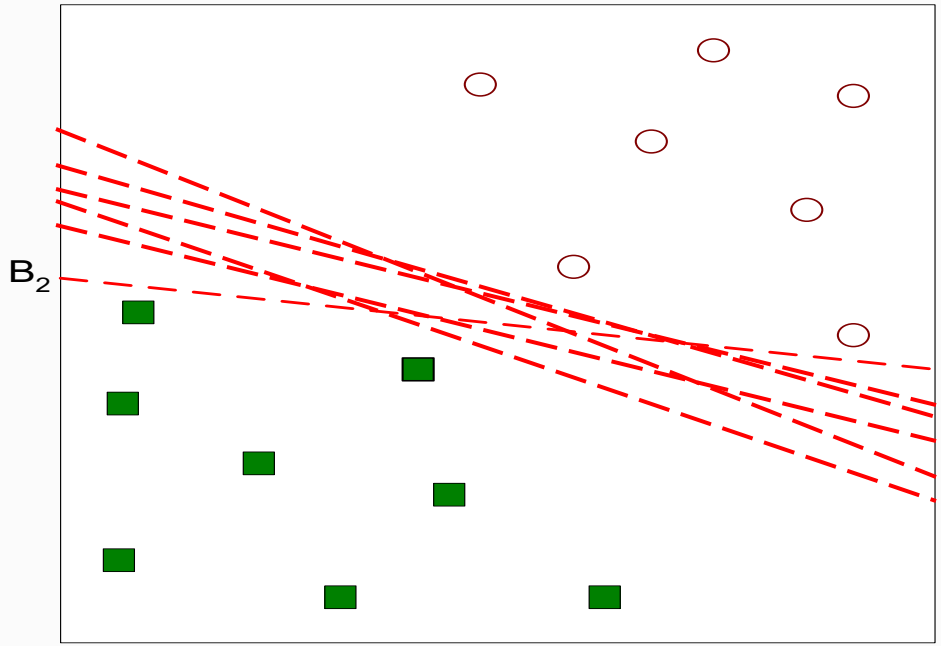
One Possible Solution

Support Vector Machines



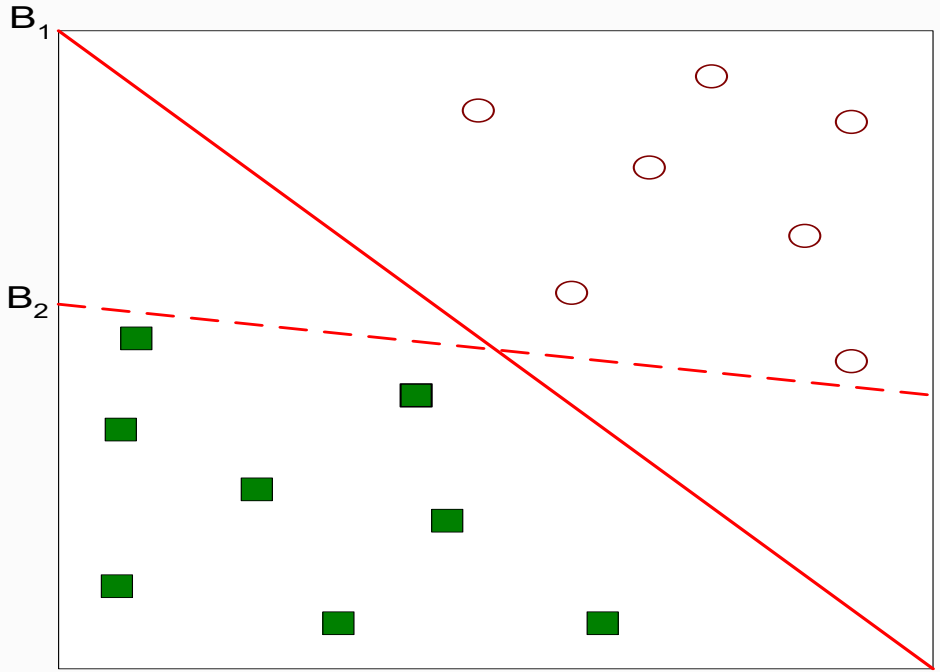
Another possible solution

Support Vector Machines



Other possible solutions

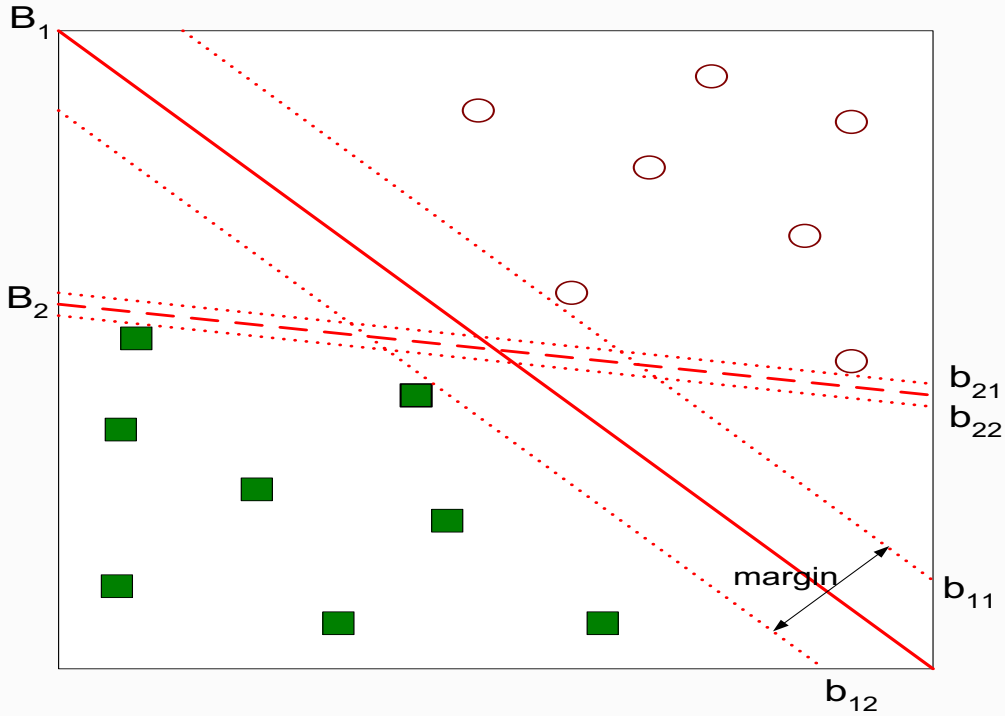
Support Vector Machines



Which one is better? B1 or B2?

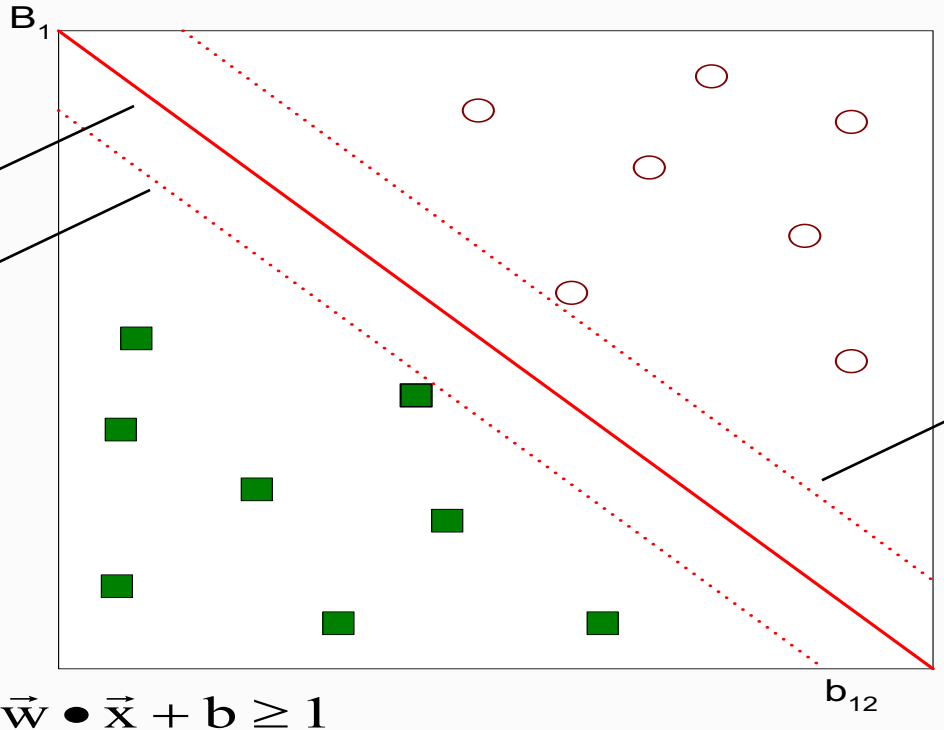
How do you define better?

Support Vector Machines



Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$\vec{w} \bullet \vec{x} + b = 0$

$\vec{w} \bullet \vec{x} + b = -1$

$\vec{w} \bullet \vec{x} + b = +1$

$$\text{Margin} = \frac{2}{\| \vec{w} \|}$$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

Support Vector Machines

We want to maximize: $$\text{Margin} = \frac{2}{\| \vec{w} \|}$$

- Which is equivalent to minimizing: $$L(w) = \frac{\| \vec{w} \|^2}{2}$$

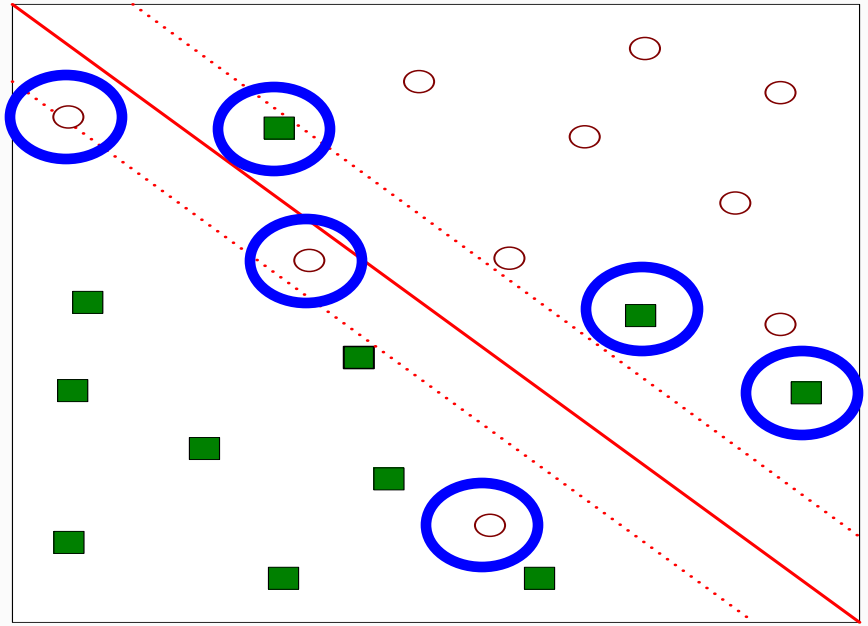
- But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

  - This is a constrained optimization problem
    - Numerical approaches to solve it (e.g., quadratic programming)

Support Vector Machines

What if the problem is not linearly separable?



Support Vector Machines

What if the problem is not linearly separable?

- Introduce slack variables
  - Need to minimize:

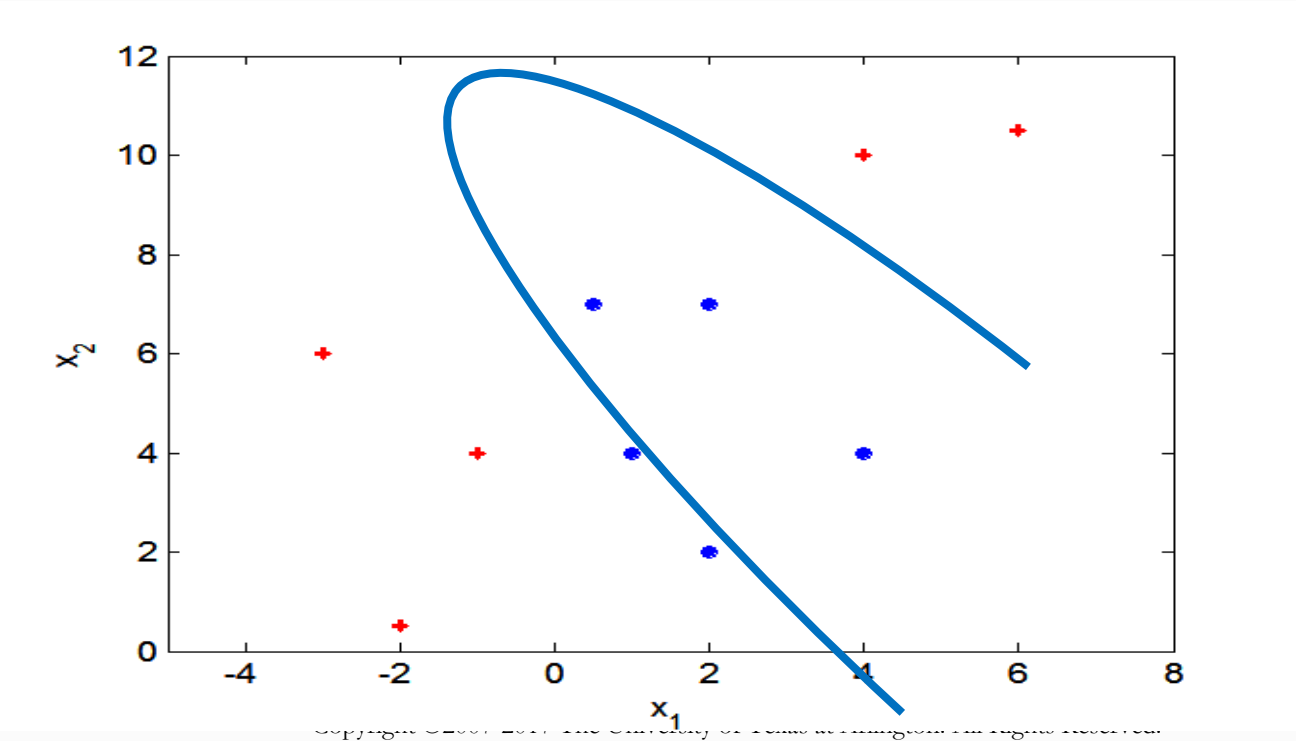
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

  - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$

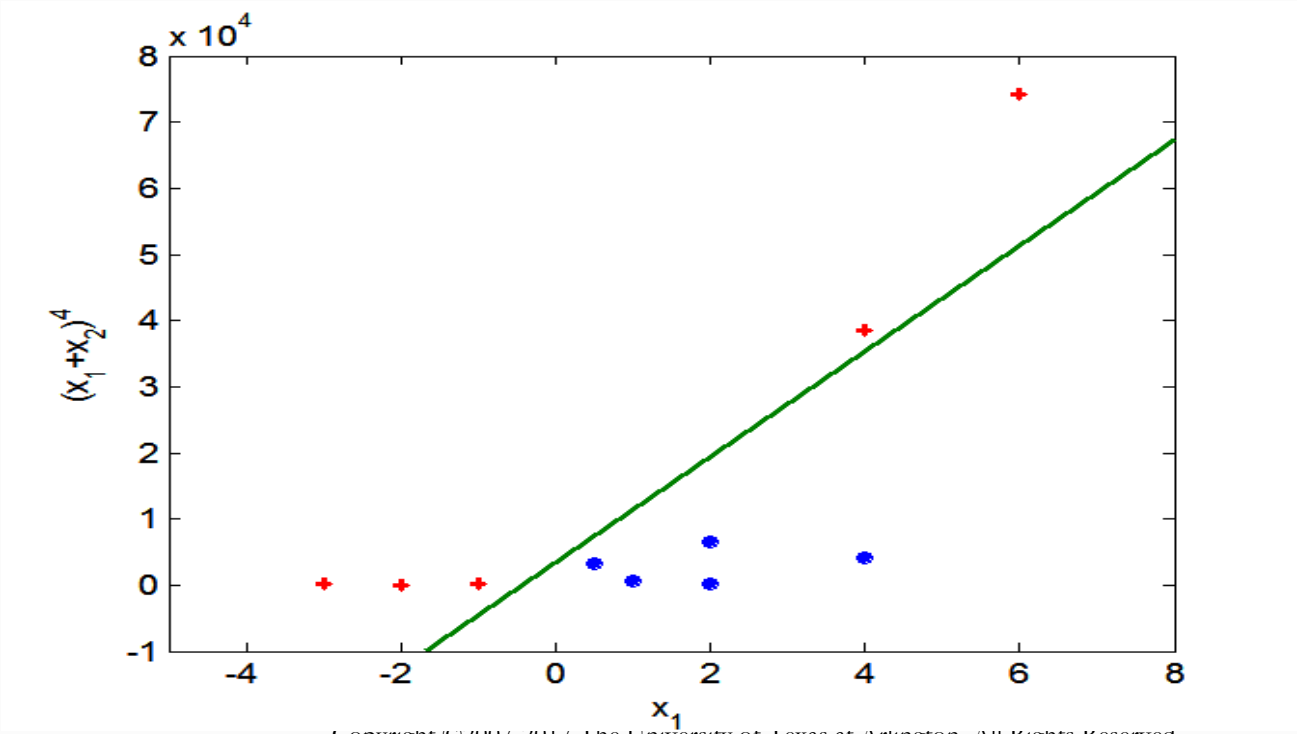
Nonlinear Support Vector Machines

What if decision boundary is not linear?



Nonlinear Support Vector Machines

Transform data into higher dimensional space



Exercise

Consider the following 5 points in a 2-dimension space (x, y): (-2,0), (0,-1) (0,0), (0,1), (2,1). Of these, (0,1) and (2,1) belong to class C1 and (-2,0) (0,-1), and (0,0) belong to class C2. Use support vector machine to classify. Provide the support vectors, the decision boundary, and the maximized margin.