CSX4202_ITX4202: Data Mining
Lecture 5

Asst. Prof. Dr. Rachsuda Setthawong
Computer Science Department
Assumption University

Outlines

- Instance-based classifier
- Naïve Bayes classifier
- Rule-based classifier
- Ensemble classifier
- ANN classifier
- SVM classifier

Asst. Prof. Dr. Rachsuda Setthawong

Instance-Based Classifiers

- How does it work?
- Characteristics of the algorithm?
- How to construct a model? (algorithms)
- Improvement? – Distance Weighted Voting
- Advantages / Disadvantages?

Instance-Based Classifiers

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

Set of Stored Cases

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

Unseen Case

Atr1	……...	AtrN

Asst. Prof. Dr. Rachsuda Setthawong

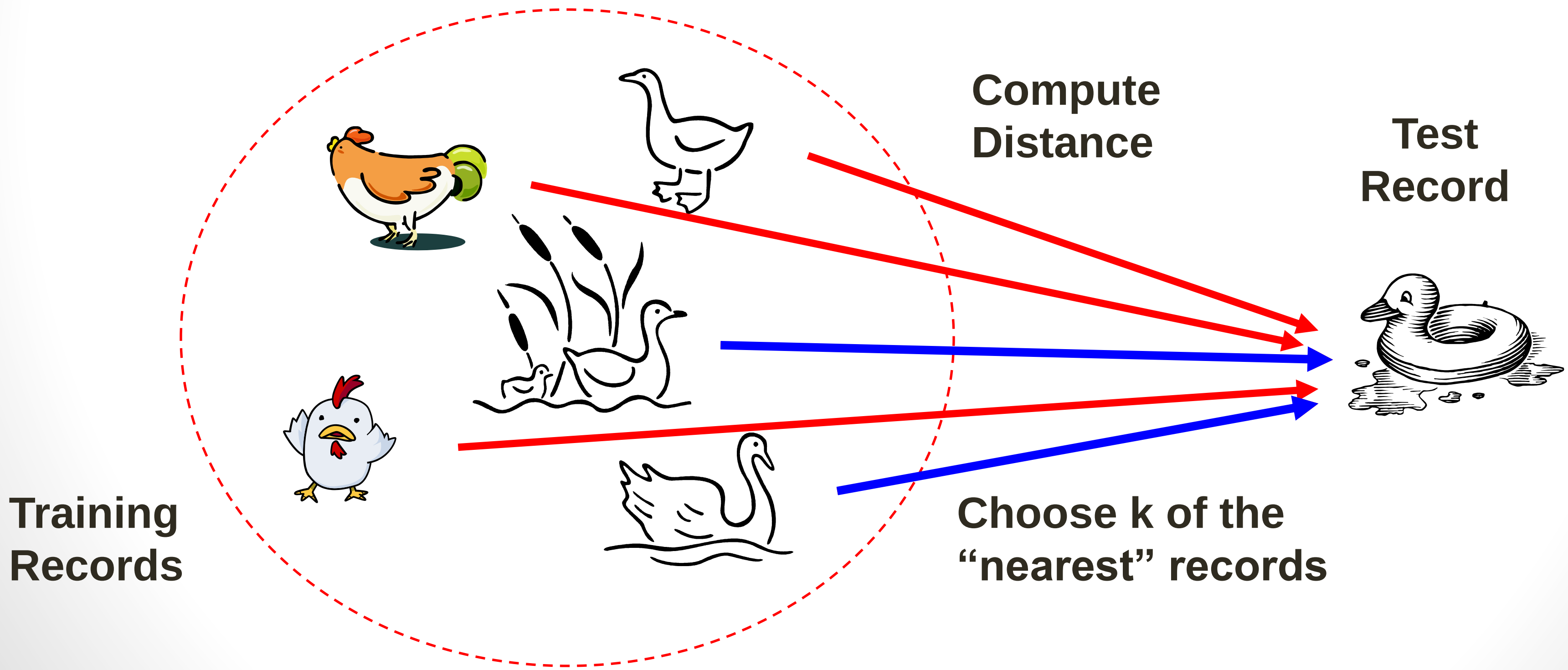
Examples of Instance Based Classifiers

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

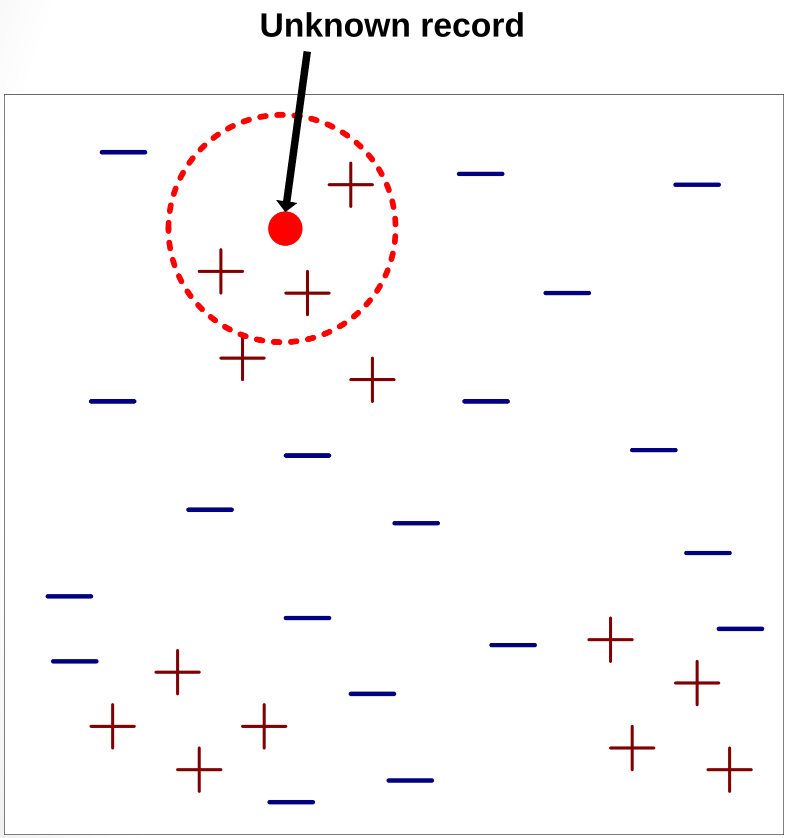
- Nearest neighbor
  - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

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



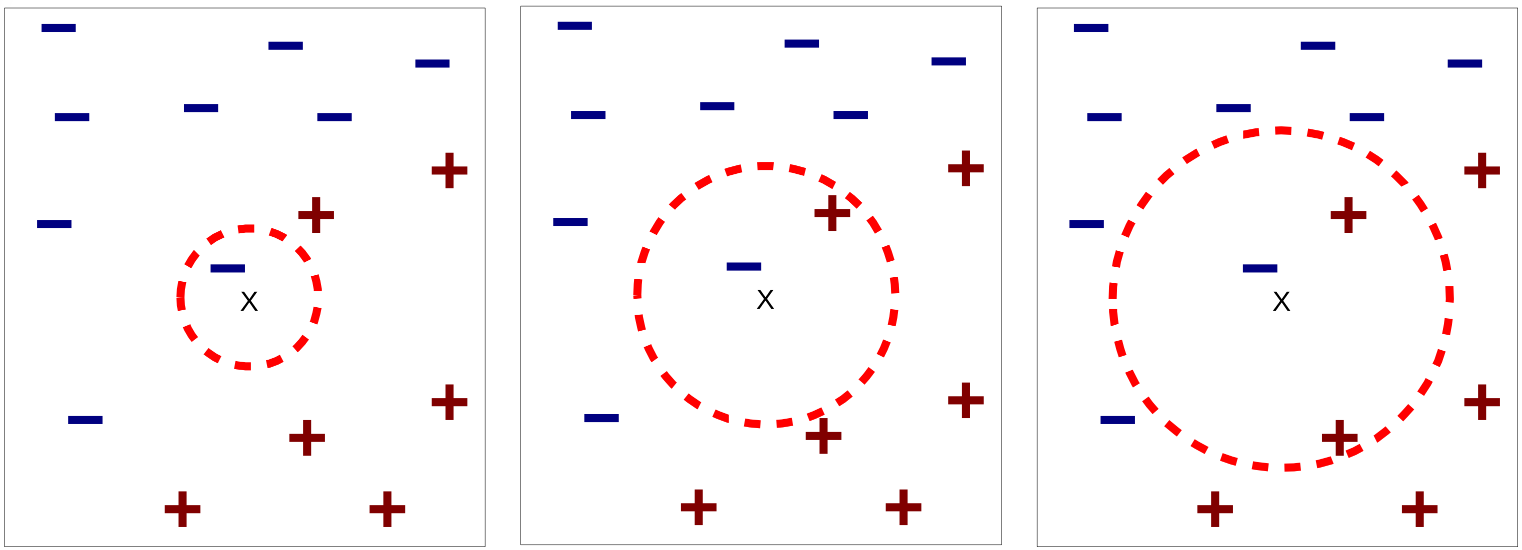
Nearest-Neighbor Classifiers



- Entities required:
  - A data set (records)
  - Distance Metric
  - The number of nearest neighbors (k)

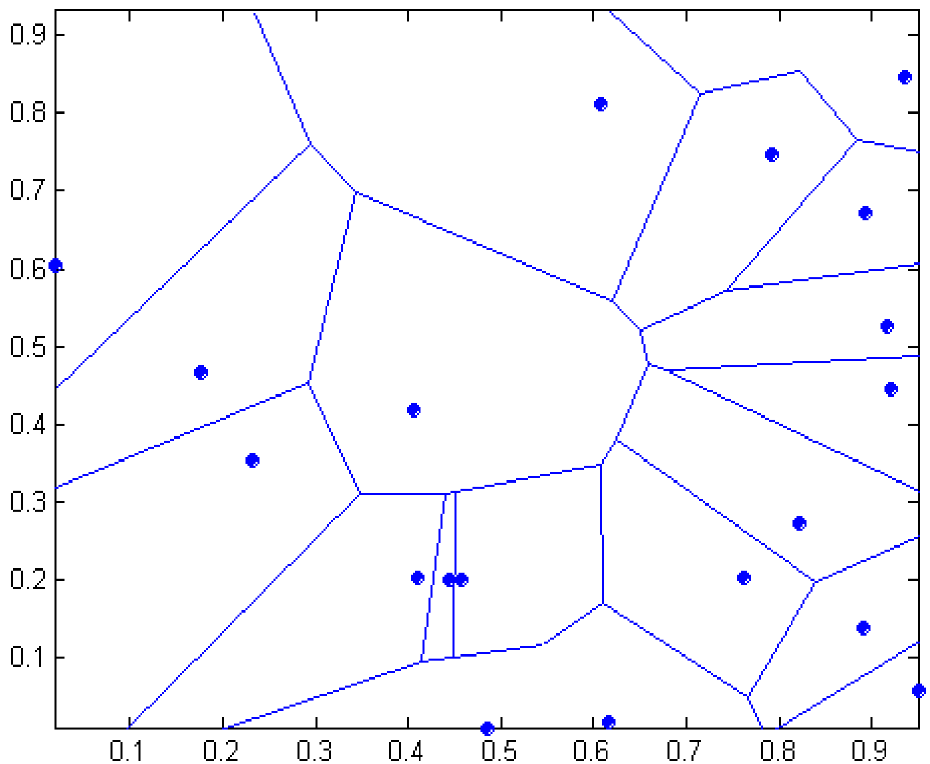
- To classify an unknown record:
  1. Compute distance to other training records
  2. Identify *k* nearest neighbors
  3. Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor



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

Decision Boundary: 1 Nearest-Neighbor



Voronoi Diagram

Nearest Neighbor Classification

- Compute distance between two points (x and x') :
  - Euclidean distance

$$d(x, x') = \sqrt{\sum_i (x_i - x'_i)^2}$$

- Determine the class from nearest neighbor list

$$Majority\ Voting\ y' = argmax_v \sum_{(x_i,y_i)\epsilon D_z} I(v = y_i)$$

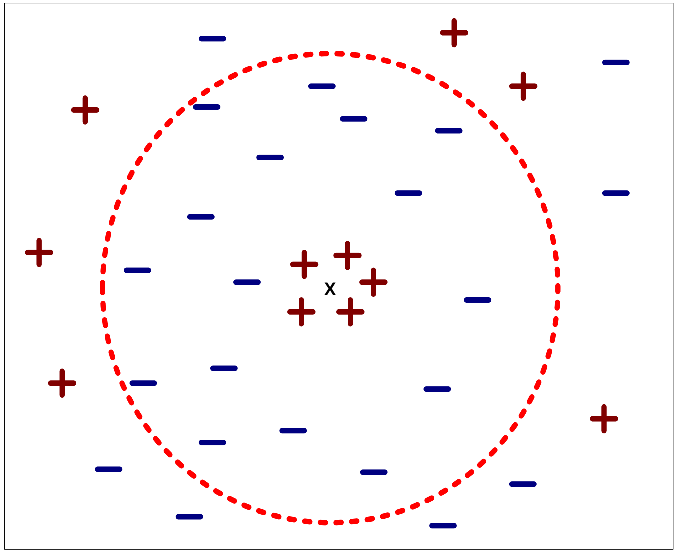
$$Distance\ Weighted\ Voting\ y' = argmax_v \sum_{(x_i,y_i)\epsilon D_z} w_i \times I(v = y_i)$$

where,

weight Factor, $w = \frac{1}{d(x,x')^2}$

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

- Re-scale attributes to prevent distance measures from being dominated by one of the attributes

	A	B	C	D	E	F
1	An example to demonstrate scaling issue					
2						
3	ID	Age	Income	Class		
4	1	15	50000	Yes		
5	2	26	100000	No		
6	3	27	70000	?		
7						
8		Distance(1, 3)	20000.0036	closer		
9		Distance(2, 3)	30000.00002			
10						
11	Normalized		min(age)	15	max(age)	27
12			min(Income)	50000	max(incor	100000
13	ID	Age	Income	Class		
14	1	0	0	Yes		
15	2	0.916666667	1	No		
16	3	1	0.4	?		
17						
18		Distance(1, 3)	1.077032961			
19		Distance(2, 3)	0.605759395	closer		
20						

Nearest Neighbor Classification: Limitations 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 0		1 0 0 0 0 0 0 0 0 0 0 0	
	vs		
0 1 1 1 1 1 1 1 1 1 1 1		0 0 0 0 0 0 0 0 0 0 0 1	

d = 1.4142 **d = 1.4142**

Example: PEBLS

- PEBLS: Parallel Exemplar-Based Learning System (Cost & Salzberg)
  - Works with both **continuous** and **nominal** features
    - For **nominal** features, *distance* between two nominal values is computed using Modified Value Difference Metric (MVDM)

$$d(v_a, v_b) = \sum_{i \in class} \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

See an example in the next slide

  - Each record is assigned *a weight factor*
  - Number of nearest neighbor, k = 1

Asst. Prof. Dr. Rachsuda Setthawong

Example: Using MVDM for calculating distance between nominal attribute values

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

$$d(v_a, v_b) = \sum_{i \in class} \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

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*
Total	4	4	2

Class	Refund	
	Yes	No
Yes	**0**	**3**
No	*3*	*4*
Total	3	7

Asst. Prof. Dr. Rachsuda Setthawong

Example: PEBLS

Tid	Refund	Marital Status	Taxable Income	Cheat
X	Yes	Single	125K	**No**
X'	No	Married	100K	**?**

Distance between record X and X':

$$Dist(X, X') = w_X w_{X'} \sum_{i=1}^{d} d(X_i, X'_i)$$

where:

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

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

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

Asst. Prof. Dr. Rachsuda Setthawong

Nearest neighbor Classification: Characteristics and Performance

- k-NN classifiers are lazy learners
  - It does not build models explicitly
    - Unlike eager learners such as decision tree induction and rule-based systems
  - Selection of right proximity measure is essential
- Pros:
  - Can produce *arbitrarily shaped decision boundaries*
  - Pretty good performance in classification
- Cons:
  - Classifying unknown records are relatively expensive
  - Superfluous or redundant attributes can create problems
  - Missing attributes are hard to handle

Bayes Classifier

- How does it work?
- Characteristics of the algorithm?
- How to construct a model? (algorithms)
- Improvement? – Distance Weighted Voting
- Advantages / Disadvantages?

Bayes Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

$$P(C \mid A) = \frac{P(A,C)}{P(A)}$$

$$P(A \mid C) = \frac{P(A,C)}{P(C)}$$

P(A,C) = joint prob.

P(C), P(A) = prior prob.

P(C|A) = posterior prob.

- Bayes theorem:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

A posterior probability distribution is an example of a conditional probability. GIVEN that we observed a particular set of data, the posterior probability tells us how likely the values of the parameters are.

Example of Bayes Theorem

- Given:
  - A doctor knows that *50% of the time meningitis causes stiff neck* P(S|M)
  - Prior probability of *any patient having meningitis* P(M) is 1/50,000
  - Prior probability of *any patient having stiff neck* P(S) is 1/20

- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Bayesian Classifiers: Overview

- Consider each attribute (A_i) and class label (C) as random variables
- Given a record with attributes (A_1, A_2,…,A_n)
  - **Goal:** predict class C
  - **Task:** find the value of C that maximizes P(C| A_1, A_2,…,A_n)
- How to estimate P(C| A_1, A_2,…,A_n) directly from data?

Bayesian Classifiers: Approach

- Compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ for all values of C using the Bayes theorem

$$\boxed{P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}}$$

- Choose value of C that maximizes $P(C \mid A_1, A_2, \ldots, A_n)$ equivalent to choosing value of C that maximizes $P(A_1, A_2, \ldots, A_n \mid C)\, P(C)$

- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

- Assume independence among attributes A_i when class is given:
  - Then $P(A_i \mid C_j)$ for all A_i and C_j is estimated by using the following formula.

$$\boxed{P(A_1, A_2, \ldots, A_n \mid C) = P(A_1 \mid C_j)\, P(A_2 \mid C_j) \ldots P(A_n \mid C_j)}$$

- **Usage of Naïve Bayes Classifier**:
  - New point (with unknown class label) is classified to C_j if

    $[\, \prod P(A_i \mid C_j)\,] \times P(C_j)$ is maximal.

How to Estimate Probabilities from Data? Discrete Attributes

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Class: $P(C) = N_c/N$
  - e.g., P(No) = 7/10,
    P(Yes) = 3/10
- For discrete attributes:

  $P(A_i \mid C_k) = |A_{ik}| / N_c$

  where
  - $|A_{ik}|$ is number of k instances having attribute A_i and belongs to class C_k
  - Examples:

    P(Status=Married|No) = 4/7

    P(Refund=Yes|Yes) = 0

How to Estimate Probabilities from Data?

- For continuous attributes:
  - Discretization (continuous -> ordinal)

  - Probability density estimation:
    - Assume attribute follows a normal distribution
    - Use data to estimate parameters of distribution
      (e.g., mean and standard deviation)
    - Once probability distribution is known, can use it to estimate the conditional probability $P(A_i|c)$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Assume normal distribution:

$$P(A_i \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}}} e^{-\frac{(A_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - Calculate one for each (A_i, c_i) pair

- Example, for (Income, Class=No):
  - If Class=No
    - sample mean (μ) = (770/7) = 110
    - sample SD (σ^2) = 2975
    - sample variance (σ) = 54.54

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Asst. Prof. Dr. Rachsuda Setthawong

Naïve Bayes Classifier: Example 1

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/3
P(Marital Status=Divorced|Yes)= 1/3
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|**Class=No**) = P(Refund=No|Class=No)
  $\times$ P(Married| Class=No)
  $\times$ P(Income=120K| Class=No)
  = $4/7 \times 4/7 \times 0.0072$ = **0.0024**

- P(X|**Class=Yes**) = P(Refund=No| Class=Yes)
  $\times$ P(Married| Class=Yes)
  $\times$ P(Income=120K| Class=Yes)
  = $1 \times 0 \times 1.2 \times 10^{-9}$ = **0**

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)
=> Class = No

Naïve Bayes Classifier: Problem and Solution

- Problem: if one of the conditional probability is zero, then the entire expression becomes zero
- Solution: need to use other estimates of conditional probabilities than simple fractions
  - Probability estimation:

$$\text{Original}: P(A_i \mid C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace}: P(A_i \mid C) = \frac{N_{ic} + 1}{N_c + c}$$

$$\text{m - estimate}: P(A_i \mid C) = \frac{N_{ic} + mp}{N_c + m}$$

c: number of classes

p: prior probability

m: parameter

N_c: number of instances in the class

N_{ic}: number of instances having attribute value A_i in class c

Naïve Bayes Classifier: Example 2

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributes

M: mammals

N: non-mammals

$$P(A|M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

Naïve Bayes Classifier: Summary

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)

Rule-Based Classifier

- How does it work?
- Characteristics of the algorithm?
- How to construct a model? (algorithms)
- Improvement? -- Rule simplification
- Advantages / Disadvantages?

Asst. Prof. Dr. Rachsuda Setthawong

Rule-Based Classifier

- Classify records by using a collection of "if…then…" rules
- Rule:

$$(Condition) \rightarrow y$$

[Antecedent] [Consequent]

  - where
    - *Condition* is a conjunctions of attributes
    - *y* is the class label
- Rule Examples:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) $\rightarrow$ Evade=No

Asst. Prof. Dr. Rachsuda Setthawong

Rule-based Classifier (Example)

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

How does Rule-based Classifier Work?

- A rule *r* covers an instance **x** if the attributes of the instance satisfy the condition of the rule

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

Bird (R1)

Mammal (R3)

Asst. Prof. Dr. Rachsuda Setthawong

How is Unusual Case Handled?

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

Mammal (R3)

Reptiles (R4) / Amphibian (R5)?

??

Characteristics of (Good) Rule-Based Classifier

- Mutually exclusive rules
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered by <u>at most one</u> rule

- Exhaustive rules
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered by <u>at least one</u> rule

Ordered Rule Set

Every record is covered by at most one rule

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a decision list
- When a test record is presented to the classifier
  - It is assigned to the class label of the highest ranked rule it has triggered
  - If none of the rules fired, it is assigned to the default class *(Each record is covered by at least one rule)*

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?

Asst. Prof. Dr. Rachsuda Setthawong

Rule Ordering Schemes

Alternative approaches to sort the rules

- Rule-based ordering *(previous slide)*
  - Individual rules are ranked based on their quality
- Class-based ordering *(alternative)*
  - Rules that belong to the same class appear together

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

Asst. Prof. Dr. Rachsuda Setthawong

Building Classification Rules

- Direct Method:
  - Extract rules directly from data
  - e.g.: RIPPER, CN2, Holte's 1R

- Indirect Method:
  - Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  - e.g: C4.5 rules

How to Define “Best Rule”?

- Coverage of a rule:
  - Fraction of records that satisfy the antecedent of a rule

- Accuracy of a rule:
  - Fraction of records that satisfy both the antecedent and consequent of a rule

Tid	Refund	Marital Status	Taxable Income	Class
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**

E.g., (Status=Single) $\rightarrow$ No

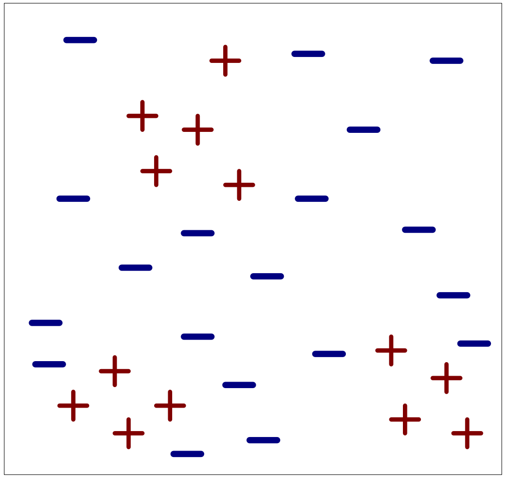
Coverage = 4/10 = 40%

Accuracy = 2/4 = 50%

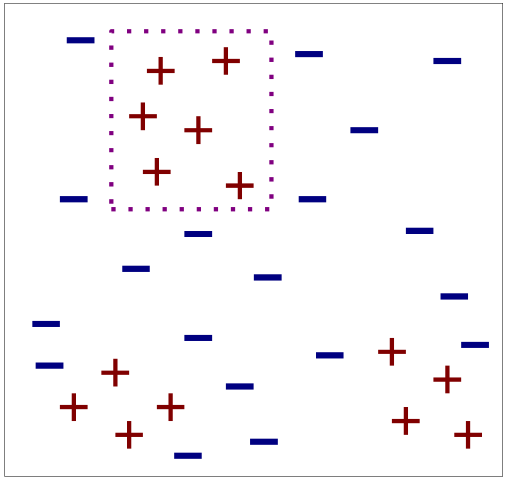
Direct Method: Sequential Covering

1. Start from an empty rule
2. Grow a rule using the **Learn-One-Rule** function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) until stopping criterion is met

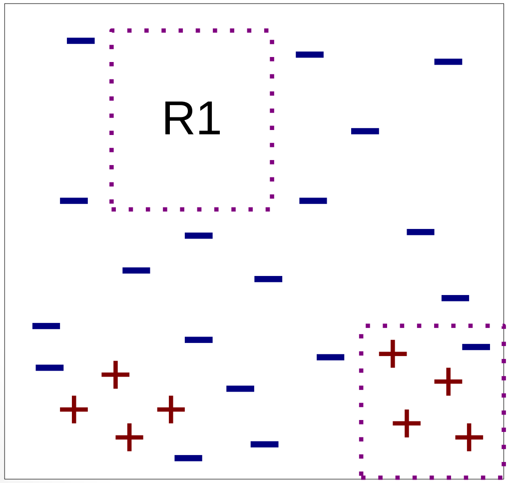
Example of Sequential Covering



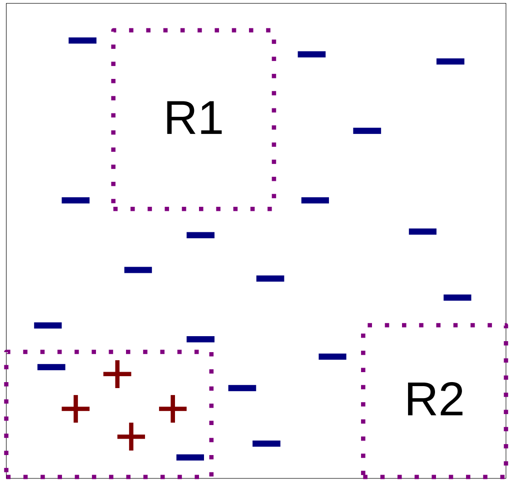
(i) Original Data



(ii) Step 1



(iii) Step 2



(iv) Step 3

Asst. Prof. Dr. Rachsuda Setthawong

Learn-One-Rule Function

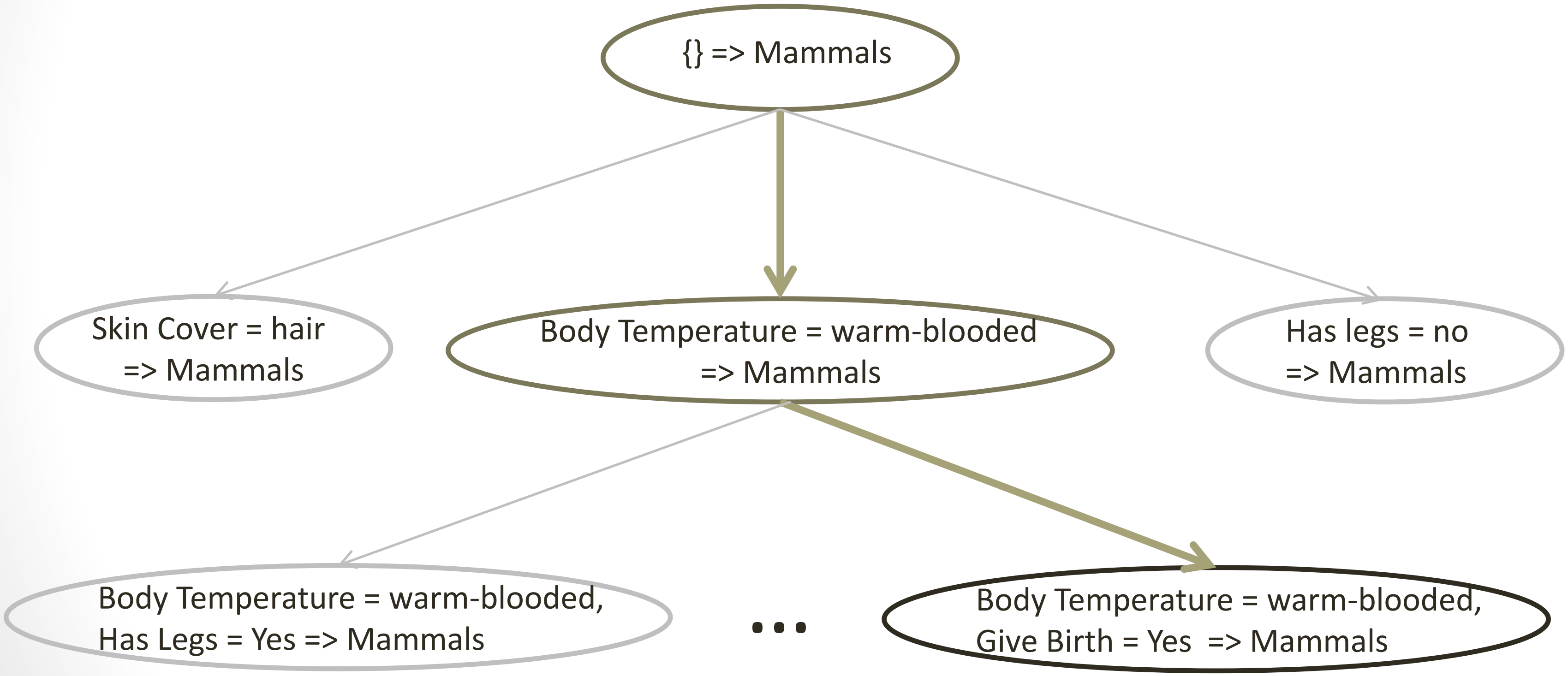
- **Goal:** extract a classification rule that
  - Max. # of positive examples
  - Min. # of negative examples

- **Greedy approach:**
  - Generates an initial rule r
  - Keep refining it until meeting a stopping criterion
  - Prune it (improve its generalization error)

Aspects of Sequential Covering

1. Rule Growing
2. Instance Elimination
3. Rule Evaluation
4. Stopping Criterion
5. Rule Pruning

1. Rule Growing: General-to-Specific



1. Rule Growing:
Specific-to-General

A randomly chosen (+ve) example

Body Temperature = warm-blooded,
Skin Cover = hair, Gives Birth = yes,
Aquatic creature = no, Aerial Creature = no
Has Legs = yes, Hibernates = no => Mammals

Skin Cover = hair, Gives Birth = yes,
Aquatic creature = no, Aerial Creature = no
Has Legs = yes, Hibernates = no => Mammals

...

Body Temperature = warm-blooded,
Skin Cover = hair, Gives Birth = yes,
Aquatic creature = no, Aerial Creature = no
Has Legs = yes => Mammals

Rule Growing (Examples)

- CN2 Algorithm:
  - Start from an empty conjunct: {}
  - Add conjuncts that minimizes the entropy measure: {A}, {A,B}, …
  - Determine the rule consequent by taking majority class of instances covered by the rule

- RIPPER Algorithm: Start from an empty rule: {} => class
  - Add conjuncts that maximizes FOIL's information gain measure:

$$\text{Gain}(R0, R1) = p1 \times [\ \log (p1/(p1+n1)) - \log (p0/(p0 + n0))\ ]$$

where

R0: {} => class (initial rule)

R1: {A} => class (rule after adding conjunct)

p0: # of +ve instances covered by R0

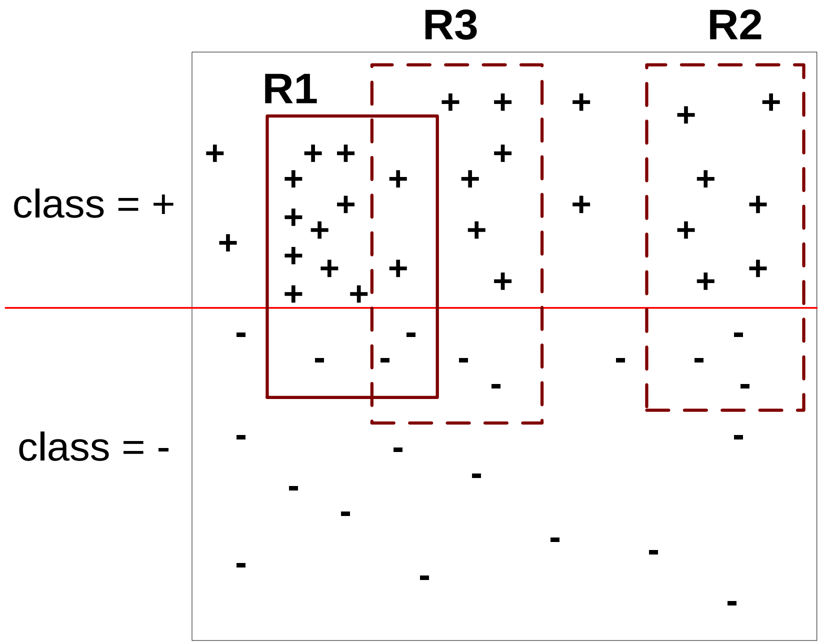
n0: # of –ve instances covered by R0

p1: # of +ve instances covered by R1

n1: # of –ve instances covered by R1

2. Instance Elimination

- Why do we need to eliminate instances?
- Why do we remove positive instances?
- Why do we remove negative instances?



Round 1:

- Accuracy(R1) = 12/15 = 0.8 (Selected R1)

Round 2:

- Accuracy(R2) = 7/10 = 0.7 (Selected R2 if R3 removes used instances)
- Accuracy(R3 w/o removing) = 8/12 = 0.67
- Accuracy(R3 removing) = 6/8 = 0.75 (Selected R3 if R3 *doesn't* remove used instances)

Rachsuda Jiamthapthaksin, Ph.D.

3. Rule Evaluation

- Metrics:
  - Accuracy $= \dfrac{n_c}{n}$
  - Laplace $= \dfrac{n_c + 1}{n + k}$
  - M-estimate $= \dfrac{n_c + kp}{n + k}$

n : # of instances covered by rule

n_c : # of +ve instances covered by rule

k : # of classes

p : the Prior probability for the +ve class

4. Stopping Criterion and Rule Pruning

- Stopping criterion (stop adding a new rule)
  - Compute the gain
  - If gain is not significant, discard the new rule

- Rule Pruning (to simplify a rule)
  - Similar to post-pruning of decision trees
  - Reduced Error Pruning:
    - Remove one of the conjuncts in the rule
    - Compare error rate on validation set before and after pruning
      - If error improves, prune the conjunct

Summary of Direct Method

- Grow a single rule
- Remove Instances from rule
- Prune the rule (if necessary)
- Add rule to Current Rule Set
- Repeat

Direct Method: RIPPER (1)

- **For 2-class problem:**
  - Learn rules for positive class (interested class)
  - Negative class will be default class

- **For multi-class problem:**
  - Order the classes according to increasing class prevalence (fraction of instances that belong to a particular class)
  - Learn the rule set for smallest class first, treat the rest as negative class
  - Repeat with next smallest class as positive class

Direct Method: RIPPER (2)

- Building a Rule Set:
  - Use sequential covering algorithm
    - Finds the best rule that covers the current set of positive examples
    - Eliminate both positive and negative examples covered by the rule
  - Each time a rule is added to the rule set, compute the new description length
    - Stop adding new rules when the new description length is *d* bits (64 bits) longer than the smallest description length obtained so far
      - If DL’ < (DL + 64) Then Stop

Direct Method: RIPPER (3)

- Growing a rule (LearnRule(Pos,Neg)):
  - Start from empty rule
  - Add conjuncts as long as they improve FOIL's information gain *(Slide 47)*
  - Stop when rule start covers negative examples
  - Prune the rule using the validation set.
  - Measure for pruning: v = (p-n)/(p+n)
    - p: # of +ve examples covered by the rule in the validation set
    - n: #of –ve examples covered by the rule in the validation set
  - E.g., ABCD → y
    - Determine to remove D, CD, BCD, etc., sequentially as long as v is improved.

RIPPER algorithm

```
Ripper(Pos,Neg,k)
  RuleSet ← LearnRuleSet(Pos,Neg)
  For k times
    RuleSet ← OptimizeRuleSet(RuleSet,Pos,Neg)
LearnRuleSet(Pos,Neg)
  RuleSet ← ∅
  DL ← DescLen(RuleSet,Pos,Neg)
  Repeat
    Rule ← LearnRule(Pos,Neg)
    Add Rule to RuleSet
    DL' ← DescLen(RuleSet,Pos,Neg)
    If DL'>DL+64
      PruneRuleSet(RuleSet,Pos,Neg)
      Return RuleSet
    If DL'<DL DL ← DL'
      Delete instances covered from Pos and Neg
  Until Pos = ∅
  Return RuleSet
```

Learn a Rule

Stop learning

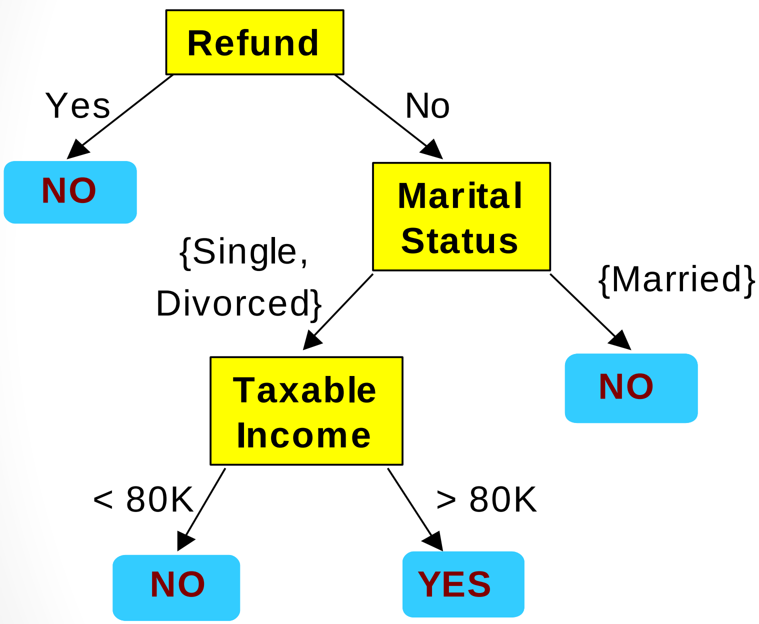
```
PruneRuleSet(RuleSet,Pos,Neg)
  For each Rule ∈ RuleSet in reverse order
    DL ← DescLen(RuleSet,Pos,Neg)
    DL' ← DescLen(RuleSet-Rule,Pos,Neg)
    IF DL'<DL Delete Rule from RuleSet
  Return RuleSet
OptimizeRuleSet(RuleSet,Pos,Neg)
  For each Rule ∈ RuleSet
      DL0 ← DescLen(RuleSet,Pos,Neg)
      DL1 ← DescLen(RuleSet-Rule+
        ReplaceRule(RuleSet,Pos,Neg),Pos,Neg)
      DL2 ← DescLen(RuleSet-Rule+
        ReviseRule(RuleSet,Rule,Pos,Neg),Pos,Neg)
      If DL1=min(DL0,DL1,DL2)
        Delete Rule from RuleSet and
          add ReplaceRule(RuleSet,Pos,Neg)
      Else If DL2=min(DL0,DL1,DL2)
        Delete Rule from RuleSet and
          add ReviseRule(RuleSet,Rule,Pos,Neg)
  Return RuleSet
```

Ref: *E. Alpaydin, Lecture Notes for An Introduction to machine Learning, 2004, MIT Press. (Chapter 9)*

Indirect Method: C4.5 Rules

- Extract rules from an unpruned decision tree
- (Prune a rule) For each rule, r: A $\rightarrow$ y,
  - Consider an alternative rule r': A' $\rightarrow$ y where A' is obtained by removing one of the conjuncts in A
  - Compare the pessimistic error rate for r against all r's
  - Prune if one of the r's has lower pessimistic error rate
  - Repeat until we can no longer improve generalization error
- Use class based ordering to order the extracted rules.

From Decision Trees To Rules



Classification Rules

(Refund=Yes) ==> No

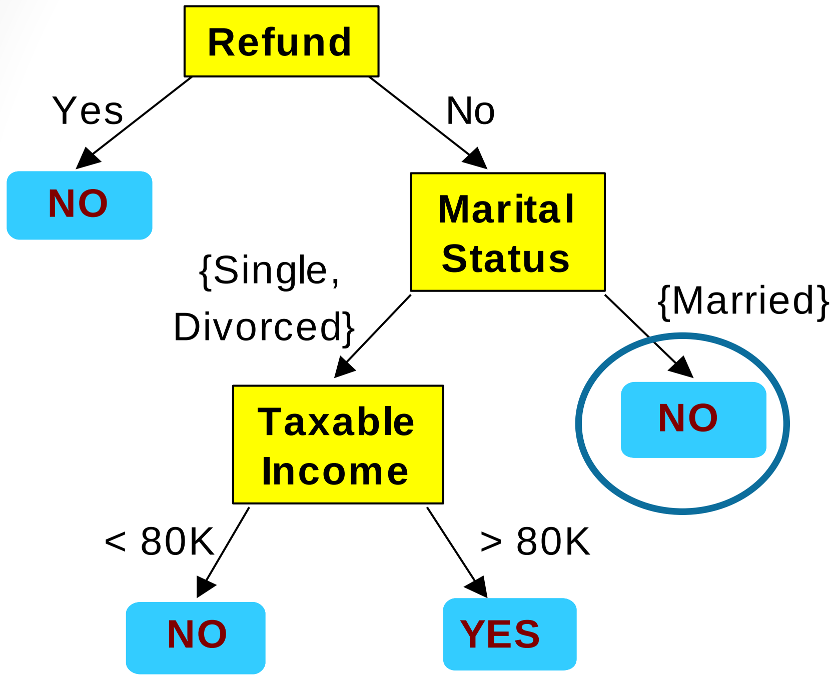
(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Rules are mutually exclusive and exhaustive.

Rules Can Be Simplified



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

Initial Rule: **(Refund=No) $\wedge$ (Status=Married) $\rightarrow$ No**

Simplified Rule: **(Status=Married) $\rightarrow$ No**

Asst. Prof. Dr. Rachsuda Setthawong

Effect of Rule Simplification

- Rules are no longer mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes

- Rules are no longer exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class

Advantages of Rule-Based Classifiers

- As highly expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

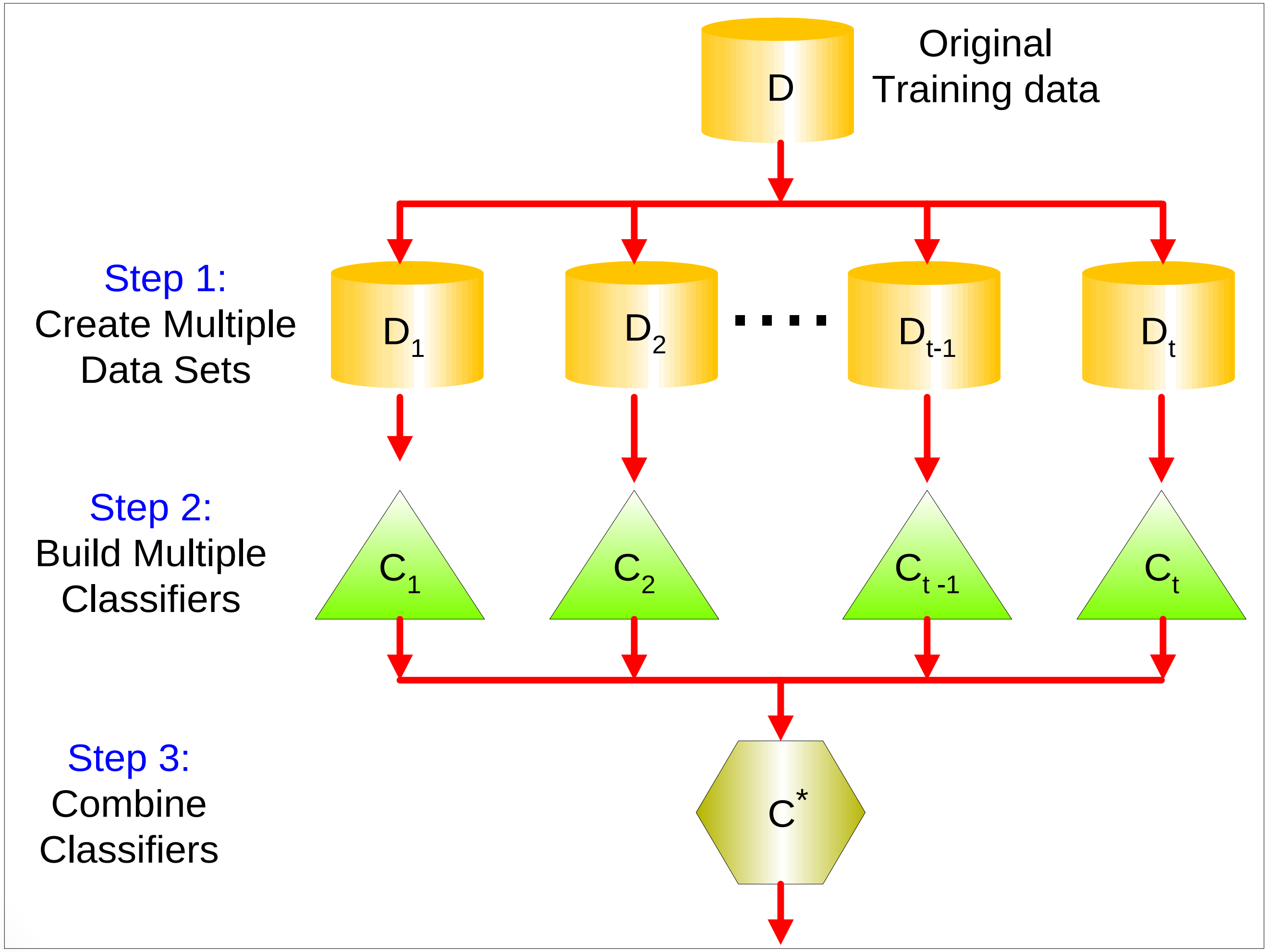
Ensemble Methods

- How does it work?
- Characteristics of the algorithm?
- How to construct a model? (algorithms)
- Advantages / Disadvantages?

Ensemble Methods

- Construct a set of classifiers from the training data
- Predict class label of previously unseen records by aggregating predictions made by multiple classifiers

General Idea



Why does it work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

How to generate an ensemble of classifiers?

1. Manipulating the training set (sampling distribution)
   - Bagging
   - Boosting

2. Manipulating the input features (sampling feature subset)

3. Manipulating the learning algorithms (varying parameters)

Asst. Prof. Dr. Rachsuda Setthawong

Bagging

- Sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Build classifier on each bootstrap sample (same size as original data)

- Each sample has probability $(1 – 1/n)^n$ of being selected

$$Voted\ class\ C^*(x) = argmax_y \sum_i \delta(C_i(x) = y)$$

$$\delta(.) = 1\ if\ its\ argument\ is\ true\ and\ 0\ otherwise.$$

Asst. Prof. Dr. Rachsuda Setthawong

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of boosting round

Asst. Prof. Dr. Rachsuda Setthawong

Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Boosting Example: AdaBoost's Concept (1)

- Base classifiers: $C_1, C_2, \ldots, C_T$

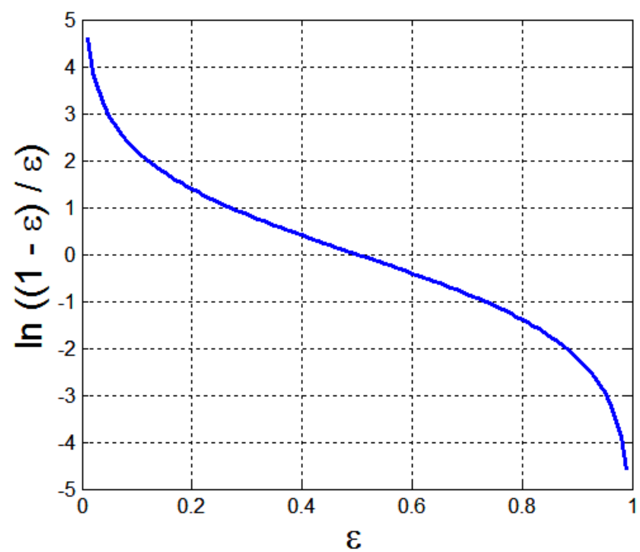
- Error rate:

$$\varepsilon_i = \frac{1}{N}\sum_{j=1}^{N} w_j \delta\left(C_i(x_j) \neq y_j\right)$$

N: # of training examples

- Importance of a classifier:

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$



Rachsuda Jiamthapthaksin, Ph.D.

Boosting Example: AdaBoost's Concept (2)

- Weight update:

$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \times \begin{cases} \exp^{-\alpha_j} & \text{if } C_j(x_i) = y_i \\ \exp^{\alpha_j} & \text{if } C_j(x_i) \neq y_i \end{cases}$$

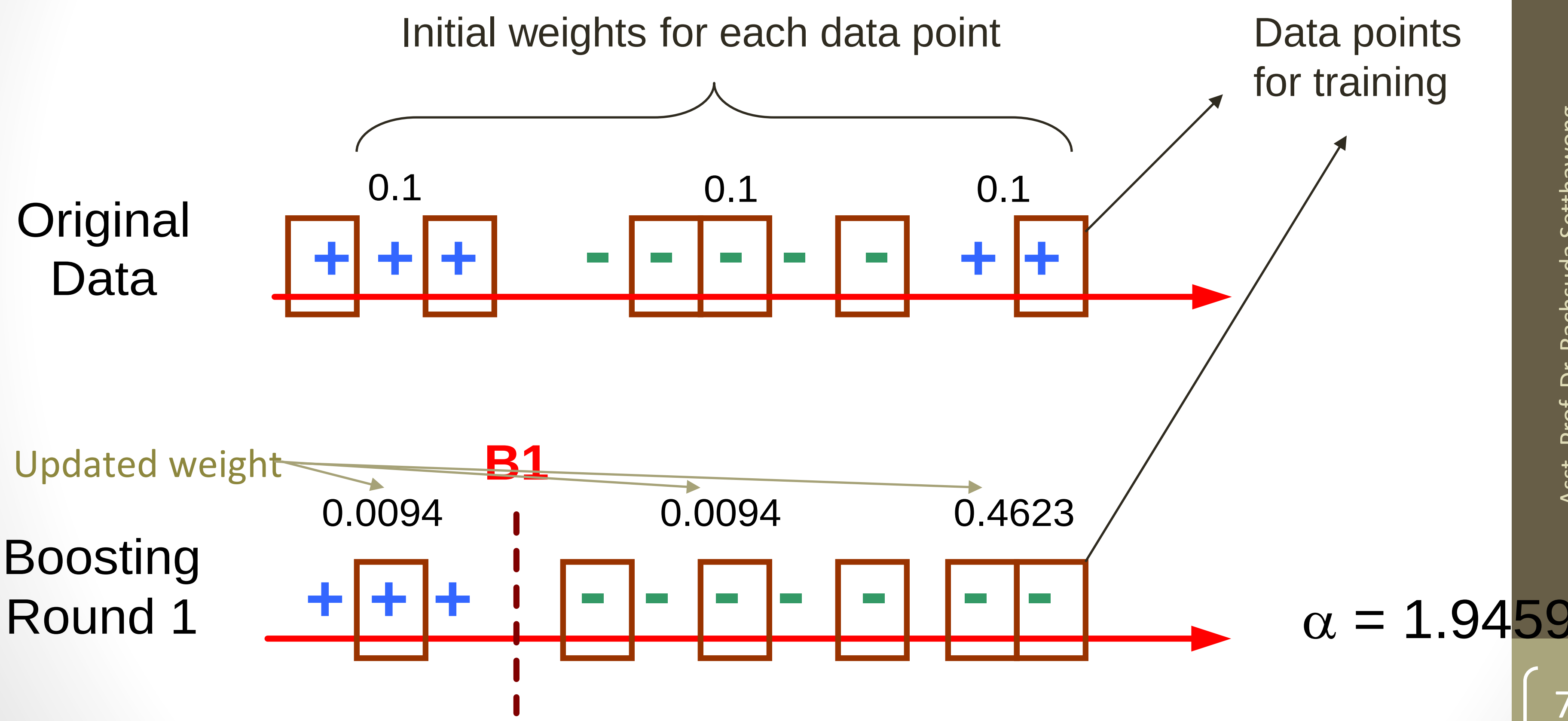
where Z_j is the normalization factor

- Increase the weight of incorrectly classified examples
- Decrease the weight of correctly classified examples

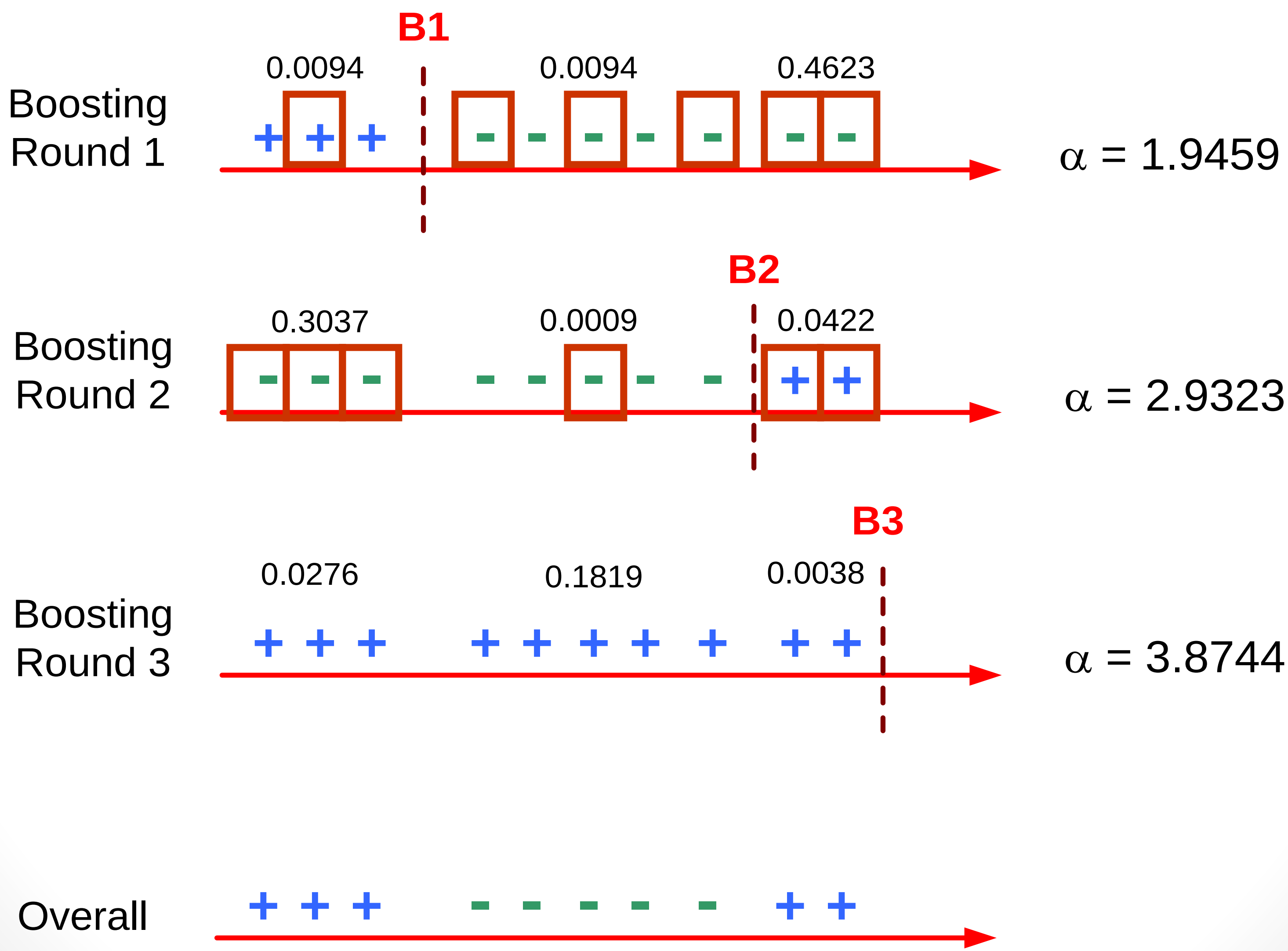
AdaBoost Algorithm

- Initial the weights w_i for all N examples
- Repeat k rounds
  - Create training set D_i by sampling (with replacement) from D according to w.
  - Train a base classifier C_i on D_i.
  - Apply C_i to all examples in the original training set, D.
  - Calculate the weighted error ε_i
  - If ε_i > 0.5 then
    - Reset the weights for all examples.
    - Go back to Create training set's step
  - End if
  - Calculate importance of a classifier α_i
  - Update the weight of each example $w_i^{(j+1)}$
- Classification: $C^*(x) = \arg\max_y \sum_{j=1}^{T} \alpha_j \delta\left(C_j(x) = y\right)$

Illustrating AdaBoost



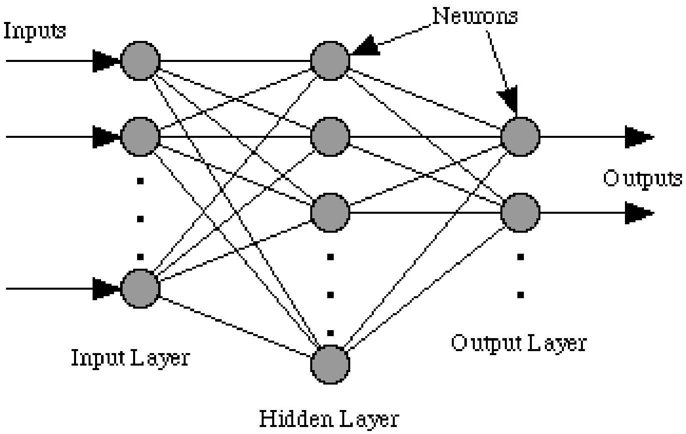
Illustrating AdaBoost



Asst. Prof. Dr. Rachsuda Setthawong

Artificial Neural Network (ANN)

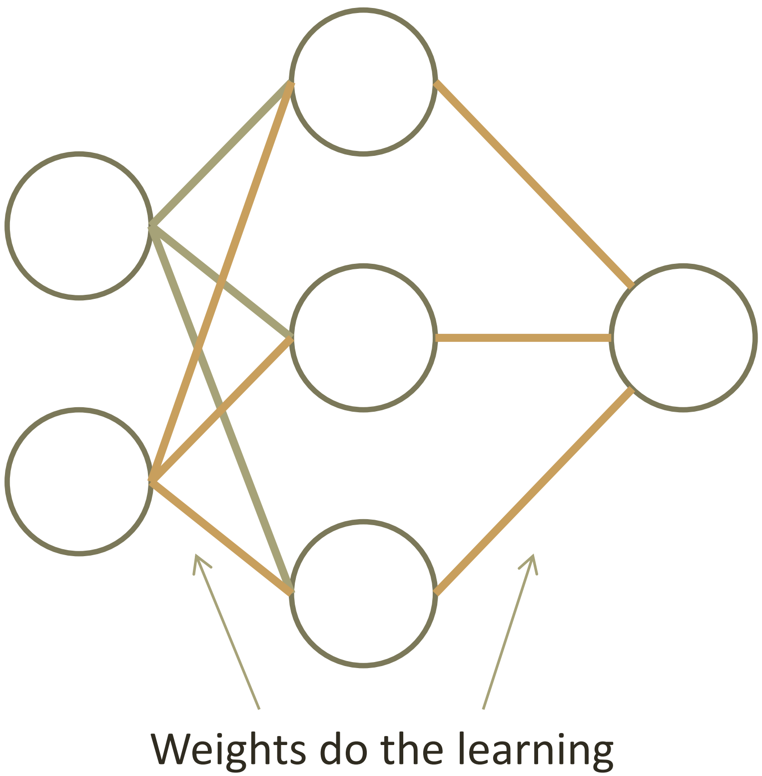
- Computing systems vaguely inspired by the biological neural networks that constitute animal brains.
- A collection of connected units or nodes (artificial neurons).
  - Each connection (synapse) between artificial neurons can transmit a signal from one to another.
  - The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.



Ref: https://en.wikipedia.org/wiki/Artificial_neural_network

Hyperparameters

- Fixed structure of neural network that update weights when training the network.



Prediction's Example
Supervised Regression

	x (Hr. Sleep, Hr. Study)	y (Score on Test)
Training data	(3, 5)	75
	(5, 1)	82
	(10, 2)	93
Test data	(8, 3)	?

General Steps

1. Scale the data (e.g., standardization, normalization)
2. Construct a model from training data
3. Predict results for unknown data (test data)

Asst. Prof. Dr. Rachsuda Setthawong

Scale the Data

- $x_{norm} = \frac{x}{\max(x)}$
- $y_{norm} = \frac{y}{\max(y)}$

x (Hr. Sleep, Hr. Study)	y (Score on Test)
(3, 5)	75
(5, 1)	82
(10, 2)	93
(8, 3)	?

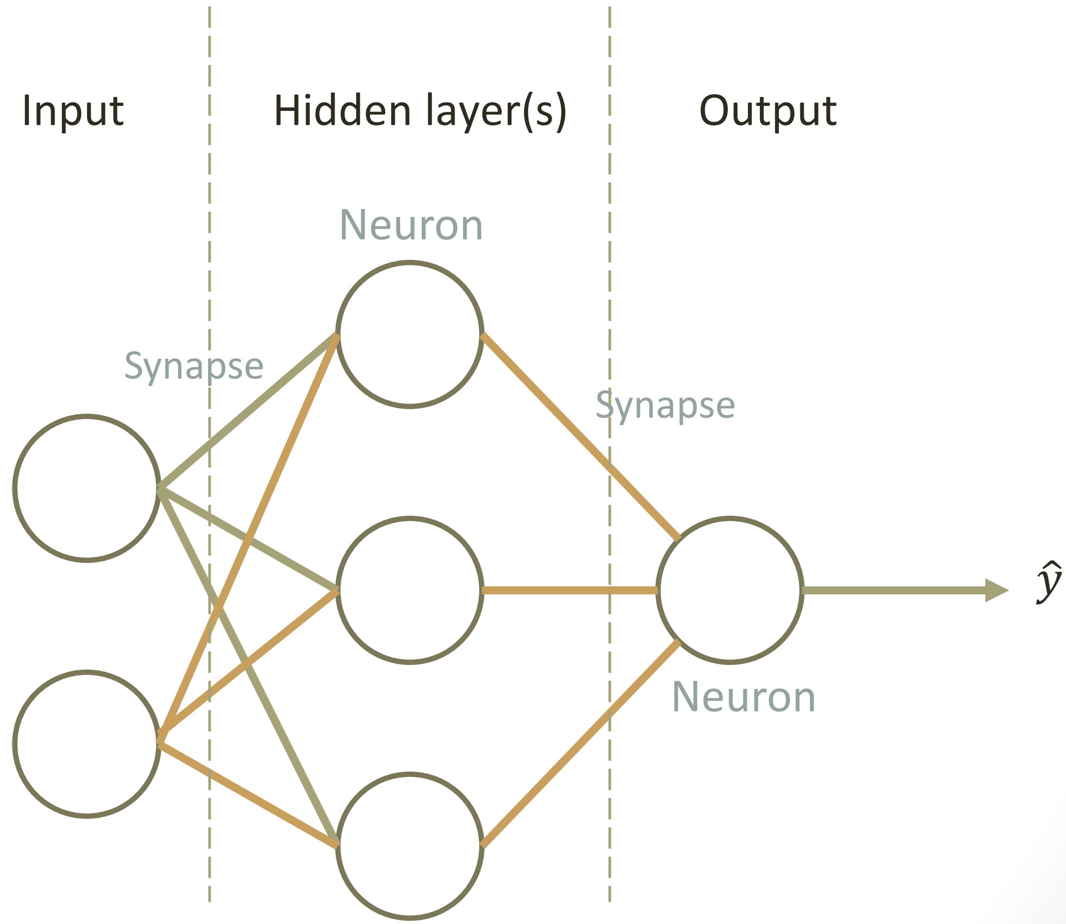
Suppose that
max(x) = 10
max(y) = 100

x (Hr. Sleep, Hr. Study)	y (Score on Test)
(0.3, 0.5)	0.75
(0.5, 0.1)	0.82
(1.0, 0.2)	0.93
(0.8, 0.3)	?

Range = [0, 1]

Artificial Neural Network

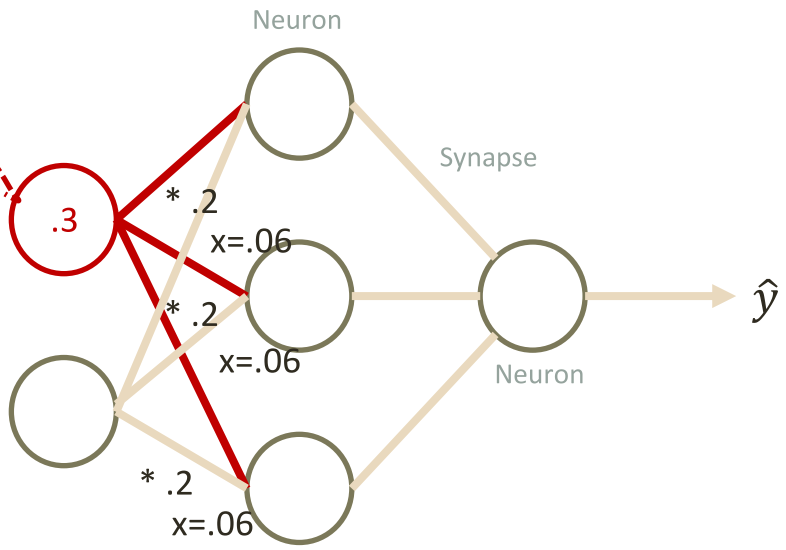
x (Hr. Sleep, Hr. Study)	y (Score on Test)
(0.3, 0.5)	0.75
(0.5, 0.1)	0.82
(1.0, 0.2)	0.93
(0.8, 0.3)	?



How does it work? (1)

- Synapse: Take an input and multiply it by weight (equal to 0.2 in the example) and transfer to neuron(s).

x (Hr. Sleep, Hr. Study)	y (Score on Test)
(0.3, 0.5)	0.75
(0.5, 0.1)	0.82
(1.0, 0.2)	0.93
(0.8, 0.3)	?

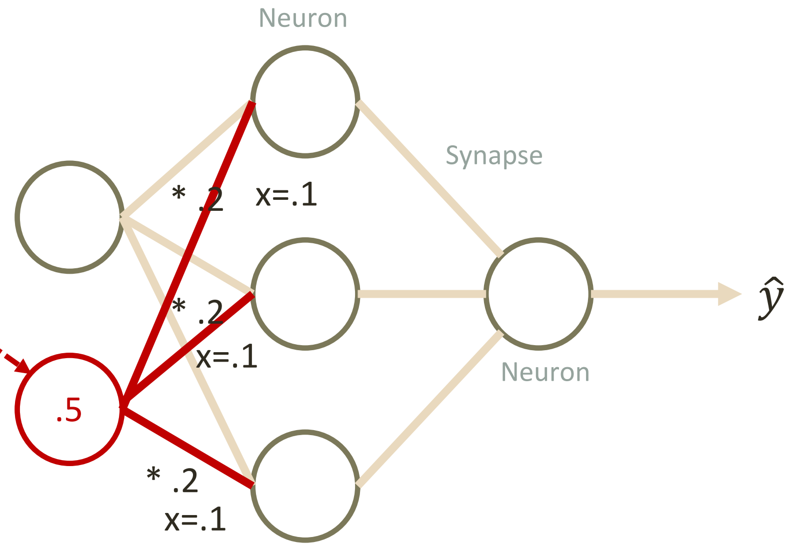


Asst. Prof. Dr. Rachsuda Setthawong

How does it work? (2)

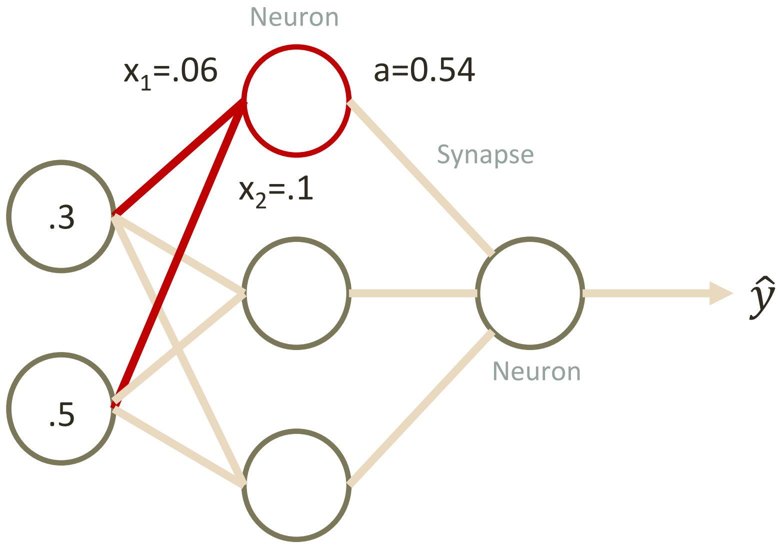
- Synapse: Take an input and multiply it by weight (equal to 0.2 in the example) and transfer to neuron(s).

x (Hr. Sleep, Hr. Study)	y (Score on Test)
(0.3, 0.5)	0.75
(0.5, 0.1)	0.82
(1.0, 0.2)	0.93
(0.8, 0.3)	?



How does it work? (3)

- Neurons: Integrate them [Eq.1] and activate the result [Eq.2].

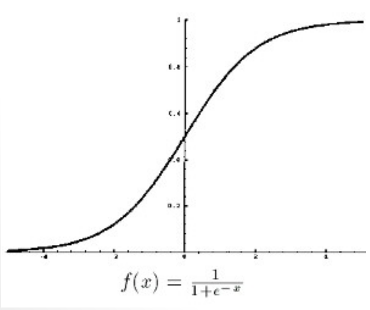


[Eq. 1] $z = \sum x_i = x_1 + x_2$

$z = .06 + .1 = 0.16$

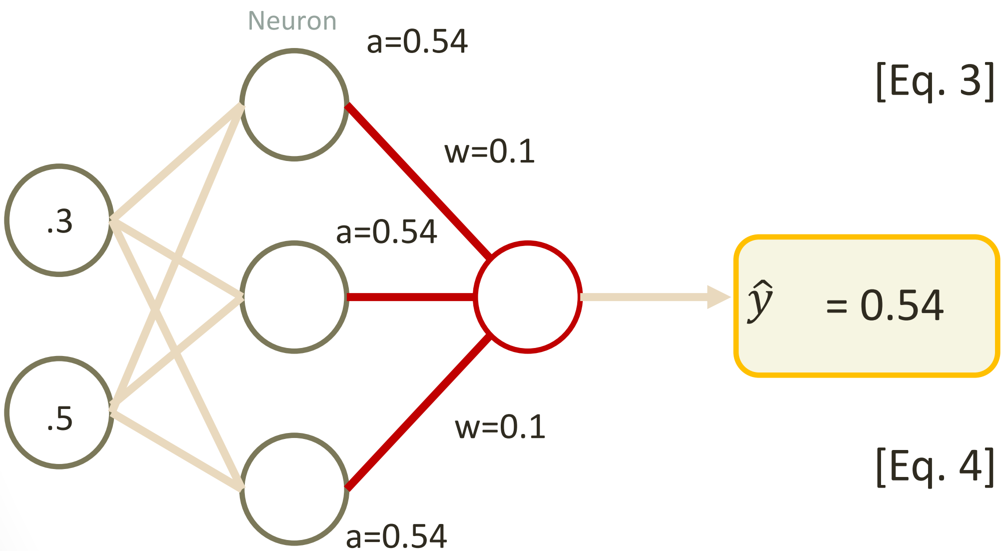
[Eq. 2] $a = \frac{1}{1+e^{-z}}$

$a = \frac{1}{1+e^{-0.16}} = 0.54$



How does it work? (4)

- Neurons: Integrate them [Eq. 3] and activate the result [Eq. 4].



[Eq. 3] $z = a \cdot w$

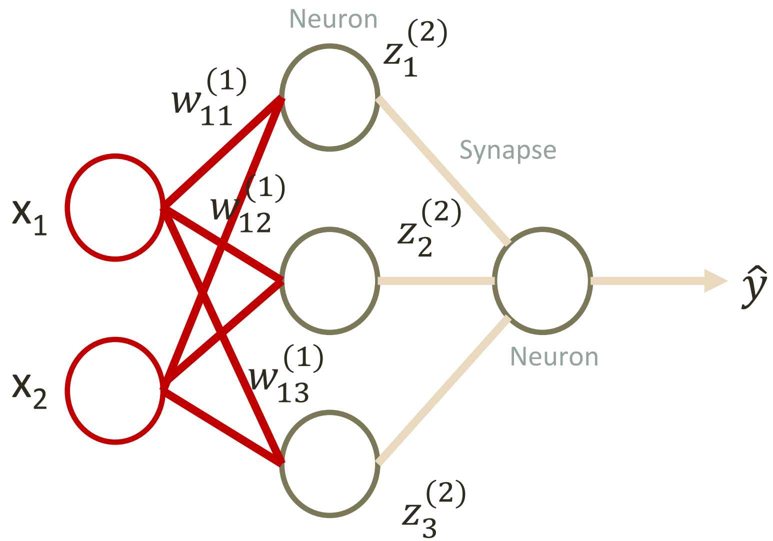
$z = 0.54 * .1 + 0.54 * .1 + 0.54 * .1$
$= 0.162$

[Eq. 4] $\hat{y} = \frac{1}{1+e^{-z}}$

$= \frac{1}{1+e^{-0.162}} = 0.54$

Big Picture (Matrix):

Step 1: Propagate inputs to network



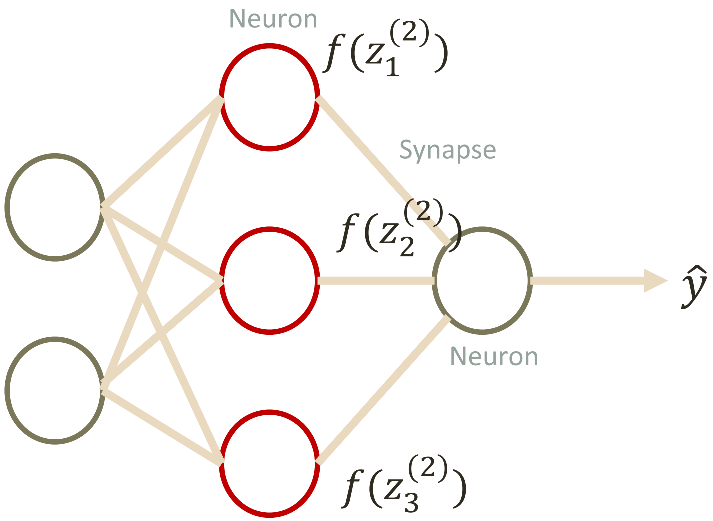
x (Hr. Sleep, Hr. Study)	y (Score on Test)
(0.3, 0.5)	0.75
(0.5, 0.1)	0.82
(1.0, 0.2)	0.93
(0.8, 0.3)	?

$$X \quad . \quad W^{(1)} \quad = \quad Z^{(2)} \qquad \text{[Eq. 1]}$$

$$\begin{bmatrix} .3 & .5 \\ .5 & .1 \\ 1 & .2 \end{bmatrix} \cdot \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix} = \begin{bmatrix} .3w_{11}^{(1)} + .5w_{21}^{(1)} & .3w_{12}^{(1)} + .5w_{22}^{(1)} & .3w_{13}^{(1)} + .5w_{23}^{(1)} \\ .5w_{11}^{(1)} + .1w_{21}^{(1)} & .5w_{12}^{(1)} + .1w_{22}^{(1)} & .5w_{13}^{(1)} + .1w_{23}^{(1)} \\ 1w_{11}^{(1)} + .2w_{21}^{(1)} & 1w_{12}^{(1)} + .2w_{22}^{(1)} & 1w_{13}^{(1)} + .2w_{23}^{(1)} \end{bmatrix}$$

Big Picture (Matrix):

Step 2: Apply Sigmoid Activation Function



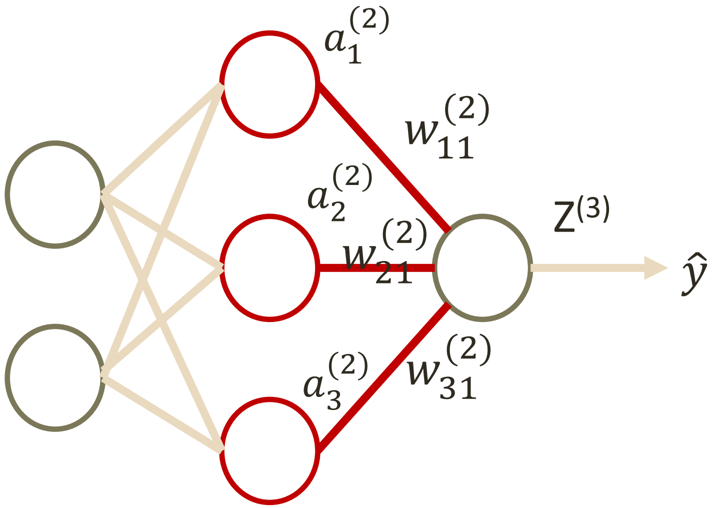
$$f(Z^{(2)}) = \frac{1}{1+e^{-z}}$$

$$a^{(2)} = f(Z^{(2)}) \qquad \text{[Eq. 2]}$$

$$= \begin{bmatrix} f(.3w_{11}^{(1)} + .5w_{21}^{(1)}) & f(.3w_{12}^{(1)} + .5w_{22}^{(1)}) & f(.3w_{13}^{(1)} + .5w_{23}^{(1)}) \\ f(.5w_{11}^{(1)} + .1w_{21}^{(1)}) & f(.5w_{12}^{(1)} + .1w_{22}^{(1)}) & f(.5w_{13}^{(1)} + .1w_{23}^{(1)}) \\ f(1w_{11}^{(1)} + .2w_{21}^{(1)}) & f(1w_{12}^{(1)} + .2w_{22}^{(1)}) & f(1w_{13}^{(1)} + .2w_{23}^{(1)}) \end{bmatrix}$$

Big Picture (Matrix):

Step 3: Propagate intermediate result to the next neuron(s)



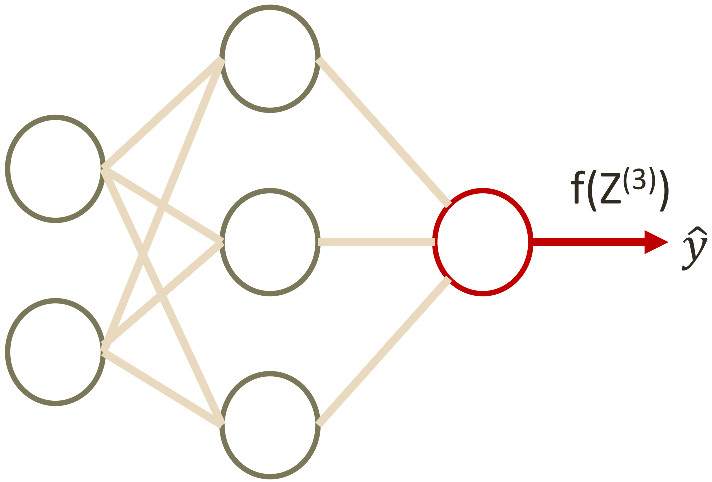
$$Z^{(3)} = A^{(2)} . W^{(2)} \quad \text{[Eq. 3]}$$

$$w^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

Asst. Prof. Dr. Rachsuda Setthawong

Big Picture (Matrix):

Step 4: Apply Sigmoid Activation Function



$$\hat{y} = f(Z^{(3)})$$ [Eq. 4]

Perceptron Learning Algorithm

Let D = {x_i, y_i) | I = 1,2,…,N} be the set of training examples

Initialize the weight vector with random values, $w^{(0)}$

Repeat

for each training example (x_i, y_i) ∈ D do

compute the predicted output $\hat{y}_i^{(k)}$

for each weight w_j do

Update the weight,

$$w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$$

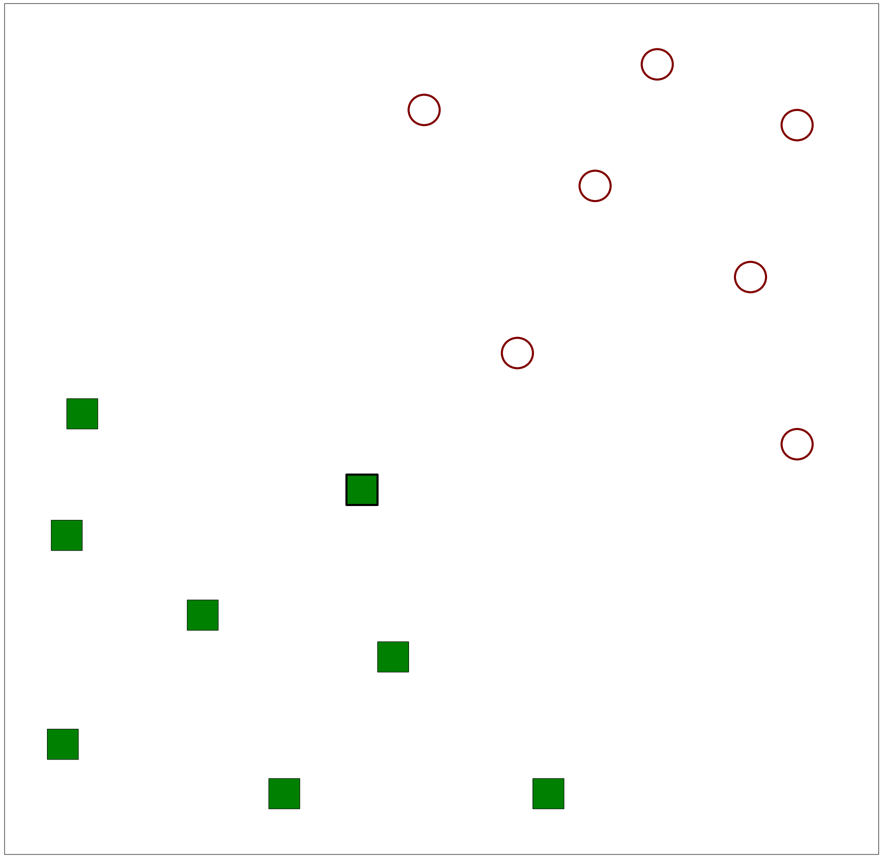
end for

end for

Until stopping condition is met.

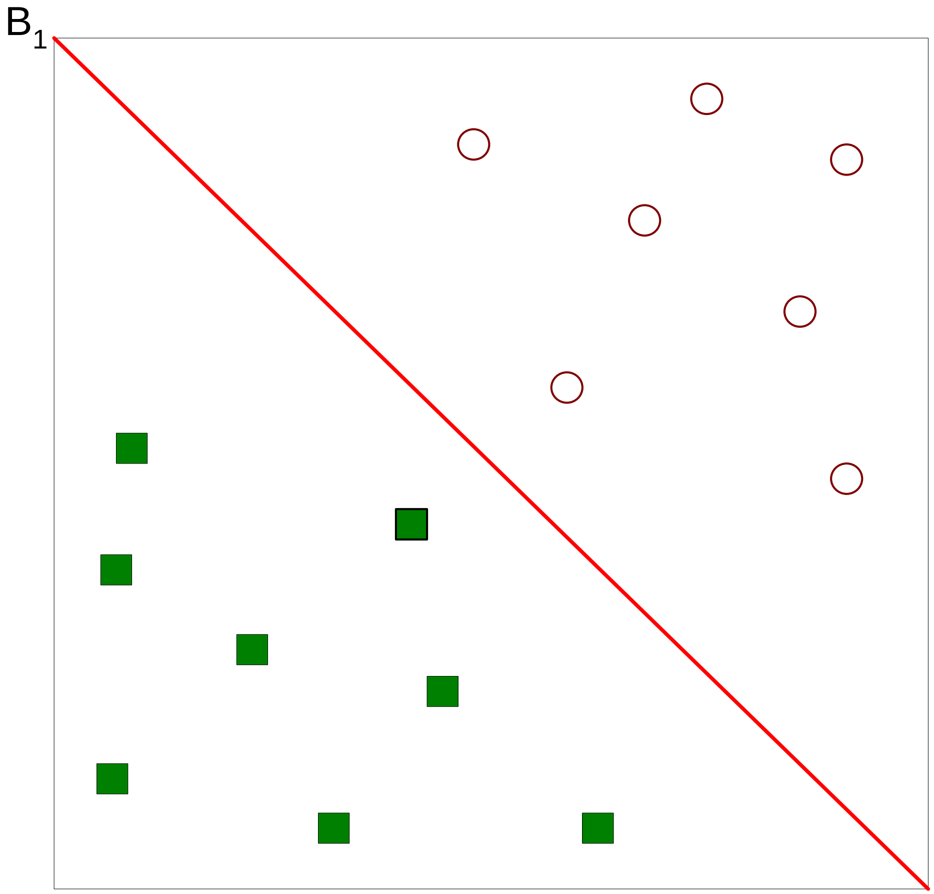
λ: learning rate

Support Vector Machines



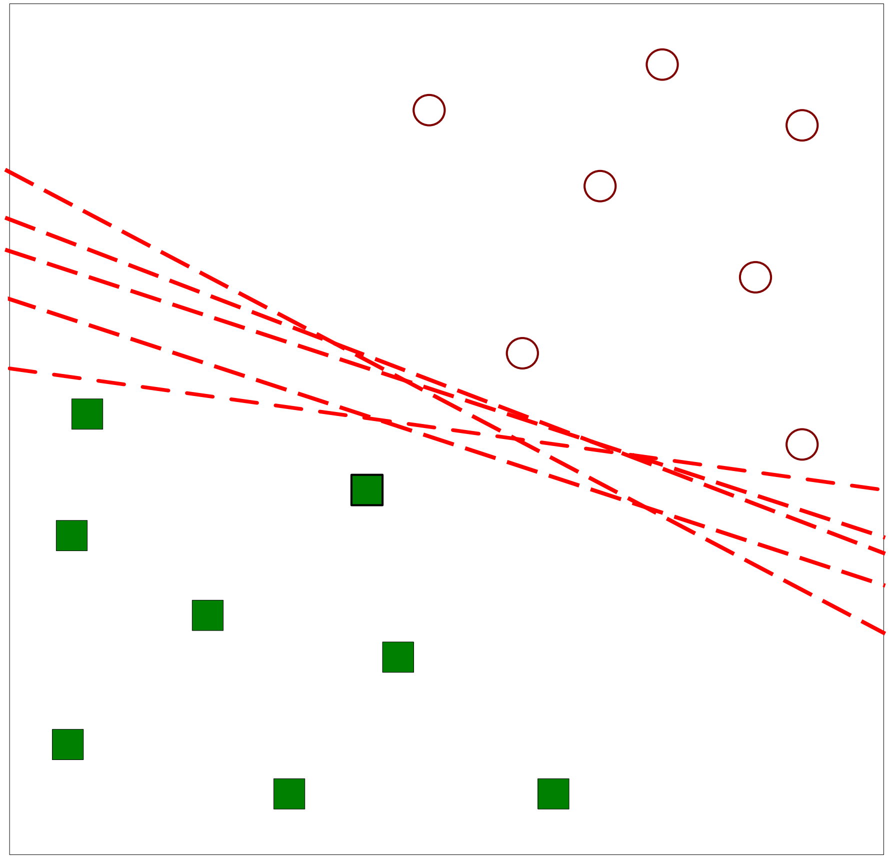
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



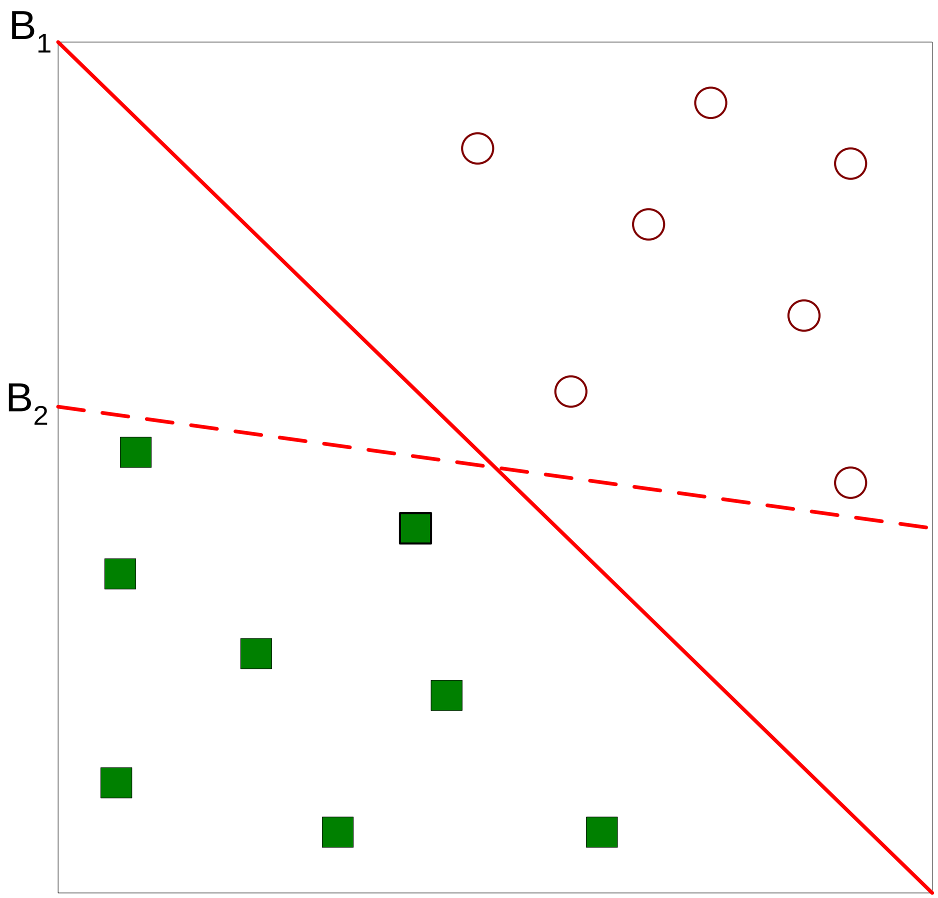
- One Possible Solution

Support Vector Machines



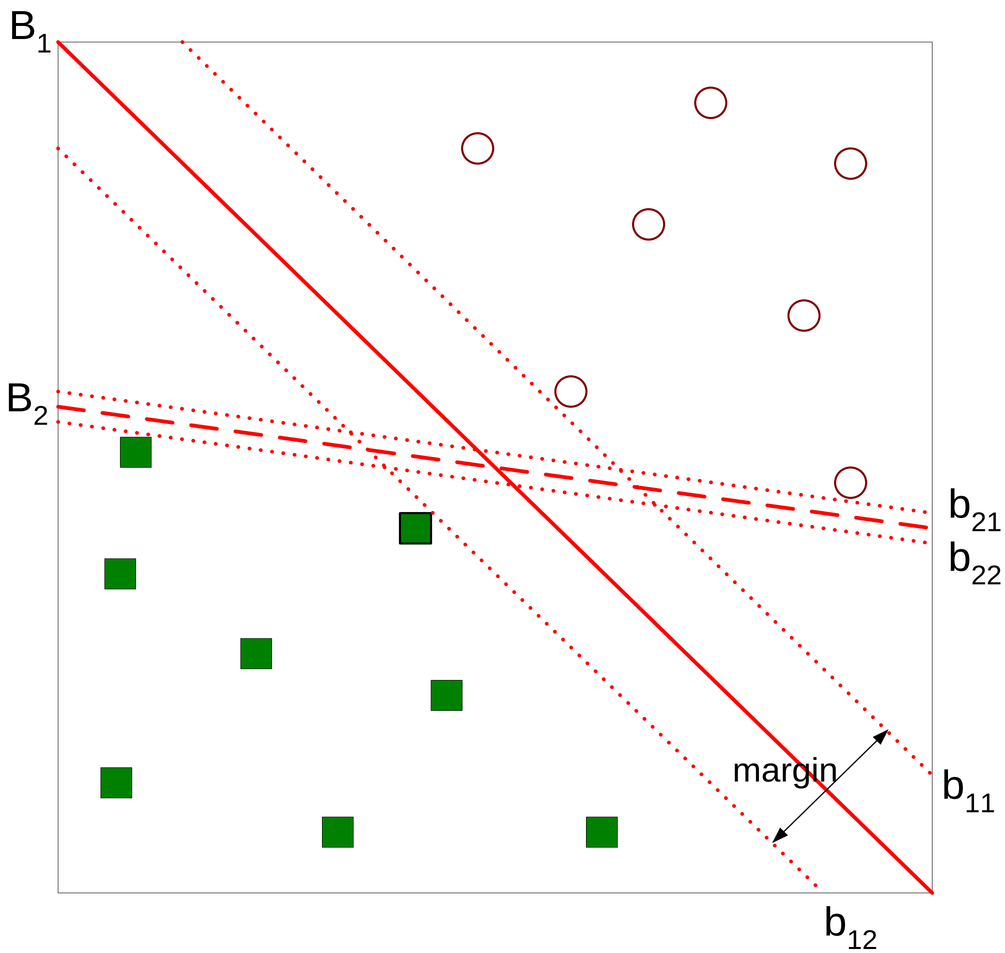
- Other possible solutions

Support Vector Machines



- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



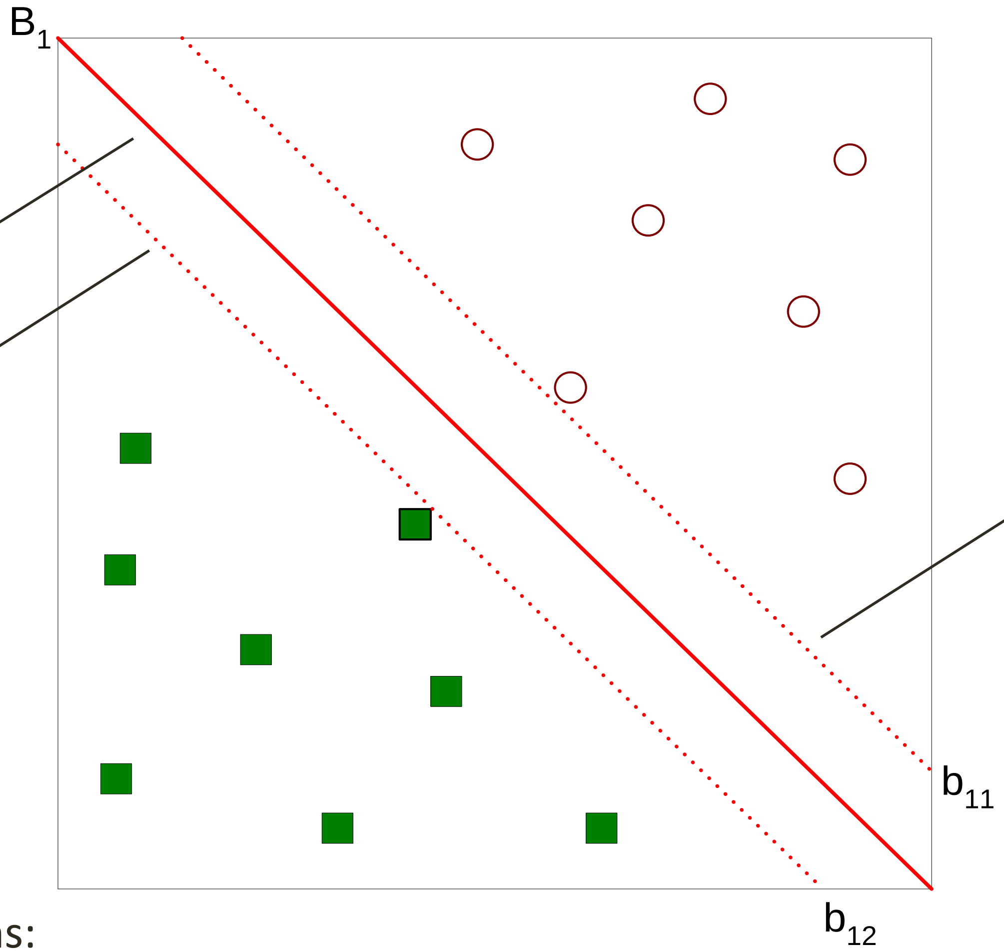
- Find hyperplane maximizes the margin => B1 is better than B2

Asst. Prof. Dr. Rachsuda Setthawong

Support Vector Machines

$w \cdot x + b = 0$

$w \cdot x + b = -1$



$w \cdot x + b = +1$

Decision Functions:

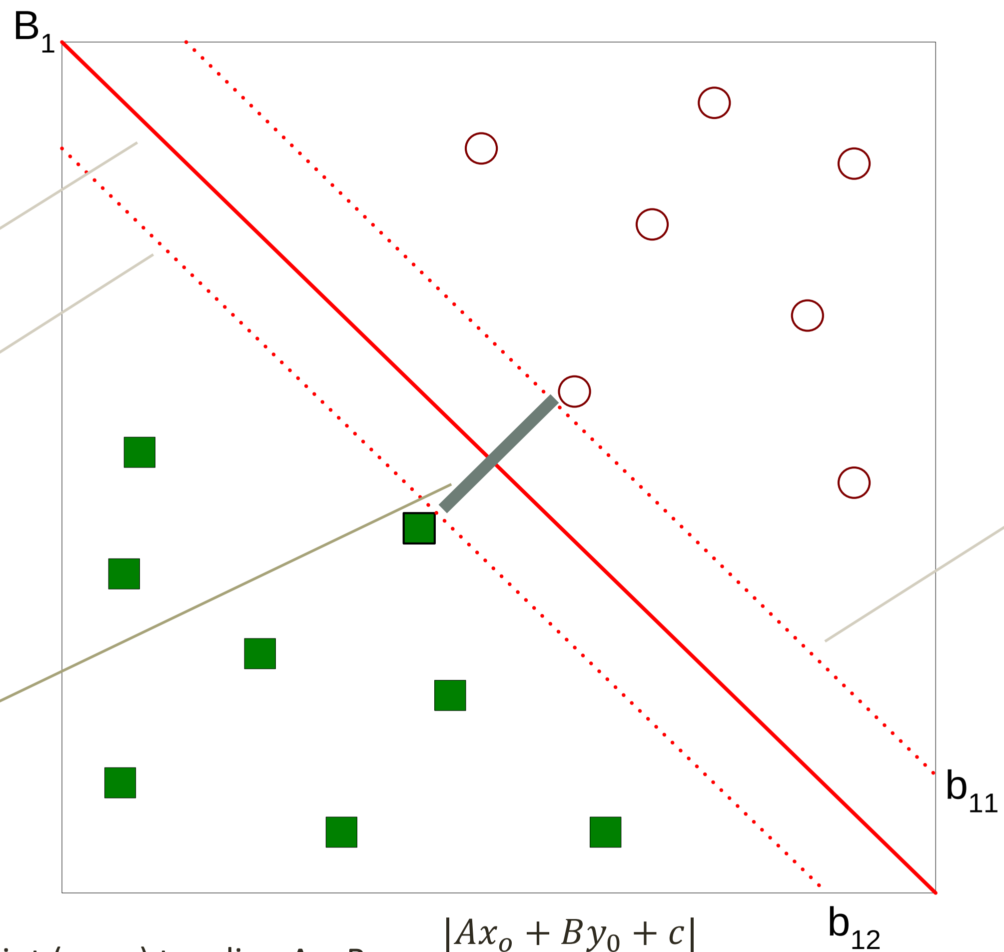
$$f(x) = \begin{cases} 1 & if\ w \cdot x + b \geq 1 \\ -1 & if\ w \cdot x + b \leq -1 \end{cases}$$

Asst. Prof. Dr. Rachsuda Setthawong

Support Vector Machines

$w \cdot x + b = 0$

$w \cdot x + b = -1$



$w \cdot x + b = +1$

Find Margin (d):

Distance between a point (x_0, y_0) to a line Ax+By+c: $\frac{|Ax_o + By_0 + c|}{\sqrt{A^2 + B^2}}$

Distance between B1 and b11 is then: $\frac{|w \cdot x + b|}{\|w\|} = \frac{1}{\|w\|}$

Total distance between b11 and b12 is then: $\frac{2}{\|w\|}$

Support Vector Machines

- We want to maximize: Margin = $\frac{2}{\|w\|}$
  - Equivalent to minimizing: $L(w) = \frac{\|w\|^2}{2}$
  - But subjected to the following constraints:

$$y_i(\vec{w} \cdot \vec{x_i} + \mathrm{b}) \geq 1, i = 1, 2, \ldots, N$$

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

Convex Optimization Problem

- Given
  - The objective function is quadratic.

$$min_w \frac{\|w\|^2}{2}$$

  - The constraints are linear in the parameter w and b.

$$y_i(\vec{w} \cdot \vec{x_i} + \text{b}) \geq 1, i = 1, 2, \ldots, N$$

- Solve **w** and **b** using **Lagrange multiplier** method
  - New objective function (Lagrangian)

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \lambda_i (y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1)$$

where $\boldsymbol{\lambda_i}$ is the **Lagrange multiplier**. ($\lambda_i \geq 0$)

Remark: able to obtain ***w*** *and b with any variation of* λ_i

Asst. Prof. Dr. Rachsuda Setthawong

Convex Optimization Problem

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \lambda_i (y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1) \qquad (1)$$

To minimize the Lagrangian,

$$\frac{\partial L_P}{\partial \boldsymbol{w}} = 0 \Rightarrow \boxed{\boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i}} \qquad (2)$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{N} \lambda_i y_i = 0} \qquad (3)$$

Substitute (2) and (3) in (1),

$$L_P = \frac{1}{2}\left(\sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i}\right) \cdot \left(\sum_{i=1}^{N} \lambda_j y_j \boldsymbol{x_j}\right) - \left(\sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i}\right) \cdot \left(\sum_{i=1}^{N} \lambda_j y_j \boldsymbol{x_j}\right) - \sum_{i=1}^{N} \lambda_i y_i\, b + \sum_{i=1}^{N} \lambda_i$$

Convex Optimization Problem

- **Dual formulation of the optimization problem**:

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j \boldsymbol{x_i x_j} \qquad (4)$$

- **Use quadratic programming to solve $\boldsymbol{\lambda_i}$** and obtain the feasible solutions for **w** and **b**.

- The decision boundary

$$\sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i} \cdot \boldsymbol{x} + \mathrm{b} = 0 \qquad (5)$$

An Example

Support vectors

x_1	x_2	y	Lagrange Multiplier
0.3858	**0.4687**	**1**	**65.5261**
0.4871	**0.611**	**-1**	**65.5261**
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

$$\overbrace{\sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i}}^{\mathbf{w}} \cdot \boldsymbol{x} + \mathrm{b} = 0$$

$$\mathrm{w}_1 = \sum_i \lambda_i y_i x_i = (65.5621 \times 1 \times 0.3858) + (65.5621 \times -1 \times 0.4871) = -6.64$$

$$\mathrm{w}_2 = \sum_i \lambda_i y_i x_i = (65.5621 \times 1 \times 0.4687) + (65.5621 \times -1 \times 0.611) = -9.32$$

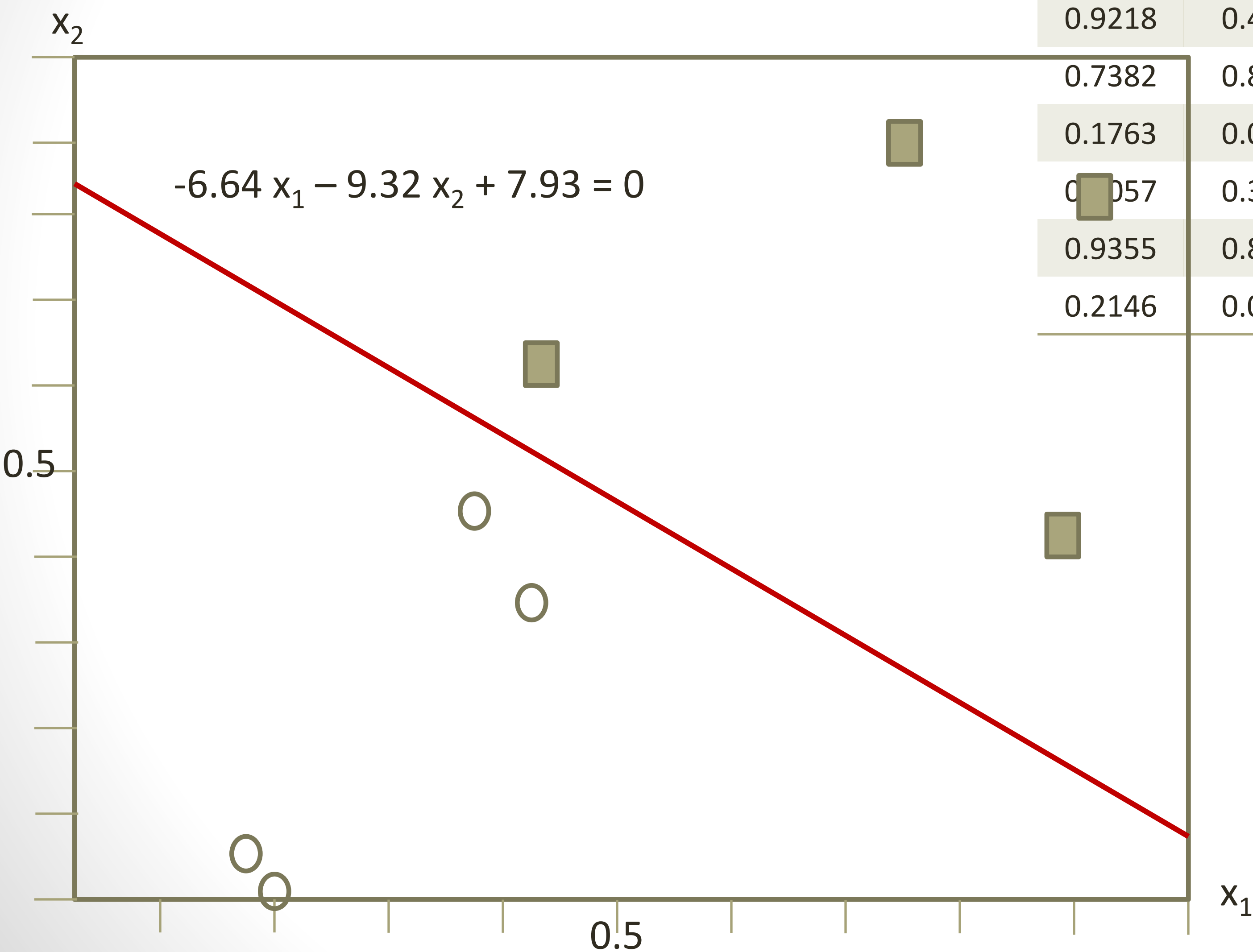
$b^{(1)} = 1 - \mathrm{w} \cdot x_1 = 1 - (-6.64)(0.3858) - (-9.32)(0.4687) = 7.93$

$b^{(2)} = 1 - \mathrm{w} \cdot x_2 = 1 - (-6.64)(0.4871) - (-9.32)(0.611) = 7.94$

$b_{avg} = 7.93$

An Example

x_1	x_2	y	Lagrange Multiplier
0.3858	**0.4687**	**1**	**65.5261**
0.4871	**0.611**	**-1**	**65.5261**
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0[illegible]057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0



SVM Classifier

x_1	x_2	y	Lagrange Multiplier
0.3858	**0.4687**	**1**	**65.5261**
0.4871	**0.611**	**-1**	**65.5261**
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

f(**z**) = sign(w · z + b)

$= \text{sign}\ (\sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x_i} \cdot \boldsymbol{x} + \text{b})$

f(**z**) = + → positive class

Otherwise, negative class

Given the decision boundary:

e.g.,

$-6.64\ x_1 - 9.32\ x_2 + 7.93 = 0$

z1 = sign[-6.44(0.3858) + -9.32(0.4687) + 7.93] = +(1.0771)

Positive class

z2 = sign[-6.44(0.4871) + -9.32(0.611) + 7.93] = -(0.9014)

Negative class

Asst. Prof. Dr. Rachsuda Setthawong

References

- Neural Networks Demystified [Part 1: Data and Architecture], Stephen Welch, https://www.youtube.com/watch?v=bxe2T-V8XRs
- Neural network tutorial: The back-propagation algorithm, Ryan harris, https://www.youtube.com/watch?v=zpykfC4VnpM
- Learning: Support Vector Machines, MIT OpenCourseWare, https://www.youtube.com/watch?v=_PwhiWxHK8o
- Slides: Lecture Notes for Chapter 5, Introduction to Data Mining by Tan, Steinbach, Kumar