



Lecture 4 Classifiers

Dr. Patrick Chan
patrickchan@ieee.org
South China University of Technology, China



Agenda

- ◆ Linear Discriminant Function (LDF)
- ◆ Decision Tree (DT)
- ◆ K nearest neighbour (k-NN)
- ◆ Support Vector Machine (SVM)



2

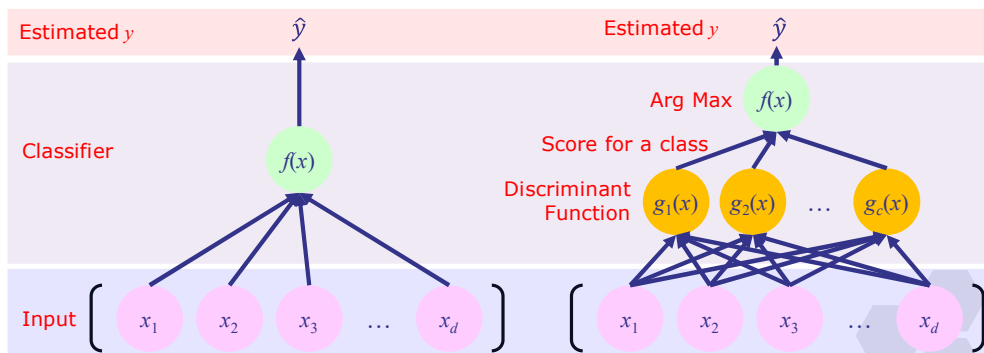
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Classifier

- ◆ Generally, a classifier $f(x)$ return a class label
- ◆ Some classifiers estimate the probability of x belongs to a class
 - Contains a set of discriminant functions $g_i(x)$, $i = 1, \dots, c$ indicates how likely x belongs to y_i
 - x is assigned to class y_i if $g_i(x)$ is max for $i = 1 \dots c$



3

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Linear Discriminant Function

- ◆ LDF: a linear combination of x

$$g(x) = \theta^T x$$

w : is the weight vector

* the last feature of x can be set as 1 if bias term is needed

- One feature value in x can be fixed when a constant term is needed

$$x = [1, x_1, x_2, \dots, x_d]^T$$

$$g(x) = \theta^T x + \theta_0$$



4

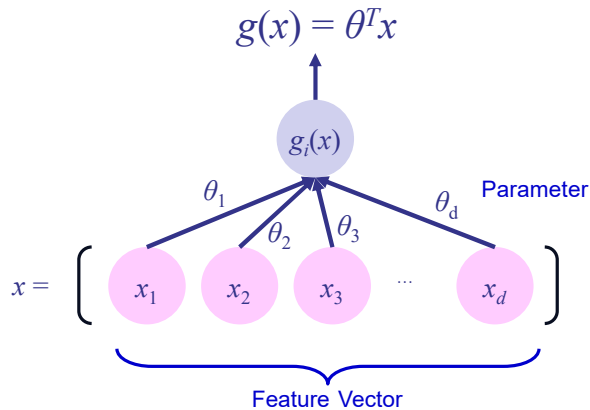
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Linear Discriminant Function

- ◆ For each class



5

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



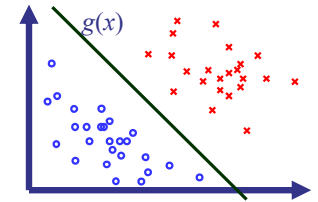
Linear Discriminant Function

- ◆ For a 2-class problem

- Only one classifier is needed
- $g(x) = g_1(x) - g_2(x)$

$$= \left((\theta^{(1)})^T - (\theta^{(2)})^T \right) x$$

$$= \theta^T x$$
 - If $g(x) > 0$, decide y_1
 - If $g(x) < 0$, decide y_2
 - If $g(x) = 0$, ambiguity



- All training samples are used,
 $y = \{1, -1\}$

Original Dataset

x	y
23	1
42	1
52	2
12	2

$g(x)$

x	y
23	1
42	1
52	-1
12	-1

6

Dr. Patrick Chan @ SCUT

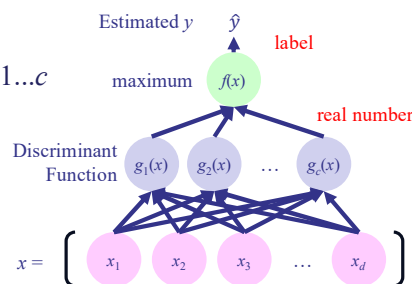
Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Linear Discriminant Function

- ◆ For a multi-class problem

- x is y_i if $g_i(x)$ is maximum for $i = 1 \dots c$
- c classifiers are required
- $g_i(x)$ represents the probability that x belongs to class i
 - We will discuss this later
- For $g_i(x)$, class $i = 1$, the rest = 0



Original Dataset

x	y
23	1
42	2
52	3
12	4



$g_1(x)$

x	$y^{(1)}$
23	1
42	0
52	0
12	0

$g_2(x)$

x	$y^{(2)}$
23	0
42	1
52	0
12	0

...

7

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Linear Discriminant Function

- ◆ How to train $g(x)$?

- LMS discussed in Regression previously can be used
- **Cost Function:** $J(\theta) = \frac{1}{2n} \sum_{i=1}^n (h_{\theta}(x^{(i)}) - y^{(i)})^2$
- Minimize by
 - Pseudoinverse
 - Gradient descent

8

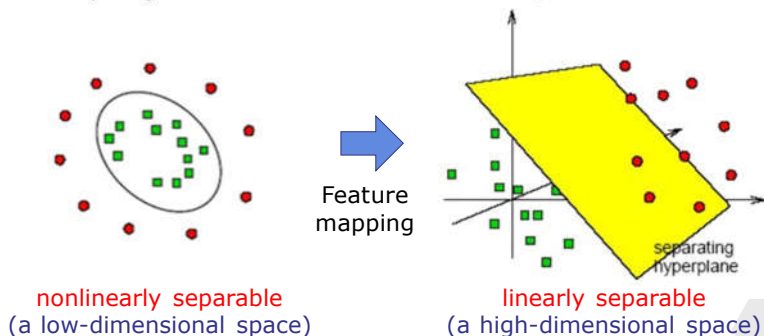
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Linear Discriminant Function Mapping

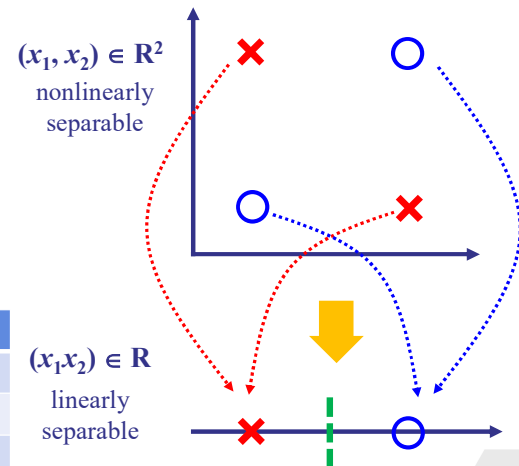
- Practically, a problem is seldom linearly separable
- How can LDF handle a non-linearly separable problem?
 - Map nonlinearly to linearly separable problem
 - Usually high-dimensional space is required



Linear Discriminant Function Mapping

XOR Example

x_1	x_2	y
1	1	1
-1	1	-1
1	-1	-1
-1	-1	1



x_1	x_2	$x_1 x_2$	y
1	1	1	1
-1	1	-1	-1
1	-1	-1	-1
-1	-1	1	1

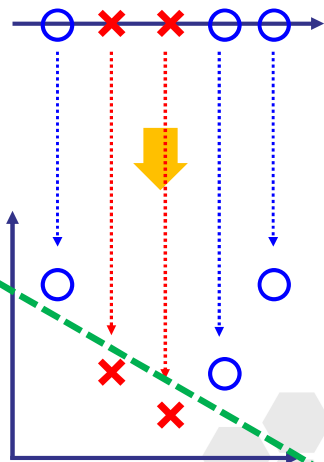


Linear Discriminant Function Mapping

Another Example

x	y
-2	1
-1	-1
0	-1
1	1
2	1

$(x) \in \mathbb{R}$
nonlinearly separable



x	x	x^2	y
-2	-2	4	1
-1	-1	1	-1
0	0	0	-1
1	1	1	1
2	2	4	1



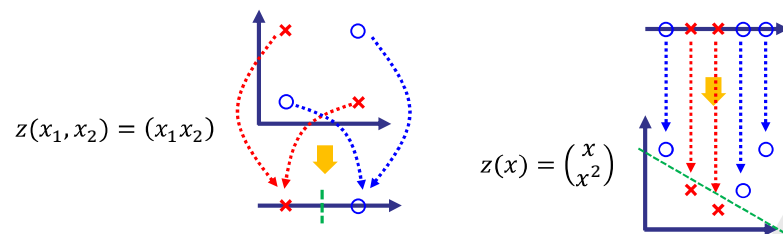
Linear Discriminant Function Mapping

Generalized Linear Discriminant Function

$$g(x) = w^t z(x)$$

- z is a mapping from x to $z(x)$
- g is not linear in x , but linear in z

For example



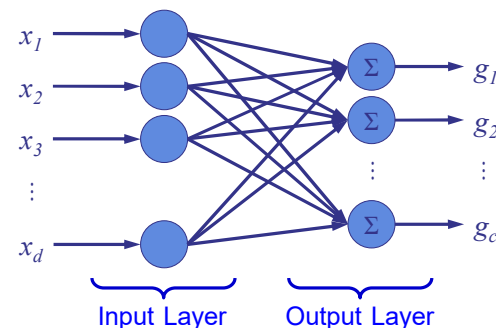


Mapping

- ◆ How to determine the feature space?
 - How to design a proper high dimensional space automatically by learning?
 - ANN, SVM...

Multi-Layer Perceptron
Introduction

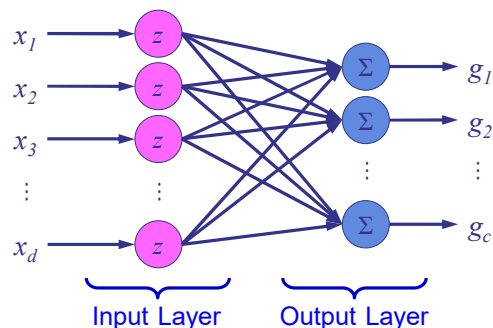
- ◆ Recall, **Linear Discriminant Functions**:



- **Limited** generalization capability
- Cannot handle the non-linearly separable problem

Multi-Layer Perceptron
Introduction

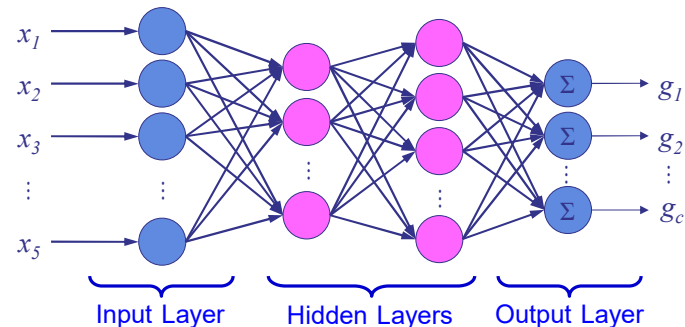
- ◆ Solution 1: **Mapping Function $\phi(x)$**



- Pro: Simple structure (still using LDF)
- Cons: Selection of $z(x)$ and its parameters

Multi-Layer Perceptron
Introduction

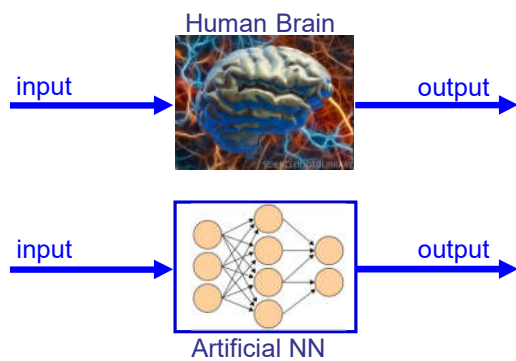
- ◆ Solution 2: **Multi-Layer Neural Network**



- Standard structure
 - Hidden layers serve as mapping
- No prior knowledge is required (no need to choose $z(x)$)



- ANN is inspired biologically by human brain



- Most classifiers are black-box
- DT provides explanation on decisions
- One of the most widely used and practical methods for inductive inference
- Approximates discrete-valued functions (including disjunctions)



- Do we go to play tennis today?

If Outlook is Sunny AND Humidity is Normal

Yes

If Outlook is Overcast

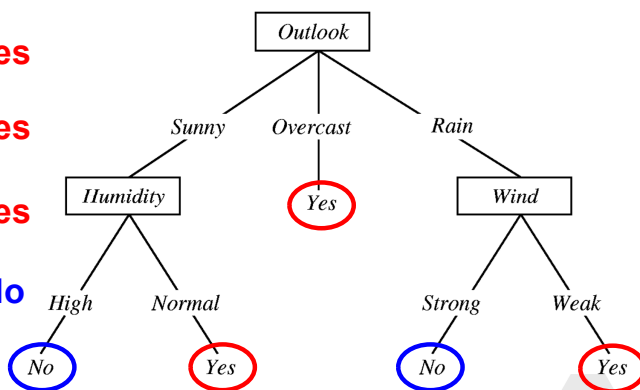
Yes

If Outlook is Rain AND Wind is Weak

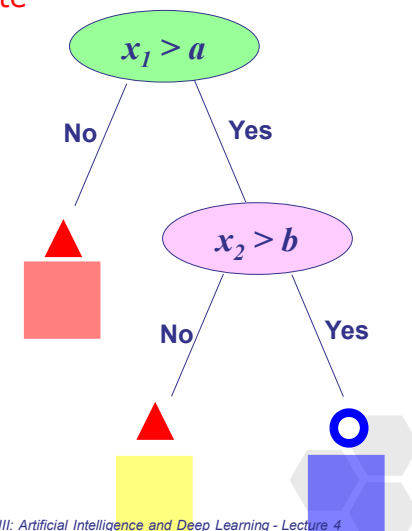
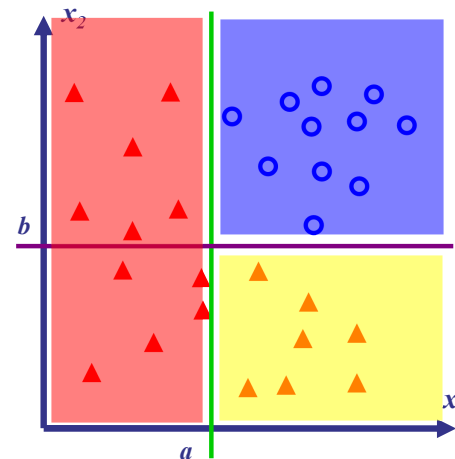
Yes

Other situation?

No



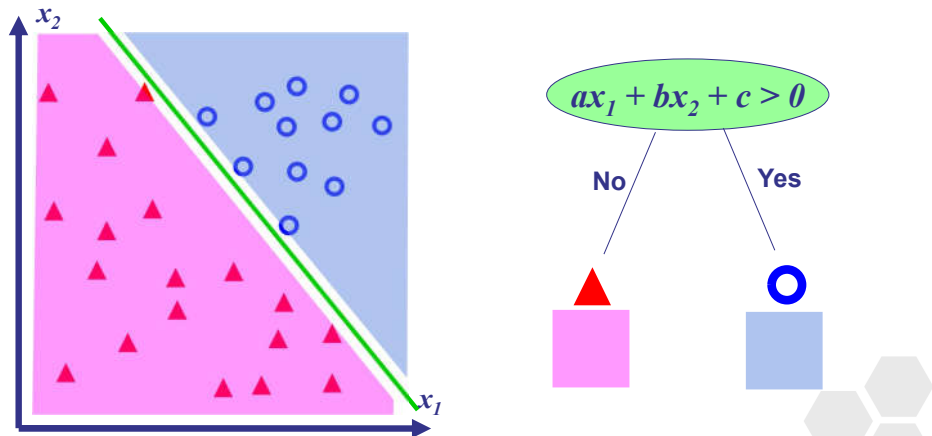
- Decision Region:
 - Internal nodes can be univariate
 - (Only one feature is used)





DT: Classification

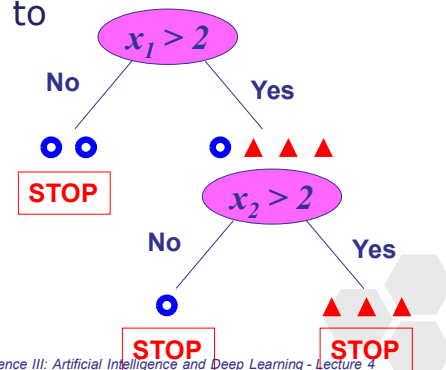
- ◆ **Internal nodes** can be **multivariate**
 - More than one features are used
 - Shape of Decision Region is irregular



DT: Learning Algorithm

- ◆ **LOOP:**
 1. Select the **best feature (A)**
 2. For each value of A, **create new descendant of node**
 3. **Sort** training samples to **leaf nodes**
- ◆ **STOP** when training samples perfectly classified

#	x_1	x_2	x_3	
1	1	3	5	○
2	1	4	2	○
3	3	1	5	○
4	3	5	6	▲
5	3	3	4	▲
6	4	5	7	▲



DT: Learning Algorithm

- ◆ **Observation**
 - Many trees may code a training set without any error
 - Finding the **smallest tree** is a **NP-hard problem**
- ◆ **Local search algorithm** to find **reasonable solutions**
 - What is the best feature?



DT: Feature Measurement

- ◆ **Entropy** is used to **evaluate features**
 - Measure of **uncertainty**
 - Range: **0 - 1**
 - **Smaller value, less uncertainty**

$$H(X) = - \sum_{i=1}^n p(x_i) \log(p(x_i))$$

where

X : a random variable with n outcomes, $X = \{x_i | i = 1, 2, \dots, n\}$
 $p(x)$: the probability mass function of outcome x .

- If all samples belongs to x_i , then $p(x_i) = 1$, and other $p(x_j) = 0$, $i \neq j$
 - Thus, $H(X) = 0$ (no uncertainty)



DT: Feature Measurement

◆ Information Gain

- Reduction in entropy (reduce uncertainty) due to sorting on a feature A

$$Gain(X, A) = H(X) - H(X|A)$$

Current entropy
Entropy after using feature A



DT: Example

◆ Which feature is the best?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$H(X) = - \sum_{i=1}^n p(x_i) \log_2 p(x_i)$$

$$x_1 = \text{yes} \quad x_2 = \text{No}$$

$$\begin{aligned}
 H(X) &= - \sum_{i=1}^2 p(x_i) \log_2 p(x_i) \\
 &= - \frac{9}{14} \log_2 \left(\frac{9}{14} \right) - \frac{5}{14} \log_2 \left(\frac{5}{14} \right) \\
 &= 0.410 + 0.531 \\
 &= 0.941
 \end{aligned}$$

Current: $H(X) = 0.941$

Uncertainty is high without considering any feature



DT: Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Let A = Outlook



Recall:

$$Gain(X, A) = H(X) - H(X|A)$$

$$\begin{aligned}
 H(X|A) &= H(X|A = \text{sunny})P(A = \text{sunny}) + \\
 &H(X|A = \text{Rain})P(A = \text{Rain}) + \\
 &H(X|A = \text{overcast})P(A = \text{overcast})
 \end{aligned}$$



DT: Example

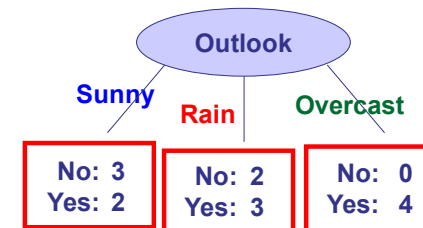
$$\begin{aligned}
 H(X|A) &= H(X|A = \text{sunny})P(A = \text{sunny}) + \\
 &H(X|A = \text{Rain})P(A = \text{Rain}) + \\
 &H(X|A = \text{overcast})P(A = \text{overcast})
 \end{aligned}$$

$$H(X| \text{sunny}) = -\frac{3}{5} \log_2 \left(\frac{3}{5} \right) - \frac{2}{5} \log_2 \left(\frac{2}{5} \right) = 0.971$$

$$H(X| \text{Rain}) = -\frac{2}{5} \log_2 \left(\frac{2}{5} \right) - \frac{3}{5} \log_2 \left(\frac{3}{5} \right) = 0.971$$

$$H(X| \text{overcast}) = -\frac{0}{4} \log_2 \left(\frac{0}{4} \right) - \frac{4}{4} \log_2 \left(\frac{4}{4} \right) = 0$$

Let A = Outlook



$$\begin{aligned}
 H(X|A) &= 0.971 \times \left(\frac{5}{14} \right) + 0.971 \times \left(\frac{5}{14} \right) + 0 \times \left(\frac{4}{14} \right) \\
 &= 0.694
 \end{aligned}$$



DT: Example

Outlook is the best feature and
Should be used as the first node

Current: $H(X) = 0.941$

Similarly, for each feature

$$H(X | \text{Outlook}) = 0.694$$

$$H(X | \text{Temperature}) = 0.911$$

$$H(X | \text{Humidity}) = 0.789$$

$$H(X | \text{Wind}) = 0.892$$

Information Gain is:

$$\text{Gain}(X, \text{Outlook}) = 0.247$$

$$\text{Gain}(X, \text{Temperature}) = 0.030$$

$$\text{Gain}(X, \text{Humidity}) = 0.152$$

$$\text{Gain}(X, \text{Wind}) = 0.049$$

Recall:

$$\text{Gain}(X, A) = H(X) - H(X | A)$$

29

Dr. Patrick Chan @ SCUT

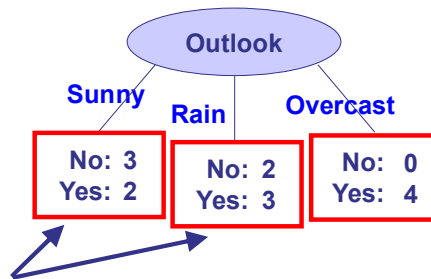
Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



DT: Example

Next Step

- Repeat the steps for each sub-branch
- Until there is no ambiguity
(all samples are of the same class)



Continues to select next features

Done

30

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



DT: Continuous-Valued Feature

- So far, we handle features with categorical values
- How to build a decision tree whose features are numerical?

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	29.9	High	Weak	No
D2	Sunny	28.2	High	Strong	No
D3	Overcast	35.2	High	Weak	Yes
D4	Rain	26.4	High	Weak	Yes
D5	Rain	18.9	Normal	Weak	Yes
D6	Rain	21.2	Normal	Strong	No
D7	Overcast	20.4	Normal	Strong	Yes
D8	Sunny	24.4	High	Weak	No
D9	Sunny	17.0	Normal	Weak	Yes
D10	Rain	25.1	Normal	Weak	Yes
D11	Sunny	24.0	Normal	Strong	Yes
D12	Overcast	24.5	High	Strong	Yes
D13	Overcast	27.7	Normal	Weak	Yes
D14	Rain	25.5	High	Strong	No

31

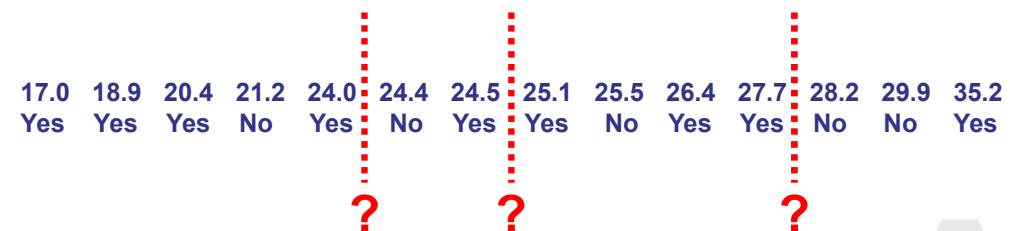
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



DT: Continuous-Valued Feature

- Accomplished by partitioning the continuous attribute value into a discrete set of intervals
- A new Boolean feature A_c ($A < c$) can be created, c is the threshold
- How to select the best value for c ?



32

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



DT: Continuous-Valued Feature

- ◆ Objective is to **minimize the entropy** (or **maximize the information gain**)
- ◆ Entropy only needs to be evaluated between points of different classes



- ◆ Best c is: $c^* = \arg \max_c \text{Gain}(X, A_c)$

33

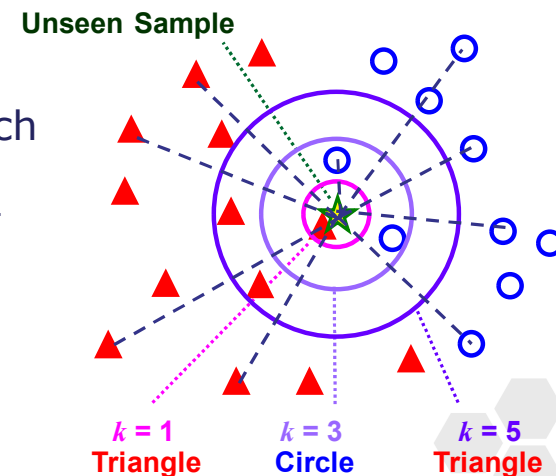
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-Nearest Neighbor (K-NN)

- ◆ A new pattern is classified by a majority vote of its k nearest neighbors (training samples)
- ◆ n distances are calculated for each new sample
 - n : the number of training samples



34

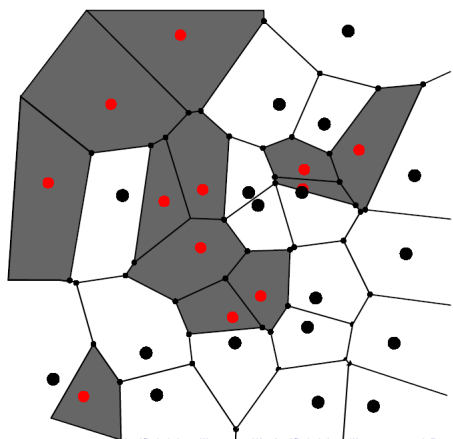
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-Nearest Neighbor (K-NN)

- ◆ Target function for the entire space may be described as a combination of less complex local approximations



35

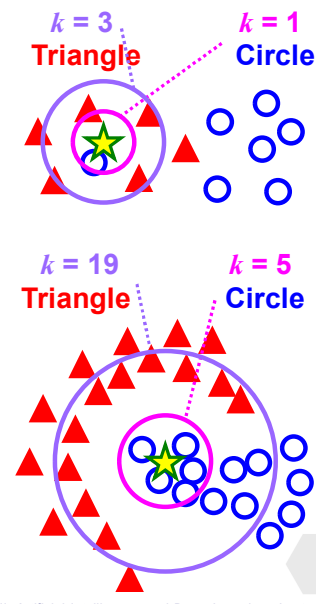
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-Nearest Neighbor (K-NN)

- ◆ How to determine k ?
 - Small k
 - Noise Sensitive
 - Large k
 - Neighbours may be too far away from the unseen sample
 - Less representative



36

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-NN: Characteristic

- ◆ **Advantages:**
 - Very **simple**
 - **No training** is needed
 - All computations deferred until classification
- ◆ **Disadvantages:**
 - **Difficult** to determine k
 - **Affected** by **noisy** training data
 - **Classification** is **time consuming**
 - Need to calculate the distance between the unseen sample and each training sample

37

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-NN: Advanced Discussion

- ◆ **No need for training** is a significant advantage
- ◆ Which applications need it?
- ◆ **Classes update frequently**
- ◆ E.g. Face Recognition
 - When face images of a new user is added, will the system be re-trained?
 - How can adding a new class without retraining be possible?

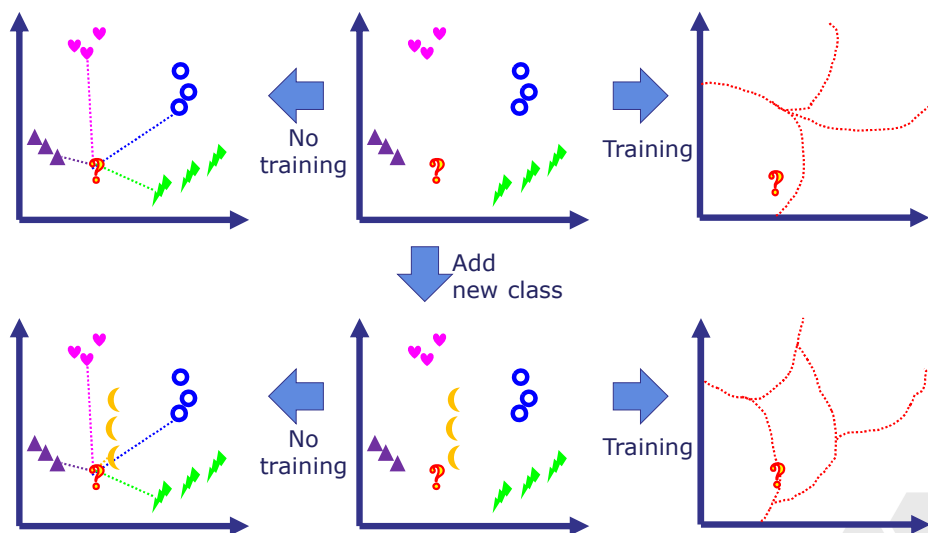
38

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



K-NN: Advanced Discussion



39

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4

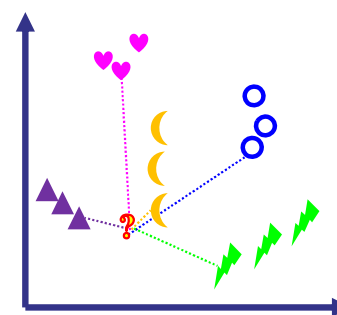


K-NN: Advanced Discussion

Query-based Framework

◆ Query-based framework

- K-nn mechanism



Gallery: reference samples, which are few good quality samples representing a class

Query: unseen sample

40

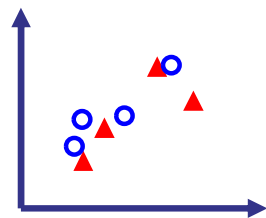
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Space

- ◆ However, in the original feature space, are samples that are **more similar** to each other **more likely to belong to the same class**?



- ◆ K-NN may not be useful in the original feature space

41

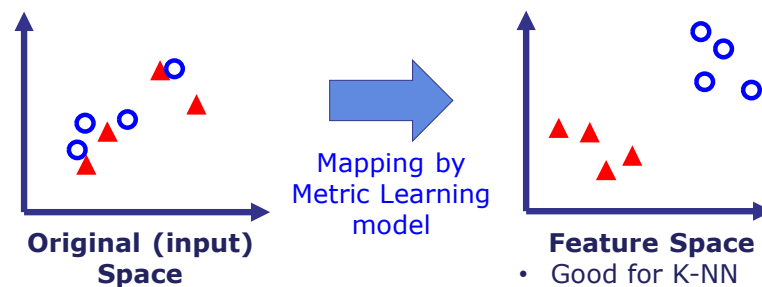
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Metric Learning

- ◆ **Metric Learning**: Aim to obtain a space which fulfills a certain measure
 - E.g. get closer > higher chance to be the same class



- K-NN may not work well here

42

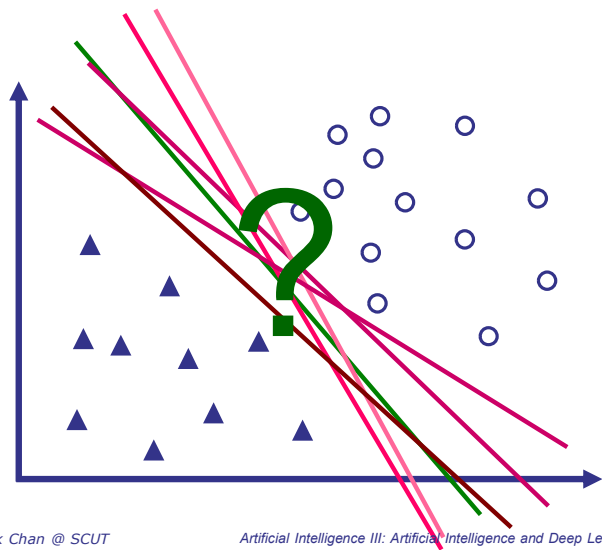
Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Support Vector Machine (SVM)

- ◆ Which one is the **best** linear separator?



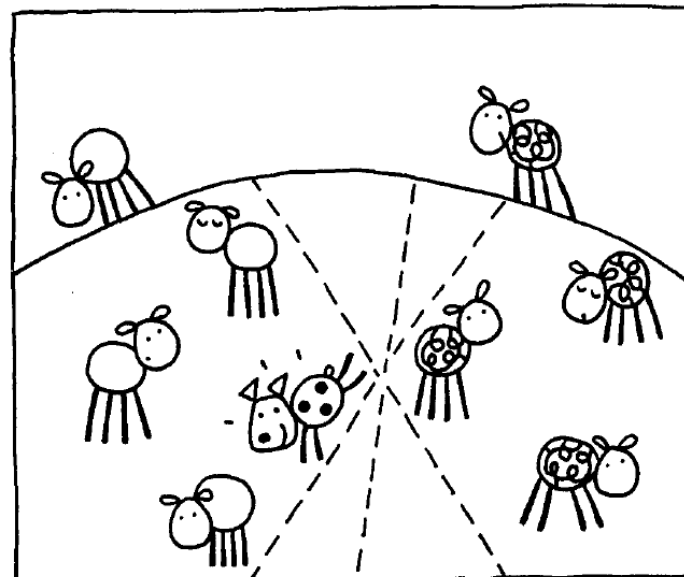
43

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Support Vector Machine (SVM)



A clever sheep dog who was herding his sheep...

It runs between the sheep and tries to separate the black sheep and white sheep



Clever Sheep dog



White Sheep



Black Sheep

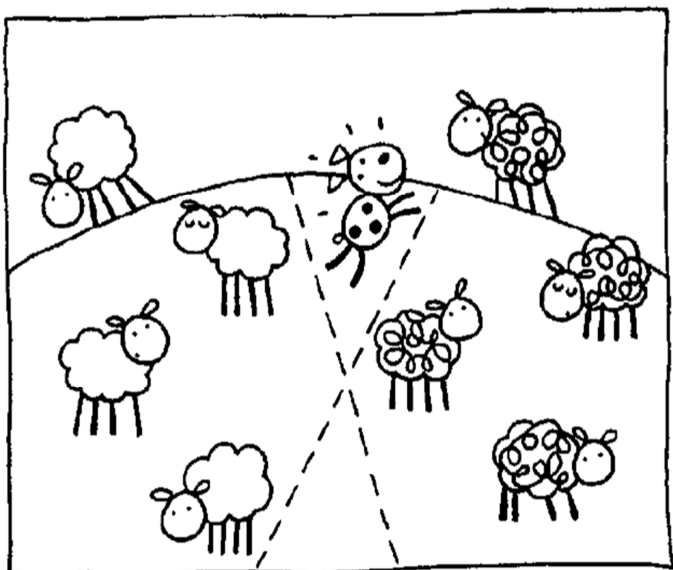
44

Dr. Patrick Chan @ SCUT

Artificial Intelligence III: Artificial Intelligence and Deep Learning - Lecture 4



Support Vector Machine (SVM)



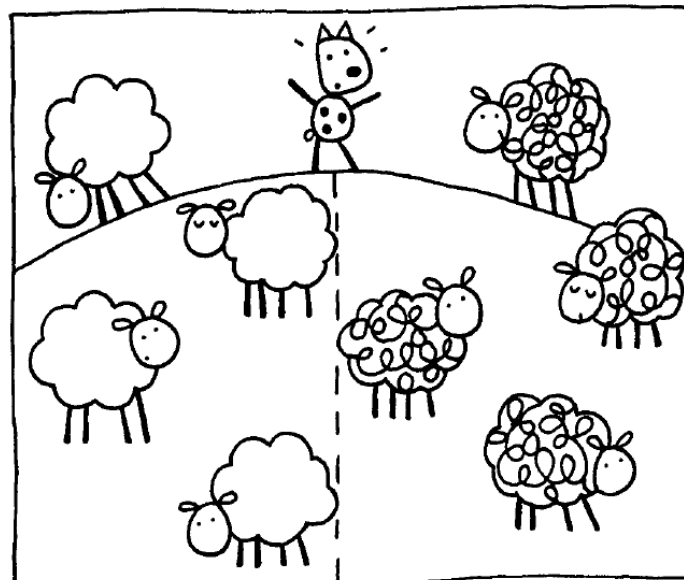
The sheep dog keeps running...

The sheep start to grow wools...

The dog feel the gap between black sheep and white sheep is narrower...



Support Vector Machine (SVM)

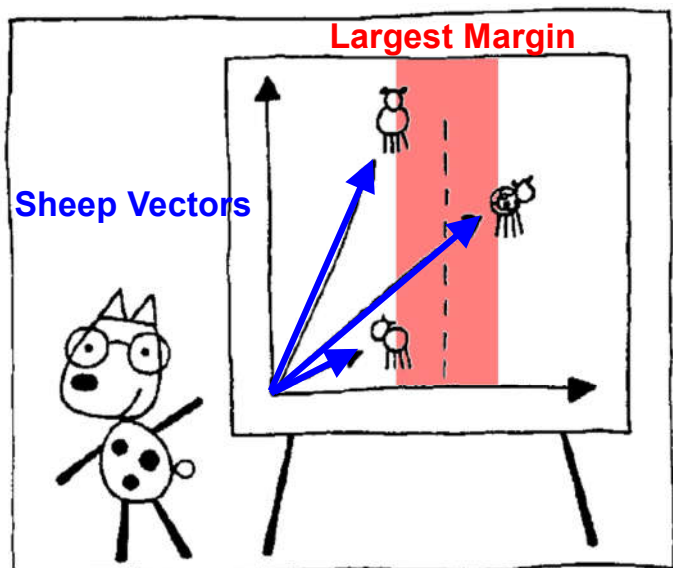


The wools become bigger and bigger...

Finally, only one path is left..



Support Vector Machine (SVM)



Largest Margin

The sheep found out that the single path relies only on the some sheep.

These sheep are "sheep vectors"

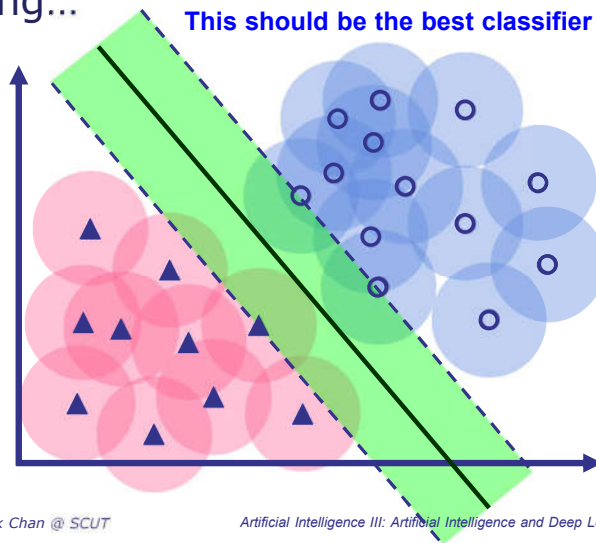
* "Support vector" in SVM



Support Vector Machine (SVM)

◆ Similar to the sheep, if a sample is growing...

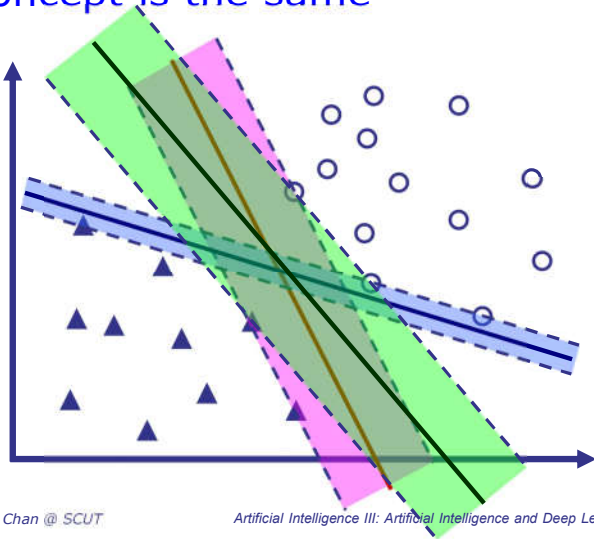
This should be the best classifier





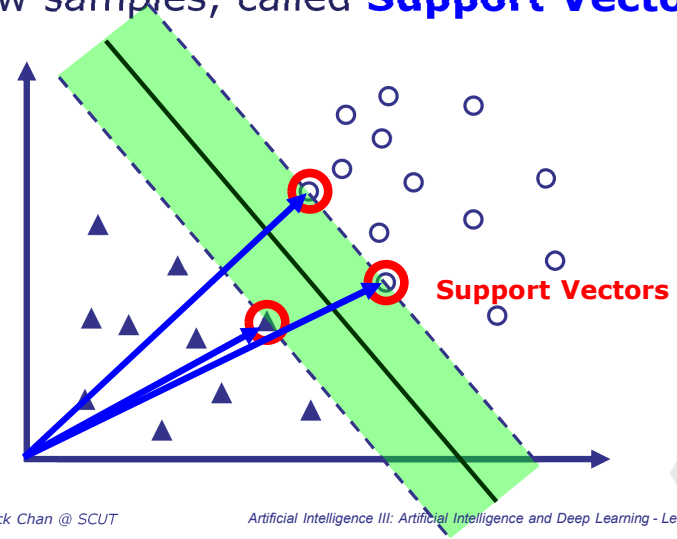
Support Vector Machine (SVM)

- ◆ A classifier with the **largest margin**
- ◆ The concept is the same



Support Vector Machine (SVM)

- ◆ **Maximum Margin Classifier** ONLY depends on few samples, called **Support Vectors**



Support Vector Machine (SVM)

- ◆ SVM has a solid and strong mathematics background
- ◆ Only the concept of SVM is focused in this course

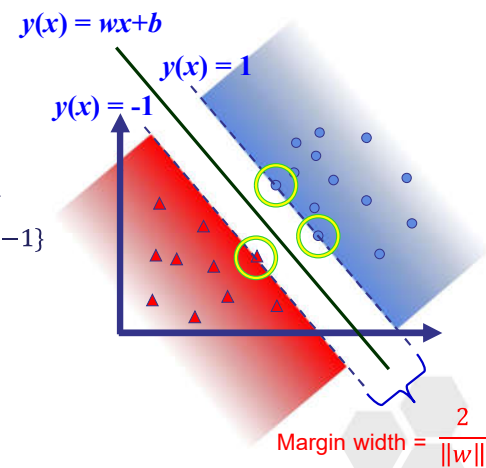


SVM: Linearly Separable

- ◆ Problem can be formulated as **Quadratic Optimization Problem** and solve for w and b

$$\begin{aligned} &\text{minimize}_{w, b} \quad \frac{1}{2} \|w\|^2 \\ &\text{subject to} \quad y_i(w^T x_i + b) \geq 1 \\ &\quad \text{where} \quad i = 1 \dots n \quad \text{and} \quad y = \{1, -1\} \end{aligned}$$

All samples should be behind the margin



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



SVM: Linearly Separable

- ◆ This **optimization problem** can be formulated as **Dual Problem** using Lagrangian method:

$$\begin{aligned} & \underset{w, b}{\text{minimize}} && \frac{1}{2} \|w\|^2 \\ & \text{subject to} && y_i(w^T x_i + b) \geq 1 \\ & \text{where} && i = 1 \dots n \quad \text{and} \\ & && y = \{1, -1\} \end{aligned}$$

$$\begin{aligned} & \text{Maximum } \alpha \\ & \text{subject to} \quad \alpha_i \geq 0 \quad i = 1 \dots n \quad \text{and} \quad \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

$$\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j$$

- ◆ Weight is determined by: $w = \sum_{i=1}^n \alpha_i y_i x_i$

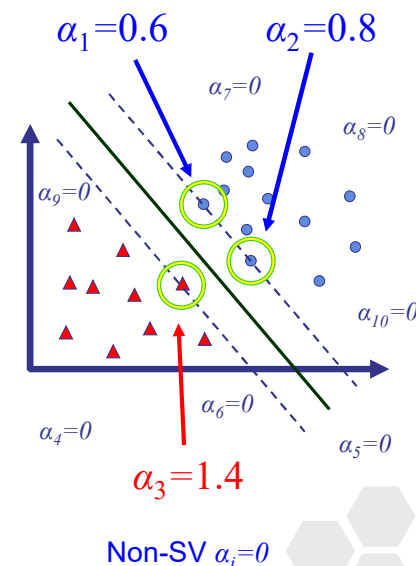


SVM: Linearly Separable

$$w = \sum_{i=1}^n \alpha_i y_i x_i \quad \sum_{i=1}^n \alpha_i y_i = 0$$

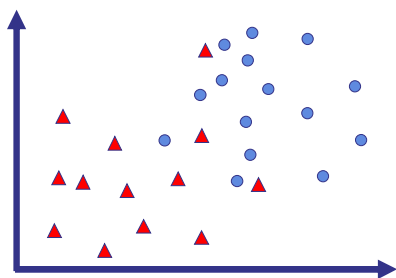
- ◆ Many α_i are zero
- ◆ x_i with non-zero α_i are **support vectors (SV)**
 - The decision boundary is determined only by the SV
- ◆ Let t_j ($i = 1, \dots, l$) be the indices of the support vectors.

$$w = \sum_{j=1}^l \alpha_{t_j} y_{t_j} x_{t_j}$$



SVM: Non-Linearly Separable

- ◆ How about **Non-Linearly Separable Case**?
 - The margin cannot be defined anymore



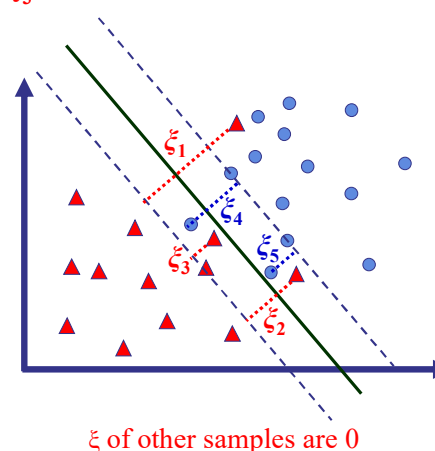
- ◆ Two approaches:
 - Add a slack variables
 - Use a **kernel** (Non-Linear SVM)



SVM: Non-Linearly Separable Slack Variable

$\xi_1 > 1$ **Error** $\xi_4 > 1$ **Error**
 $\xi_2 > 1$ **Error** $\xi_5 < 1$ **Correct**
 $\xi_3 < 1$ **Correct**

- ◆ **Slack Variable (ξ)** is added as a **punishment** to allow a sample in / far away from the margin



- ◆ **Optimization:**

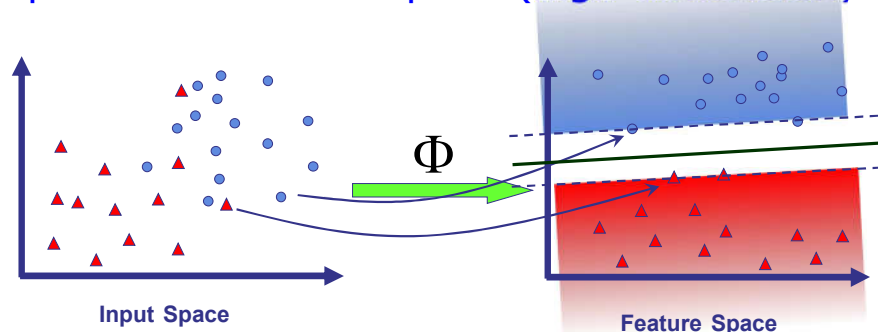
$$\begin{aligned} & \underset{w, b}{\text{Minimize}} && \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i \\ & \text{subject to} && y_i(w^T x_i + b) \geq 1 - \xi_i \quad i = 1 \dots N \\ & && \xi_i \geq 0 \end{aligned}$$

where C : tradeoff parameter between error and margin

Punishment allow a sample not behind the margin



- ◆ **Kernel** is a function which maps input space into feature space (high dimension)



- ◆ Construct linear SVM in feature space



- ◆ Similar to the linearly separable case but change all inner products to kernel functions

$$\begin{aligned} &\underset{w, b}{\text{minimize}} && \frac{1}{2} \|w\|^2 \\ &\text{subject to} && y_i(w^T \varphi(x_i) + b) \geq 1 \quad i = 1 \dots N \end{aligned}$$

$$\begin{aligned} &\underset{\alpha}{\text{maximum}} && \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ &\text{subject to} && \alpha_i \geq 0 \quad i = 1 \dots n \quad \text{and} \quad \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$



SVM: Multi-Class Problem

- ◆ SVM is a binary classifier
- ◆ **How to handle multi-class problem?**
 - g in LDF can be formulated as the estimation on posterior probability to a class
 - However, **SVM must consider two classes**
 - Do not estimate the probability of a class
 - Max method cannot be applied to SVM

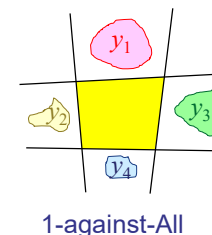


SVM: Multi-Class Problem

- ◆ How to handle multi-class problem?

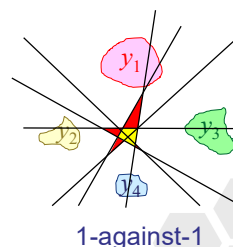
- **1-against-All**

- 4-class example
 - C1 vs Non-C1, C2 vs Non-C2, C3 vs Non-C3, C4 vs Non-C4
- c classifiers



- **1-against-1**

- 4-class example
 - C1 vs C2, C1 vs C3, C1 vs C4, C2 vs C3, C2 vs C4, C3 vs C4
- $c(c-1)/2$ classifiers





SVM: Characteristic

◆ Advantages

- Training is relatively easy
 - No local optimal
- It scales relatively well to high dimensional data (inner product)
- Tradeoff between classifier complexity and error can be controlled explicitly

◆ Disadvantage

- Slow when the number of samples is large
- Need to choose a “good” kernel function

