Pattern Recognition

Lecture 13. Linear Discriminant Functions and decision hyperplanes

Dr. Shanshan ZHAO

shanshan.zhao@xjtlu.edu.cn

School of AI and Advanced Computing

Xi'an Jiaotong-Liverpool University

Academic Year 2021-2022

Table of Contents

Notations

- w : a scalar
- $\mathbf{w}$: a vector
- c : denotes the class

Introduction

Generative methods

- Parametric Methods
- non-Parametric Methods

Discriminative methods

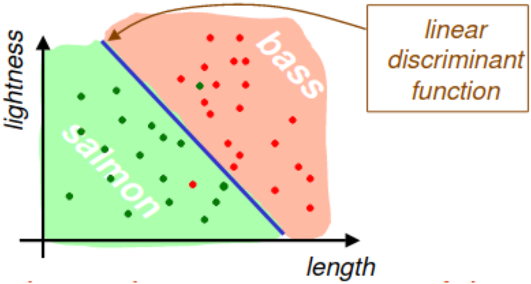
- Distance-based methods
- Linear Discriminant Functions
  - Hyperplane Geometry
- Artificial Neural Networks
- Support Vector Machines

Role of Linear Discriminant Functions

- A Discriminative Approach, as apposed to Generative approach of Parameter Estimation
- Leads to perceptrons and Artificial Neural Networks
- Leads to Support Vector Machines

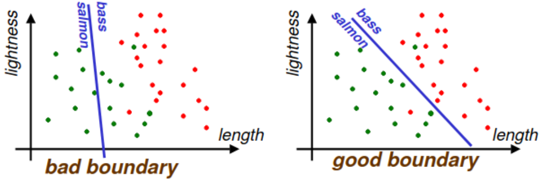
Preliminaries

- No probability distribution (no shape or parameters are known).
- Data with labels.
- The shape of discriminant functions is known.



- Need to estimate parameters of the discriminant functions.
- The problem of finding a linear discriminant function will be formulated as a problem of minimizing a criterion function.
- For classification purposes, the obvious criterion function is **sample risk** or **training error**.

LDF: Basic idea



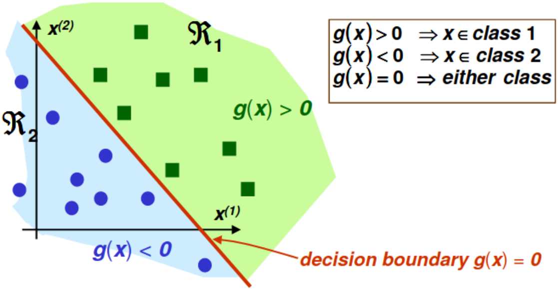
- Have samples from 2 classes $x_1, x_2, ..., x_n$.
- Assume 2 classes can be separated by a linear boundary $l(\theta)$ with some unknown parameters θ.
- Fit the "best" boundary to data by optimizing over parameters θ.
  - Minimize classification error on training data is an option

LDF: 2 Classes

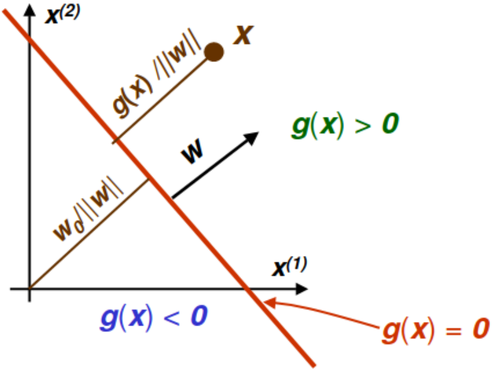
A discriminant function is linear if it can be written as

$$g(x) = w^t x + w_0 \tag{1}$$

- w is called the weight vector, and w_0 called bias or threshold



LDF: 2 Classes



$$g(x) = w^t x + w_0 = 0 \tag{2}$$

- w determines orientation of the decision hyperplane
- w_0 determines location of the decision surface

LDF: 2 Classes

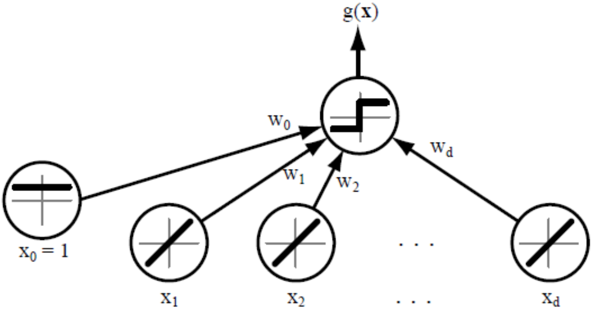


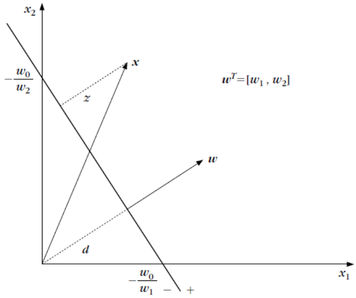
Figure: A simple linear classifier having d input units, each corresponding to the values of the components of an input vector. Each input feature value x_i is multiplied by its corresponding weight w_i; the output unit sums all these products and emits a +1 if $w^t x + w_0 > 0$ or a -1 otherwise.[1]

LDF: 2 Classes

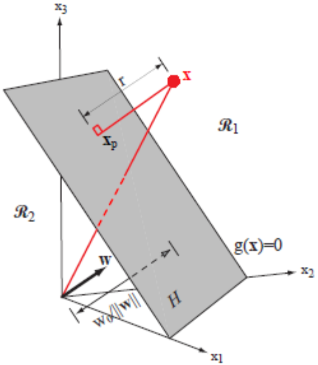
Decision boundary $g(x) = w^t x + w_0 = 0$ is

- a point in 1D
- a line in 2D
- a plane in 3D

LDF: 2 Classes



(a) 2D [2]



(b) 3D [1]

Figure: The linear decision boundary H, where $g(x) = w^t x + w_0 = 0$, separates the feature space into two half-spaces R1 (where $g(x) > 0$) and R2 (where $g(x) < 0$).

LDF: Multiple Classes

- We have M classes
- Define M linear discriminant functions

$$g_i(x) = w_i^T x + w_{i0} \quad i = 1, ..., M \tag{3}$$

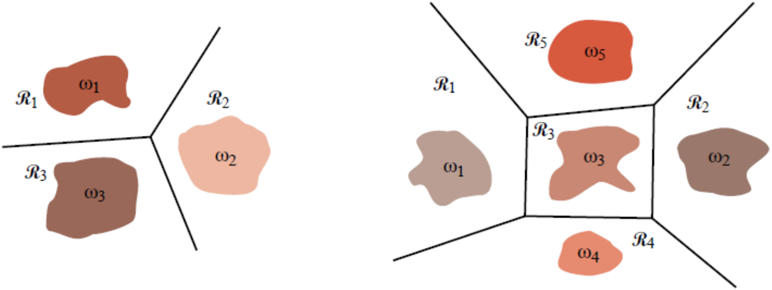
- Given x, assign class c_i if

$$g_i(x) \geq g_j(x) \quad \forall j \neq i \tag{4}$$

- Such classifier is called a **linear machine**
- A linear machine divides the feature space into M decision regions, with $g_i(x)$ being the largest discriminant if x is in the regions R_i.

LDF: Multiple Classes

Linear machine



LDF: Multiple Classes

- For a two contiguous regions R_i and R_j; the boundary that separates them is a portion of hyperplane H_{ij} defined by:

$$g_i(x) = g_j(x) \Longleftrightarrow w_i^T x + w_{i0} = w_j^T x + w_{j0} \tag{5}$$

$$\Longleftrightarrow (w_i - w_j)^T x + (w_{i0} - w_{j0}) = 0 \tag{6}$$

- $w_i - w_j$ is normal to H_{ij}
- Distance from x to H_{ij} is given by

$$d(x, H_{ij}) = \frac{g_i(x) - g_j(x)}{||w_i - w_j||} \tag{7}$$

LDF: Multiple Classes

applicability of linear machine to mostly limited to unimodal conditional densities $p(x|\theta)$

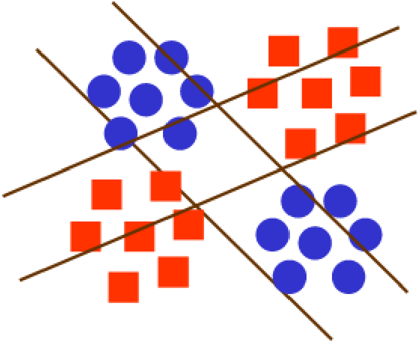


Figure: This an example where linear machine will fail

LDF: Augmented feature vector

- Linear discriminant function: $g(x) = w^T x + w_0$
- It can be rewritten as :

$$g(x) = \begin{bmatrix} w_0 & w^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = a^T y = g(y) \tag{8}$$

- y is called the augmented feature vector
- Add a dummy dimension to get a completely equivalent new Homogeneous problem

old problem: $g(x) = w^T x + w_0$

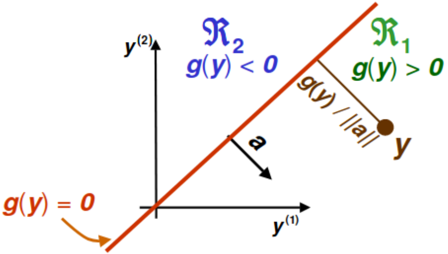
$$\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$$

new problem: $g(y) = a^T y$

$$\begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$$

LDF: Augmented feature vector

Given samples $x_1, x_2, ..., x_n$, convert them to augmented samples $y_1, y_2, ..., y_n$ by adding a new dimension of value 1.



The homogeneous discriminant at y separates points in this transformed space by a hyperplane passing through the origin.

LDF: Train Error

- For the rest of lecture, we assume we have 2 classes
- Samples $y_1, ..., y_n$ belongs to either class 1 or class 2.
- Our goal is to use these samples to determine weights a in the discriminant function $g(y) = a^T y$
- We need to decide which criterion for determining a.
  For now, suppose we want to minimize the training error, which means the number of misclassified samples $y_1, ..., y_n$
- Recall that
  - $g(y_i) > 0 \Rightarrow y_i$ classified c_1
  - $g(y_i) < 0 \Rightarrow y_i$ classified c_2
- The training error is 0 if
  - $g(y_i) > 0 \quad \forall y_i \in c_1$
  - $g(y_i) < 0 \quad \forall y_i \in c_2$

LDF: Problem "Normalization"

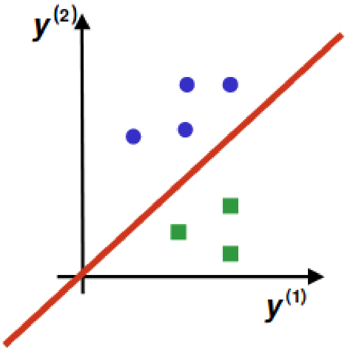
- Equivalently, training error is 0 if

$$\begin{cases} a^T y_i > 0 & \forall y_i \in c_1 \\ a^T(-y_i) > 0 & \forall y_i \in c_2 \end{cases}$$

- This suggest problem "normalization"
  - Replace all examples from class c_2 by their negative
    $y_i \Rightarrow -y_i \quad \forall y_i \in c_2$
  - seek weight vector a
    $a^T y_i > 0 \quad \forall y_i$
    - If such a exists, it is called a separating or solution vector
    - original samples $x_1, ..., x_n$ can indeed be separated by a line

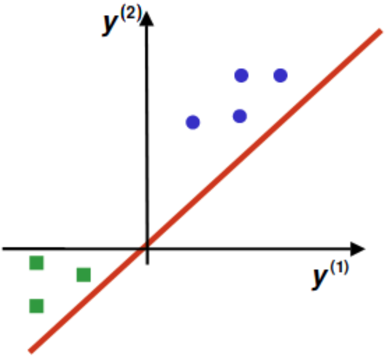
LDF: Problem "Normalization"

Before Normalization



Seek a hyperplane that separates patterns from different categories

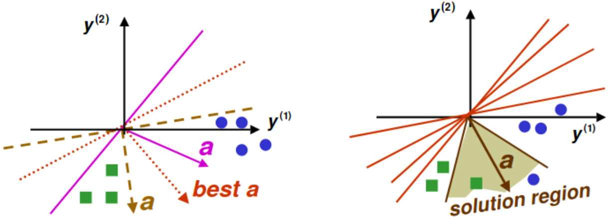
After Normalization



Seek a hyperplane that puts normalized patterns on the same side (should be positive)

LDF: Solution Region

- Find weight vector a, for all samples $y_1, ..., y_n$:
  $a^T y_i = \sum_{k=0}^{d} a_k y_i^{(k)} > 0$



- In general, there are many such solutions a

Optimization

We need a criterion function $J(a)$

$J(a)$ is minimized if a is a solution vector.
Regarding the exact form of J(a), we will talk about it on week4 day2.

This reduces our problem to one of minimizing a scalar function :

a problem that can often be solved by a gradient descent procedure.

Optimization: Gradient Descent

Basic idea of Gradient Descent

Gradient Descent

For minimizing any function $J(x)$ set k = 1 and $x^{(1)}$ to some initial guess for the weight vector

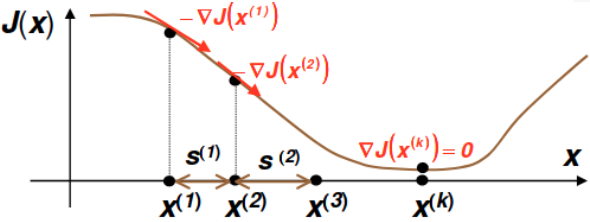
while $\eta^{(k)}|\nabla J(x^{(k)})| > \epsilon$ **do**

choose learning rate $\eta^{(k)}$

$x^{(k+1)} = x^{(k)} - \eta^{(k)}\nabla J(x^{(k)})$

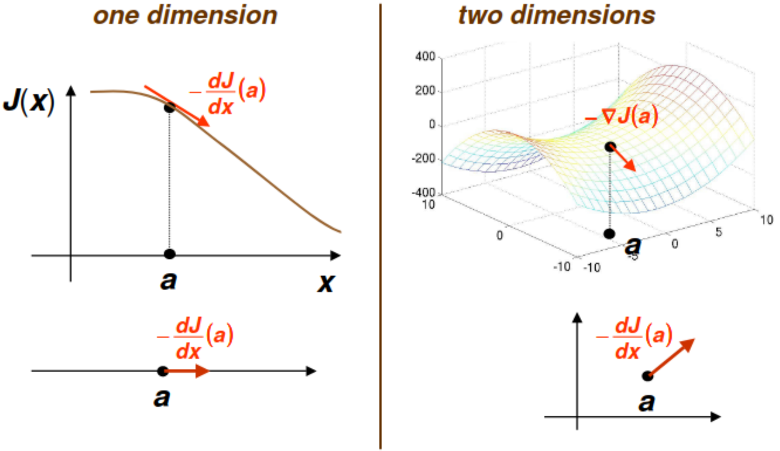
$k = k + 1$

end



Optimization: Gradient Descent

- Gradient $\nabla J(x)$ points in direction of steepest increase of $J(x)$, and $-\nabla J(x)$ in direction of steepest decrease



Reference I

[1] Richard O Duda, Peter E Hart, et al. **Pattern Classification**. 2nd ed. Wiley New York, 2000.

[2] Sergios Theodoridis and Konstantinos Koutroumbas. **Pattern Recognition**. Elsevier, 2009.

Thank You !

Q & A