



Classification Part II

Maximal margin classifier, support vector machines

송 준

고려대학교
통계학과 / 융합데이터과학 대학원

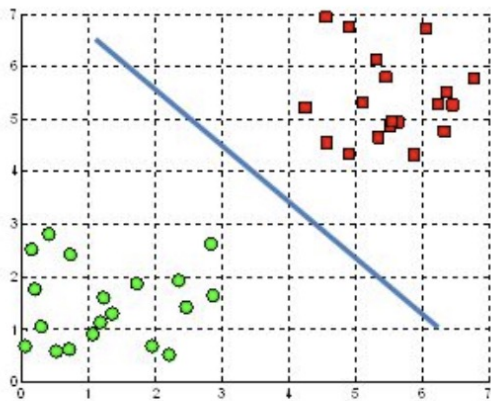
Maximal Margin Classifier

hard margin

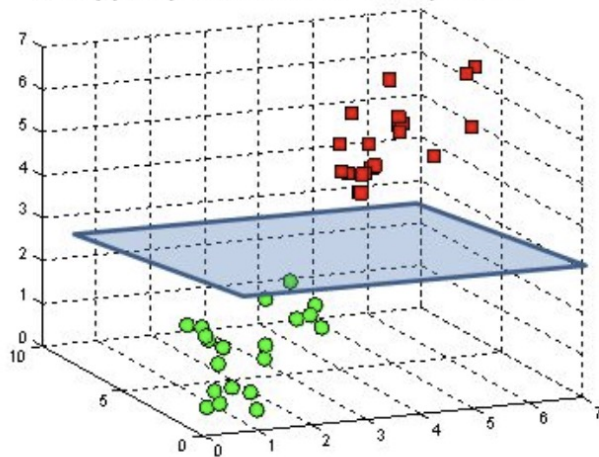
Hyperplane

- A **hyperplane in p -dimensional space** is a flat affine subspace of dimension $p-1$

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane

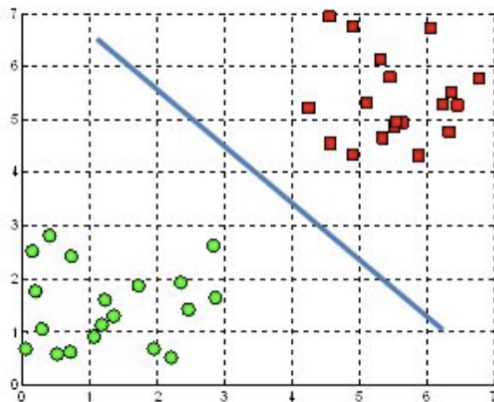


- p -차원에 있는 set 이지만, $p-1$ 차원 공간과 본질적으로 같음

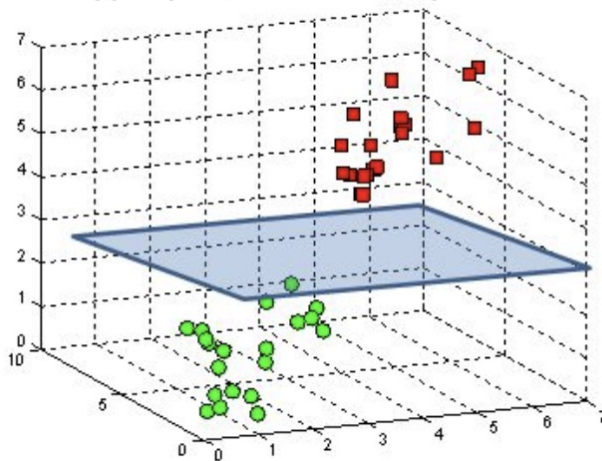
Hyperplane

- A hyperplane in p -dimensional space is a flat affine subspace of dimension $p-1$

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane



Given $\beta_0, \beta_1, \dots, \beta_p$, Hyperplane은 아래와 같이 표현 가능.

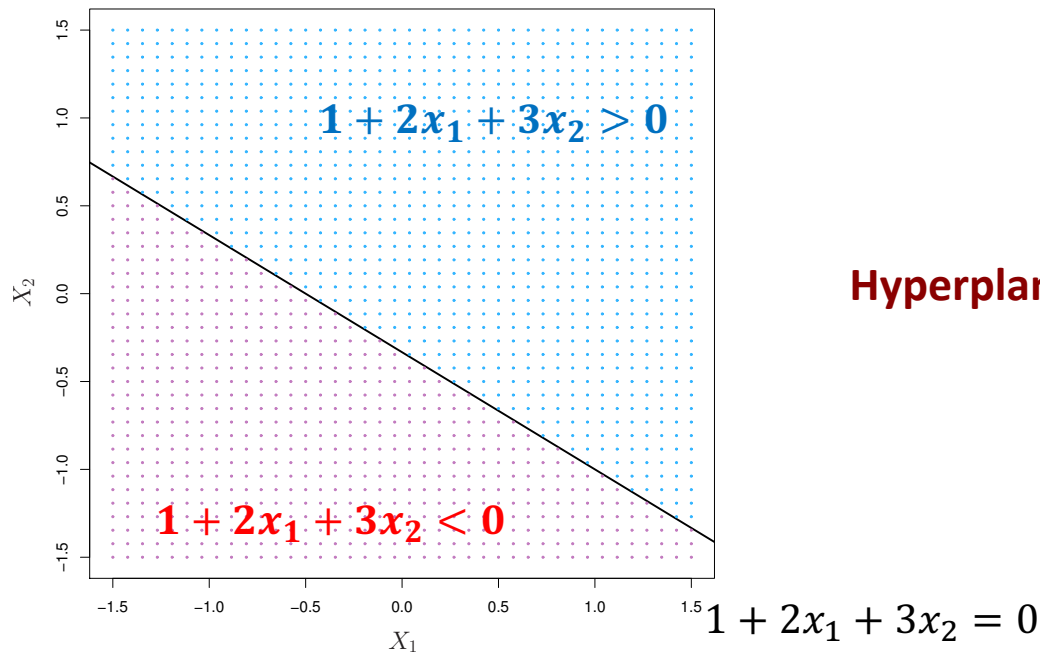
$$\text{Hyperplane} = \{(x_1, \dots, x_p) \in \mathbb{R}^p : \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0\}$$

Hyperplane : Example

예: $p = 2$, $\beta_0 = 1, \beta_1 = 2, \beta_2 = 0$

$$\text{Hyperplane} = \{(x_1, x_2) \in \mathbb{R}^2 : 1 + 2x_1 + 3x_2 = 0\}$$

The hyperplane: $1 + 2x_1 + 3x_2 = 0 \Leftrightarrow x_2 = -\frac{2}{3}x_1 - \frac{1}{3}$



Hyperplane 은 공간을 두 부분으로 나눔

Classification : Separating Hyperplane

- Binary Classification Problem

- X : p -variables
- y : -1 or 1

- Given $\beta_0, \beta_1, \dots, \beta_p$, Hyperplane은 아래와 같이 표현 가능.

$$\text{Hyperplane} = \{(x_1, \dots, x_p) \in \mathbb{R}^p : \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0\}$$

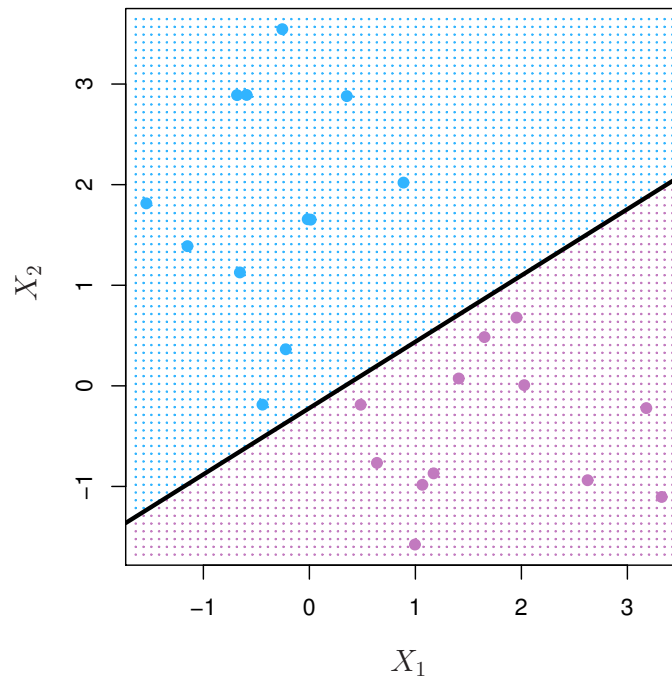
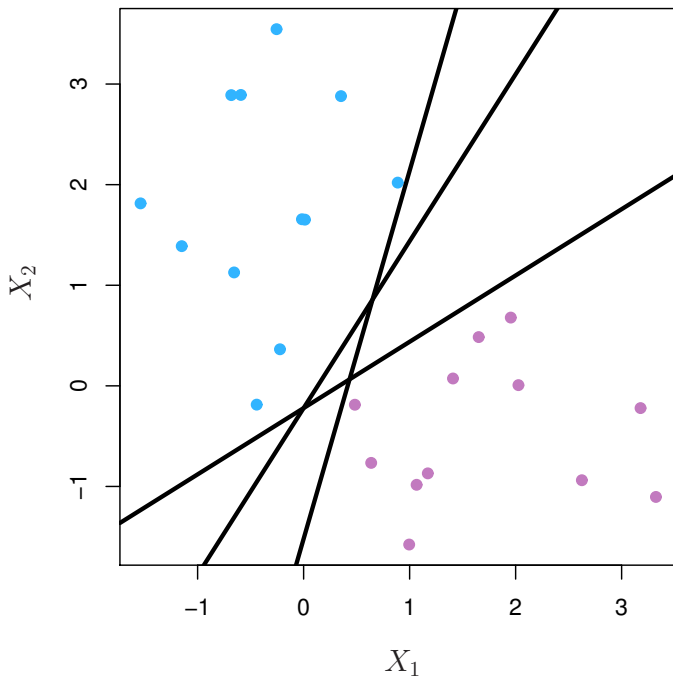
편의상 $f_\beta(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$ 라고 하면,

- Classifier: Given $X = x_i$,
 - Assign 1 if $f_\beta(x_i) > 0$
 - Assign -1 if $f_\beta(x_i) < 0$

- Equivalently, the above is

$$y_i f_\beta(x_i) > 0$$

Maximal Margin Classifier

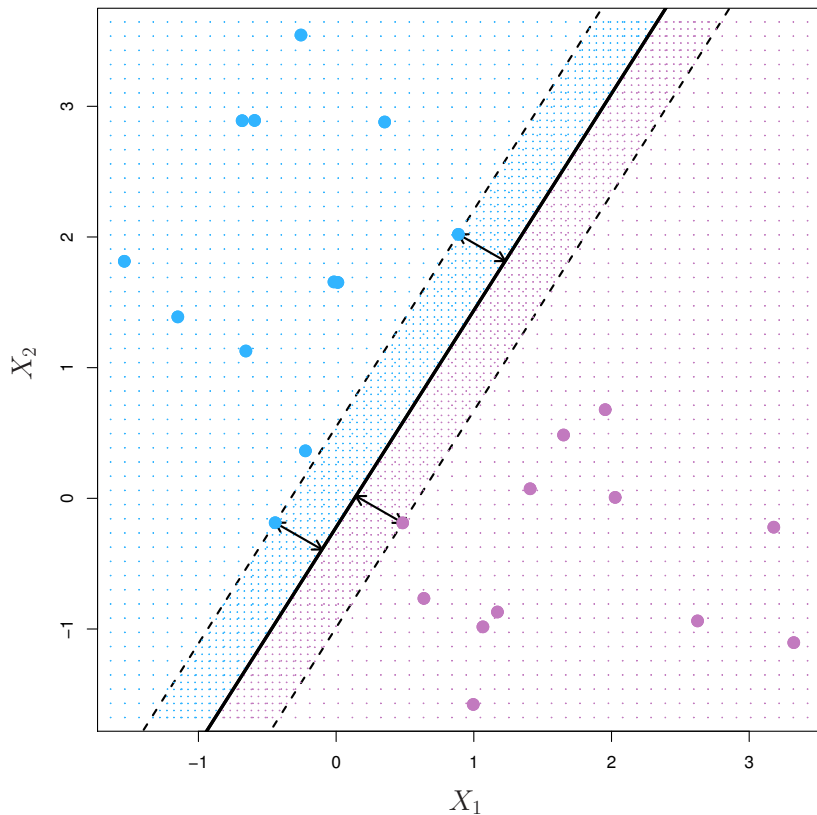


- Which hyperplane will you choose? Any Justification?

Maximal Margin Classifier : Margin

$$f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0$$

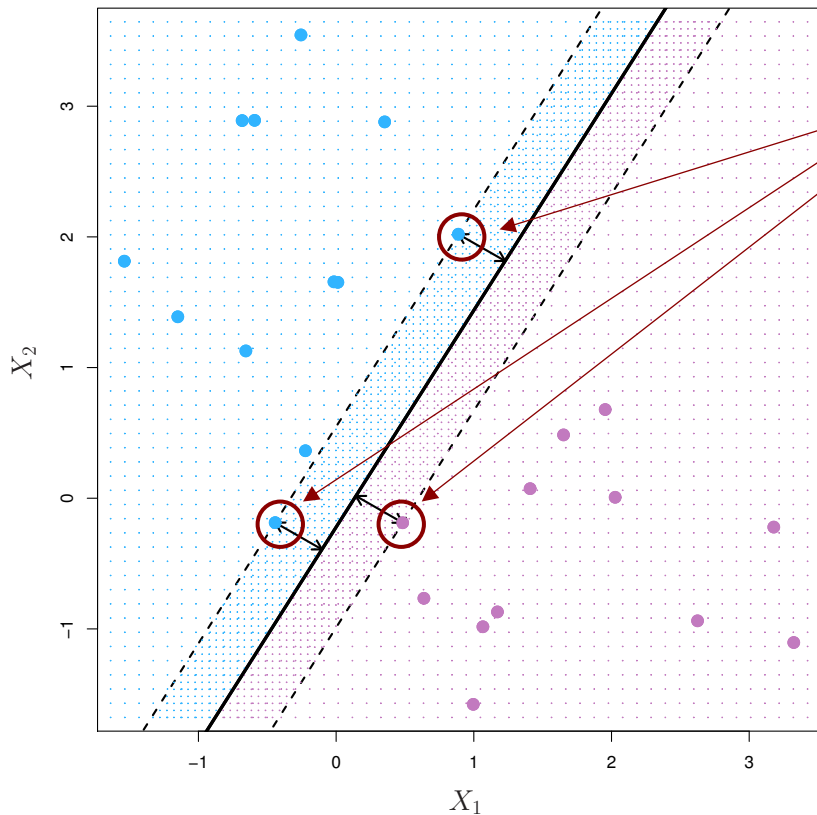
$\|\beta\| = 1$, 방향만 고려



- 직선이 이미 주어졌다고 가정(β)
- 각 data point마다, 직선과의 거리를 계산
 $d_i = |f_{\beta}(x_i)|$
- **M: margin (or gap)**: 가장 가까운 점과 직선과의 거리 (거리들 중 가장 작은 거리)
- **거리가 멀다 = more confident**
- **Maximal margin classifier**: 이 margin을 가장 크게, gap을 가장 크게, 만드는 hyperplane 찾기 (= β 찾기)

Maximal Margin Classifier : Support Vectors

$$f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p = \beta_0 + \langle \beta, x \rangle = 0$$



- **Support Vectors**: the points that “support” the maximal margin hyperplane
- 이 (얼마 안되는) Data 포인트들이 조금만 움직여도 hyperplane이 바뀜
- 다른 점들은 hyperplane에 직접적으로 영향을 주지 않음

Maximal Margin Classifier : Optimization

- maximize M
- subject to
 - $\|\beta\| = 1$
 - $d_i = |f_\beta(x_i)| = y_i f_\beta(x_i) \geq M, \quad i = 1, \dots, n$
- Direct 한 계산이 어렵다.

Maximal Margin Classifier : Optimization

- maximize M
- subject to
 - $\|\beta\| = 1$
 - $d_i = |f_\beta(x_i)| = y_i f_\beta(x_i) \geq M, \quad i = 1, \dots, n$
- Direct 한 계산이 어렵다.
- x_i 와 separating hyperplane ($\{x: f_\beta(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p\}$)과의 거리
$$\left| \frac{f_\beta(x_i)}{\|\beta\|} \right|$$

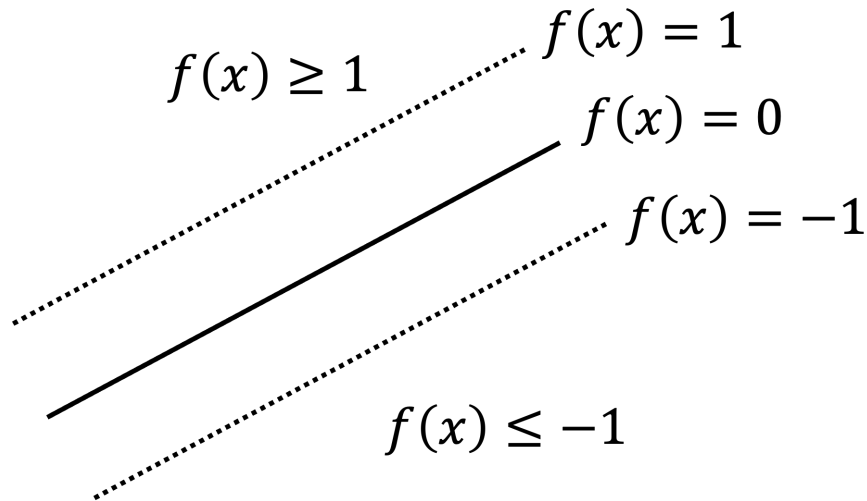
Maximal Margin Classifier: Different Formulation

- Margin=1 이라는 조건으로 변경(대신 $\|\beta\| = 1$ 이라는 조건 제외)
- Maximize $\frac{1}{\|\beta\|}$
- Subject to
 - $y_i f_\beta(x_i) \geq 1 \quad i = 1, \dots, n$

Equivalently

- Minimize $\|\beta\|^2 = \beta^T \beta$
- Subject to
 - $y_i f_\beta(x_i) \geq 1 \quad i = 1, \dots, n$

where $f_\beta(x_i) = \beta_0 + \langle \beta, x \rangle$



Maximal Margin Classifier : Different Formulation

Equivalently

- Minimize $\frac{1}{2} \|\beta\|^2 = \frac{1}{2} \beta^T \beta$
- Subject to
 - $y_i f_\beta(x_i) \geq 1 \quad i = 1, \dots, n$

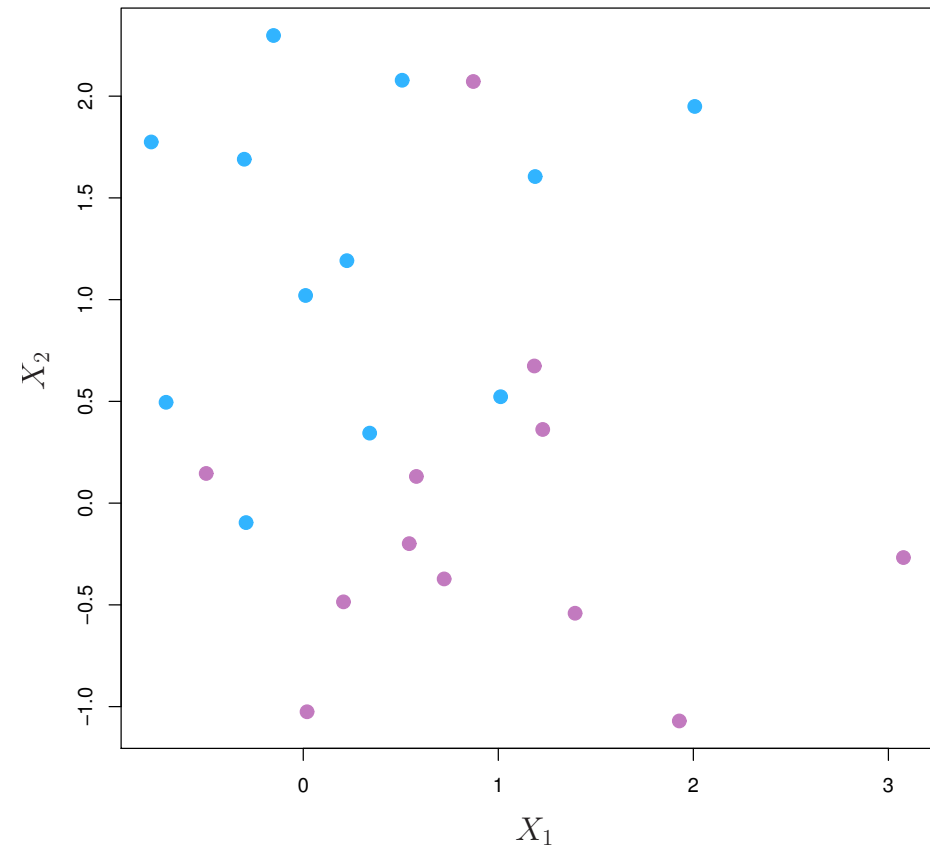
where $f_\beta(x_i) = \beta_0 + \langle \beta, x_i \rangle$

with the Lagrangian multiplier, the **loss function** is

$$L(\beta; \alpha_i) = \frac{1}{2} \beta^T \beta + \sum_{i=1}^n \alpha_i [y_i (\beta_0 + \langle \beta, x_i \rangle) - 1]$$

$[y_i (\beta_0 + \langle \beta, x_i \rangle) - 1]$ is a form of a **hinge loss** (or **svm loss**)

Non-separable case

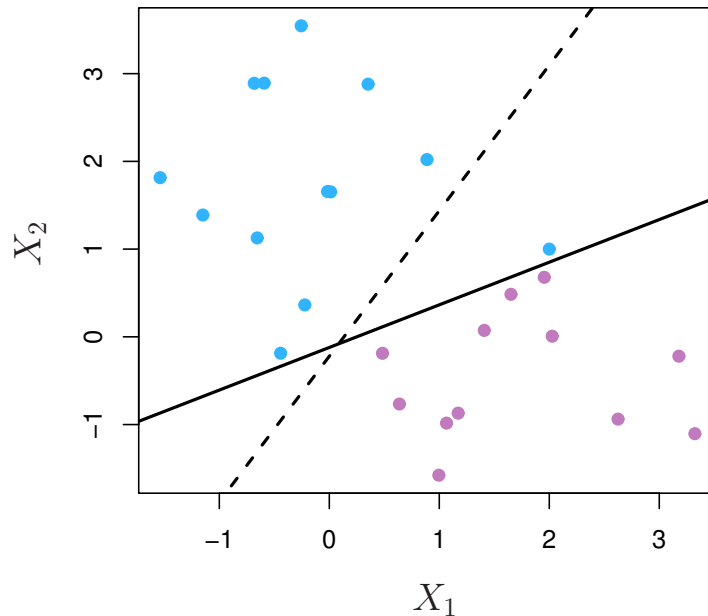
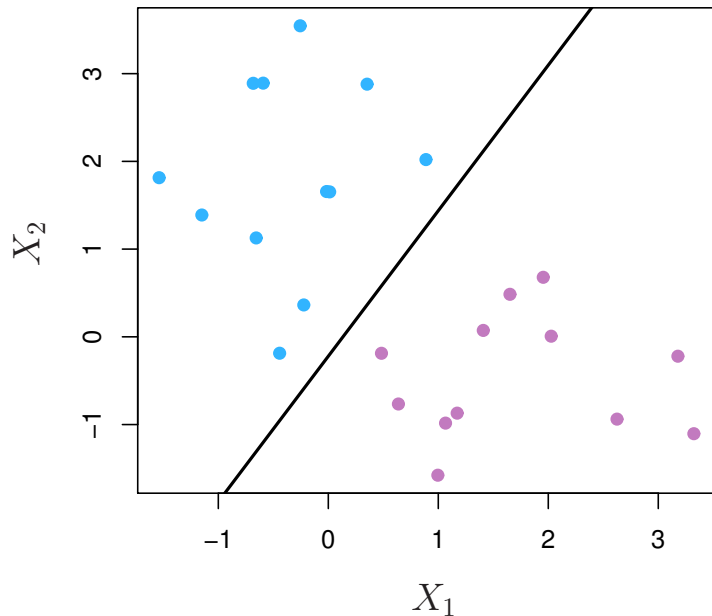


- 많은 경우 Non-separable
- There is no maximal margin classifier
- Instead of the hard margin,
- Use soft margin (일부 오류 허용)

Support Vector Classifier

soft margin

Maximal Margin Classifier



- A change in a single observation \rightarrow affect the classifier a lot
- Potential overfitting

Support Vector Classifiers

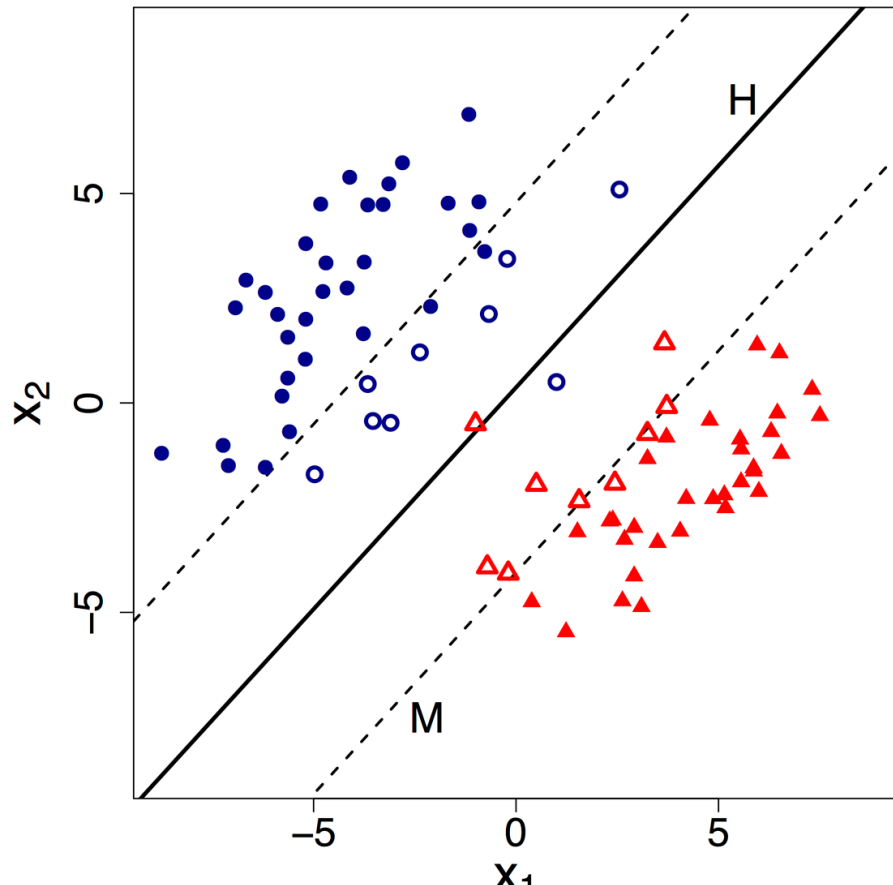
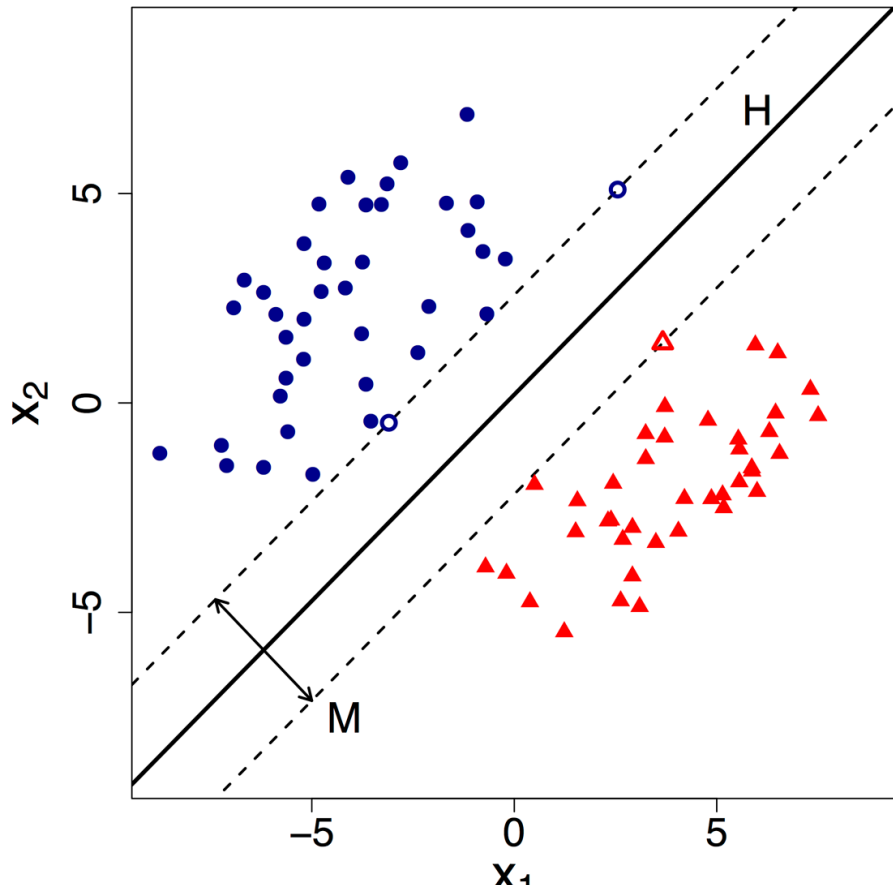
- Find a hyperplane that **does not perfectly separate** the two classes.
- Separate almost all the data correctly, **allow a few mis-classification**.
- Gain:
 - **Robustness** to individual observations
 - **Better** classification of **most** of the training observations

Support Vector Classifiers

- maximize M
- subject to
 - ✓ $\|\beta\| = 1$
 - ~~✓ $d_i = |f_\beta(x_i)| = y_i f_\beta(x_i) \geq M, i = 1, \dots, n$~~
 - ✓ $y_i f_\beta(x_i) \geq M(1 - \epsilon_i), \quad i = 1, \dots, n$
 - ✓ $\epsilon_i \geq 0, \sum \epsilon_i \leq B$

where $f_\beta(x) = \beta_0 + \langle \beta, x \rangle$, B : a budget

Support Vector Classifiers



Support Vector Classifier : Optimization

Maximal Margin Classifier

- Minimize $\frac{1}{2} \|\beta\|^2 = \frac{1}{2} \beta^T \beta$
- Subject to
 - $y_i f_\beta(x_i) \geq 1 \quad i = 1, \dots, n$

where $f_\beta(x_i) = \beta_0 + \langle \beta, x \rangle$

C: penalty on $\sum_{i=1}^n \epsilon_i$

- cost of violation
- C 무한대 ~ maximal margin classifier ($\epsilon_i=0$)

Support Vector Classifier

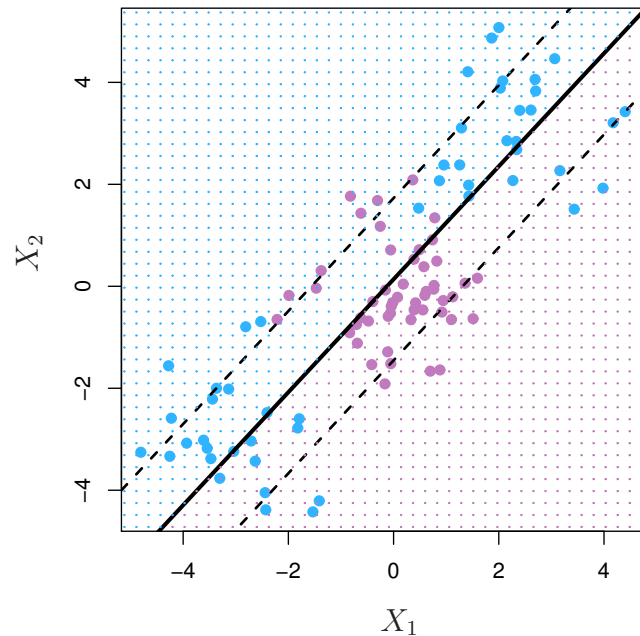
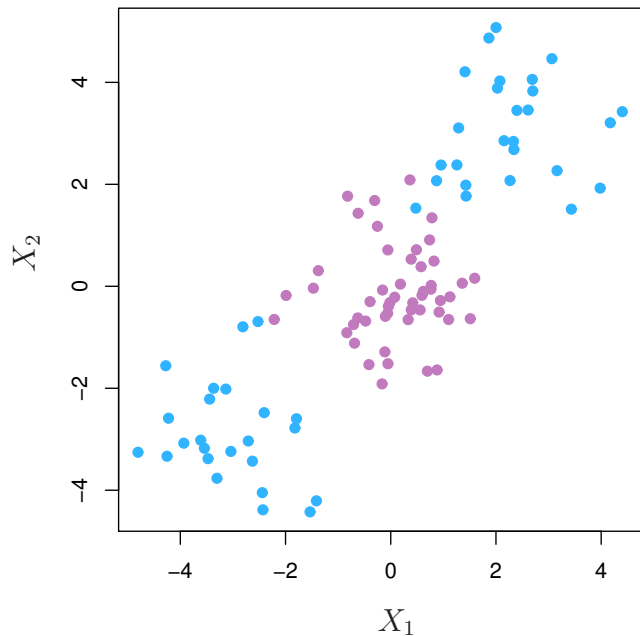
- Minimize $\frac{1}{2} \|\beta\|^2 = \frac{1}{2} \beta^T \beta + C \sum_{i=1}^n \epsilon_i$
- Subject to
 - $y_i f_\beta(x_i) \geq 1 - \epsilon_i \quad i = 1, \dots, n$
 - $\epsilon_i \geq 0$

where $f_\beta(x_i) = \beta_0 + \langle \beta, x \rangle$

The Support Vector Machines

Nonlinear classifier

Linear Boundary Can Fail



(Optional) SVM Optimization Detail

The linear SVM is

$$\text{minimize } \frac{1}{2} \beta^T \beta + C \sum_{i=1}^n \epsilon_i$$

Subject to $y_i(\beta_0 + \langle \beta, x_i \rangle) \geq 1 - \epsilon_i$ for all $i = 1, \dots, n$.

Or, $\epsilon_i \geq 1 - y_i(\beta_0 + \langle \beta, x_i \rangle)$ tells

$$\text{minimize } \frac{\lambda}{2} \|\beta\|^2 + \sum_{i=1}^n \max(0, 1 - y_i f(x_i))$$

Subject to $f(x) = \beta_0 + \langle \beta, x \rangle$

Considering $\frac{\lambda}{2} = C^{-1}$

(Optional) SVM Optimization Detail

With the linear constraints

$$\text{minimize } \frac{\lambda}{2} \|\beta\|^2 + \sum_{i=1}^n \max(0, 1 - y_i f(x_i))$$

$$\text{Subject to } f(x) = \beta_0 + \langle \beta, x \rangle$$

Removing the **linear constraints**,

$$\text{minimize } \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 + \sum_{i=1}^n \max(0, 1 - y_i f(x_i))$$

$$\text{Subject to } f \in \mathcal{H}$$

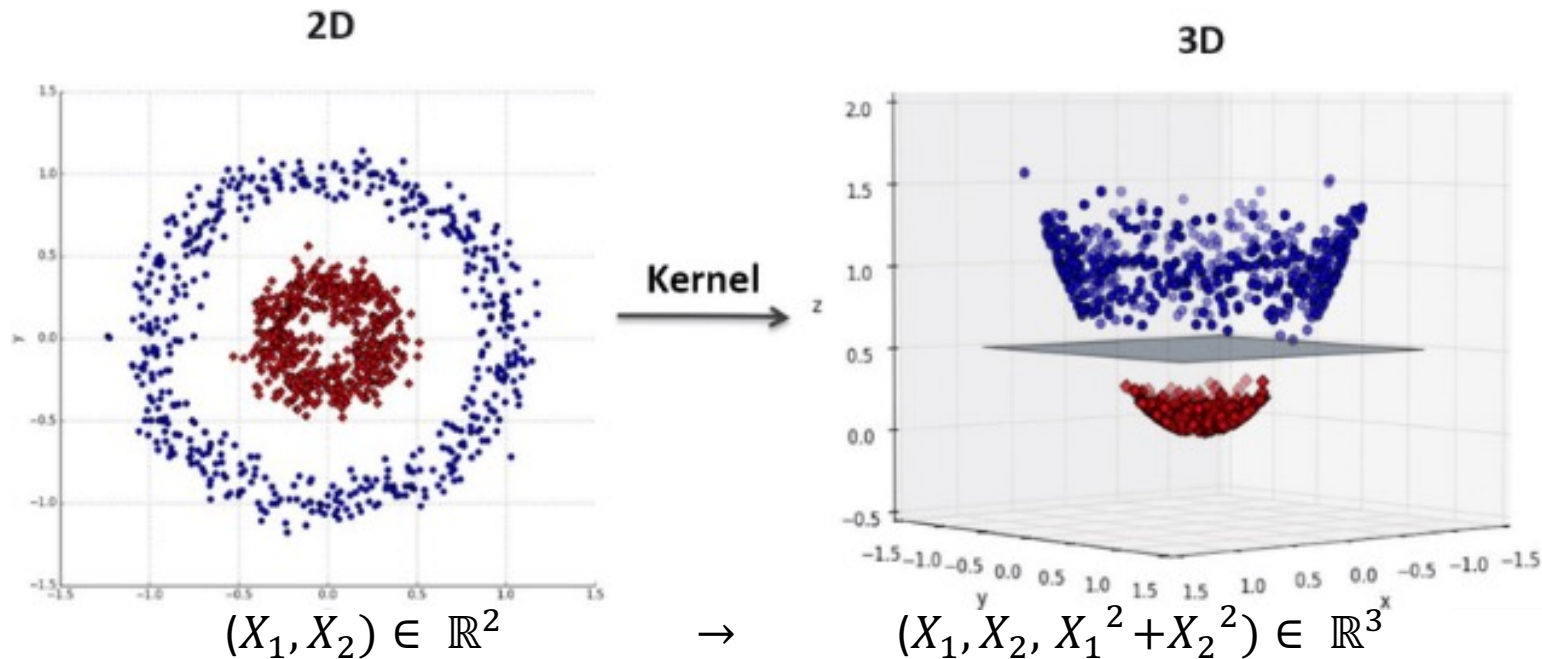
How? Feature Mapping!

Recall: Feature Mapping

- $\hat{y} = \hat{A}(x)$ 라는 선형방법론이 있다고 가정
- \hat{A} 은 데이터 $(x_1, y_1), \dots, (x_n, y_n)$ 을 통해 만들어짐.

Feature Mapping: $\phi: \mathbb{R}^p \rightarrow \mathcal{H}$ 가 원래 데이터를 더 고차원으로 보내는 mapping일 때
 $(\phi(x_1), y_1), \dots, (\phi(x_n), y_n)$
을 이용하여 만든다면, 최종 \hat{A} 은 $\phi(x_1), \dots, \phi(x_n)$ 와 선형관계. x 에 대한 비선형 함수!

Recall: Feature Mapping



Decision boundary: becomes **nonlinear** in terms of x_1, x_2 .

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0, \quad v.s. \quad c_0 + c_1 x_1 + c_2 x_2 + c_3 (x_1^2 + x_2^2) = 0$$

Kernel SVM

- Feature Mapping: $\phi: \mathbb{R}^p \rightarrow \mathcal{H}$ 에서 \mathcal{H} 가 특별한 성질을 만족하는 함수공간
- Reproducing kernel Hilbert Space (RKHS) with a reproducing kernel

$$\begin{aligned}\phi(x_i) &= \kappa(\cdot, x_i) \\ \langle \kappa(\cdot, x_i), \kappa(\cdot, x_j) \rangle &= \kappa(x_i, x_j)\end{aligned}$$

Gaussian Radial Basis Function(GRBF)

$$\kappa(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2)$$

Polynomial Kernel

$$\kappa(x_1, x_2) = (a + \langle x_1, x_2 \rangle)^2$$

The Support Vector Machine

- The **Support Vector Machine (SVM)** is an extension of the support vector classifier that results from enlarging feature space to be a RKHS.
- It is well known that RKHS generated by the **GRBF kernel** is rich enough space to cover any nonlinear function of x .
- Note
 - Why do we use the kernel functions? What is the RKHS?
 - Choice of kernel?
 - What is the role of γ ? in the kernel function?

Summary

- SVM: Find a separating hyperplane with a soft-margin
- Hyperparameters
 - Cost parameter: 얼마나 hard하게 soft-margin 구성?
 - Kernel function: Gaussian RBF, Polynomial, Brownian, ...
 - Parameter of the kernel: Gaussian: γ , Polynomial: degree
- 다른 Classifiers (두번째 모듈)
 - Tree-based models (decision trees)
 - Ensemble of trees
 - parallel: Random Forest
 - sequential: adaboost
 - GBM계열: GBM, XGBoost, LightGBM, CatBoost, ...