Inclass 10: Support Vector Machine

[SCS4049] Machine Learning and Data Science

Seongsik Park (s.park@dgu.edu)

AI Department, Dongguk University

Optimization (최적화).

	linear regression	perceptron.	logistic regression
cost.	• SSE, MSE, ∈ RMSE ∈	$-\sum_{n}^{M} w_n^T \phi(x_n) t_n$	• cross entropy.
Constraint. (model)	$\hat{y} = \theta^T x.$ $= w^T \phi(x)$	model.	model. $\sigma(w^T x)$
Solution	• closed form (normal equation)	• gradient descent.	grad. St. g.d. mini - ...
			grad. descent.

cost 함수 정의.

SVM: Maximum Margin Classifier

constraint.

↓

convex,

Lagrange.

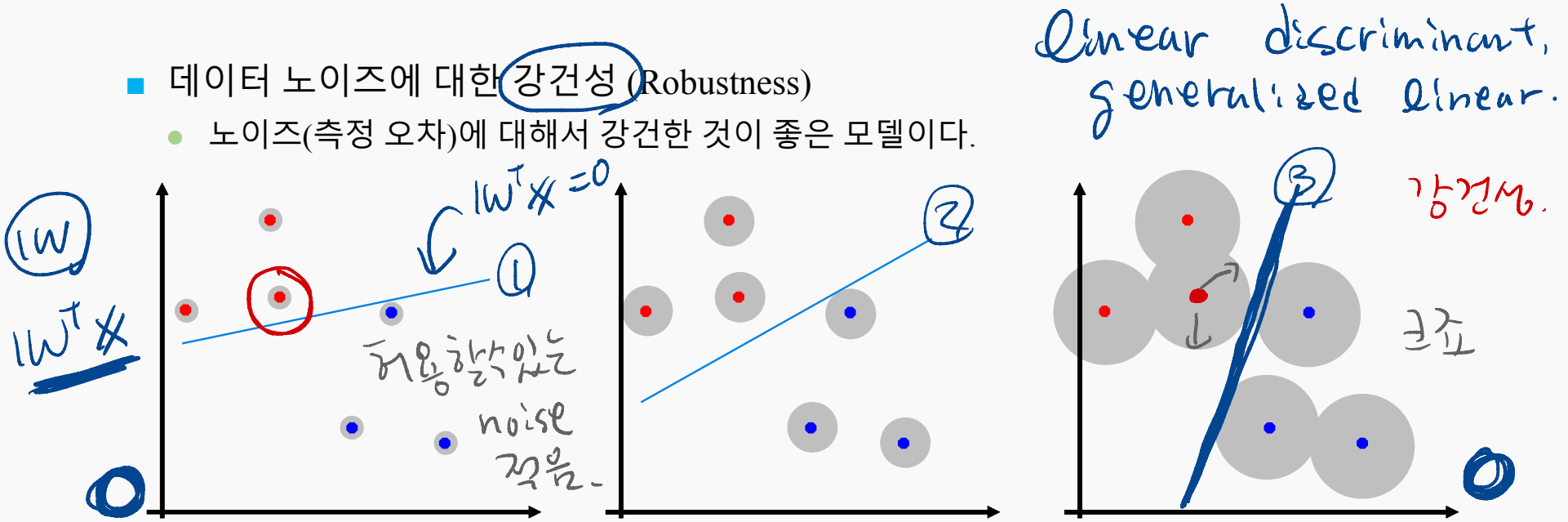
↓

Solution

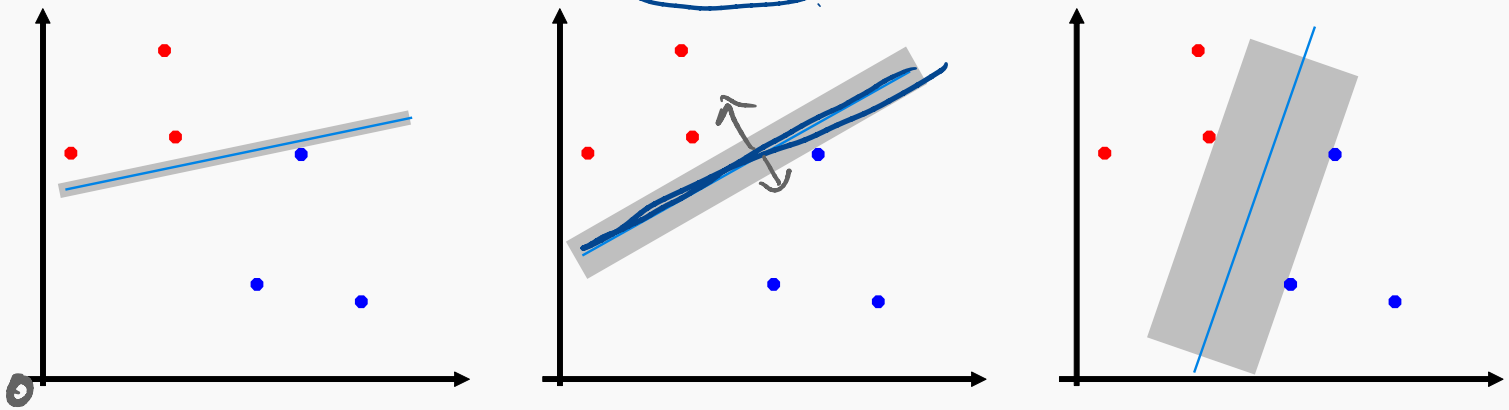
⇒ Support vector.
(complementary slackness).

What is a good decision boundary?

- 데이터 노이즈에 대한 강건성 (Robustness)
  - 노이즈(측정 오차)에 대해서 강건한 것이 좋은 모델이다.



- 여유로운 것이 더 강건하다 ⇒ 넓은 통로가 좋다 ⇒ Large Margin Classification



decision boundary → 가장가까운 sample까지의 거리. = margin

What is a good decision boundary?

→ 의사 결정은 경계의 데이터(support vectors)에 의해서 결정됨

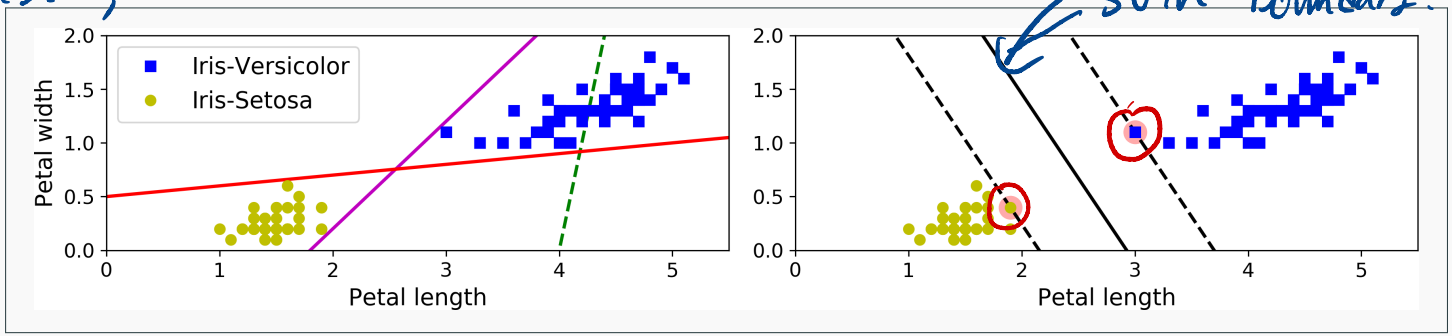


Figure 5-1. Large margin classification

→Input feature의 scale에 민감한 support vector machine

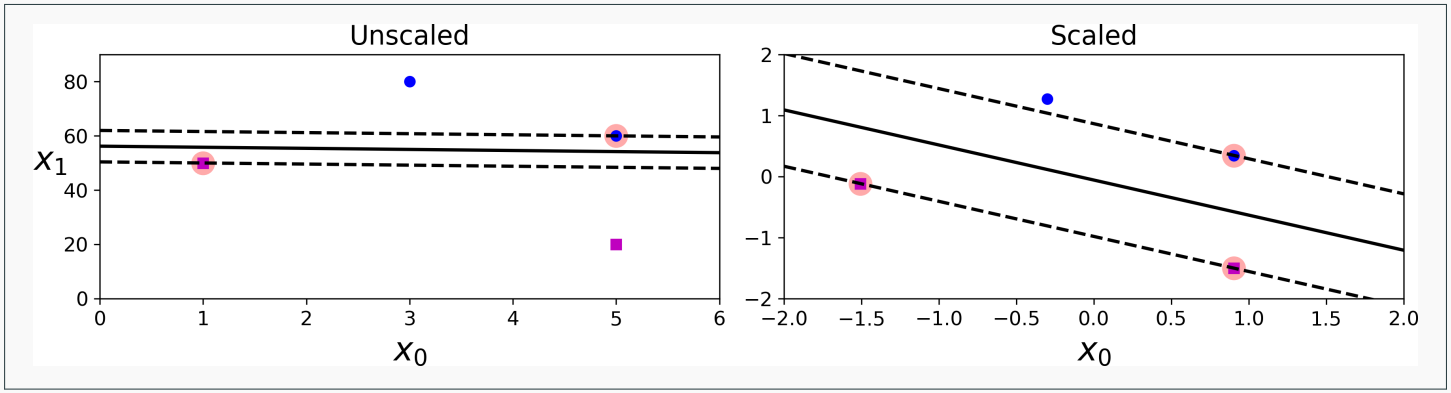
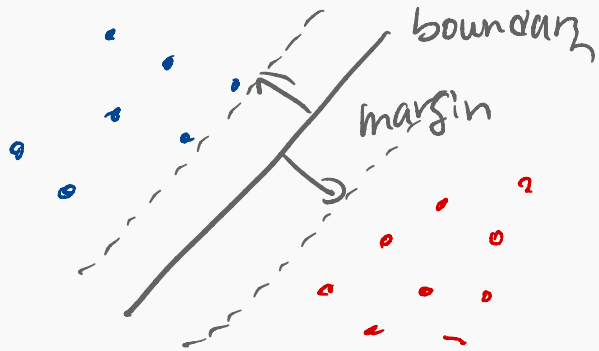


Figure 5-2. Sensitivity to feature scales

Hard margin vs. soft margin



Hard margin classification (hard-SVM)

- 모든 데이터들이 margin 밖에 위치하도록 boundary를 설정
- 데이터가 선형적으로 분리 가능(linearly separable)할 때만 적용 가능
- outlier에 매우 민감

✗ Soft margin classification (soft-SVM)

- margin을 가능한 넓게 하면서도 margin 안쪽으로 들어오는 것을 허용
- hyperparameter C: 클수록 좁아짐 (엄격), 작을 수록 넓어짐 (위반 허용)

Hard margin vs. soft margin

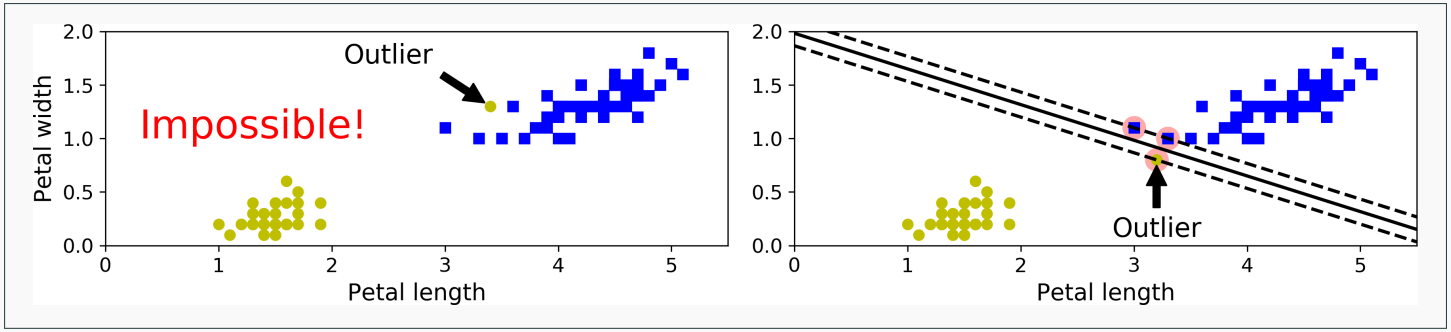


Figure 5-3. Hard margin sensitivity to outliers

soft SVM.

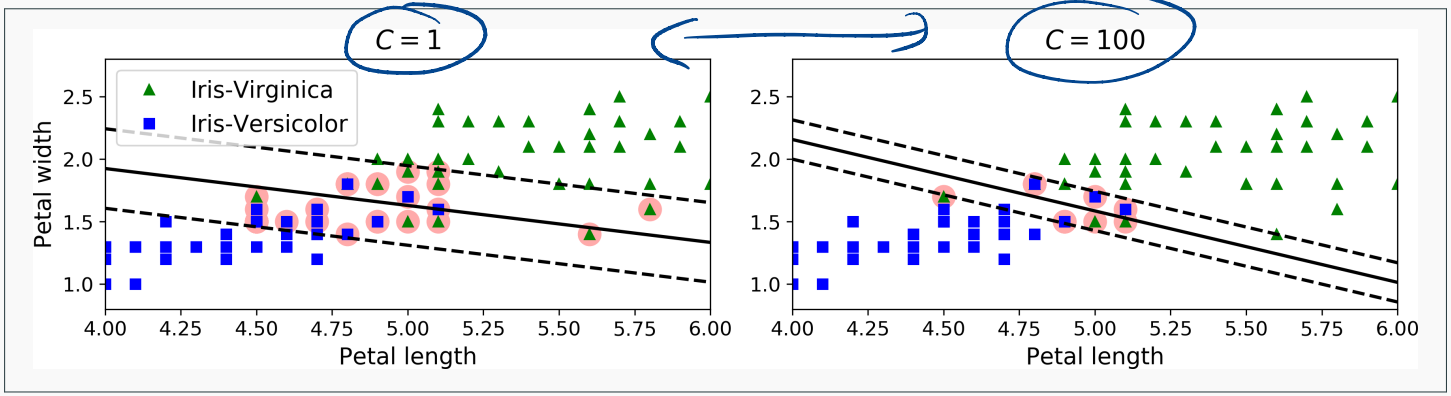


Figure 5-4. Large margin (left) versus fewer margin violations (right)

A brief history of SVM

- SVM은 1992년 Boser, Guyon and Vapnik에 의해서 소개됨
- Statistical Learning Theory에 이론적 바탕을 둔 알고리즘 (Vapnik & Chervonenkis)
- 손글씨 숫자 인식에서 뛰어난 성능을 보이면서 널리 쓰이게 됨
- SVM으로 1.1% Test error rate ≈ 신중히 설계된 신경망과 맞먹음
- 실용적으로 우수한 성능
- Bioinformatics, text, image recognition 등을 포함한 많은 성공 사례
- 선형/비선형 분류 뿐 아니라 회귀, outlier detection 도 수행
- 복잡한 소규모/중규모 데이터셋의 분류에 특히 잘 맞음
- Kernel 방법을 사용하는 대표적 알고리즘
- Liblinear & libsvm: Scikit-Learn 에서 liblinear 및 libsvm 을 사용

Hard margin SVM: Theory

Maximum margin classifier

training data → $\mathbf{w}, b$ 를 찾게 된다.

We begin our discussion of support vector machines to the two-class classification problem using linear models of the form

$$y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \qquad \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} \tag{1}$$

where $\phi(\mathbf{x})$ denotes a fixed feature-space transformation, and we have made the bias parameter b explicit.

The training data set comprises N input vectors $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$, with corresponding target values $t_1, t_2, ..., t_N$ where $t_n \in \{-1, 1\}$, and new data points $\mathbf{x}$ are classified according to the sign of $y(\mathbf{x})$

$y(\mathbf{x}) \gtrless 0 \leftrightarrow t_n = \begin{matrix} +1 \\ -1 \end{matrix}$

Maximum margin classifier

We shall assume that the training data set is linearly separable in feature space, so that by definition there exists at least one choice of the parameters $\mathbf{w}$ and b such that a function satisfies $y(\mathbf{x}_n) > 0$ for points having $t_n = +1$ and $y(\mathbf{x}_n) < 0$ for points having $t_n = -1$, so that $t_n y(\mathbf{x}_n) > 0$ for all training data points.

Classification 관점에서.

$y(x_n) > 0, \quad t_n = +1.$
$y(x_n) < 0, \quad t_n = -1.$
$\Longleftrightarrow \quad y(x_n) \cdot t_n > 0.$

misclass

$y(x_n) \cdot t_n < 0$

Maximum margin classifier: optimality criterion

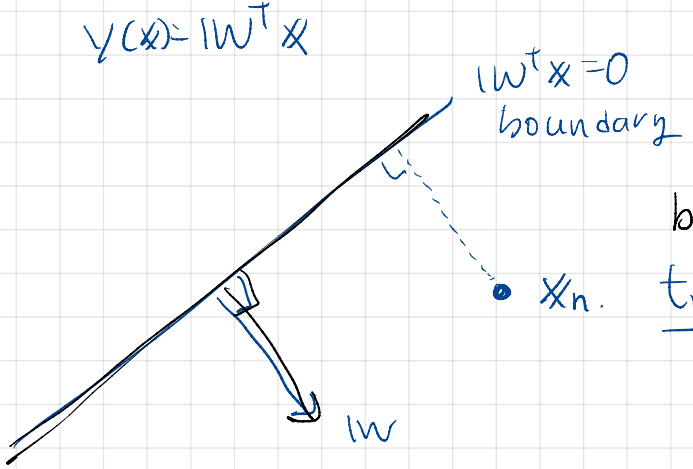
Thus the distance of a point $\mathbf{x}_n$ to the decision surface is given by

$$\min_{n=1,\ldots,N} \left[\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \frac{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)}{\|\mathbf{w}\|}. \right] \quad (2)$$

The margin is given by the perpendicular distance to the closest point $\mathbf{x}_n$ from the data set, and we wish to optimize the parameters $\mathbf{w}$ and b in order to maximize this distance. Thus the maximum margin solution is found by solving

$$\arg\max_{\mathbf{w},b} \left\{ \frac{1}{\|\mathbf{w}\|} \min[t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b)] \right\} \quad (3)$$

where we have taken the factor $1/\|\mathbf{w}\|$ outside the optimization over n because $\mathbf{w}$ does not depend on n.



boundary, decision surface ~ x_n까지의 거리.

$$\frac{t_n \cdot y(x_n)}{|w|} = \frac{t_n \cdot (w^T x_n + b)}{|w|}$$

Dual problem for convex optimization

Primal optimization problem for hard SVM → opt. prob.

$$\text{maximize} \quad \frac{1}{\|\mathbf{w}\|} \min[t_n(\mathbf{w}^T\phi(\mathbf{x}_n)+b)] \tag{4}$$

$$\text{subject to} \quad t_n(\mathbf{w}^T\phi(\mathbf{x}_n)+b) \geq 0 \tag{5}$$

$$n = 1, \ldots, N \tag{6}$$

Equivalently,

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 \tag{7}$$

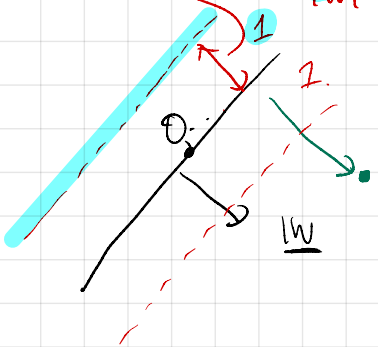
$$\text{subject to} \quad t_n(\mathbf{w}^T\phi(\mathbf{x}_n)+b) \geq 1 \tag{8}$$

Direct solution of this optimization problem would be very complex, so we shall convert it into an equivalent problem that is much easier to solve.

$w \rightsquigarrow \tilde{w}$

$w \updownarrow$

$\min \frac{t_n y(x_n)}{\|w\|} = 1$ (normalize)



Scaling.

$\|w_1\|$ $[1,2]^T$

$\|w_2\|$ $[2,4]^T$ = 같은 boundary

$\frac{t_n \cdot (w^T x_n)}{\|w\|}$ = dist. 같아짐.

Lagrangian function with constraint

In order to solve this constrained optimization problem, we introduce Lagrange multipliers $a_n \geq 0$, with one multiplier a_n for each of the constraints, giving the Lagrangian function

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n(\mathbf{w}^T\phi(\mathbf{x}_n) + b) - 1\} \tag{9}$$

where $\mathbf{a} = (a_1, ..., a_N)^T$. Note the minus sign in front of the Lagrange multiplier term, because we are minimizing with respect to $\mathbf{w}$ and b, and maximizing with respect to $\mathbf{a}$.

Setting the derivatives of $L(\mathbf{w}, b, \mathbf{a})$ with respect to $\mathbf{w}$ and b equal to zero, we obtain the following two conditions

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) \tag{10}$$

$$0 = \sum_{n=1}^{N} a_n t_n \tag{11}$$

Lagrangian function with constraint

Eliminating $\mathbf{w}$ and b from $L(\mathbf{w}, b, \mathbf{a})$ using these conditions then gives the *dual representation* of the maximum margin problem in which we maximize

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \tag{12}$$

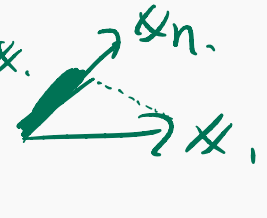
with respect to $\mathbf{a}$ subject to the constraints

$$a_n \geq 0, \qquad n = 1, ..., N \tag{13}$$

$$\sum_{n=1}^{N} a_n t_n = 0. \tag{14}$$

Here the kernel function is defined by $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$.

Prediction for a new sample: support vector machine



In order to classify new data points using the trained model, we evaluate the sign of $y(\mathbf{x})$. This can be expressed in terms of the parameter $\{a_n\}$ and the kernel function by substituting for $\mathbf{w}$ to give

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b \tag{15}$$

$$y(x) = \sum a_n t_n \phi(x_n)^T \phi(x) + b.$$

$k(x, x_n)$.

$\Rightarrow$ 두개의 vector의 유사성 비교하는 것, 함수.

kernel function.

KKT condition: complementary slackness

We show that a constrained optimization of this form satisfies the *Karush-Kuhn-Tucker* (KKT) conditions, which in this case require that the following three properties hold

$$a_n \geq 0 \tag{16}$$

$$t_n y(\mathbf{x}_n) - 1 \geq 0 \tag{17}$$

$$a_n \{t_n y(\mathbf{x}_n) - 1\} = 0 \tag{18}$$

Thus for every data point, either $a_n = 0$ or $t_n y(\mathbf{x}_n) = 1$. Any data point for which $a_n = 0$ will not appear in the sum and hence plays no role in making predictions for new data points. The remaining data points are called *support vectors*, and because they satisfy $t_n y(\mathbf{x}_n) = 1$, they correspond to points that lie on the maximum margin hyperplanes in feature space.

KKT condition: complementary slackness

→ 지원. margin = 1.

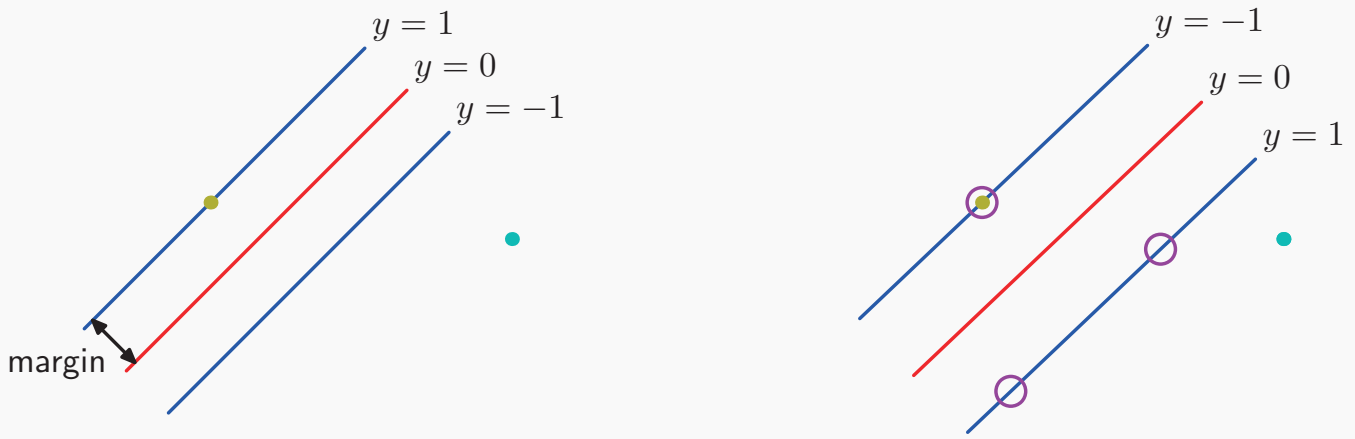


Figure 7.1 The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points, as shown on the left figure. Maximizing the margin leads to a particular choice of decision boundary, as shown on the right. The location of this boundary is determined by a subset of the data points, known as support vectors, which are indicated by the circles.

KKT condition: complementary slackness

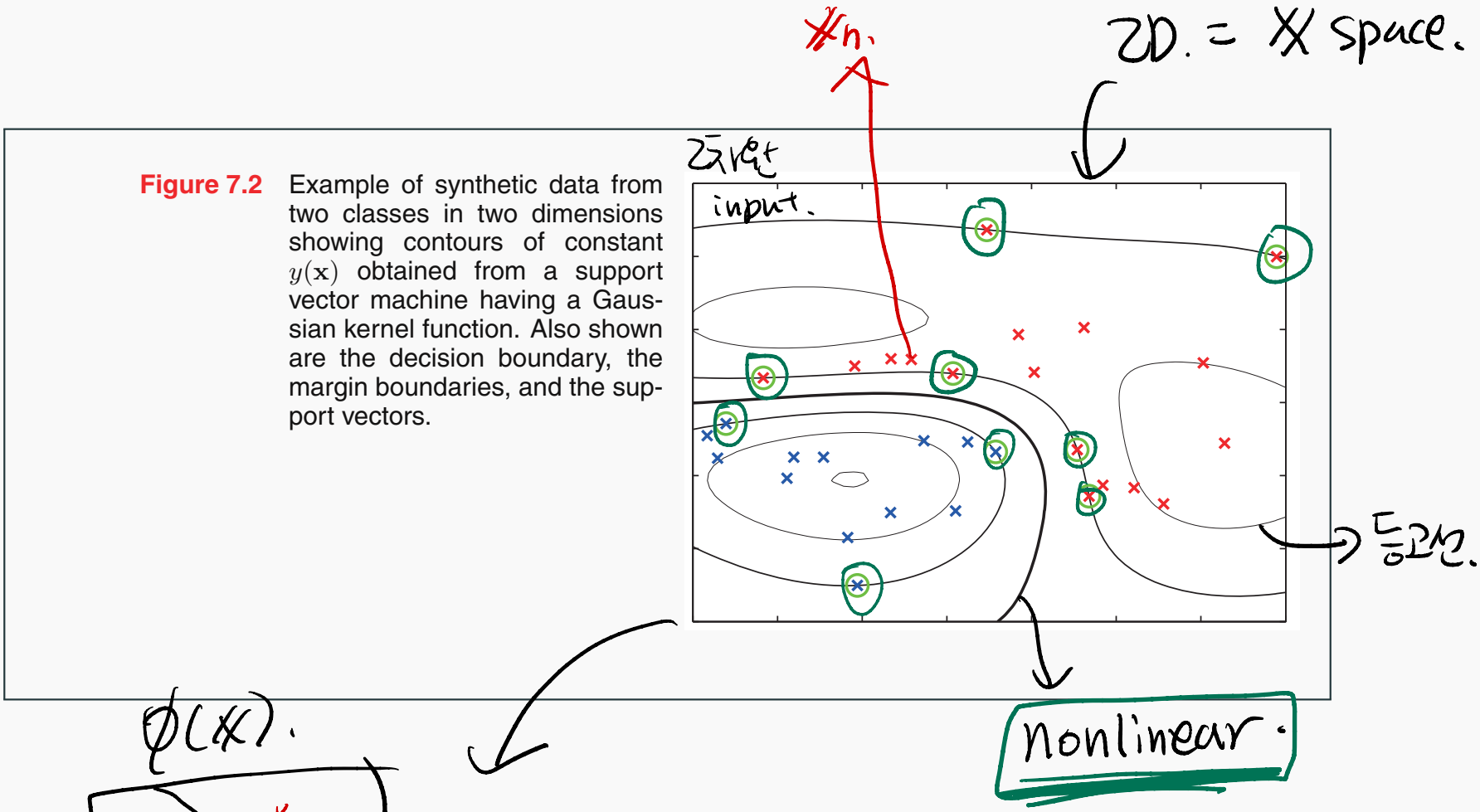
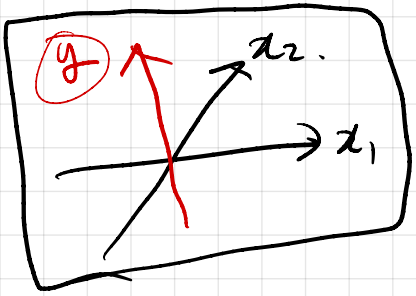


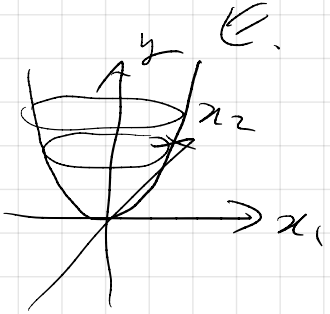
Figure 7.2 Example of synthetic data from two classes in two dimensions showing contours of constant $y(\mathbf{x})$ obtained from a support vector machine having a Gaussian kernel function. Also shown are the decision boundary, the margin boundaries, and the support vectors.

pcolormesh.

$y(x) \geq 0$.

2D → 1D.

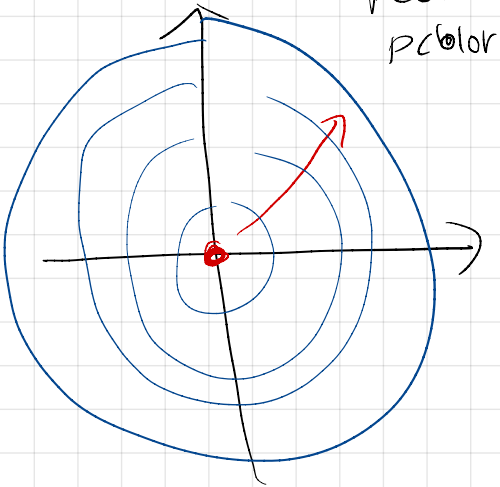


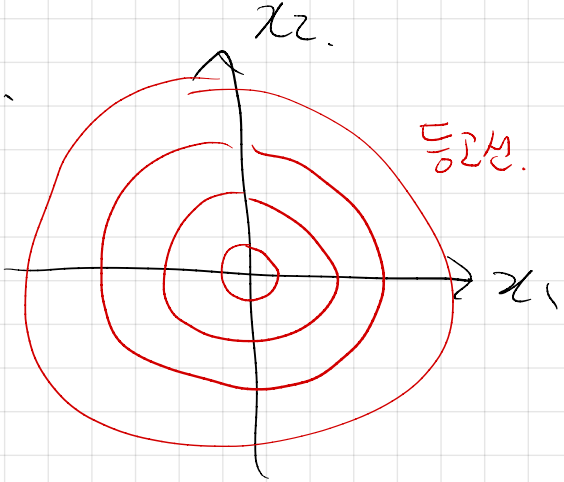


$y = x_1^2 + x_2^2$.

pcolor

pcolormesh.





Kernel Method

Kernel method

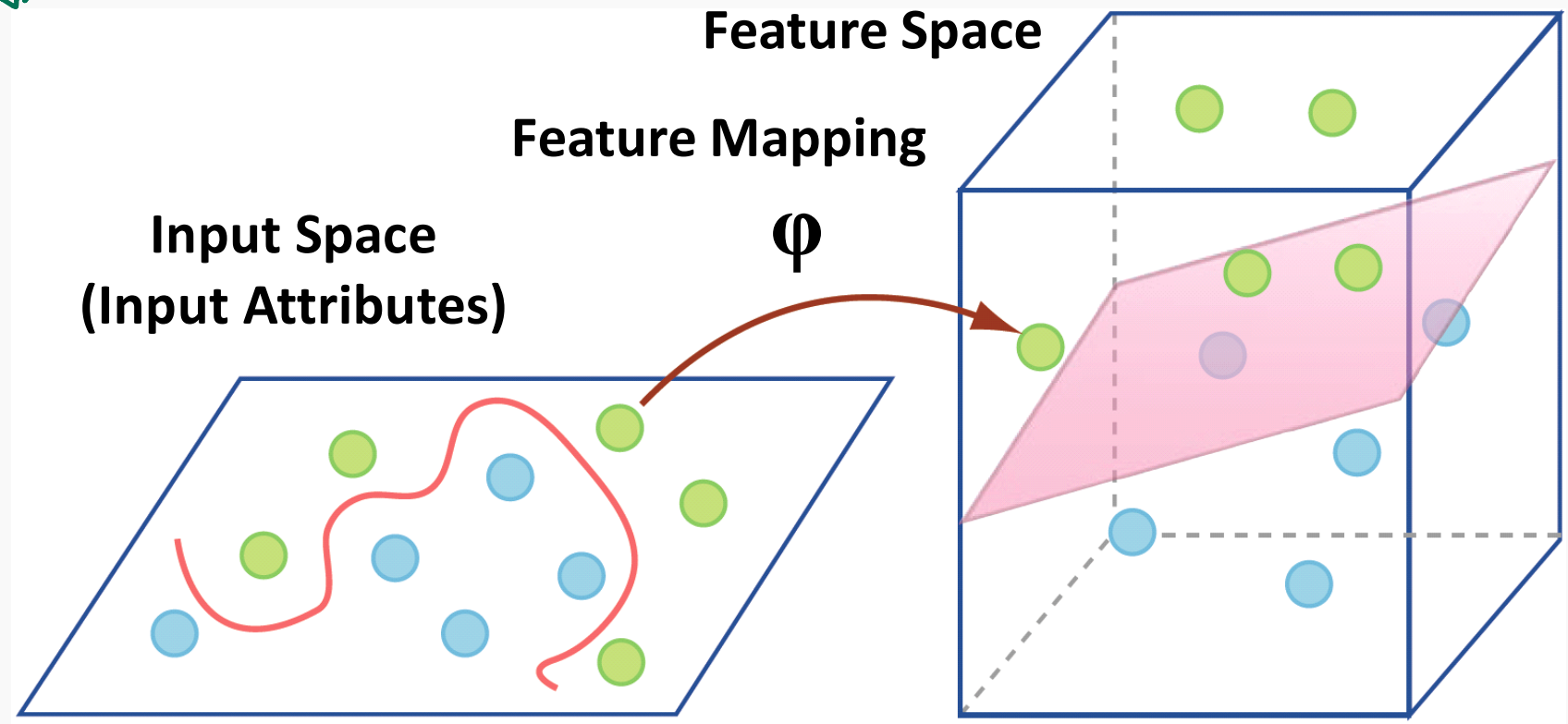


Image by MIT OpenCourseWare

Kernel method

Kernel function: Kernel function이란 어떤 feature space에서 feature의 inner product와 동등한 함수를 말함. Feature mapping $\boldsymbol{\phi}(\mathbf{x})$에 대한 kernel function은 다음과 같이 정의할 수 있음.

$$k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}') \tag{19}$$

Similarity: Kernel function $k(\mathbf{x}, \mathbf{x}')$는 $\mathbf{x}$와 $\mathbf{x}'$ 간의 similarity로 볼 수 있음.

또한 일반적으로 feature mapping을 통한 $\boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}')$에 비해 $k(\mathbf{x}, \mathbf{x}')$의 계산이 훨씬 간단함. 예를 들어

$$k(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^2 \tag{20}$$

$$\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} 1 & x_1^2 & \sqrt{2}x_1 x_2 & x_2^2 & \sqrt{2}x_1 & \sqrt{2}x_2 \end{bmatrix} \tag{21}$$

Construction of kernel

X 안함, 시험X.

임의의 kernel function 대해서 $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}')$가 성립하는 feature mapping이 존재하는지 검사하는 것은 매우 까다로움.

Mercer's theorem

Every positive semi-definite symmetric function is a kernel.

다음과 같이 정의되는 Gram matrix K가 positive semi-definite symmetric matrix면 function k는 kernel function임.

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & k(\mathbf{x}_N, \mathbf{x}_2) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \tag{22}$$

Gaussian radial basis function (RBF) kernel

많이 쓰는 kernel.

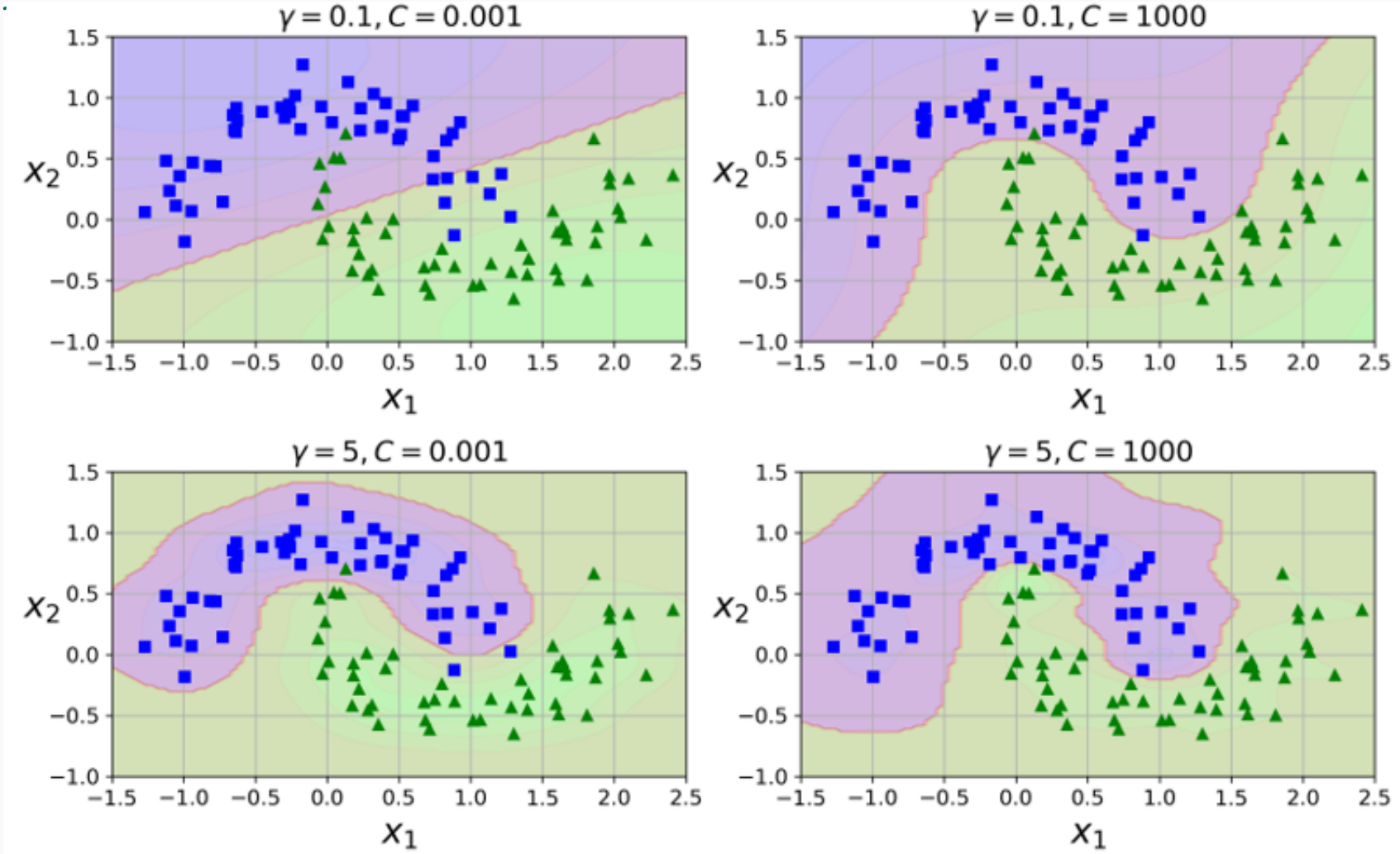
Gaussian RBF kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2) \tag{23}$$

유사도: $\mathbf{x}, \mathbf{x}'$ 가까움 → ↑
멀면 → ↓.

- kernel value는 $\mathbf{x}'$가 landmark $\mathbf{x}$와 가까울수록 큰 값
- γ 값이 커지는 경우 landmark에서 멀어질수록 kernel 값이 급격하게 작아짐 = 영향력의 범위가 좁아짐

Hyperparameter of soft SVM-RBF



SVM: strength and limitations

장점

- 강력하고 우수한 이론을 바탕으로 함
- 많은 블랙박스 알고리즘과는 대조적으로 비교적 직관적인 해석과 이해가 가능
- 학습이 상대적으로 쉬움
- 신경망처럼 지역 최적값에 빠지는 일이 없음
- 학습 시간이 차원에 의존하지 않으며 kernel trick 덕분에 고정된 입력에만 의존
- 과적합이 잘 조절되는 경향
- 많은 분야에서 신경망 및 기타 알고리즘과 필적하는 성능
- 데이터가 작은 조건이나 고차원 공간에서도 잘 일반화

SVM: strength and limitations

Cost ∈ max. margin.

hard SVM → linearly separable.
Soft SVM → 완화.
↕
다른 알고리즘.

단점

1. 노이즈에 민감: 비교적 적은 수의 잘못된 label로 성능이 심각하게 악화
   - 커널 함수의 선택/구축하는 방법에 대한 정리된 원칙이 없음
   - hyperparameter C의 적정값을 정하기 위한 원칙이 없음
2. 컴퓨터의 메모리와 계산 시간 측면에서 비용이 높은 편이며 multiclass에서 더욱더 심화됨

ECOC.

Support vector 개수.
- 항상 들고다녀야함.
- 보통 → S.V 들고다고 kernel을 전부 비교.

Appendix

Reference and further reading

- “Chap 7 | Sparse Kernel Machines” of C. Bishop, Pattern Recognition and Machine Learning
- “Chap 5 | Support Vector Machines” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow
- “Chap 4 | Convex Optimization Problems”, “Chap 5 | Duality” of S. Boyd, Convex Optimization
- “Lecture 6 | Support Vector Machines” of Kwang Il Kim, Machine Learning (2019)