

13기 정규세션

ToBig's 12기 박진혁

Support Vector Machine

Contents

Unit 01 | Support Vector Machine이란?

Unit 02 | Soft Margin VS Hard Margin




Unit 03 | Non-Linear SVM

Unit 04 | 마무리

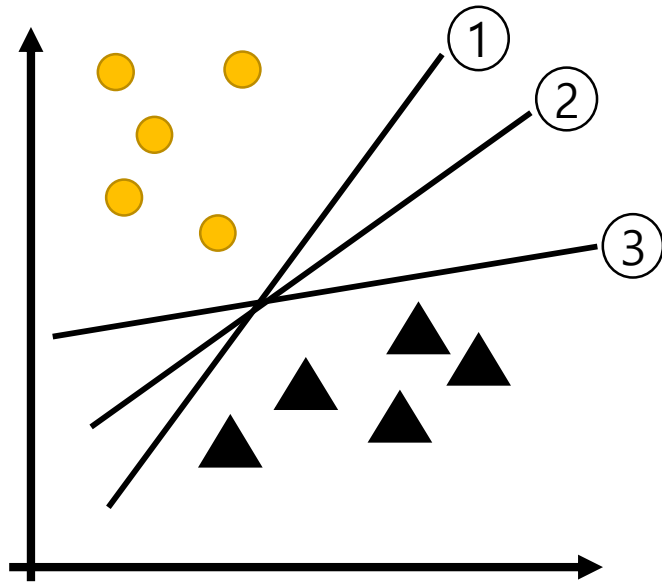
Unit 01 | Support vector machine이란?

Support Vector machine

: 주로 **바이너리 분류**를 하기 위해 사용되는 기법 (어떤 건지는 차근차근 설명해 드릴게요!)

- Bell 연구소의 Vladimir Vapnik이 1963년에 개발하고, 1992년에  커널 트릭의 사용으로 발전
- Deep Learning이 사용되기 전까지 **가장 성능이 좋은 모델로 평가됨**. 다만 그만큼  연산이 오래걸리는 편
-  **분류 문제에서는 뛰어난 성능을 보임!** 머신러닝으로 문제를 해결한다면 몇 번쯤 접해볼만한 방법

Unit 01 | Support vector machine이란?

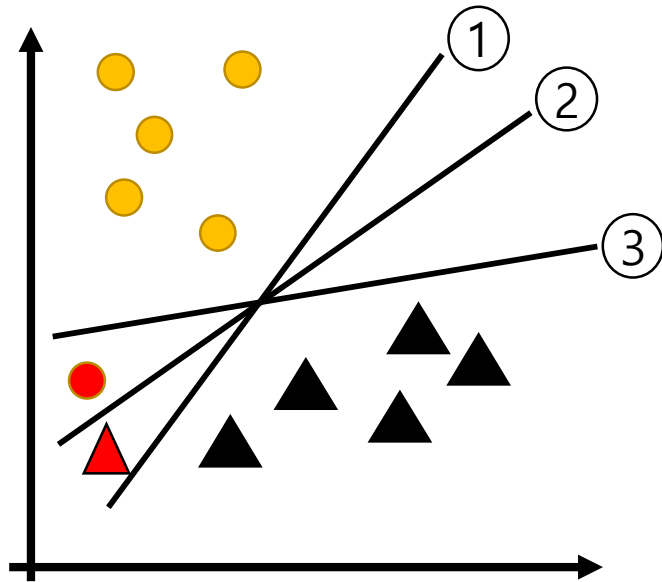


**어떤 선이 가장 데이터를 잘 나누는 걸까?
(전지적 Support Vector Machine 관점에서)**

Unit 01 | Support vector machine이란?

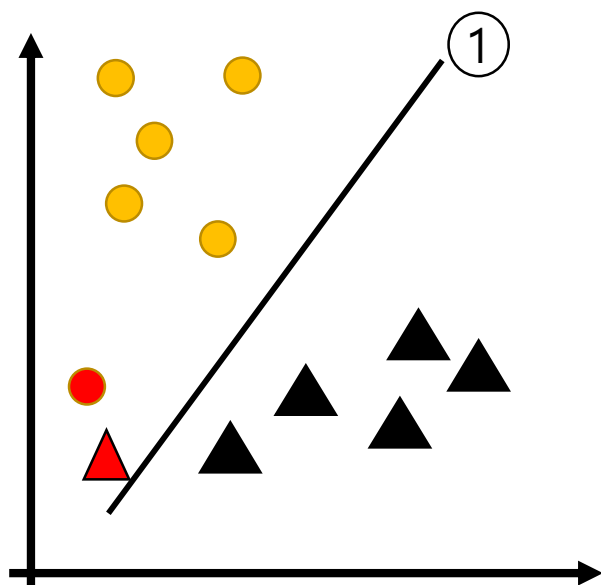
정답은 2번

Unit 01 | Support vector machine이란?



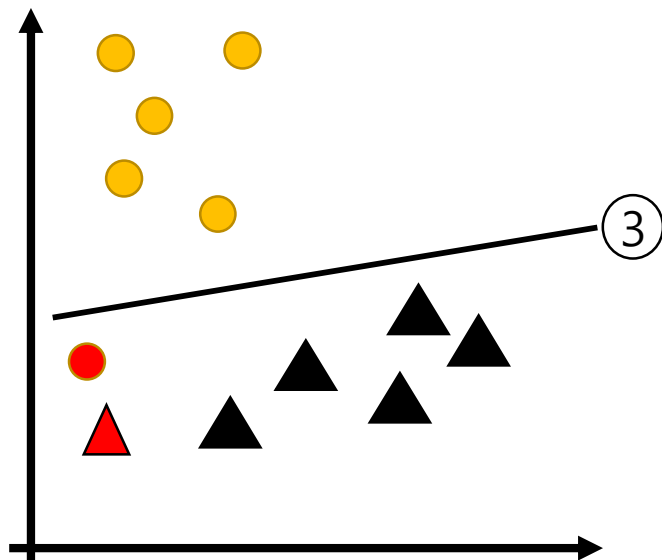
저 두 데이터가 추가 됐다고 합시다

Unit 01 | Support vector machine이란?



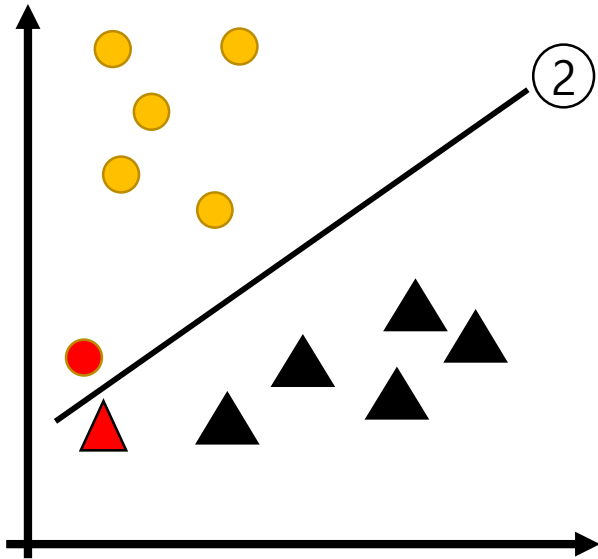
1번 분류기 삼각형 분류 실패

Unit 01 | Support vector machine이란?



3번 분류기 동그라미 분류 실패

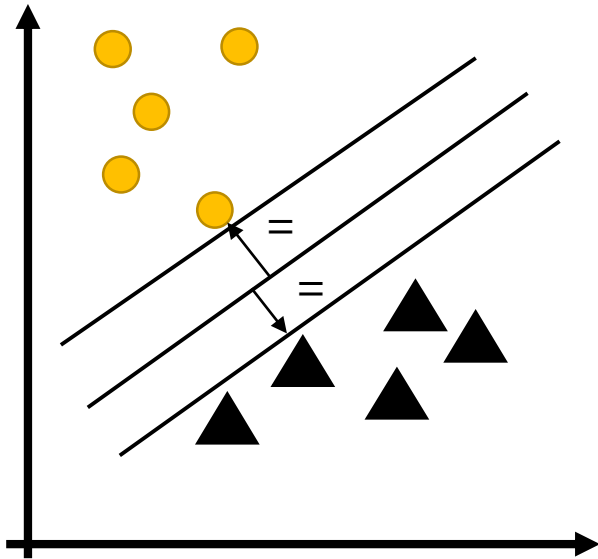
Unit 01 | Support vector machine이란?



2번이 좀 더 잘 분류할 가능성이 높아보인다!

왜냐? 원래 있던 데이터들과 경계선이 충분히 많이 떨어져 있으니까 좀 더 모호한 데이터가 나와도 구분을 잘 한다!

Unit 01 | Support vector machine이란?

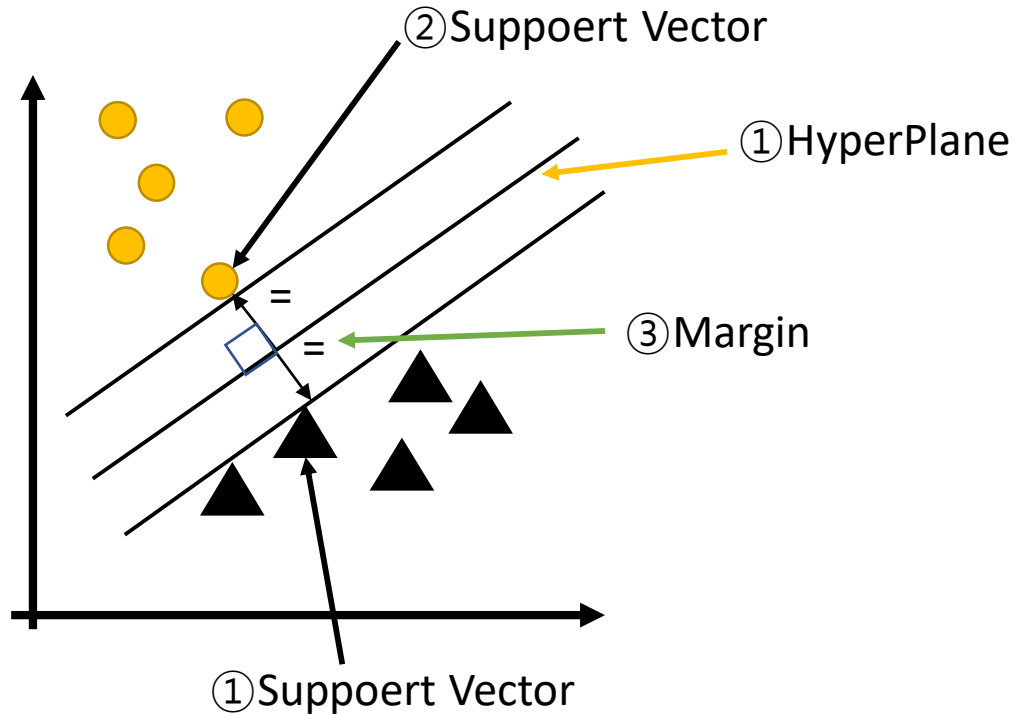


Support Vector Machine은 결국 위 내용을 기준으로
어떤 데이터를 가장 잘 분류하는 Hyper Plane을 찾는것!

Unit 01 | Support vector machine이란?

용어정리

● 레이블 A
▲ 레이블 B



① Hyper Plane : 여러 데이터를 나누는 기준이 되는 경계, 평면

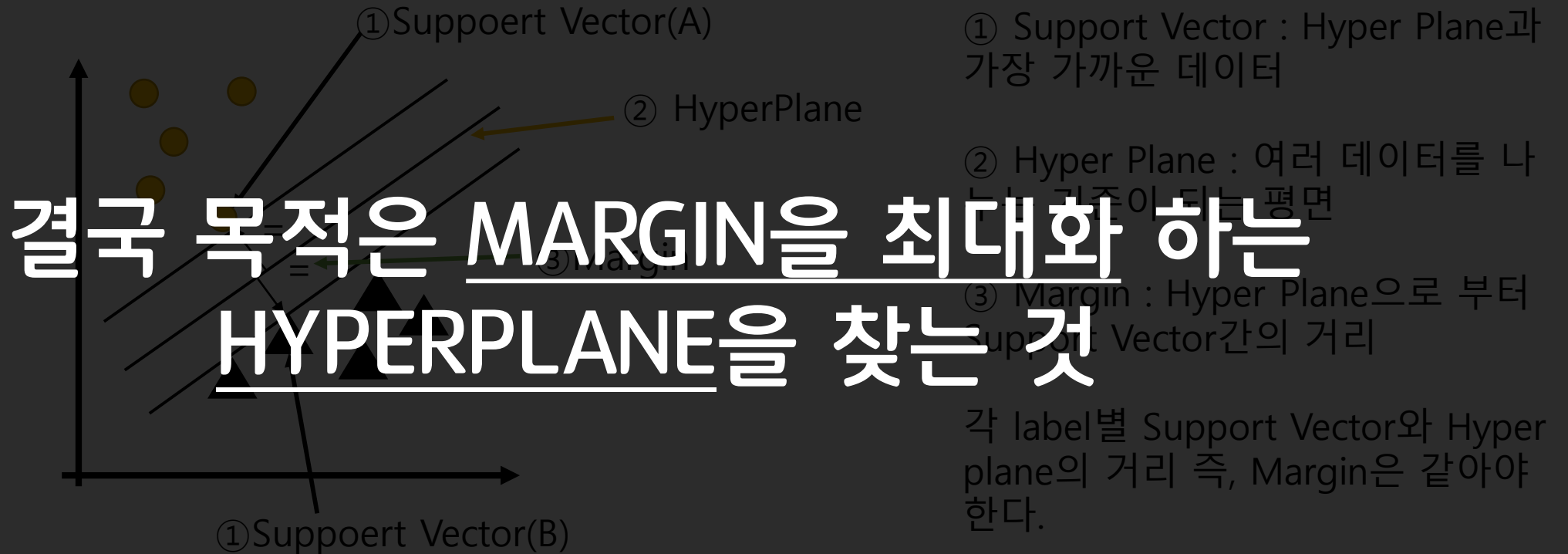
② Support Vector : Hyper Plane과 가장 가까운 데이터

③ Margin : Hyper Plane으로 부터 Support Vector간의 거리 $\times 2$

각 label별 Support Vector와 Hyper plane의 거리 즉, Margin은 같아야 한다.

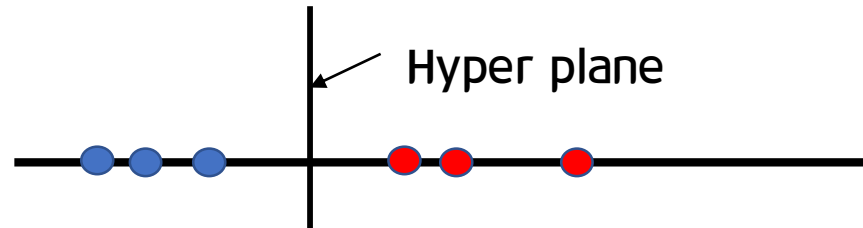
=> 데이터를 분류할 수 있는 여러 경계(Hyper Plane) 중 **마진을 최대화 하는 경계가** 제일 데이터를 잘 분류한다
는게 SVM

Unit 01 | Support vector machine이란?



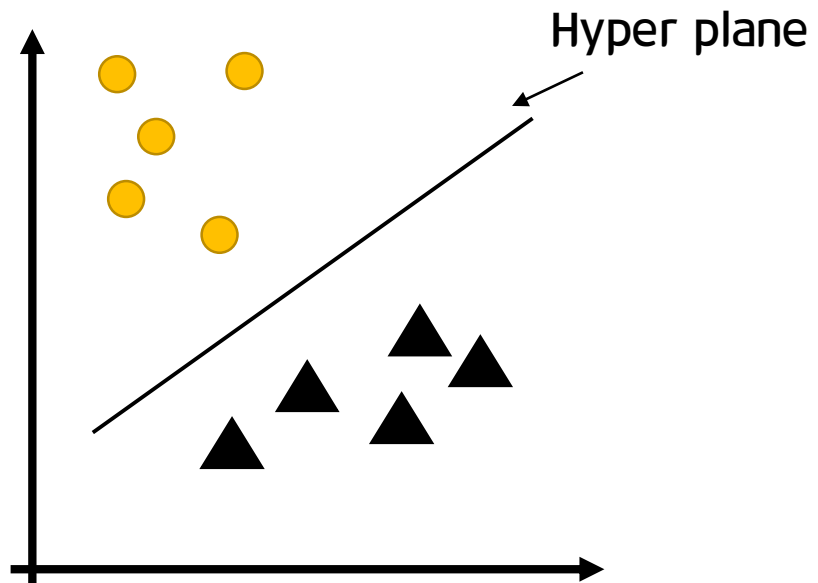
Unit 01 | Support vector machine이란?

1차원 : 선

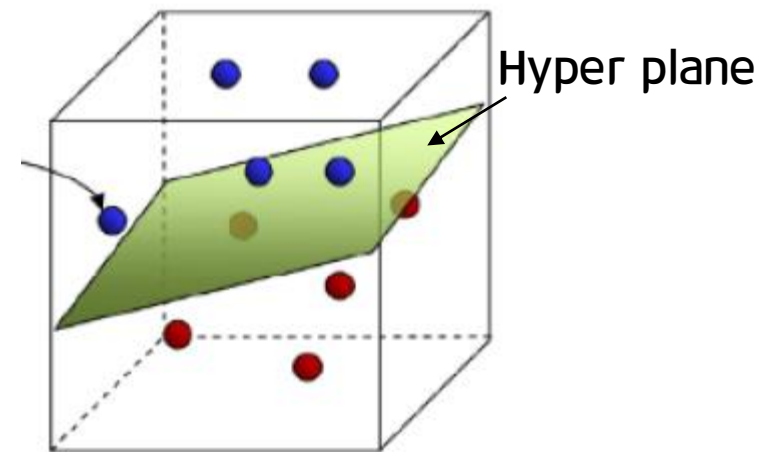


여러 차원에서의 분류되는 모습

2차원 : 평면

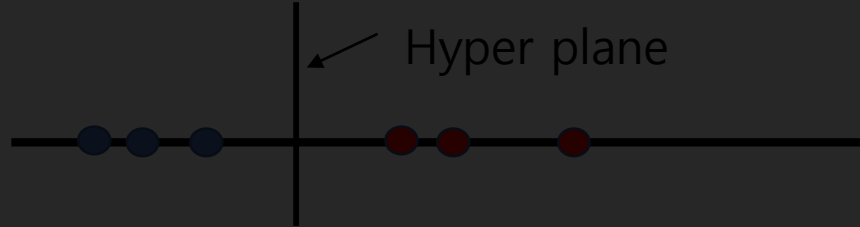


3차원



Unit 01 | Support vector machine이란?

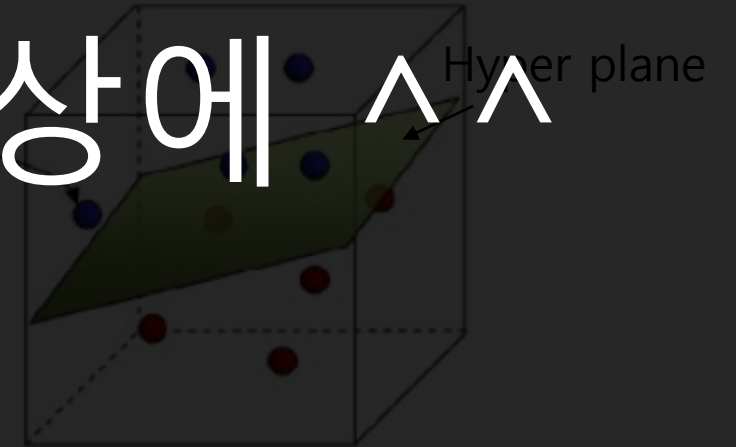
1차원 : 선



2차원 : 평면



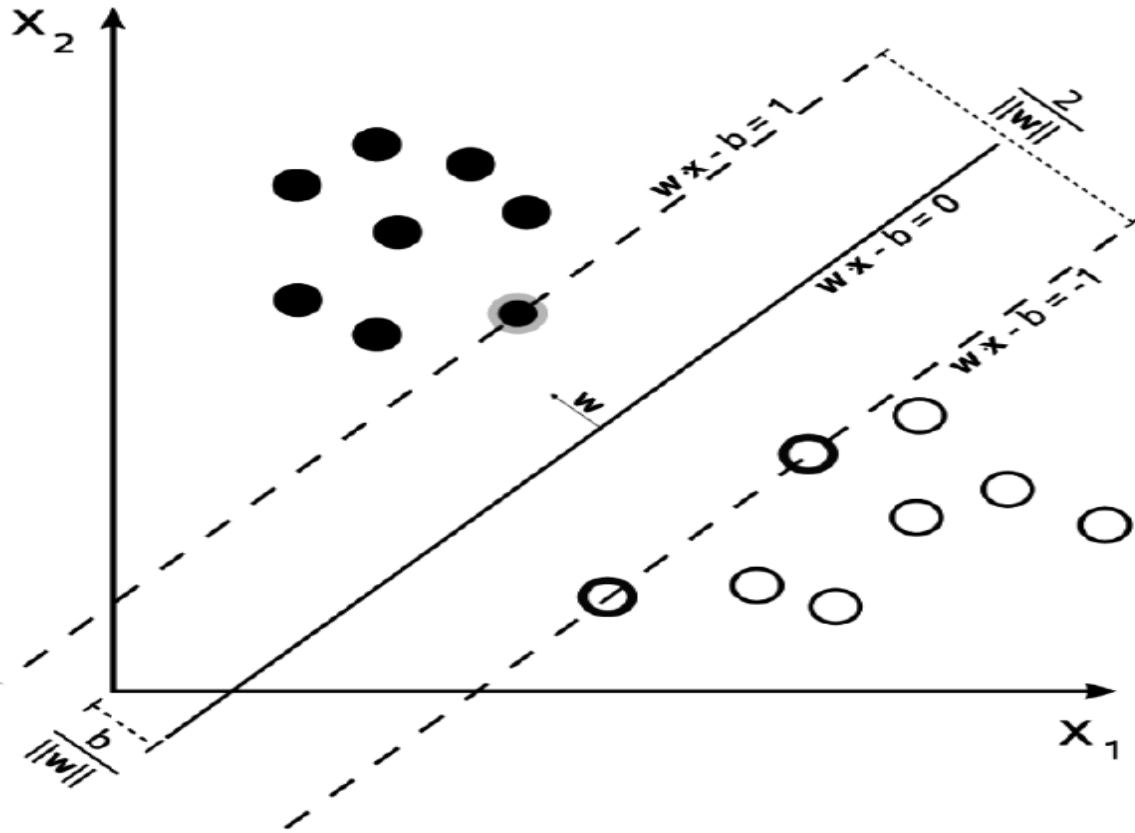
3차원



N차는 여러분의 상상에 ^^

Unit 01 | Support vector machine이란?

여기서 잠깐 고등학교 수학시간~



Machine Learning with R, Brett Lantz

모든 plane 은 $W * X - b = 0$ 로 표현된다Ex) $y = ax + b$ $y = a * x + b \rightarrow$ 축이 x_1, x_2 임으로 축 변경 $x_2 = a * x_1 + b \rightarrow$ 왼쪽으로 다 넘겨주면 $a * x_1 - x_2 + b = 0 \rightarrow$ 일반화를 위해 계수모양을 바꾸자 $a' * x_1 - a'_2 * x_2 + b' = 0 \rightarrow$ 행렬로 표현하면

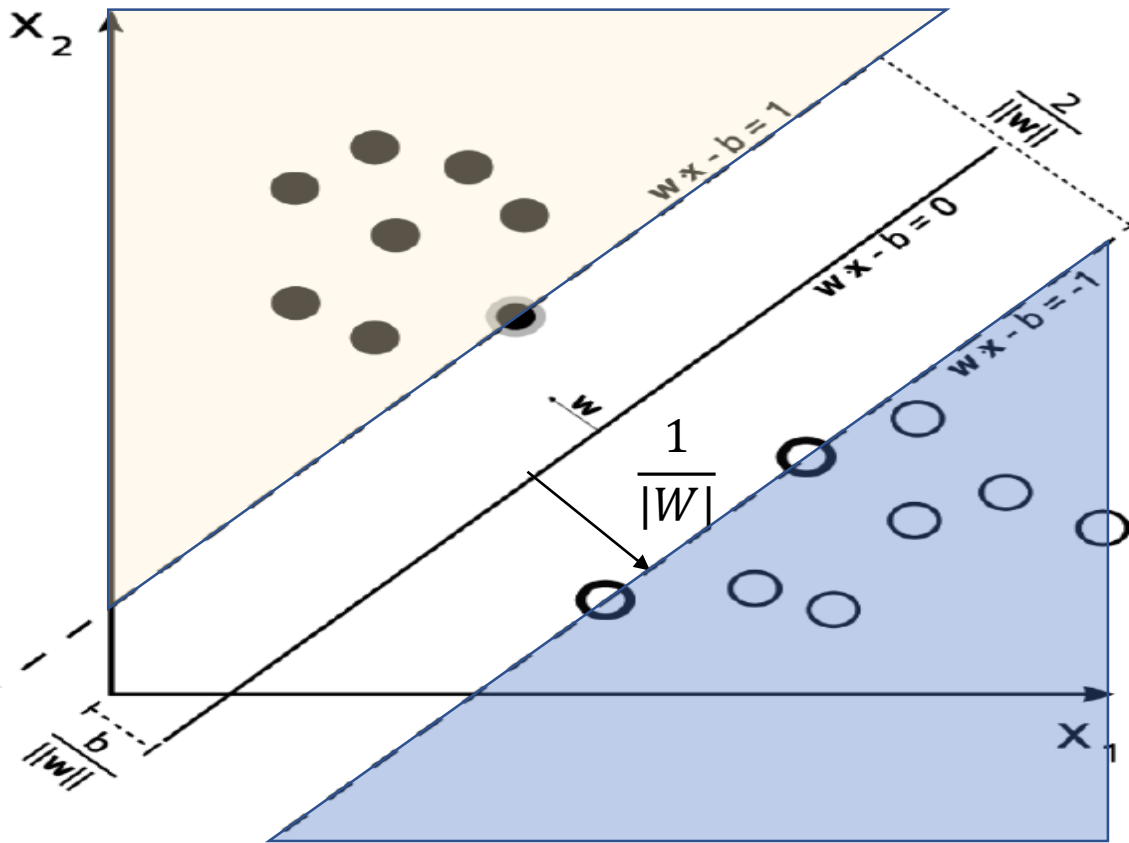
$$[a' \quad -a'_2] * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + b = 0 \rightarrow [a' \quad a'_2] = W \quad \begin{bmatrix} x_1 \\ -x_2 \end{bmatrix} = X$$

$$W * X - b = 0$$

모든 plane은 $W * X - b = 0$ 로 표현할 수 있다.

$$\text{ex) 3차원} \rightarrow ax + by + cz + d = 0 \rightarrow [a \quad b \quad c] * \begin{bmatrix} x \\ y \\ z \end{bmatrix} + d = 0$$

Unit 01 | Support vector machine이란?



Machine Learning with R, Brett Lantz

아직 헷갈리시면 $W^*X - b = 0$ 을 $-x + y + 1 = 0$ 이라 두고
 $W = [-1 \ 1]$ $X = [x \ y]$ $b = -1$ 으로 대입해서 생각해 보세요!

$W^*X - b < -1$: 파란 부분

$W^*X - b > 1$: 노란 부분

$W^*X - b = -1$: 파란 부분 경계선

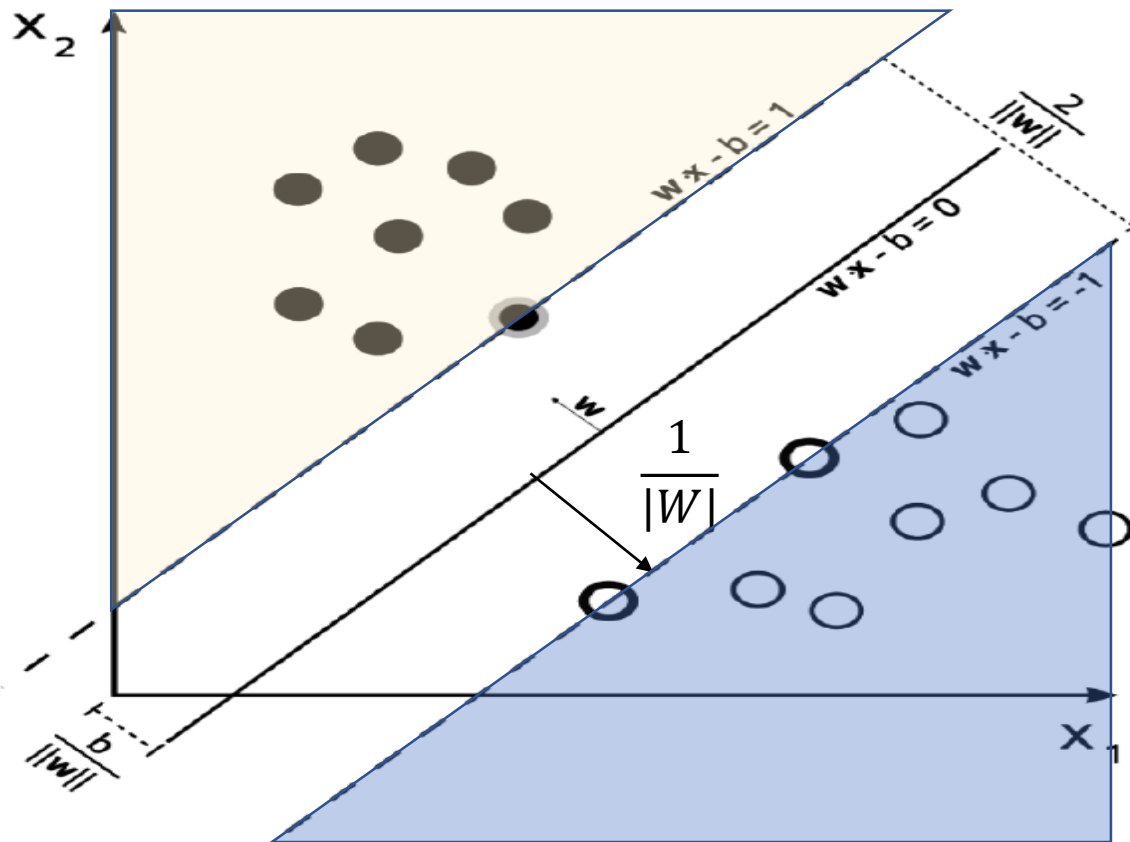
$W^*X - b = 1$: 노란 부분 경계선

$W^*X - b = 0$ 과 $W^*X - b = 1$ 사이의 거리 : $\frac{1}{|W|}$

-> 영역에 대한 감이 오시나요?

Unit 01 | Support vector machine이란?

다시 본론으로 돌아와보면



Machine Learning with R, Brett Lantz

모든 데이터를 잘 분류하면서 마진을 최대화 하는 hyper plane의미?

$-1 < W \cdot X - b < 1$ 영역에는 데이터가 없고
 label이 A인 데이터는 $W \cdot X - b \geq 1$ 에 있고
 label이 B인 데이터는 $W \cdot X - b \leq -1$ 에 있게 만들자
 이것을 수식으로 나타내면 다음과 같다.

Label : Label이 A면 +1, B면 -1의 값을 가진다!
 $y_i(w \cdot x_i - b) \geq 1, \text{ for all } 1 \leq i \leq n.$

이 때, 마진의 크기는 ($WX - b = 0$ 과 $WX - b = 1$ 사이의 거리)는 $\frac{1}{|W|}$ 이 된다.

따라서 위 조건을 만족하는 W와 b에 대해 $\frac{1}{|W|}$ 가 최대가 되는 쌍, 즉 W가 최소가 되는 쌍을 찾으면 그게 Hyper plane이 된다는 것이다.

Unit 01 | Support vector machine이란?

한 번만 더 정리하면..

조건(제약식): $y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1$, for all $1 \leq i \leq n$. : 모든 데이터들은 결정경계안에 잘 나뉘들어가있어야한다

목적식 : 위 조건을 만족하면서 $\frac{2}{|\mathbf{w}|}$ 가 최대가 되게하는(margin이 최대가 되는) \mathbf{w} 와 b 를 찾자 !

그런데 우리는 문제 해결을 용이하게 하기 위해 $\frac{|\mathbf{w}|^2}{2}$ 을 최소화 하는걸로 바꾸자!

그런데 그 해를 찾는 방법은 조금 어렵다!

Unit 01 | Support vector machine이란?

1. 라그랑주 승수법을 이용하면 목적 + 제약을 식 하나로 바꿀 수 있다.

라그랑주 승수법?

어떤 $f(x,y)$ 를 최대화 or 최소화 하는 문제가 있다고 하자! 이 때 $g(x,y) = 0$ 이라는 제약식(조건)이 있을 때 이 둘을 하나로 합쳐서 계산하는 방법이 라그랑주 승수법이다

목적식이 f 제약식이 g 로 주어져 있으면, 이 g 에 gradient방향과 f 의 gradient 방향이 같을 때 목적식의 최적값이 결정된다. (왜 그런지는 잘 생각해보세요! 뒤에 링크도 걸어드립니다)

$$\nabla_{x,y} f = \lambda \nabla_{x,y} g,$$

따라서 L 을 다음과 같이 정의하면

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda g(x, y)$$

다음과 같은 식을 만족하는 문제를 푸는 문제로 바꿀 수 있게 된다!!

$$\nabla_{x,y,\lambda} \mathcal{L}(x, y, \lambda) = 0$$

Unit 01 | Support vector machine이란?

1. 라그랑주 승수법을 이용하면 목적 + 제약을 식 하나로 바꿀 수 있다.

그렇다면 제약식이 여러 개라면?

모든 제약식에 대해, f 의 그래디언트와, g 의 그래디언트가 같은 방향일때 최대가 된다..!

이걸 식으로 정리하면

$$\nabla f(\mathbf{x}) = \sum_{k=1}^M \lambda_k \nabla g_k(\mathbf{x})$$

어차피 방향이 중요하므로 각 $g_k(x)$ 의 그래디언트 방향만 같다면, scaling하고 더하고 하는 과정은 영향이 없다.

그렇다면 우리의 목적식과 제약식을 다시 한 번 보자

목적 $f(x)$: $\frac{1}{2} \|\mathbf{w}\|^2$ 을 최대화 하자

제약 $g(x)$: $y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1$ 모든 점들이 자기 영역으로 분류 돼있어야 한다.

그런데, 예는 라그랑주 승수법이랑 다르게 '부등식'이다 !!

Unit 01 | Support vector machine이란?

2. KKT 조건

라그랑주 승수법으로 해결할 수 없는 부등식에 대해서는 KKT 조건이라는 것이 존재하는데 이것은 최적값이기 위한 필요충분 조건이다. 이것 역시 들어가면 대학원 과정에 너무 깊어지므로 조건만 짚고 넘어가자

1. 라그랑제 승수를 제외한 변수에 대한 편 미분 값은 0이 되어야 한다.
 2. 라그랑제 승수는 무조건 0보다 크거나 같아야한다.
 3. 라그랑제 승수 혹은, 제약식중 하나는 무조건 0이 되어야한다. (즉 두개의 곱은 항상 0이되어야 한다)
- > 이것을 우리 식에 대입해보자, 변수는 W, b 라그랑제 승수를 α 제약식은 $(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1)$ 라고 할 수 있다!

$$\textcircled{1} \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \alpha) = 0, \frac{\partial}{\partial b} L(\mathbf{w}, b, \alpha) = 0$$

$$\textcircled{2} \alpha_i \geq 0, i = 1, 2, \dots, N$$

$$\textcircled{3} \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1) = 0, i = 1, 2, \dots, N$$

->조건을 잘 전개하면 우리가 얻고자 하는 식이 나온다!! 전개하는 과정은 레퍼런스 걸어드리겠습니다. 내용이 생각보다 간단하니 한 번 과제로 직접 해보셔도 될 것 같아요! (갠톡으로 질문해주셔도 좋습니다!)

Unit 01 | Support vector machine이란?

그렇다면 결론은 짚고 넘어갑시다 (**의미가 중요함**)

아까 봤던 그 조건을 적용하여 전개하면, W가 최대가 되는 W와 b를 찾는 문제는 다음과 같은 문제로 바뀌게 됩니다.

Subject to $\sum_{i=1}^N \alpha_i y_i = 0, \alpha_i \geq 0, i = 1, 2, \dots, N$ <- 이 **조건**을 만족하는 알파에 대해

Maximize $Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ <- 이 **식을 최대화** 하는 α 를 찾으면

$$\hat{\mathbf{W}} = \sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i \quad \leftarrow \text{W(하이퍼플레인의 방향)은 이렇게 된다!}$$

$$\text{sgn} \left(\sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i^T \mathbf{x} + \hat{b} \right) \quad \leftarrow \text{분류결과 는 이렇게 된다.}$$

sgn? : 함수안에 값이 양수면 1 음수면 -1 뱃어나는 함수! 즉 저 안을 계산한 값
이 양수면 라벨이 1 아니면 -1이 된다 !!!

애가 Hyper plane과의 거리! 즉 **Decision function**

$$\hat{b} = \frac{1 - y_i \hat{\mathbf{W}}^T \mathbf{x}_i}{y_i}$$

* b는 이렇게돼요

Unit 01 | Support vector machine이란?

증명하기에는 매우 시간이 오래걸려서 자세한 내용은 레퍼런스 + 추가자료 걸어드립니다! (갠톡으로 질문환영)
참고로 대학원 수업에서 PPT 45페이지 동안 다루는 내용입니다 (속닥속닥)

다만 결론은 짚고 넘어가요 (**의미가 중요함**)

Subject to $\sum_{i=1}^N \alpha_i y_i = 0, \alpha_i \geq 0, i = 1, 2, \dots, N$ <- **조건**을 만족하는 알파에 대해

Maximize $Q(\alpha) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ <- **이 식을 최대화** 하는 α 를 찾으면

$\hat{\mathbf{w}} = \sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i$ <- **W(하이퍼플레인의 방향)**은 이렇게 된다!

$\text{sgn} \left(\sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i^T \mathbf{x} + \hat{b} \right)$ <- **분류결과** 는 이렇게 된다.

α 를 찾는 문제로 바뀐다!!
컴퓨터는 이게 더 풀기 쉬움!

Unit 01 | Support vector machine이란?

한 가지 더 짚고 넘어가야할 부분! 별표 친 부분을 잘 봐주세요

Subject to $\sum_{i=1}^N \alpha_i y_i = 0, \alpha_i \geq 0, i = 1, 2, \dots, N$ <- 조건이 이러할때

Maximize $Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ <- 이 식을 최대화 하는 α 를 찾으면

$\hat{\mathbf{w}} = \sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i$ <- \mathbf{w} (하이퍼플레인)은 이렇게 된다!

분류 결과와, 분류하는 알파를 찾을 때, $\mathbf{x}_i, \mathbf{x}_j$ 벡터가 필요없이 **내적값만 필요하다**

$\text{sgn} \left(\sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i^T \mathbf{x} + \hat{b} \right)$ <- 분류결과 는 이렇게 된다.

Unit 01 | Support vector machine이란?

별표 친 부분은
뒤에 또 나오니까 기억해주세요!
물론 또 알려드릴림

Contents

Unit 01 | Support Vector Machine이란?

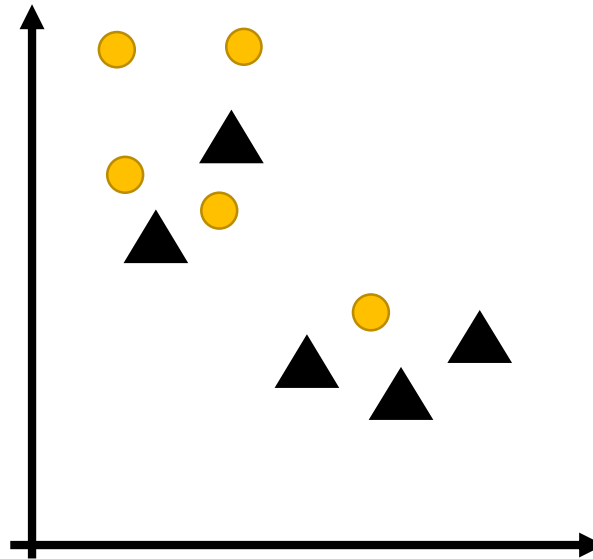
Unit 02 | Soft Margin VS Hard Margin

Unit 03 | Non-Linear SVM

Unit 04 | 마무리

Unit 02 | Soft margin vs Hard Margin

아 서포트 벡터 머신이 뭔진 알겠어
그런데 현실적으로 모든 데이터가 저렇게 예쁜 경계로 나뉠까..?
이런 애들은 어떻게 하지...???????

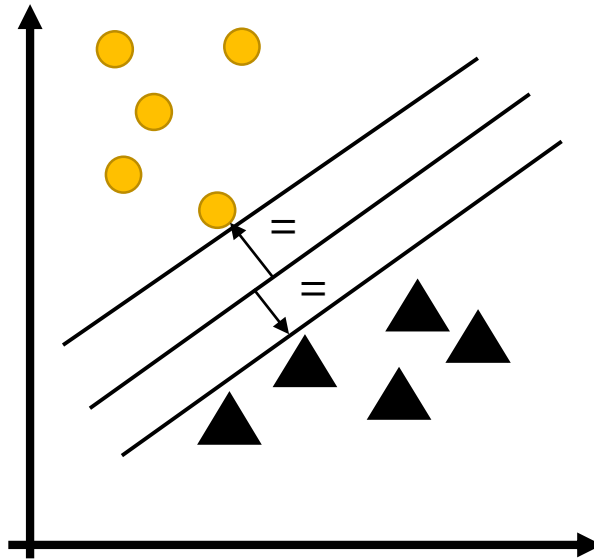


Unit 03 | Soft margin vs Hard Margin

Soft Margin VS Hard Margin

Unit 02 | Soft margin vs Hard Margin

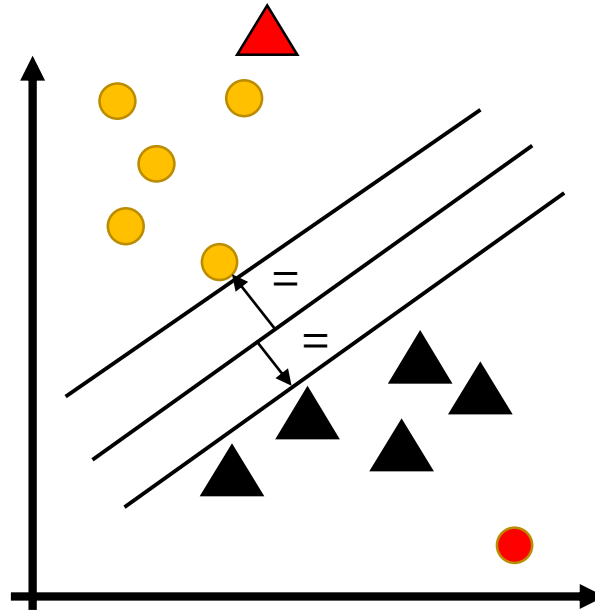
Hard Margin SVM : Error 따윈 없다. 모든 것을 정확하게 분류할래
-> 우리가 여태까지 했던 모든 그것



현실적으로 이렇게 나눌 수 있는 경우가 얼마나 될까?

Unit 02 | Soft margin vs Hard Margin

Soft Margin SVM : 저런 애들 하나 두 개 때문에 평면을 제대로 못나누는건 좀 그래
... 몇 개는 틀려도 괜찮아



에러를 허용하되, 에러에 대한 페널티를 준다! 틀려도 되는데, 틀리는만큼 감점할테
니까 감점하는거 다고려해서 잘 해보렴^^

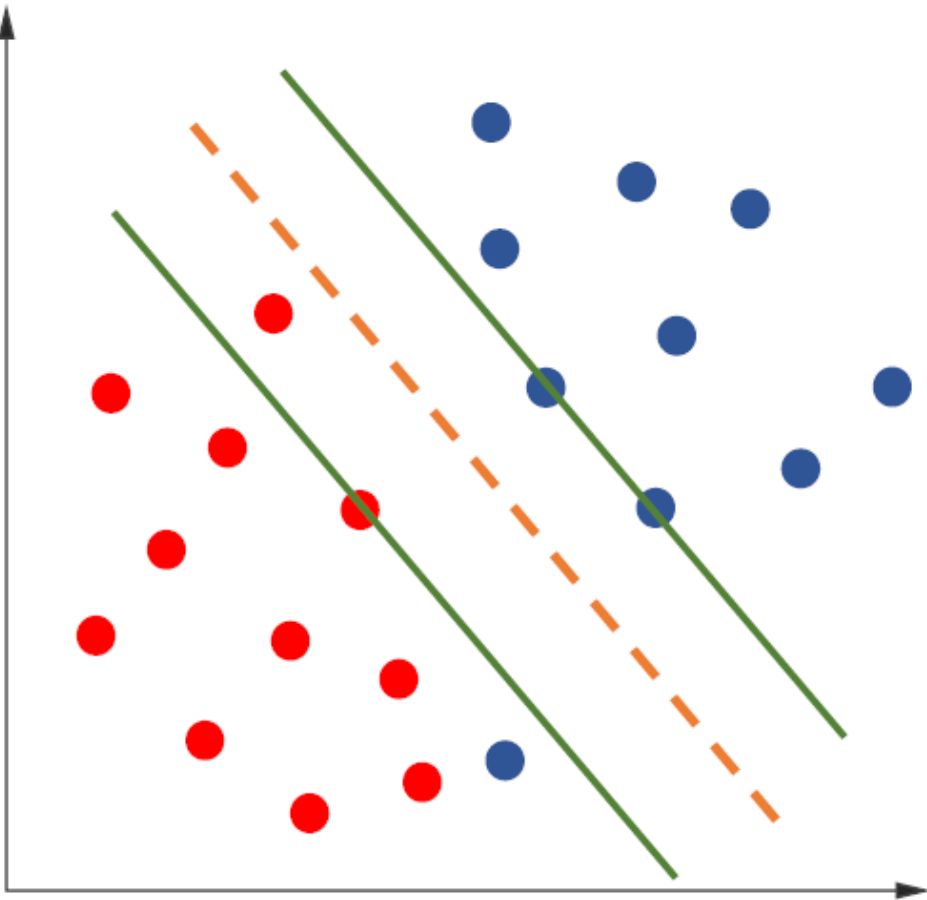
Unit 02 | Soft margin vs Hard Margin

Penalty를 주는 방법

Option 1 (0-1 LOSS) VS Option 2 (Hinge Loss)

Unit 02 | Soft margin vs Hard Margin

Soft Margin SVM



Option 1)

에러 발생 개수 만큼 패널티를 계산하자

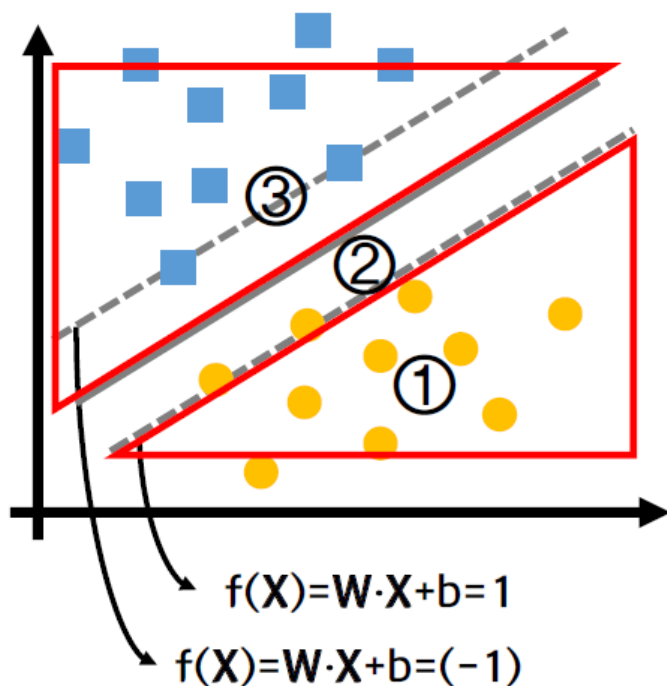
$$\min ||w|| + C * \#error$$

→ 0-1 Loss

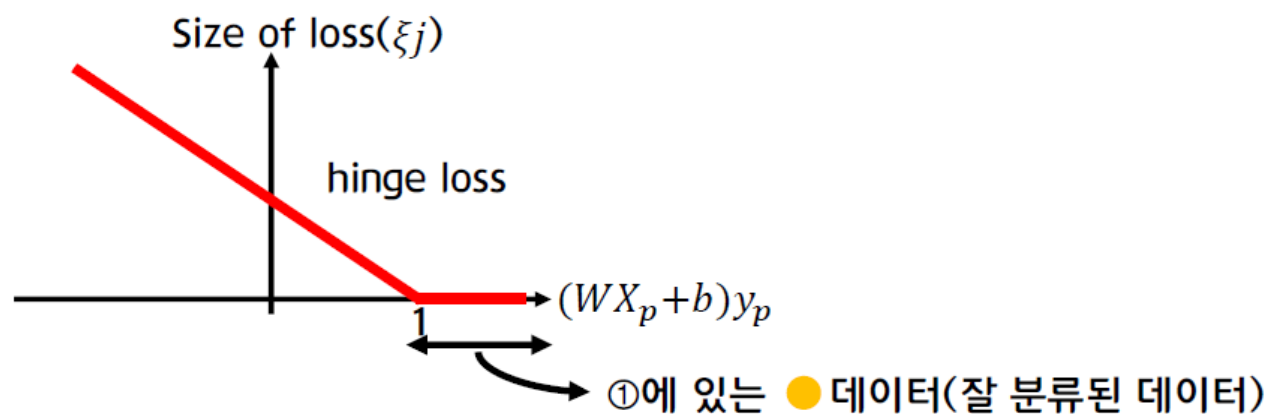
Unit 02 | Soft margin vs Hard Margin

<Soft Margin SVM>

Option 2)

Slack variable(ξ_j)

- 데이터가 분류기의
- ①번에 있는 경우: $(WX_p + b)y_p \geq 1 \rightarrow \xi_j = 0$
 - ②번에 있는 경우: $0 \leq (WX_p + b)y_p < 1 \rightarrow 0 < \xi_j \leq 1$
 - ③번에 있는 경우: $(WX_p + b)y_p < 0 \rightarrow \xi_j > 1$
- > hinge loss

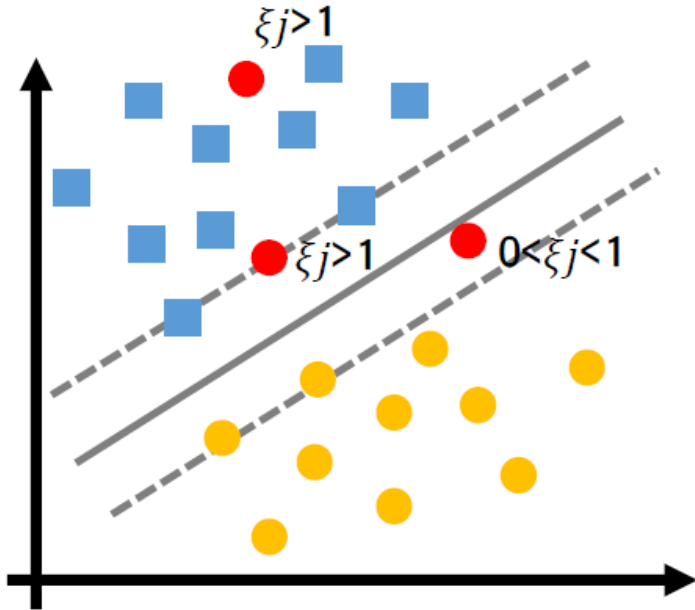


↓ hinge function

$$\max(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)).$$

Unit 02 | Soft margin vs Hard Margin

Soft margin svm



Soft margin SVM

- 기존 목적함수에 error항을 추가
- $\underset{W,b}{\operatorname{argmin}} ||W|| + C \sum \xi_j$

C는 하이퍼 파라미터!

C가 크다? 에러항이 작아져야 목적식이 더 작아지겠네 W줄이는 것 보단 error 줄이는 걸 더 신경써야 겠다

C가 작다? 에러항이 좀 커져도 영향력이 없군 에러가 좀 있더라도 W를 줄이는데 신경을 써야겠다

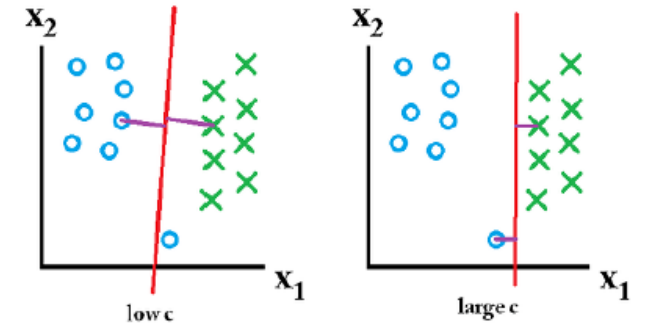


그림6. 매개변수 C의 영향

작은 C는 underfitting이 될 위험이 있고 큰 C는 overfitting의 위험이 있다!

Contents

Unit 01 | Support Vector Machine이란?

Unit 02 | Soft Margin VS Hard Margin

Unit 03 | Non-Linear SVM

Unit 04 | 마무리

Unit 03 | Non-Linear SVM

얘는 어떤 plane으로 나눌 수 있을까..?



Unit 03 | Non-Linear SVM

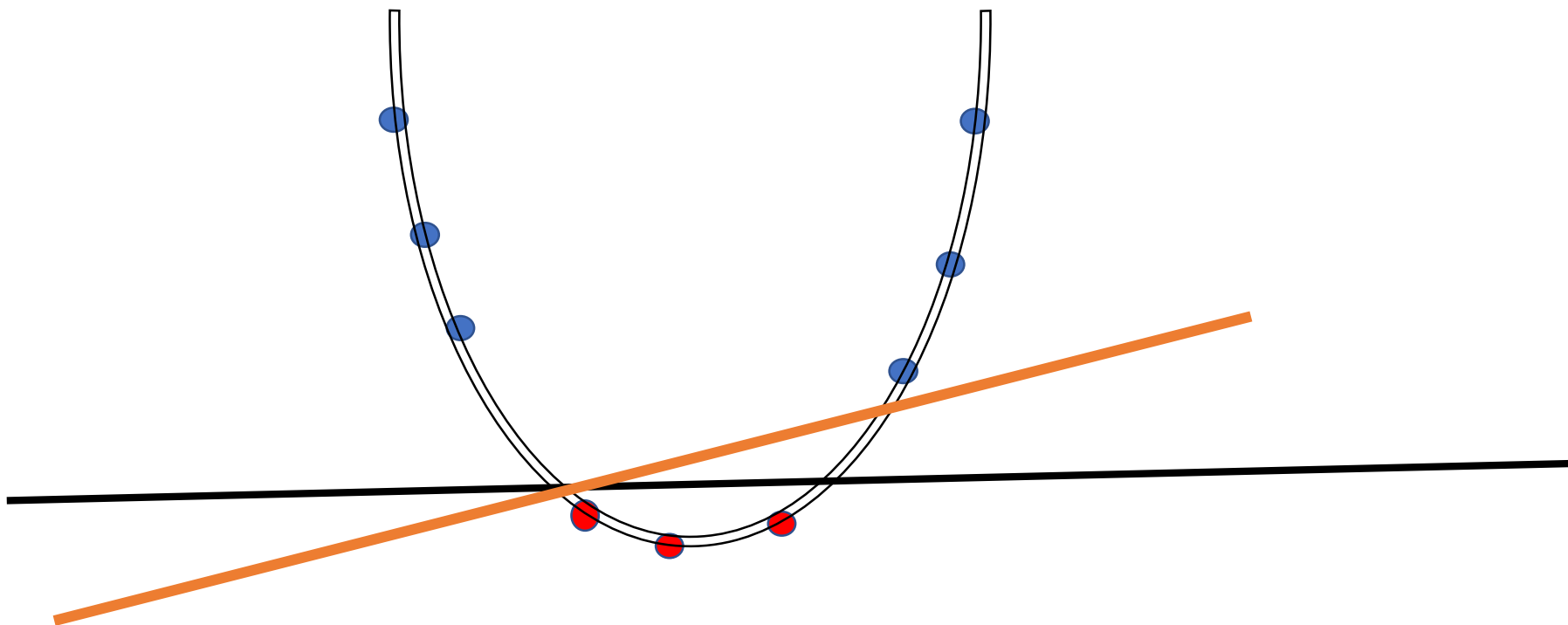
얘는 어떤 plane으로 나눌 수 있을까..?

불가능하다



Unit 03 | Non-Linear SVM

2차원으로 매핑하니 가능하다!



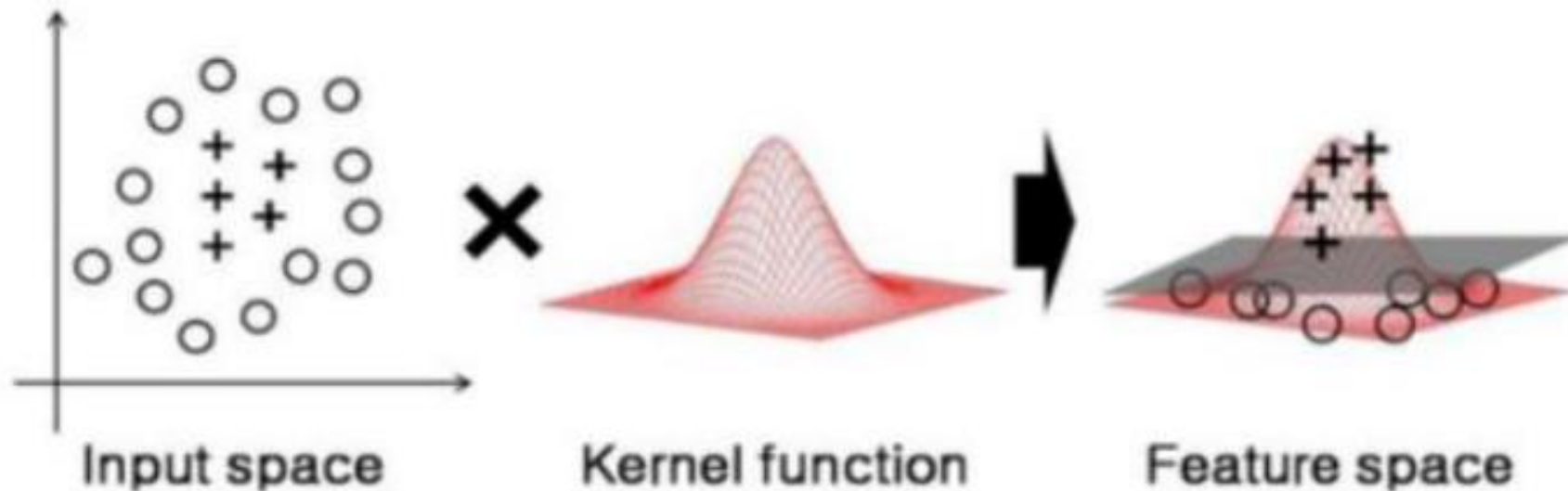
Unit 03 | Non-Linear SVM

Kernel ?

저차원 데이터를 고차원 데이터로 매핑하는 것!

Unit 03 | Non-Linear SVM

대부분의 문제들은 Linear하게 해결되지 않으므로 **고차원으로 Mapping**을 해야한다.



Unit 03 | Non-Linear SVM

그런데.. 연산량이 너무 많지 않을까?

Unit 03 | Non-Linear SVM

Kernel Trick!

$$\text{Maximize } Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \quad \leftarrow \text{이 식을 최대화 하는 } \alpha \text{를 찾으면}$$

$$\text{sgn} \left(\sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i^T \mathbf{x} + \hat{b} \right) \quad \leftarrow \text{분류결과 는 이렇게 된다.}$$

Unit 03 | Non-Linear SVM

Kernel Trick!

우리가 필요한 것은 고차원으로
매핑한 데이터가 아니라 내적값만 알면된다!

Maximize $Q(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j$ ← 이 식은 최대화 하는 α 를 찾으면

$\text{sgn} \left(\sum_{i=1}^n \alpha_i y_i x_i^T x + b \right)$ ← 분류결과 는 이렇게 된다.

Unit 03 | Non-Linear SVM

Kernel Trick!

$$\max L_D(\alpha_i) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \quad \Rightarrow \quad \max L_D(\alpha_i) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \Phi(x_i)^T \Phi(x_j)$$

Mapping 전

Mapping 후

매핑한 함수따위 필요없고 그냥 매핑한거 내적
값만 알면된다! 그 값을 쉽게 알 수있는 Trick

Unit 03 | Non-Linear SVM

Kernal Trick!


Ex) $\phi < x_1, x_2 > = < x_1^2, \sqrt{2}x_1x_2, x_2^2 >$ 로 Mapping 한다고 하자..

$$\begin{aligned} K(<x_1, x_2>, <z_1, z_2>) &= <x_1^2, \sqrt{2}x_1x_2, x_2^2> \cdot <z_1^2, \sqrt{2}z_1z_2, z_2^2> \\ &= x_1^2z_1^2 + 2x_1x_2z_1z_2 + x_2^2z_2^2 = (x_1z_1 + x_2z_2)^2 = \underline{(\mathbf{x} \cdot \mathbf{z})^2} \end{aligned}$$

=> 오잉? 2차원에서 3차원으로 매핑한걸 다시 내적했더니 그냥 2차원 내적인거 제공이네? 원래 쓰던거 제공한번만 해주면 3차원으로 매핑이 된다!

Unit 03 | Non-Linear SVM

Kernal Trick -> 이걸 만족하는 조건은 복잡함 (\therefore 생략)
-> 어차피 쓰는건 정해져있다!

linear	$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
quadratic	$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^2$
polynomial	$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^n$
sigmoid	$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(a\mathbf{x}_i^T \mathbf{x}_j + b)$
 Gaussian	$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ ^2}{\sigma^2}\right)$

<= 이 함수 쓰는 것도 튜닝이라면 튜닝
근데 그냥 Gaussain 쓰는 것을 추천!

그리고 각 함수마다 parameter가 조금씩 다르다!

Unit 03 | Non-Linear SVM

Gaussian Kernel?

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}\right) \longrightarrow K(x_1, x_2) = \exp\left\{-(x_1 - x_2)^2\right\}$$

σ^2 is circled with a blue circle and an arrow pointing to the text below.

$$= \exp(-x_1)\exp(-x_2)\exp(2x_1x_2)$$

얘는 parameter니까 그냥 임의로 두면..

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

An arrow points from the $\exp(2x_1x_2)$ term in the previous equation to this Taylor series expansion.

Exponential 한번만 하면 무한대 차원으로 kernel한걸 내적인 값을 얻을 수 있음
(by 테일러급수)

Unit 03 | Non-Linear SVM

Gaussian Kernel?

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}\right) \longrightarrow K(x_1, x_2) = \exp\left\{-(x_1 - x_2)^2\right\}$$

σ^2 is a parameter, so we can just assume it.

$$= \exp(-x_1) \exp(-x_2) \exp(2x_1 x_2)$$

애는 parameter니까 그냥 임의로 두면..

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

Exponential 한번만 하면 무한대 차원으로 kernel한걸 내적인 값 얻을 수 있음
(by 테일러급수)

Unit 03 | Non-Linear SVM

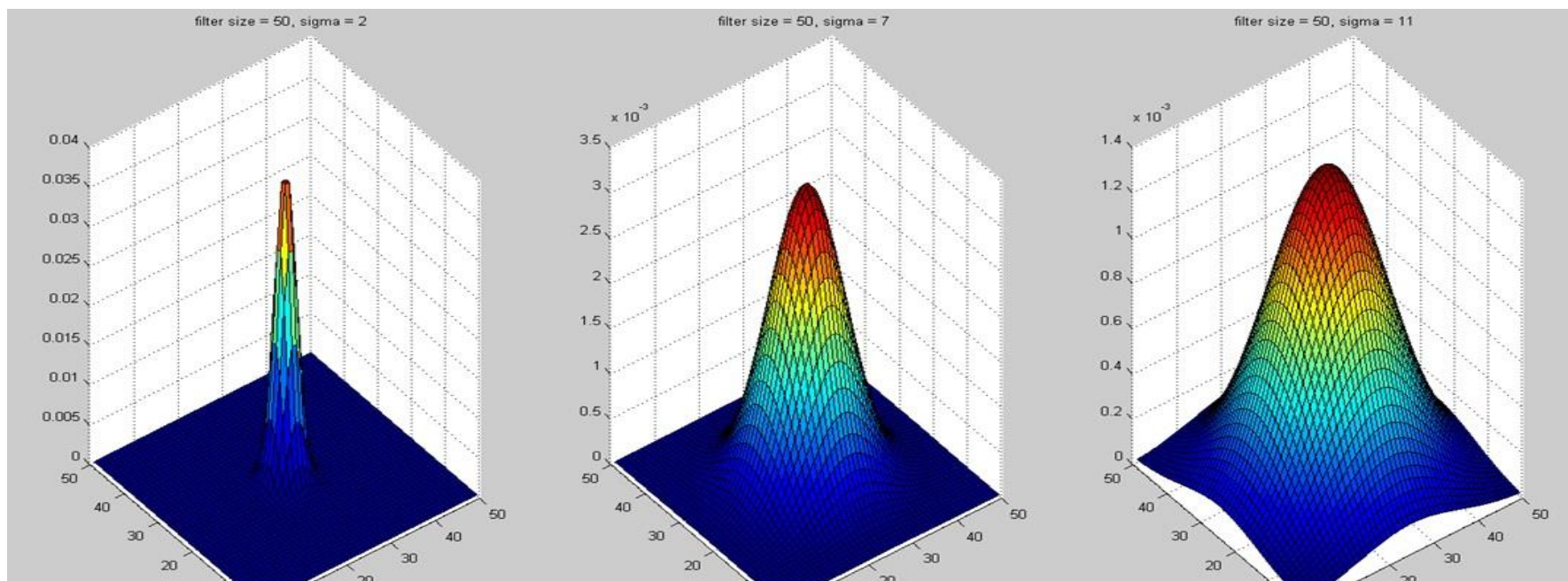
Gamma Parameter Tuning

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}\right) \longrightarrow \exp(-\delta |x_i - x_j|)$$

감마는 표준편차가 커질수록 작아짐!

Unit 03 | Non-Linear SVM

Gaussian Kernel? (RBF Kernel)

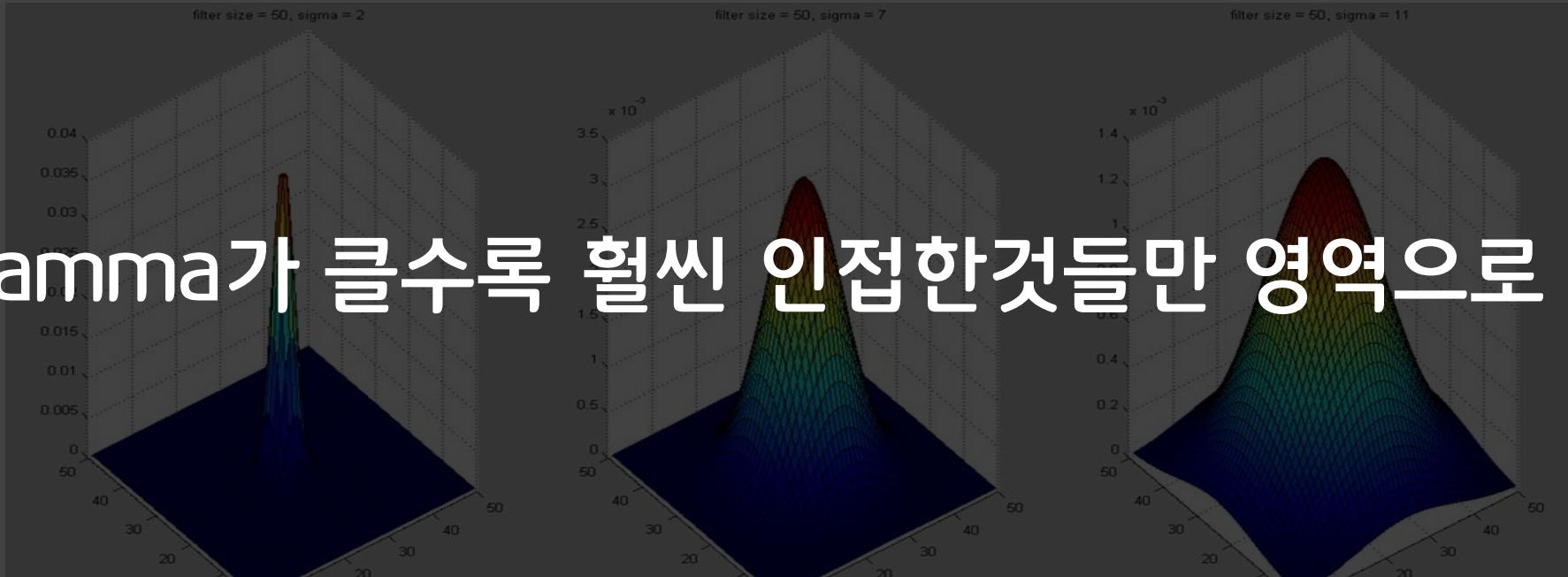


Gamma가 내려감

Unit 03 | Non-Linear SVM

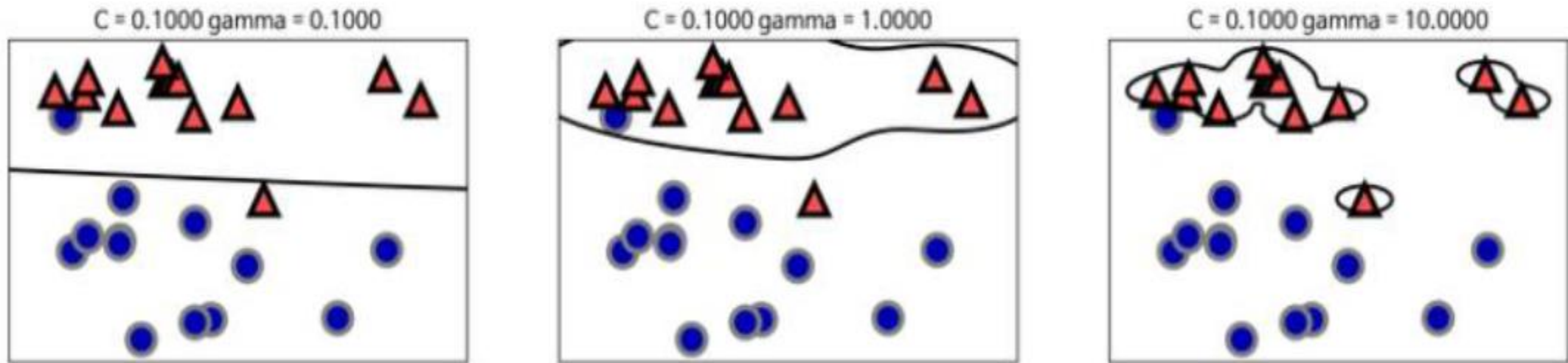
Gaussian Kernel? (RBF Kernel)

Gamma가 클수록 훨씬 인접한것들만 영역으로 본다



Gamma가 내려감

Unit 03 | Non-Linear SVM



Gamma가 커질수록 엄청 인접하지 않으면 엄청 먼 곳으로 인식
-> 원래 차원으로 돌아왔을 때 경계가 아주 촘촘하게 되어있다!
-> Hyperplane이 훨씬더 굴곡지다!

즉 감마가 커질수록! 더 Overfitting될 것이다!!

Contents

Unit 01 | Support Vector Machine이란?

Unit 02 | Soft Margin VS Hard Margin

Unit 03 | Non-Linear SVM

Unit 04 | 마무리

Unit 04 | 마무리

정리!

Hard Margin VS Soft Margin

Hard Margin : 에러를 허용하지 않음 (그닥 현실감각이 없음)

Soft Margin : 에러를 허용함

*에러를 허용하는 방법은 0-1 Loss 와 Hinge Loss

*에러를 줄이는데 초점을 둘 지 마진을 줄이는데 초점을 둘 지 결정하는 하이퍼 파라미터는 C!

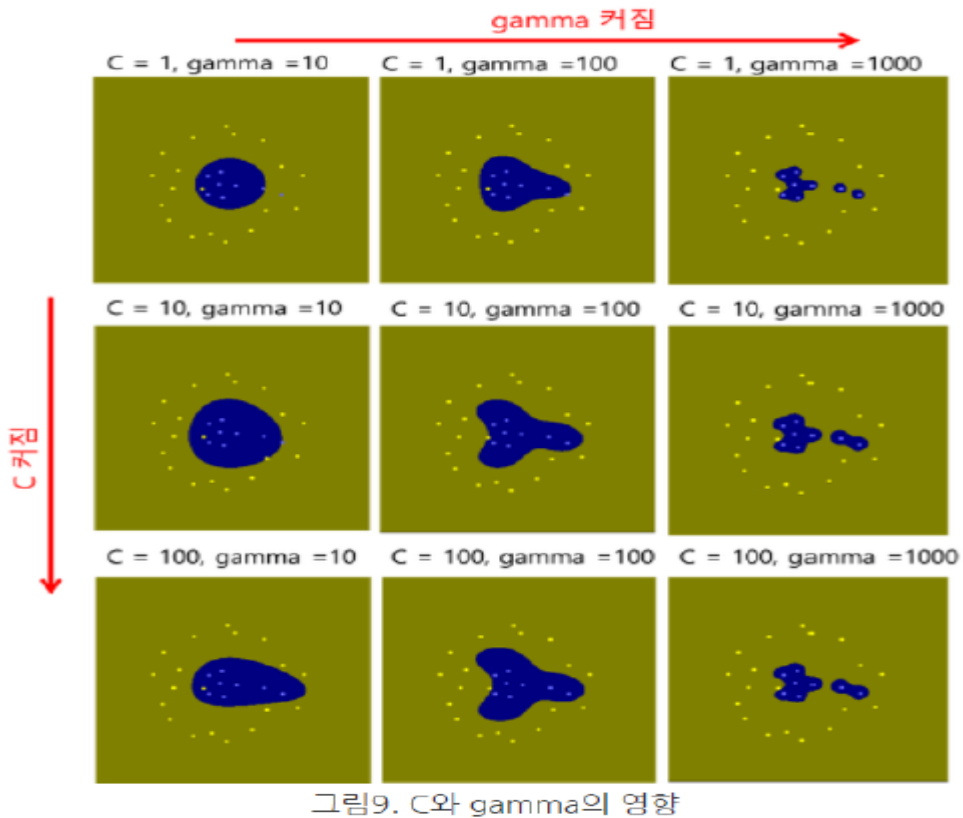
Linear SVM VS Non-linear SVM

Linear SVM : 선형으로 분류함! Feature가 많으면 kernel을 하게 되면 너무 차원이 높아지고 연산량이 많아지므로 오히려 Linear가 나옴

Non-Linear SVM : Feature가 안 많으면 그걸로 분류하기가 쉽지 않아서 차원을 올려줌! 차원을 올려주는 방법에 따라 다양한 parameter가 존재함! 제일 많이 쓰는건 Gaussian Kernel에서 Gamma

Unit 04 | 마무리

Grid Search를 이용해보자

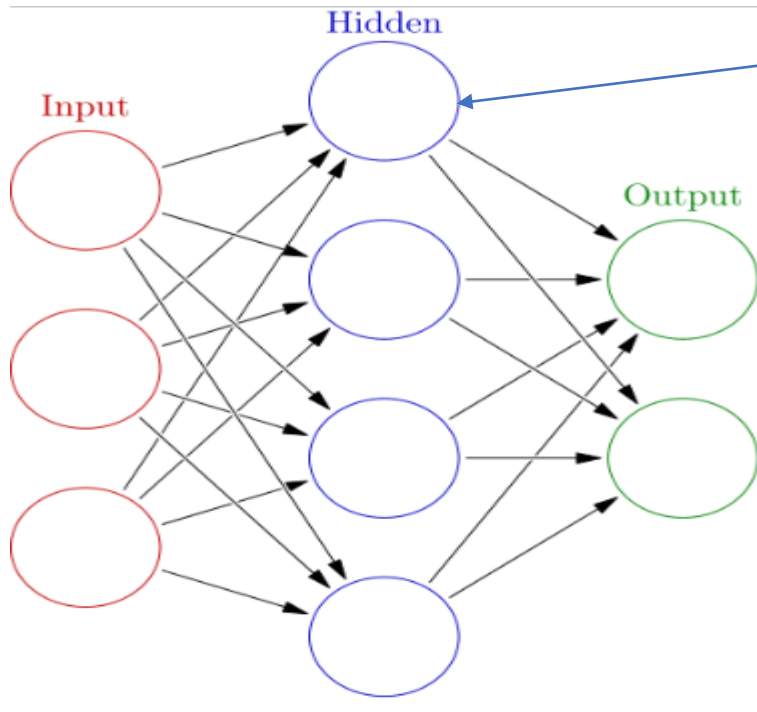


뭐가 좋은지 직관적으로 알기는 어려움!
-> Grid Search를 이용해서 감을 잡아보자

Library를 이용해서 구현가능! 실습에 예시를 넣어 둘게요

Unit 04 | 마무리

SVM과 Neural Network



Neural Network

이렇게 히든레이어를 여러 개로 만들고(차원확장) Activation Function까지 쓰면(Non-linear)기본적으로 SVM kernel이랑 원리가 같음!

대부분 NN이 성능이 더 좋아서... 요즘은 NN이 대세

* SGD Classifier를 이용해서 값을 찾는 방법이 있는데 이는 Neural Network와 학습 방법이 아주 유사하며 사이킷런에서 라이브러리도 제공합니다..

⇒ 그럼에도 불구하고 SVM이 조금 더 직관적인 해석이 가능하며 좀 더 알고리즘 최적화도 효율적이라 쓰일 '수' 있음

⇒ 그리고 이렇게 분류를 하는 방법에 대한 고민이 NN을 학습하는데 좋은 자양분이 될 것(오글...)

⇒ 그러니까 과제 대충하지마세요

Unit 04 | 마무리

Multi Class SVM

One VS one

클래스가 N개 있으면 모든 Class에 대해 1 : 1로 binary분류를 한 다음 제일 많이 승리한 것에 투표로 결정!
N개의 클래스에 대해 서로서로 다 Classifier를 가지고 있어야 하므로 $N(N-1)/2$ 개의 Classifier가 필요!

One vs rest

클래스가 N개 있으면 모든 Class에 대해 1 : N-1로 binary 분류(이 클래스 인지 아닌지)를 한 다음 투표로 결정!
총 N 개의 Classifier가 필요하다

Q & A

들어주셔서 감사합니다.

과제 설명

Assignment 1

1. Assignment1은 위에서 언급 되었던 Multiclass SVM을 직접 구현하시는 것입니다. 기본적으로 사이킷 런에 있는 SVM은 멀티클래스 SVM을 지원합니다. 그러나 과제에서는 그것을 쓰면 안됩니다! 아이리스 데이터는 총 세 개의 클래스가 있으므로 이 클래스를 one-hot 인코딩 한 뒤, 각각 binary SVM을 트레이닝하고 이 결과를 조합하여 multiclass SVM을 구현하는 것입니다.
2. 위에서 말했듯 기본적으로 one vs one, one vs rest 방법이 있으며 어떤 것을 구현하든 자유입니다. 만약 투표결과 동점이 나온경우(예를 들어 각각의 SVM의 결과가 A vs B C의 경우 A로 판별, B vs A C의 결과 B로 판별, C vs A B의 경우 C로 판별한 경우 투표를 통해 class를 결정할 수 없음) decision_function을 활용하시거나, 가장 개수가 많은 클래스를 사용하시거나 랜덤으로 하나를 뽑거나 하는 방법 등을 이용해 **동점자인 경우를 판별해주시면** 됩니다. 공식 문서를 보면 사이킷런이 어떤 방법으로 구현했는지가 글로 나와 있으므로 참조하셔도 무관합니다.
3. 과제코드에는 제가 iris 데이터를 로드하고 iris 데이터를 one hot인코딩 한 부분까지 구현해 놓았습니다. 또한 decision function을 호출해서 사용하는 예시도 하나 넣어 놓았으니 참고하시면 됩니다. 개인적으로 one vs rest가 더 구현하기 쉬울것으로 생각되며, 모르는 부분은 언제든지 질문해주세요! 생각보다 코드가 길지 않고 어렵지 않습니다.

과제 설명

Assignment 2

1. Assignment2는 제가 anomaly-detection 데이터셋(캐글에 올라와있음)을 드립니다. 이것은 해당 결제가 사기인지 아닌지 판별하는 데이터셋이며, 실습코드를 활용하고, 또 본인이 여태 배운 내용을 활용하여 자유롭게 데이터를 가지고 연습해주시면 됩니다. 정말 시간이 없으시면, 실습코드를 조금만 바꿔서 모델을 트레이닝하고 평가한 결과만 보여주셔도 과제를 돌려보내지는 않겠습니다.
2. 다만 이 데이터셋은 굉장히 imbalance한 데이터 셋입니다. 실제 사기를 치는 사례가 많지 않으므로 사기인 경우가 전체에 0.17프로밖에 되지 않습니다. 따라서, 그냥 데이터를 트레이닝 시키면 무조건 사기가 아니라고 판별해버릴 가능성이 높습니다. 또한, 그대로 트레이닝을 돌리게 되면 엄청나게 많은 데이터 양 때문에 트레이닝조차 힘들 것입니다. 그런데 실제로 정확도를 높이면서 트레이닝을 할 수 있는 방법이 있으니 고민해보세요

(아마 이미 데이터 분석을 해보신분들은 imbalance라는 단어를 보자마자 답을 떠올리셨을테지만, 처음 해보시는 분들은 고민을 해보신 뒤, 멘토분들 아무나 붙잡고 물어보시면 답을 말해줄 것입니다)

읽을거리

다른 방법과 SVM 의 비교!

	장점	단점	<div>SVM</div> <ul style="list-style-type: none"> - 결과 해석 용이 - 높은 성과 - 적은 자료 만으로도 신속한 분별학습 수행 가능
연관성 규칙	미시/거시적 관점의 데이터 분석이 가능.	생성되는 많은 양의 규칙 대부분이 실제 활용 가치가 적음.	
의사결정 나무	적용 결과 및 분석 과정에 대한 명확하고 쉬운 이해.	새로운 자료의 예측에는 불안정.	
신경망	자료에 대한 통계적 분석 없이 수행 가능하고, 실측 데이터를 처리 능력이 우수함.	모델 구축에 많은 시간이 소요되고, 모델에 대한 설명력이 부족함. 학습 진행 과정에서의 과적합화.	
베이지안 망	변수들간의 상관관계를 쉽게 이해 할 수 있으며, 노드와 화살표를 이용하여 결과의 이해가 쉬움.	실질적으로 관심 있는 속성들과 연관성이 없는 정보들로 인하여 데이터 자체의 축소 과정이 필요로 되어짐.	
로지스틱 회귀분석	통계적 기법에 근간한 모델으로 각 변수의 영향력을 정확히 설명 가능.	예측 성과가 높지 않음.	

Reference

투빅스 11기 심은선님 강의자료
투빅스 10기 박규리님 강의자료
투빅스 8기 김은서님 강의자료
투빅스 5기 최도현님 강의자료

위키피디아 Support Vector Machine :
https://en.wikipedia.org/wiki/Support_vector_machine

라그랑지안 승수법 위키피디아:
https://en.wikipedia.org/wiki/Lagrange_multiplier

랏츠고 블로그 :
<https://ratsgo.github.io/machine%20learning/2017/05/23/SVM/>
<https://ratsgo.github.io/machine%20learning/2017/05/29/SVM2/>
<https://ratsgo.github.io/machine%20learning/2017/05/30/SVM3/>

*투빅스 3기 Jason Park 님 강의자료에 위에서 말한 생략된 내용이 구체적으로 나와있으니 해당자료 3번 항목을 참고해주시면 됩니다. (파일 같이첨부해드릴게요!)