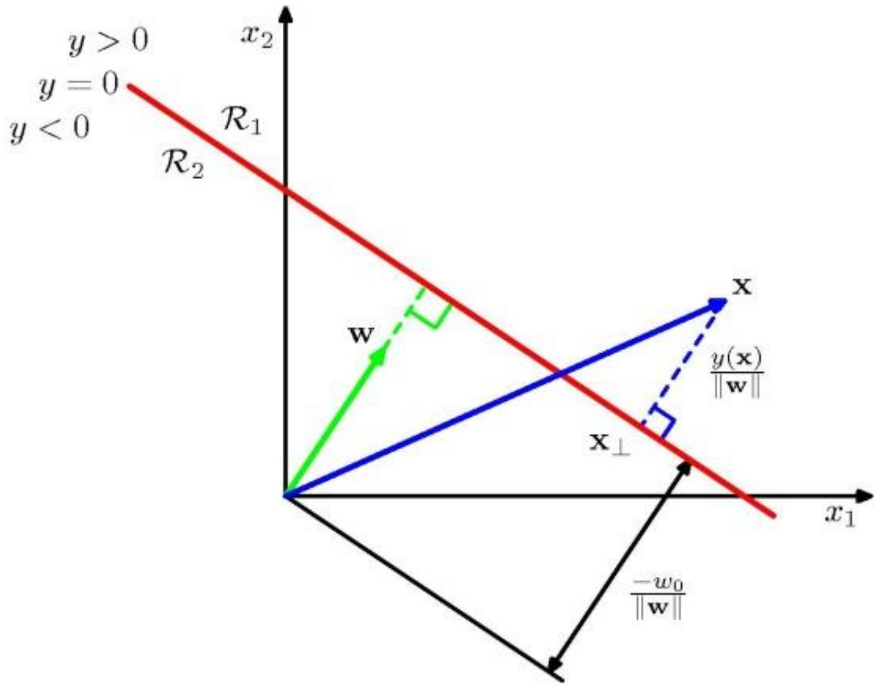
Support Vector Machine

Easy Tutorial(I wish)

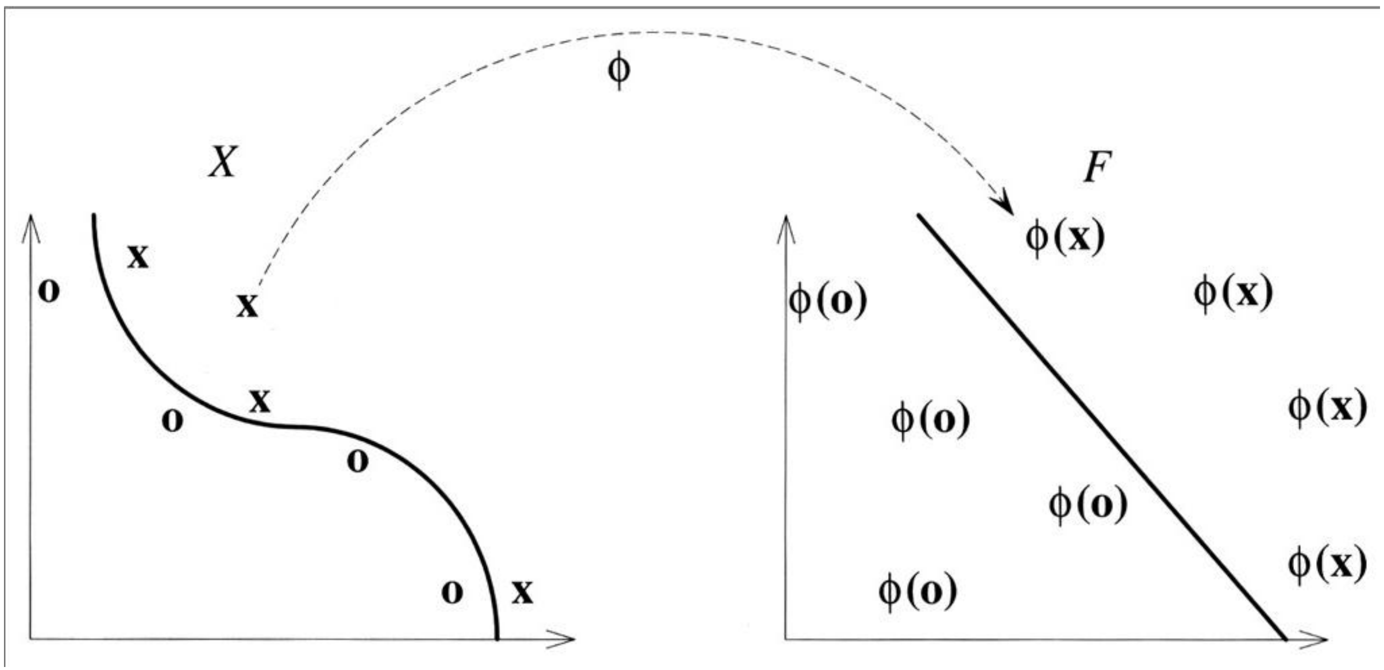


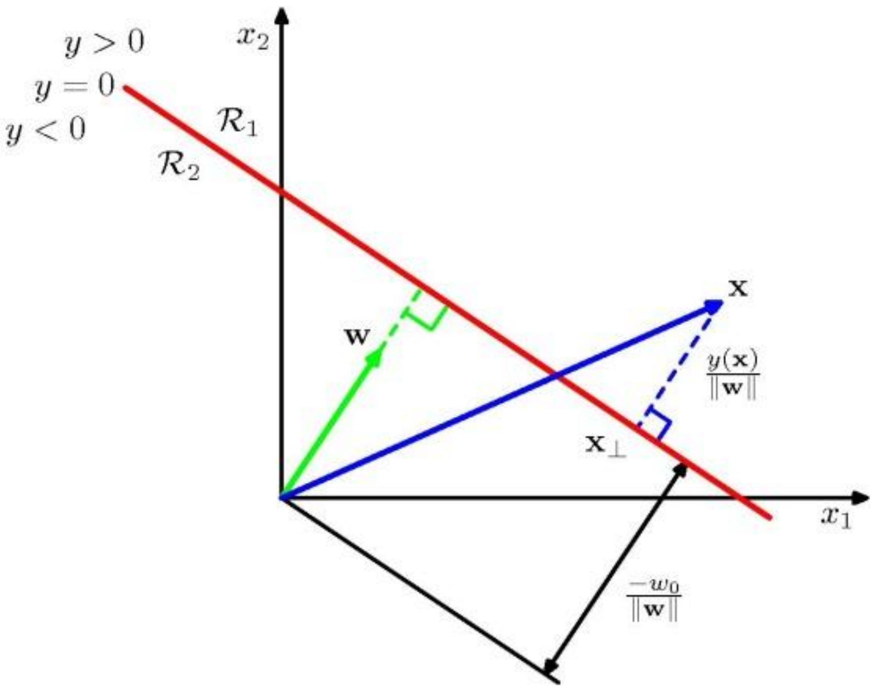
$$(1)\ y_t(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

문제를 간단히 하기 위해서 몇 가지 가정

1, 어떤 함수를 통해feature space으로의 mapping 후 linear sepeable

2, lable은 , 1과 -1 두가지





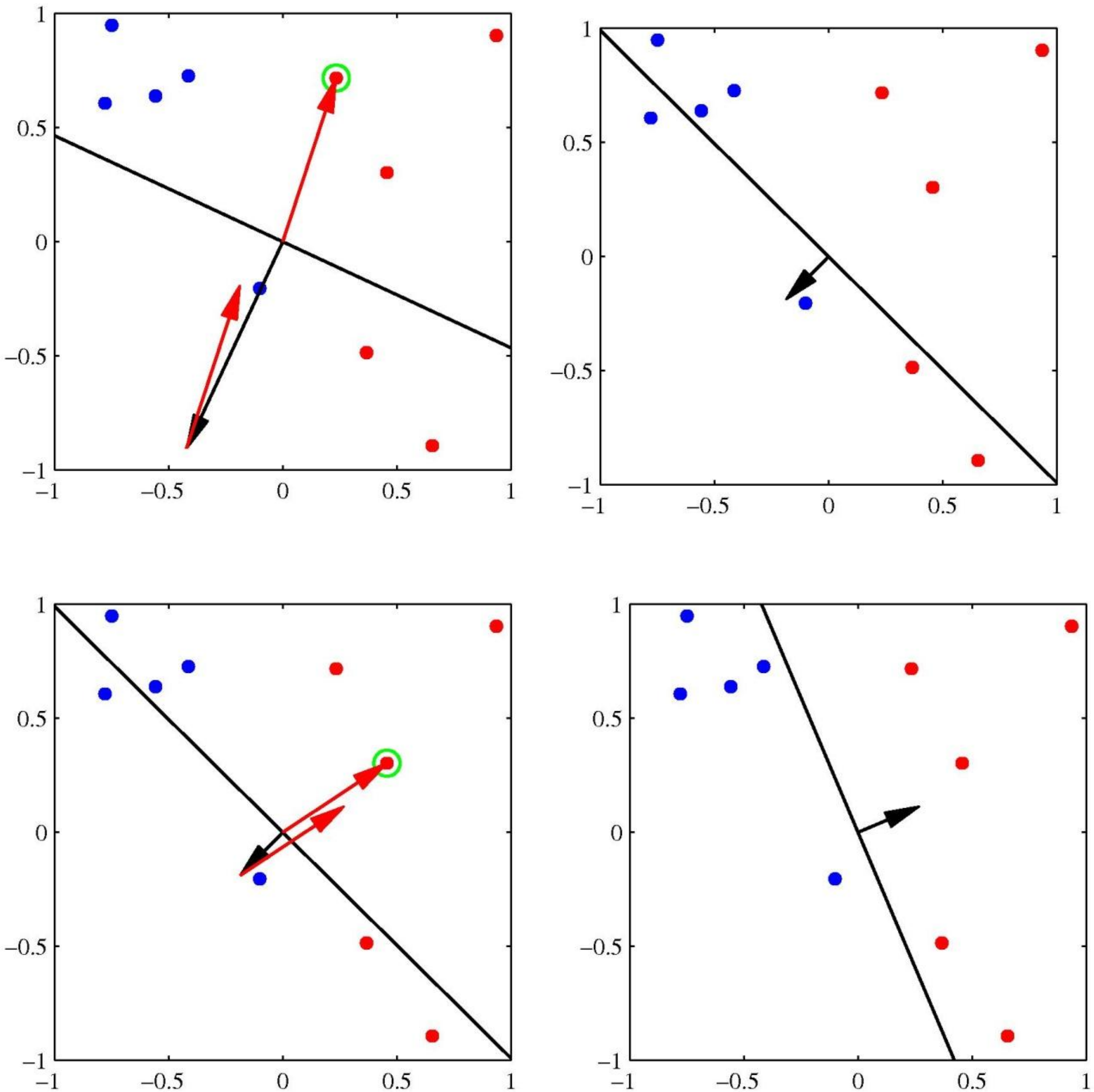
$$(1)\ y_t(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

(2)초평면과 *feature vector* 와의 거리 $= \dfrac{|y(\mathbf{x})|}{\|\mathbf{w}\|} = \dfrac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} (\because \textit{we assume linear seperable})$

$$= \frac{t_n(\mathbf{w}^T \phi(\mathbf{x}) + b)}{\|\mathbf{w}\|} (\because (1))$$

Perceptron algorithm

$$\mathbf{w}' \leftarrow \mathbf{w} + y_n \mathbf{x}_n \qquad if \;\; t_n \neq sign\{y(\mathbf{x}_n)\}$$



오분류(mistake)된 표본 (sample)들을 가지고 초평면 (hyperplane)의 법선(normal) 벡터를 조정해 나간다.

그래서 perceptron 알고리즘을 mistake driven 알고리즘이라고 한다.

Perceptron, convergence

$$\theta^{(k)} : the\ paramater\ vector\ after\ k\ update$$

- Claim : perceptron algorithm indeed convergences in a finite of updates.
- Assume that

$$\| \mathbf{x}_t \| \le R \ \ for\ \ all\ \ t\ \ and\ \ some\ \ finite\ \ R$$

$$\exists \gamma > 0 \ \ s.t \ \ y_t (\theta^*)^T \mathbf{x}_t > \gamma$$

Convergence proof

$$\theta' \leftarrow \theta + y_t \mathbf{x}_t \qquad if \;\; y_t \neq f(\mathbf{x}_t;\theta)$$

$$(\theta^*)^T \theta^{(k)} = (\theta^*)^T \theta^{(k-1)} + y_t (\theta^*)^T \mathbf{x}_t$$
$$\geq (\theta^*)^T \theta^{(k-1)} + \gamma$$
$$= (\theta^*)^T (\theta^{(k-2)} + y_t \mathbf{x}_t) + \gamma$$
$$= (\theta^*)^T \theta^{(k-2)} + y_t (\theta^*)^T \mathbf{x}_t + \gamma$$
$$\geq (\theta^*)^T \theta^{(k-2)} + 2\gamma$$
$$\ldots$$
$$\geq k\gamma$$

Assume that

$$\| \mathbf{x}_t \| \leq R \;\; for \;\; all \;\; t \;\; and \;\; some \;\; finite \;\; R$$

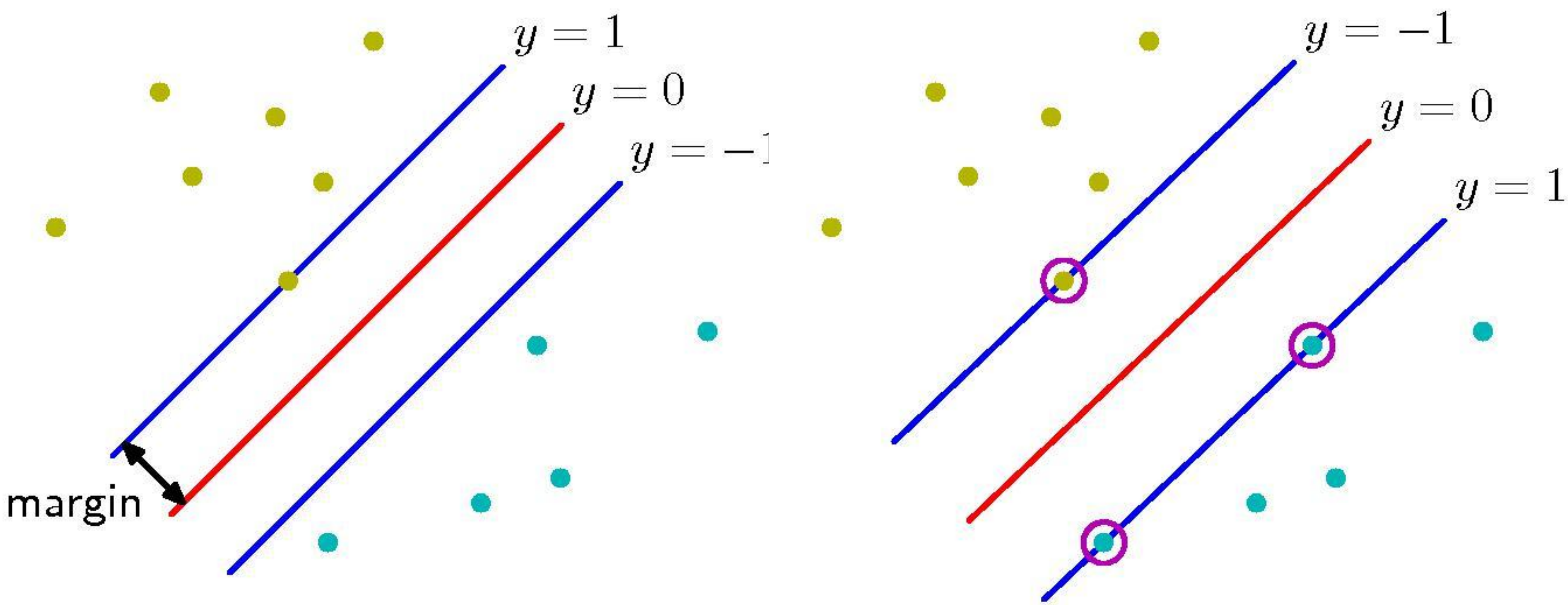
$$\exists \gamma > 0 \;\; s.t \;\; y_t (\theta^*)^T \mathbf{x}_t > \gamma$$

$$\| \theta^{(k)} \|^2 = \| \theta^{(k-1)} + y_t \mathbf{x}_t \|^2$$
$$= \| \theta^{(k-1)} \|^2 + 2 y_t (\theta^{(k-1)})^T \mathbf{x}_t + \| \mathbf{x}_t \|^2$$
$$\leq \| \theta^{(k-1)} \|^2 + \| \mathbf{x}_t \|^2$$
$$\leq \| \theta^{(k-1)} \|^2 + R^2$$
$$\ldots$$
$$\leq kR^2$$

$$1 \geq \cos(\theta^*, \theta^{(k)}) = \frac{(\theta^*)^T \theta^{(k)}}{\| \theta^{(k)} \| \| \theta^* \|} \geq \frac{k\gamma}{\| \theta^{(k)} \| \| \theta^* \|} \geq \frac{k\gamma}{\sqrt{kR^2} \| \theta^* \|}$$

$$\therefore 1 \geq \frac{k\gamma}{\sqrt{kR^2} \| \theta^* \|} \quad or \quad k \leq \frac{R^2 \| \theta^* \|^2}{\gamma^2}$$

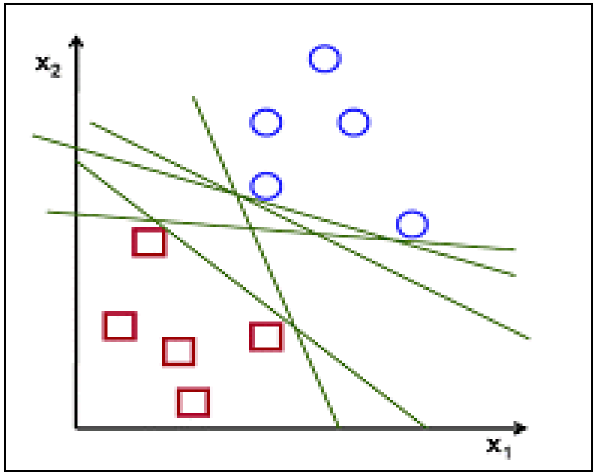
Support Vector, and Margin의 정의

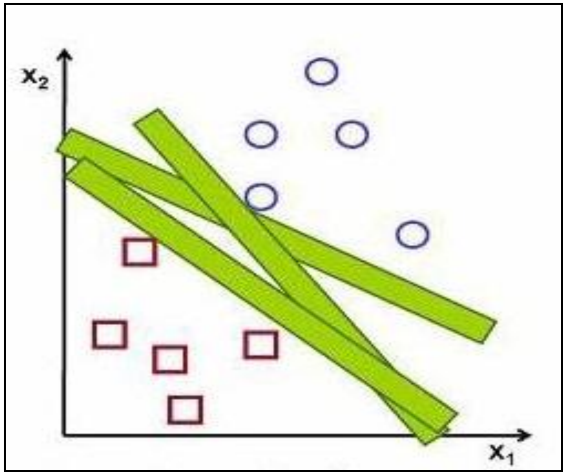


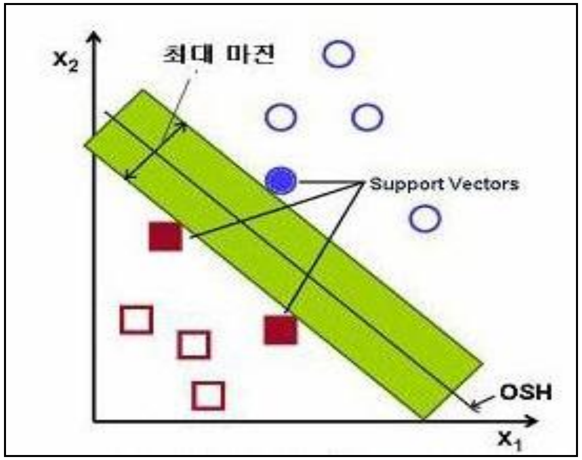
빨간색 동그라미가 Support vector이고 support vector와 초평면 (hyperplane)과의 수직거리를 margin이라고 한다.

SVM의 동기(motivation)?, 직관(intuition)?

수많은 초평면들이 있다.
어느 초평면이 가장 일반화 오류(generalization error)가 적을까?
Maximal margin classifier!!



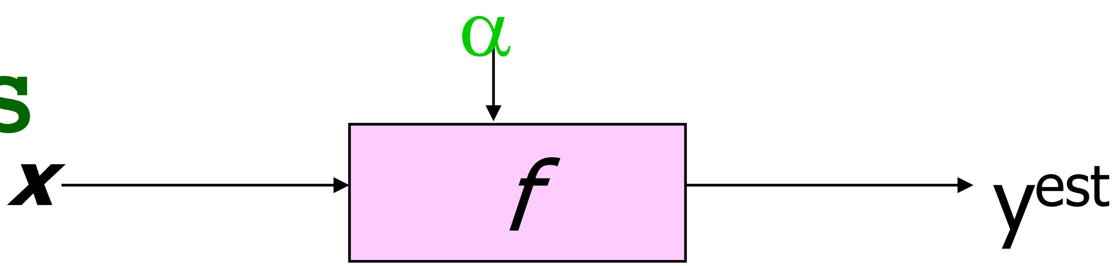




$$(3)\ \arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\mathbf{w}\|}\min_{n}\left[t_n(\mathbf{w}^T\phi(\mathbf{x})+b)\right]\right\}$$

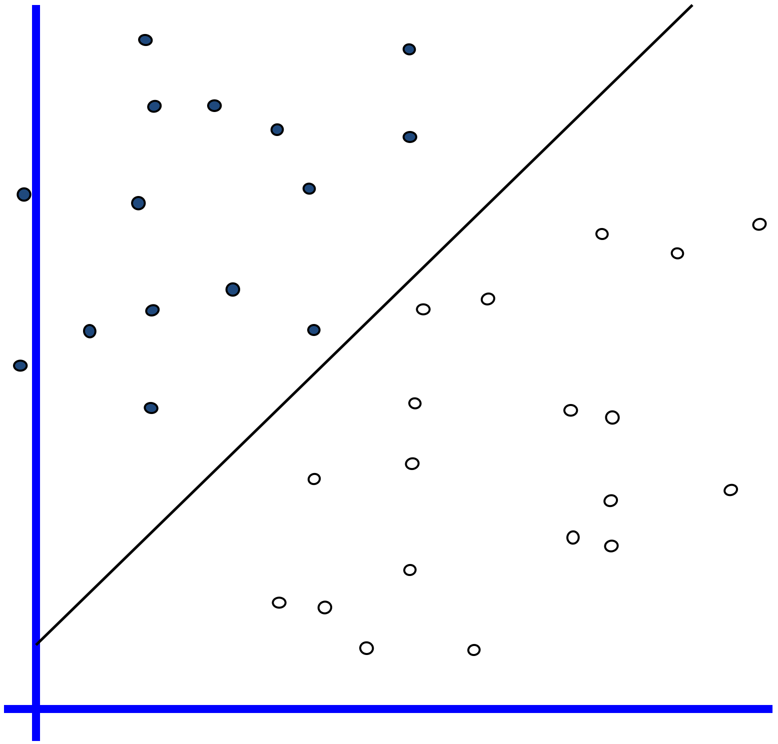
초평면에가장 가까운 샘플의마진을 최대로 하는 파라메타 구하기

Linear Classifiers



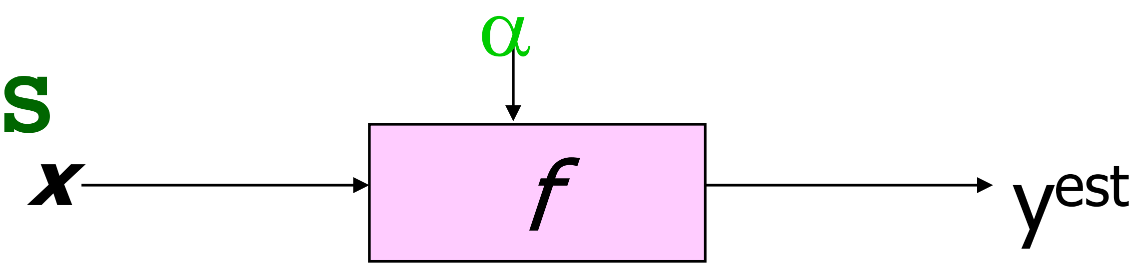
$f(\boldsymbol{x},\boldsymbol{w},b) = sign(\boldsymbol{w}.\ \boldsymbol{x} - b)$

- denotes +1
- denotes -1



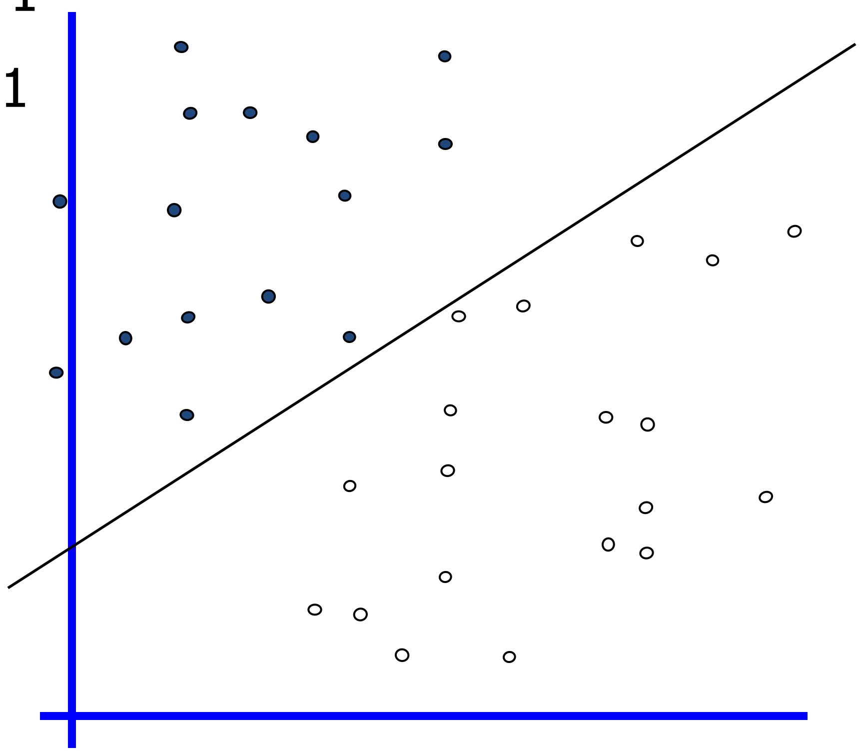
How would you classify this data?

Linear Classifiers



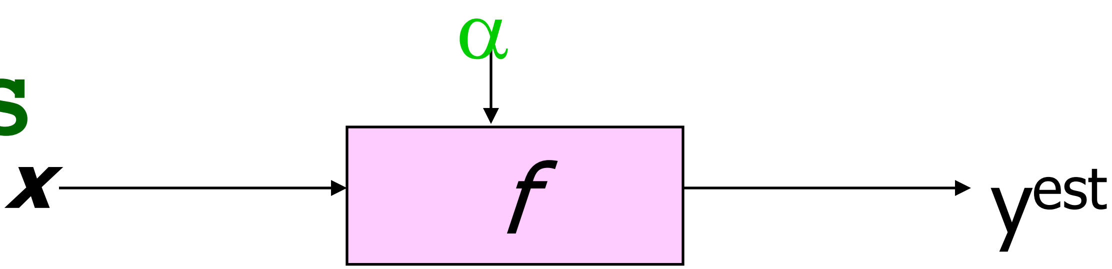
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- denotes +1
- denotes -1

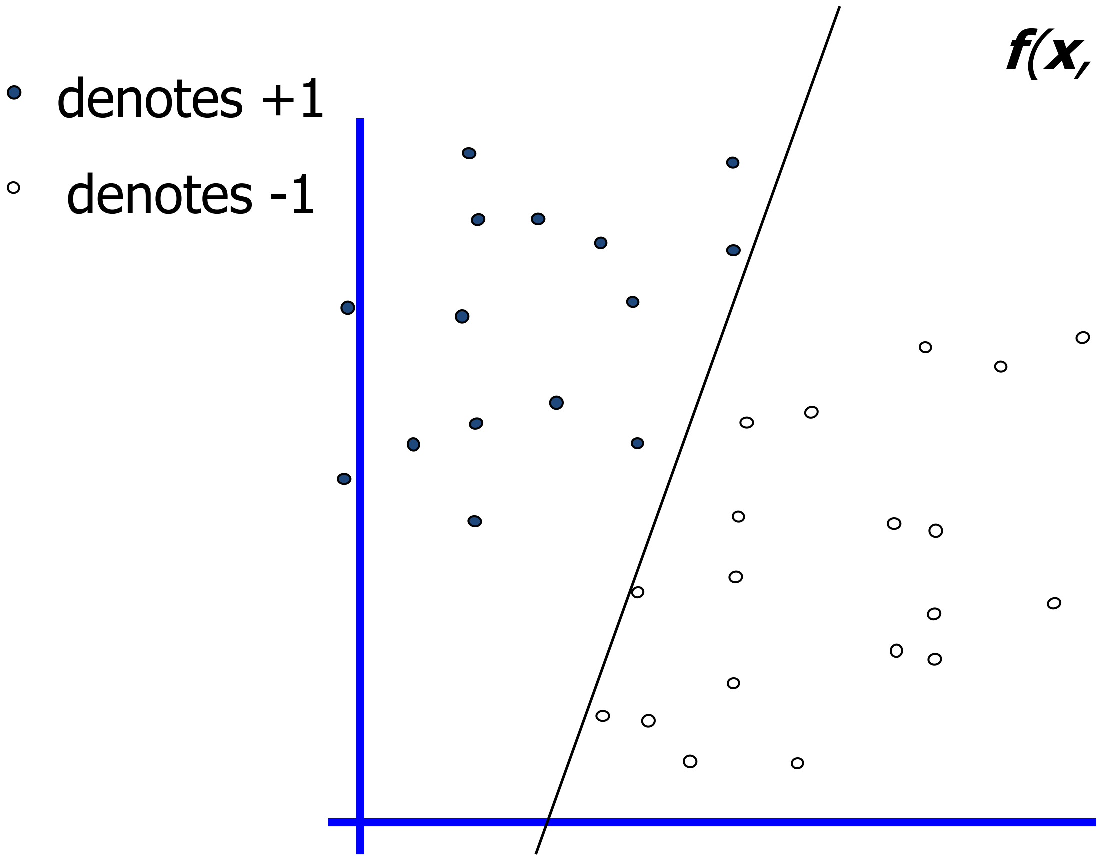


How would you classify this data?

Linear Classifiers

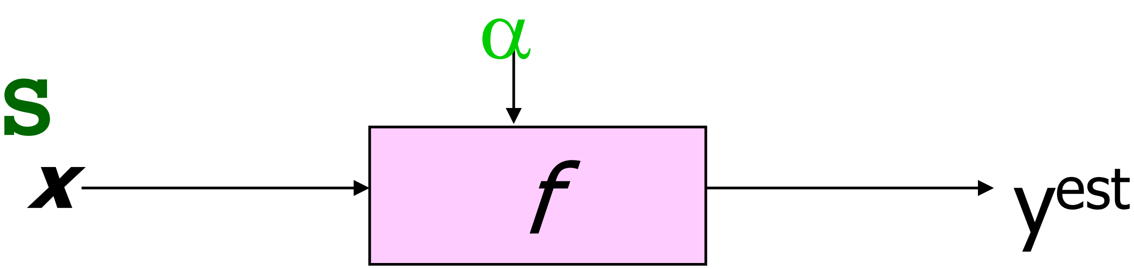


$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$



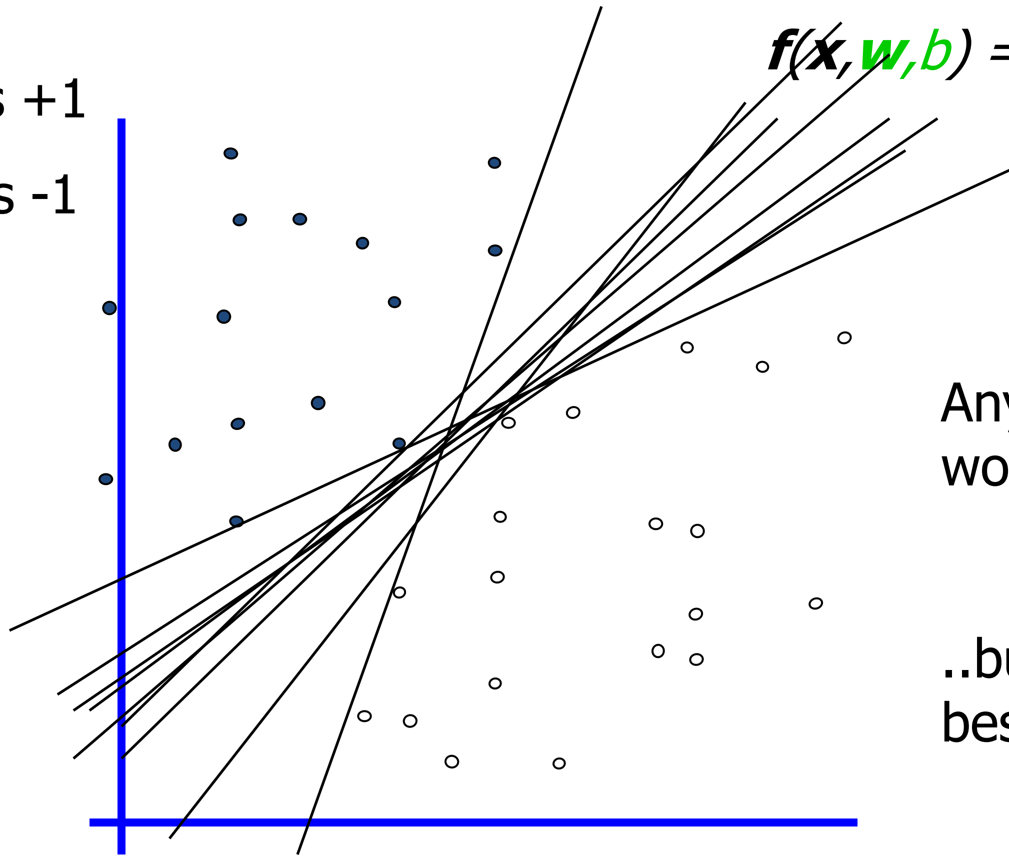
How would you classify this data?

Linear Classifiers



$\boldsymbol{f}(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

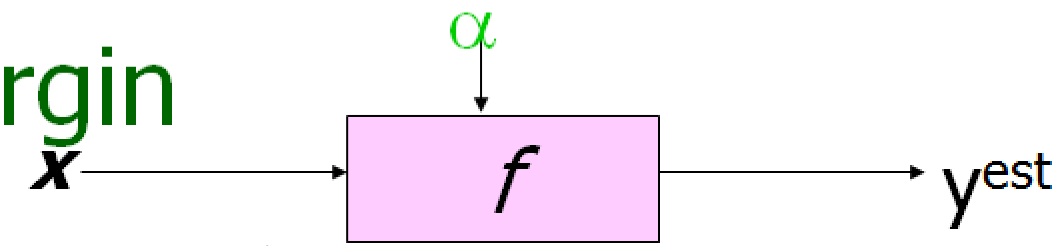
- denotes +1
- denotes -1



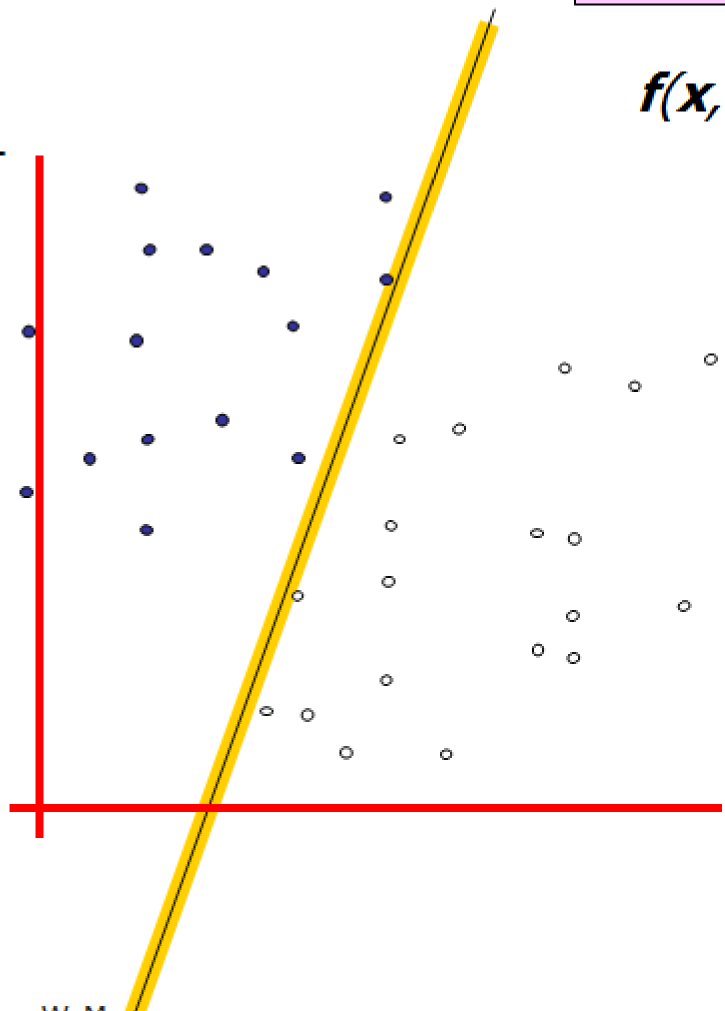
Any of these would be fine..

..but which is best?

Classifier Margin



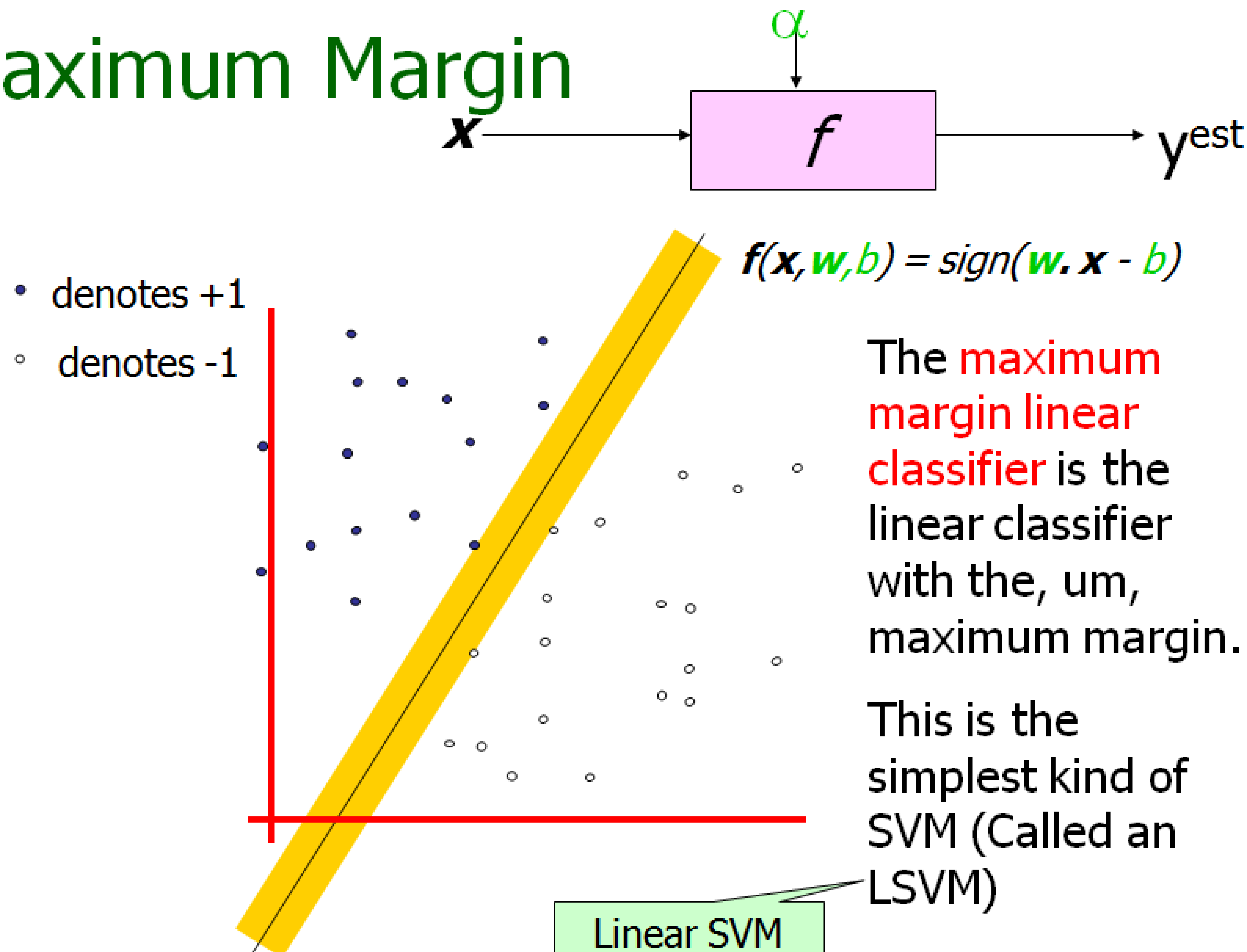
- denotes +1
- ◦ denotes -1



$f(x, w, b) = sign(w \cdot x - b)$

Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



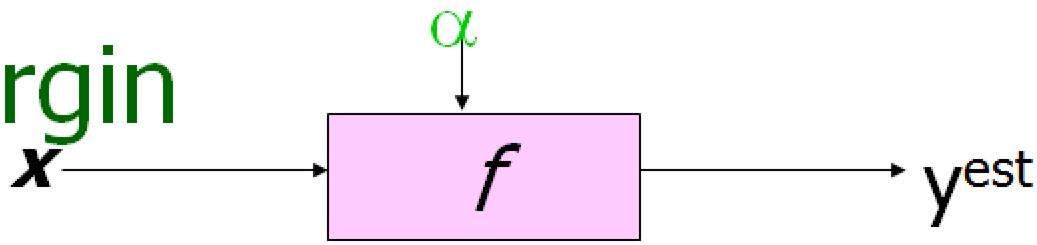
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\,\mathbf{x} - b)$

The maximum margin linear classifier is the linear classifier with the, um, maximum margin.

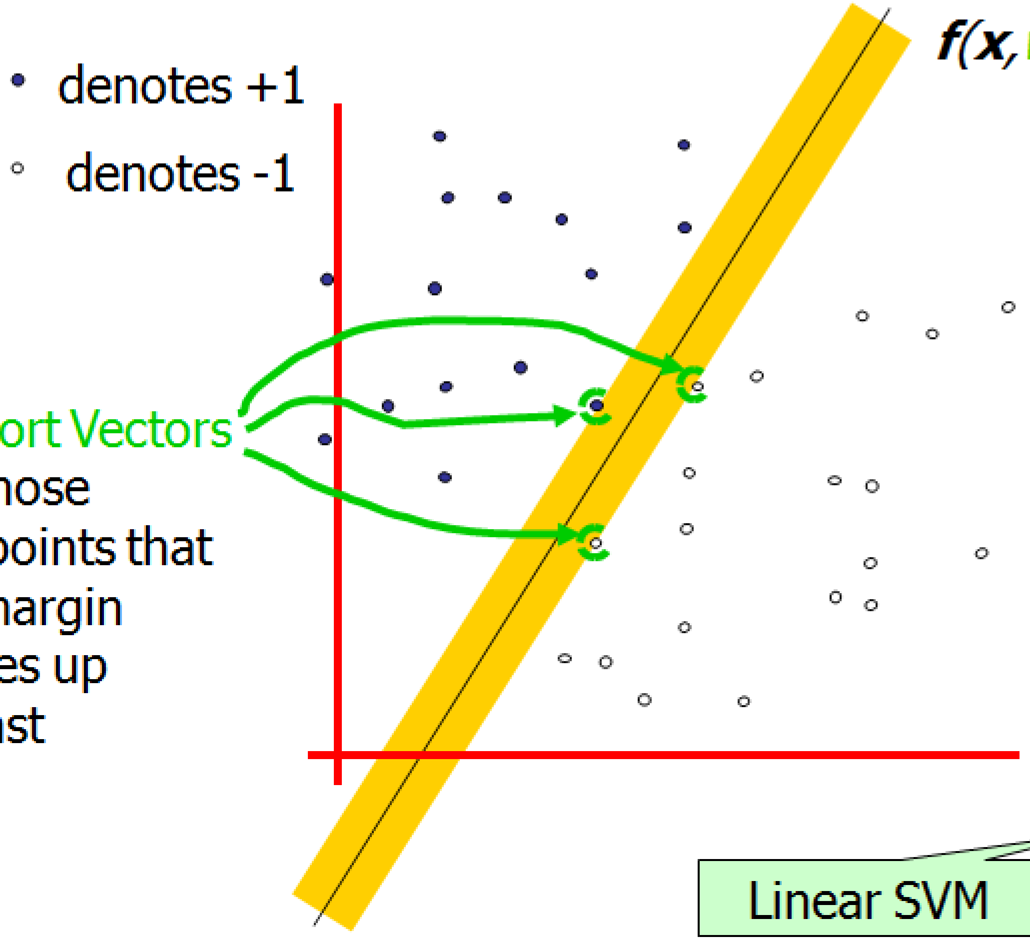
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

Maximum Margin



$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$



Support Vectors are those datapoints that the margin pushes up against

The maximum margin linear classifier is the linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Linear SVM

SVM 아이디어의 수식화를 위한 전처리?

$.\frac{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)}{\|\mathbf{w}\|};\mathbf{w}\to k\mathbf{w}, b\to kb$ 로_리스케일링해도_마진은_변함없다

$$.\frac{t_n(k\mathbf{w}^T\phi(\mathbf{x})+kb)}{\|k\mathbf{w}\|}=\frac{t_n k(\mathbf{w}^T\phi(\mathbf{x})+b)}{k\|\mathbf{w}\|}=\frac{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)}{\|\mathbf{w}\|}$$

$(4)\ t_n(\mathbf{w}^T\phi(\mathbf{x})+b)=1;$ 표면에서 가장가까운점이만족하는수식이되도록과 b를조정함

$(5)\ t_n(\mathbf{w}^T\phi(\mathbf{x})+b)\geq 1;$ 모든점이만족하는수식, *linear seperable*임을뜻함

(2)초평면과 *feature vector* 와의 거리 $=\frac{|y(\mathbf{x})|}{\|\mathbf{w}\|}=\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|}(\because \textit{we assume linear seperable})$

$$=\frac{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)}{\|\mathbf{w}\|}(\because (1))=\frac{1}{\|\mathbf{w}\|}=margin$$

$\|\mathbf{w}\|^{-1};$마진_최대화

$\Leftrightarrow \|\mathbf{w}\|^2$ 을_최소화

$\Leftrightarrow \frac{1}{2}\|\mathbf{w}\|^2$ 을최소화; $\frac{1}{2}$ 은나중의편의를위해서(미분)

$(6)\ \arg\min_{\mathbf{w},b}\frac{1}{2}\|\mathbf{w}\|^2;$ 마진의역수를최소화하는파라메터구하기

SVM 아이디어의 수식화

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \ \ subject\ to\ \ t_n(\mathbf{w}^T\phi(\mathbf{x})+b)\geq 1$$

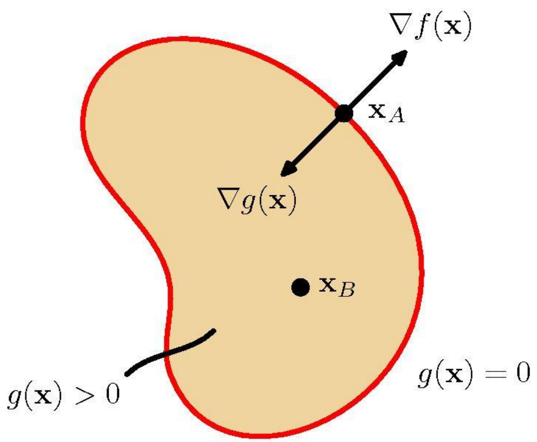
여기까지가 SVM 수식완성!!!!!!
이제 이 이 수식을 풀기만 하면 됨
최적화식은 우리가 고등학교 때 보았던 linear programming과 비슷함
*Quadratic programming*이라고 한다.

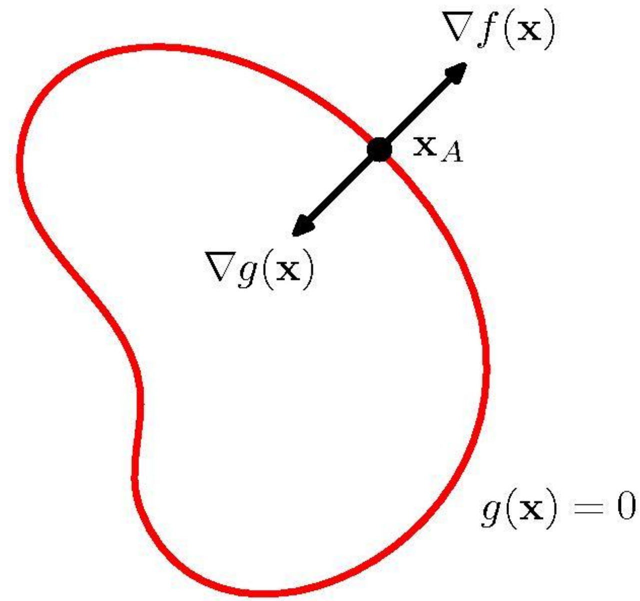
이 *Quadratic programming* 문제을 dual form으로 전환하기 풀기 위해서 Lagrange multipliers 개념을 도입한다.

Dual form으로 전환 이유
-> 커널 트릭을 위해,

위 수식은 convex 최적화 문제라는 것이다.
Convex , non convex 간단히 설명
->convex하면 좋은게 "미분해서 0" 기법으로 grobal minimum을 구할 수 있다는 것.

Lagrange multiplier





$g(\mathbf{x}+\boldsymbol{\varepsilon}) \cong g(\mathbf{x}) + \boldsymbol{\varepsilon}^T \nabla g(\mathbf{x})$;테일러시리즈

$g(x) = g(\mathbf{x}+\boldsymbol{\varepsilon})$; $x, \mathbf{x}+\boldsymbol{\varepsilon}$모두$g(\mathbf{x})$위에있다고가정

$\Rightarrow \boldsymbol{\varepsilon}^T \nabla g(\mathbf{x}) \cong 0$

if lim$\|\varepsilon\| \to 0$ then $\boldsymbol{\varepsilon}^T \nabla g(\mathbf{x}) = 0 \to \boldsymbol{\varepsilon}$는$g(\mathbf{x})$에평행하기때문에$\nabla g(\mathbf{x})$는$g(\mathbf{x})$에수직이다.

$\nabla f \perp g(\mathbf{x}), \nabla g \perp g(\mathbf{x}) \to \nabla f + \lambda \nabla g(\mathbf{x}) = 0$, *where* $\lambda \neq 0$

$L(\mathbf{x}, \lambda) = f + \lambda g(\mathbf{x})$; *KKT*조건만족

$ex)L(\mathbf{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$

$-2x_1 + \lambda = 0$

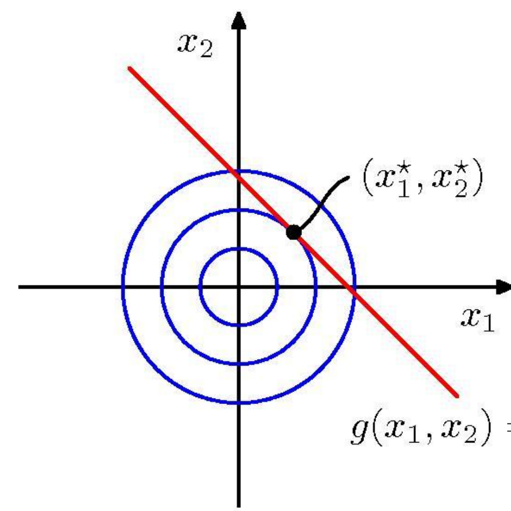
$-2x_1 + \lambda = 0$

$x_1 + x_2 - 1 = 0$

$\to x_1 = x_2 = 1/2$;다른방법으로풀기.

이 예제 풀기

두가지 방법



SVM 수식의 dual form으로 전환(SVM의 마술을 위해) 이부분 이해를 위해 천천히

$$(7)\ L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)-1\};$$ 라그랑쥬 최적화 수식

$$\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) = 0 \Leftrightarrow (8)\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial b} = \sum_{n=1}^{N} a_n t_n = 0$$

$$(8)\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$(9)0 = \sum_{n=1}^{N} a_n t_n;$$ (7)식을 미분해서 정리하면 나오는 수식

(8), (9)를 (7)에 대입해서 정리하면

$$L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x}) - \sum_{n=1}^{N} a_n t_n \mathbf{w}^T\phi(\mathbf{x}) + \sum_{n=1}^{N} a_n t_n b + \sum_{n=1}^{N} a_n$$

$$\Leftrightarrow \tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x_n},\mathbf{x_m})$$

$$(11)a_n \geq 0;$$ (10)번식의 제약식

$$(12)\sum_{n=1}^{N} a_n t_n = 0;$$ (9)번식의 제약식

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \;\; subject\;\; to \;\; t_n(\mathbf{w}^T\phi(\mathbf{x})+b) \geq 1$$

$$(7)\; L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)-1\}$$;라그랑쥬최적화수식

$$\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n) = 0 \Leftrightarrow (8)\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial b} = \sum_{n=1}^{N} a_n t_n = 0$$

$$(8)\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$(9) 0 = \sum_{n=1}^{N} a_n t_n$$;(7)식을 미분해서 정리하면 나오는 수식

$$(7)\ L(\mathbf{w},b,\mathbf{a})=\frac{1}{2}\|\mathbf{w}\|^2-\sum_{n=1}^{N}a_n\{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)-1\}$$;라그랑쥬 최적화 수식

$$(8)\mathbf{w}=\sum_{n=1}^{N}a_nt_n\phi(\mathbf{x}_n)$$

$$(9)0=\sum_{n=1}^{N}a_nt_n$$;(7)식을 미분해서정리하면나오는 수식

(8),(9)를(7)에 대입해서정리하면

$$L(\mathbf{w},b,\mathbf{a})=\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_na_mt_nt_mk(\mathbf{x}_n,\mathbf{x}_m)-\sum_{n=1}^{N}a_nt_n\mathbf{w}^T\phi(\mathbf{x})+\sum_{n=1}^{N}a_nt_nb+\sum_{n=1}^{N}a_n$$

$$\Leftrightarrow \tilde{L}(\mathbf{a})=\sum_{n=1}^{N}a_n-\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_na_mt_nt_m\phi(\mathbf{x})^{\mathbf{T}}\phi(\mathbf{x})=\sum_{n=1}^{N}a_n-\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_na_mt_nt_mk(\mathbf{x_n},\mathbf{x_m})$$

$$(11)a_n\geq 0;$$(10)번식의 제약식

$$(12)\sum_{n=1}^{N}a_nt_n=0;$$(10)번식의 제약식

SVM의 마술

이부분 이해를 위해 천천히
한번 읽어보라고 한다

(7) $L(\mathbf{w},b,\mathbf{a})=\frac{1}{2}\|\mathbf{w}\|^2-\sum_{n=1}^{N}a_n\{t_n(\mathbf{w}^T\phi(\mathbf{x})+b)-1\}$;라그랑쥬 최적화 수식

$\rightarrow$ 이 QP 최적화 문제를 푸는데 필요한 계산 복잡도 $O(M^3)$, M은 $feature$의 차원

(10) $\widetilde{L}(\mathbf{a})=\sum_{n=1}^{N}a_n-\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_n a_m t_n t_m k(\mathbf{x}_n,\mathbf{x}_m)$;(8),(9)식을이용해서얻은(7)번식의_듀얼폼

$\rightarrow$ 이 QP 최적화 문제를 푸는데 필요한 계산 복잡도 $O(N^3)$, N은 $feature$의 갯수

보통 $M \le N$이기 때문에 듀얼폼으로의 변환은 안 좋아보인다.

하지만 $M \rightarrow \infty$이라면?즉 무한차원 매핑을 할 수 있다.그러나 커널 함수(k)를 통해 적은 계산으로 그 일을 할 수 있다.이것을 커널 트릭이라고 한다.

$\rightarrow$ $e.g)\phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x})=3\mathbf{x}10000000\mathbf{x}10000000\mathbf{x}3=3\mathbf{x}3!!!!!!!;$ 차원은없어진다.

SVM의 예측기

$$(1)\ y(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b, (8)\ \mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$(13)\ y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b;$$

계산 light

*SVM*에서 *learning*이란 결국 a_n과 b를 학습하는 것이다.

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x_n}, \mathbf{x_m})$$

$$subject\ \ to\ \ a_n \geq 0\ , \sum_{n=1}^{N} a_n t_n = 0;$$

a_n은 위의 제약식을 *QP*로 풀면된다.

*QP*가 *SVM* 계산중 가장 무거운 부분이다. 그래서 *QP*를 최적화하려는 연구도있다. 그중 하나가 *SMO*(sequential minimal optimization)

MATLAB '*quadprog*' 함수 제공

$$(7)\ L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n y(\mathbf{x}) - 1\}$$;라그랑쥬 최적화 수식

위의 수식은 다음과 같은 조건을 만족한다 $(KKT\ \ condition)$

$$(14)\ a_n \geq 0$$

$$(15) t_n y(\mathbf{x}_n) - 1 \geq 0$$

$$(16) a_n \{t_n y(\mathbf{x}_n) - 1\} = 0;$$

$$a_n = 0\ or\ t_n y(\mathbf{x}_n) = 1$$

이 조건의 의미는 $a_n \neq 0$이면 $\mathbf{x}_n$은 SV라는 것이다.

즉, QP를 풀면 SV를 알 수 있다.

이 SV의 집합만 가지고 (13)식으로 분류해 낼 수 있다.

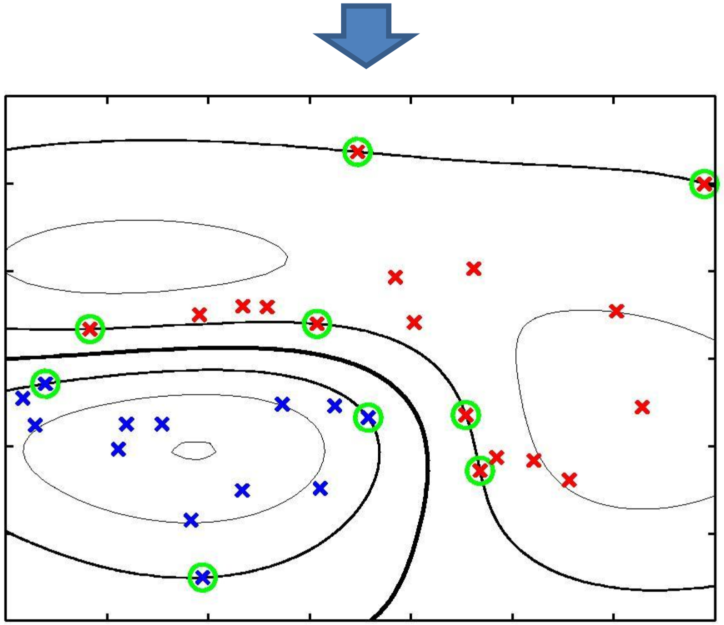
b를 계산

$$(13)\ y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b.$$ 을 $t_n y(\mathbf{x}_n) = 1$ 에 대입하면,

$$(17)\ t_n\left(\sum_{m\in SV}^{N} a_m t_m k(\mathbf{x}, \mathbf{x}_m) + b\right) = 1;$$

$$(18)\ b = \frac{1}{N_S}\left(\sum_{n\in S} t_n - \sum_{m\in S} a_m t_m k(\mathbf{x}, \mathbf{x}_n)\right)$$

Support vector만 가지고 분류평면을 만들어 낸다.

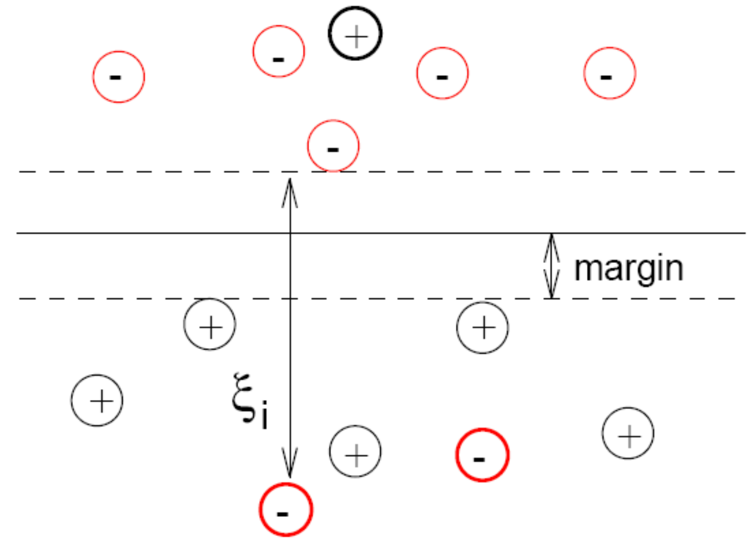


Linear seperable 하지 않다면?

여태까지는 sample들이 고차원 mapping 함수를 통해서 linear seperable 가능하다고 가정하고 이론을 전개시켜 나갔다.

하지만 실제 문제에서는 그러한 경우는 많지 않다.

그래서 데이터들이 linear seperable하지 않을 때를 해결하기 위한 최적화 수식의 변화가 필요하다.



$def.slack : \xi_n = |t_n - y(\mathbf{x}_n)|$; 실제 타겟 값과 예측값과의 차이

(20) $t_n y(\mathbf{x}_n) \geq 1 - \xi_n, n = 1,..., N$; 제약식(5)는이식으로바뀐다.

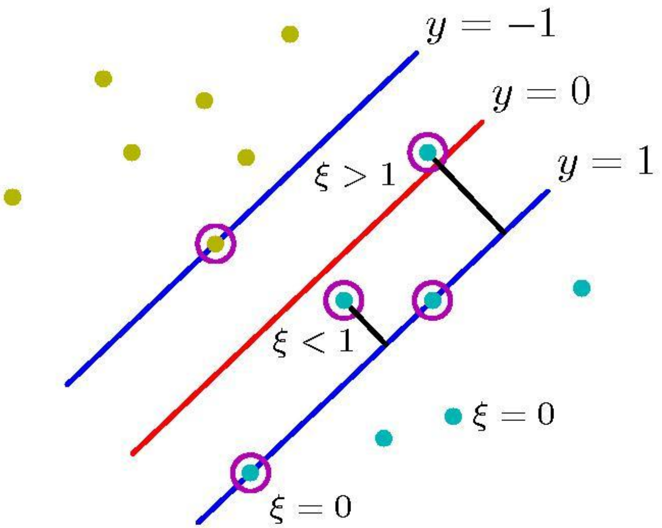
$\xi_n = 0$; 옳게분류, $0 < \xi_n \leq 1$; 마진안에, $\xi_n > 1$; 분류가틀리게됨

(21) $C\sum_{n=1}^{N} \xi_n + \frac{1}{2}\|\mathbf{w}\|^2$; 잘못분류되는샘플에대해서는페널티를가하면서마진의역을최소화

$C \to \infty$이면 (6) $\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$ 이랑 똑같아짐

(22) $L(\mathbf{w}, b, \xi, \mathbf{a}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} a_n \{t_n y(\mathbf{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N} \mu_n \xi_n$

$$\left[\begin{array}{c} (23)\, a_n \geq 0 \\ (24)\, t_n y(\mathbf{x}_n) - 1 + \xi \geq 0 \\ (25)\, a_n(t_n y(\mathbf{x}_n) - 1 + \xi) = 0 \\ (26)\, \mu_n \geq 0 \\ (27)\, \xi_n \geq 0 \\ (28)\, \mu_n \xi_n = 0 \end{array}\right] KKT$$



Linear seperable 하지 않다면?

컨디션

$$\left.\begin{cases} (29)\ \dfrac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow w = \sum_{n=1}^{N} a_n t_n \phi(x_n) \\ (30)\ \dfrac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0 \\ (31)\ \dfrac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n = C - \mu_n \end{cases}\right\}$$ 각 파라메터에 대해서 미분한 값

$$(32) L(\mathbf{w}, b, \xi, \mathbf{a}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N} a_n t_n y(\mathbf{x}_n) + \sum_{n=1}^{N} a_n - \sum_{n=1}^{N} a_n \xi_n - \sum_{n=1}^{N} \mu_n \xi_n$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n t_n y(\mathbf{x}_n) + \sum_{n=1}^{N} \xi_n (C - a_n - \mu_n) + \sum_{n=1}^{N} a_n$$

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

$$(33)(31) a_n = C - \mu_n \ \ and \ \ (23) a_n \geq 0 \ \ and \ \ (26) \mu_n \geq 0 \rightarrow 0 \leq a_n \leq C$$

$$(34) \sum_{n=1}^{N} a_n t_n = 0 (\because (30))$$

27 Polynomial-SVMs

The kernel $K(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} \cdot \boldsymbol{x}')^d$ gives the same result as the explicit mapping + dot product that we described before:

$$\Phi : R^2 \rightarrow R^3 \quad (x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt(2) x_1 x_2, x_2^2)$$

$$\Phi((x_1, x_2) \cdot \Phi((x_1', x_2') = (x_1^2, \sqrt(2) x_1 x_2, x_2^2) \cdot ({x'}_1^2, \sqrt(2) x'_1 x'_2, {x'}_2^2)$$

$$= x_1^2 {x'}_1^2 + 2 x_1 x'_1 x_2 x'_2 + x_2^2 {x'}_2^2$$

is the same as:

$$K(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} \cdot \boldsymbol{x}')^2 = ((x_1, x_2) \cdot (x'_1, x'_2))^2$$

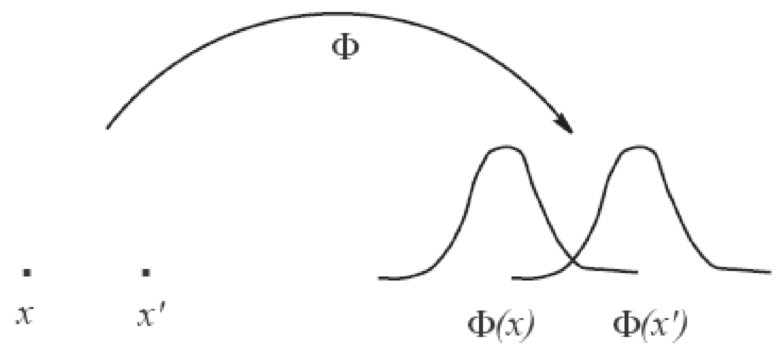
$$= (x_1 x'_1 + x_2 x'_2)^2 = x_1^2 {x'}_1^2 + x_2^2 {x'}_2^2 + 2 x_1 x'_1 x_2 x'_2$$

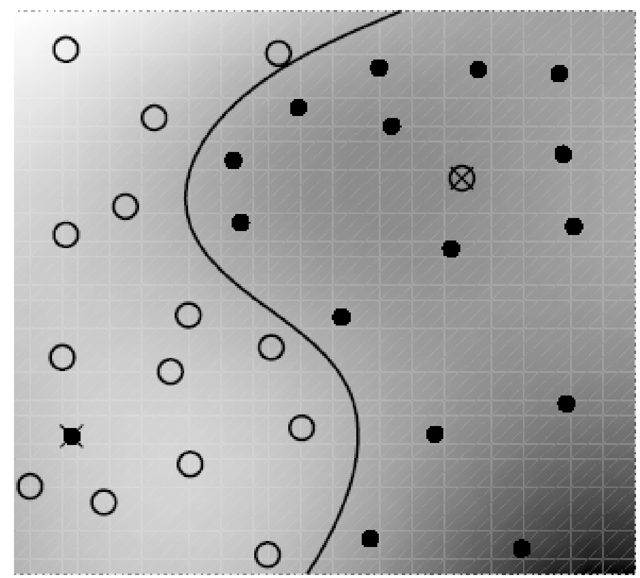
Interestingly, if d is large the kernel is still only requires n multiplications to compute, whereas the explicit representation may not fit in memory!

28 RBF-SVMs

The RBF kernel $K(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma||\boldsymbol{x} - \boldsymbol{x}'||^2)$ is one of the most popular kernel functions. It adds a "bump" around each data point:

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \exp(-\gamma||\boldsymbol{x}_i - \boldsymbol{x}||^2) + b$$





Using this one can get state-of-the-art results.

Example.

$$w_1 : (1,5)^t, (-2,-4)^t$$
$$w_2 : (2,3)^t, (-1,5)^t$$

다음점들을다음 커널과 SVM을 이용해서 분류하라.?

polynomial kernel

$$k(\mathbf{x}, \mathbf{z}) = \left(1 + \mathbf{x}^T \mathbf{z}\right)^2 = \left(1 + x_1 z_1 + x_2 z_2\right)^2$$
$$= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2$$
$$= \left(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1 x_2, x_2^2\right)\left(1, \sqrt{2}z_1, \sqrt{2}z_2, z_1^2, \sqrt{2}z_1 z_2, z_2^2\right)^T$$
$$= \phi(\mathbf{x})^T \phi(z)$$

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x_n}, \mathbf{x_m})$$

$$subject\ to\ a_n \geq 0\ , \sum_{n=1}^{N} a_n t_n = 0;$$

34. We repeat Example 2 in the text but with the following four points:

$$\mathbf{y}_1 = (1\ \sqrt{2}\ 5\sqrt{2}\ 5\sqrt{2}\ 1\ 25)^t,\quad \mathbf{y}_2 = (1\ -2\sqrt{2}\ -4\sqrt{2}\ 8\sqrt{2}\ 4\ 16)^t, z_1 = z_2 = -1$$
$$\mathbf{y}_3 = (1\ \sqrt{2}\ 3\sqrt{2}\ 6\sqrt{2}\ 4\ 9)^t,\quad \mathbf{y}_4 = (1\ -2\sqrt{2}\ 5\sqrt{2}\ -5\sqrt{2}\ 1\ 25)^t, z_3 = z_4 = +1$$

We seek the optimal hyperplane, and thus want to maximize the functional given by Eq. 109 in the text:

$$L(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}\sum_{kl}^{4} \alpha_l \alpha_k z_k z_l \mathbf{y}_k^t \mathbf{y}_l,$$

with constraints $\alpha_1 + \alpha_2 = \alpha_3 + \alpha_4$ and $\alpha_1 \geq 0$. We substitute $\alpha_4 = \alpha_1 + \alpha_2 - \alpha_3$ into $L(\alpha)$ and take the partial derivatives with respect to α_1, α_2 and α_3 and set the derivatives to zero:

$$\begin{aligned}
\frac{\partial L}{\partial \alpha_1} &= 2 - 208\alpha_1 - 256\alpha_2 + 232\alpha_3 = 0 \\
\frac{\partial L}{\partial \alpha_2} &= 2 - 256\alpha_1 - 592\alpha_2 + 496\alpha_3 = 0 \\
\frac{\partial L}{\partial \alpha_3} &= 232\alpha_1 + 496\alpha_2 - 533\alpha_3 = 0.
\end{aligned}$$

원래는 QP로 풀어야 되지만 간단한 문제이기 때문에 해석적으로 풀었다.lagrange multiplier 예제에서 보여준 것처럼.

The solution to these equations — $\alpha_1 = 0.0154$, $\alpha_2 = 0.0067$, $\alpha_3 = 0.0126$ — indeed satisfy the constraint $\alpha_i \geq 0$, as required.

Now we compute a using Eq. 108 in the text:

$$(8)\mathbf{w} = \sum_{n=1}^{N} a_n t_n \phi(\mathbf{x}_n)$$

$$\frac{\partial L}{\partial \mathbf{a}} = \mathbf{a} - \sum_{k=1}^{4} \alpha_k z_k \mathbf{y}_k = 0,$$

$\vec{a} \longrightarrow \vec{w}$

$a_0 \longrightarrow b$

which has solution

$$\begin{aligned}\mathbf{a} &= 0.0154(-\mathbf{y}_1) + 0.0067(-\mathbf{y}_2) + 0.01261\mathbf{y}_3 + 0.095\mathbf{y}_4 \\ &= (0 \;\; 0.0194 \;\; 0.0496 \;\; -0.145 \;\; 0.0177 \;\; -0.1413)^t.\end{aligned}$$

Note that this process cannot determine the bias term, a_0 directly; we can use a support vector for this in the following way: We note that $\mathbf{a}^t\mathbf{y}_k z_k = 1$ must hold for each support vector. We pick $\mathbf{y}_1$ and then

$$-(a_0 \;\; 0.0194 \;\; 0.0496 \;\; -0.145 \;\; 0.0177 \;\; -0.1413)\cdot\mathbf{y}_1 = 1,$$

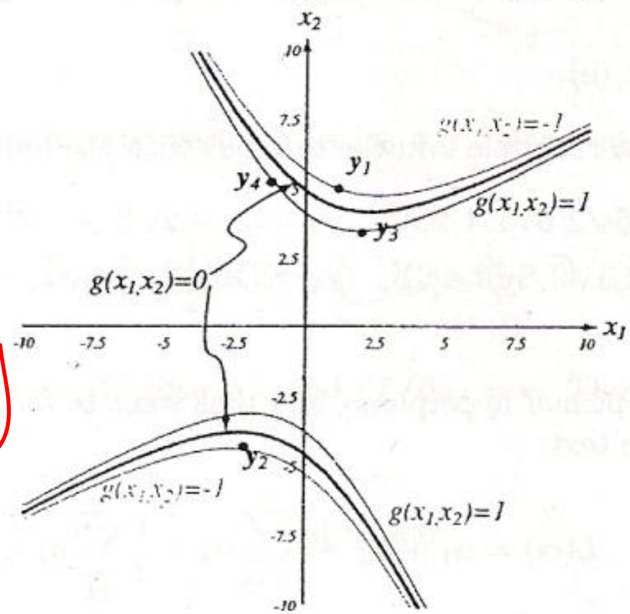
which gives $a_0 = 3.1614$, and thus the full weight vector is $\mathbf{a} = (3.1614 \;\; 0.0194 \;\; 0.0496 \;\; -0.145 \;\; 0.0177 \;\; -0.1413)^t$.

Now we plot the discriminant function in x_1–x_2 space:

고차원으로 매핑전의
차원에서의 초평면을
구하면 포물선 형태의
분류 경계가 나온다.

$$\begin{aligned}g(x_1, x_2) &= \mathbf{a}^t(1 \;\; \sqrt{2}x_1 \;\; \sqrt{2}x_2 \;\; \sqrt{2}x_1x_2 \;\; x_1^2 \;\; x_2^2) \\ &= 0.0272x_1 + 0.0699x_2 - 0.2054x_1x_2 + 0.1776x_1^2 - 0.1415x_2^2 + 3.17\end{aligned}$$

The figure shows the hyperbola corresponding to $g(x_1, x_2) = 0$ as well as the margins $g(x_1, x_2) = \pm 1$, along with the three support vectors $\mathbf{y}_2$, $\mathbf{y}_3$ and $\mathbf{y}_4$.



$g(\;) = \vec{a}^t\vec{y} = \vec{a}^T \phi(\vec{x})$

30 SVMs : software

Lots of SVM software:

- LibSVM (C++)
- SVMLight (C)

As well as complete machine learning toolboxes that include SVMs:

- Torch (C++)
- Spider (Matlab)
- Weka (Java)

All available through `www.kernel-machines.org`.

레퍼런스
1,비숍책
2,MIT 오픈강의
http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-867Fall-2006/LectureNotes/index.htm

3,수많은 인터넷 자료들?

Fast Training of Support Vector Machines using Sequential Minimal Optimization,

Platt, John.

Jung-Kyu Lee @ MLLAB

SVM revisited

- the (dual) optimization problem we covered in SVM class

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle.$$

$$\text{s.t} \quad 0 \le \alpha_{\mathrm{i}} \le C, \ \mathrm{i} = 1, \ldots, \mathrm{m}$$

$$\sum_{\mathrm{i}=1}^{\mathrm{m}} \alpha_i y_i = 0$$

- Solving this optimization problem for large-scale application by using quadratic programming is very computationally expensive.
- John Platt a colleague at Microsoft, proposed efficient algorithm called SMO (Sequential Minimal Optimization) to actually solve this optimization problem.

Digression: Coordinate ascent(1/3)

- Consider trying to solve the following unconstrained optimization problem.

$$\max_{\alpha} W(\alpha_1, \alpha_2, ..., \alpha_m)$$

- Coordinate ascent:

Loop until convergence: {

    For $i = 1, \ldots, m$, {

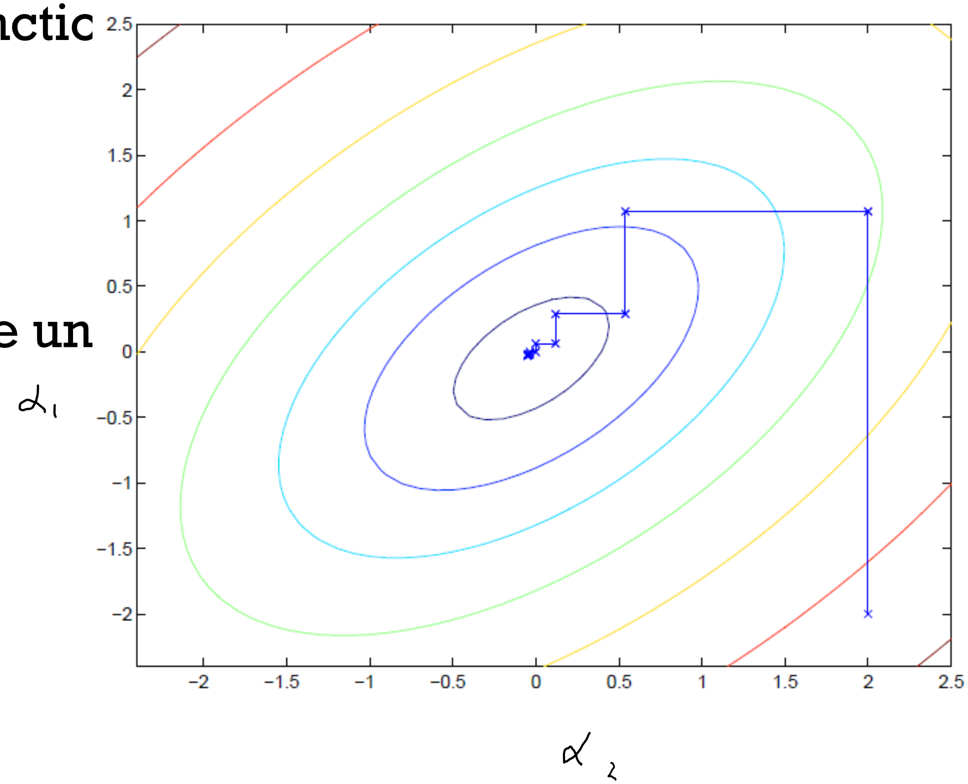
        $\alpha_i := \arg\max_{\hat{\alpha}_i} W(\alpha_1, \ldots, \alpha_{i-1}, \hat{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_m)$.

    }

}

Digression: Coordinate ascent(2/3)

- How coordinate ascent work?
- We optimize a quadratic functic
- Minimum is (0,0)
- First, minimize this w.r.t α_1
- Next, minimize this w.r.t α_2
- With same argument, iterate un convergence.



Digression: Coordinate ascent(3/3)

- Coordinate assent will usually take a lot more steps than other iterative optimization algorithm such as gradient descent, Newton's method, conjugate gradient descent etc.

- However, there are many optimization problems for which it's particularly easy to fix all but one of the parameters and optimize with respect to just that one parameter.

- It turns out that this will be true when we modify this algorithm to solve the SVM optimization problem.

SMO

- Coordinate assent in its basic form does not work to apply SVM dual optimization problem.

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle.$$

$$\text{s.t} \quad 0 \le \alpha_{\mathrm{i}} \le C,\ \mathrm{i} = 1,\ldots,\mathrm{m}$$

$$\sum_{\mathrm{i}=1}^{\mathrm{m}} \alpha_i y_i = 0$$

- As coordinate assent progress, we can't change α_i without violating the constraint.

$$\sum_{\mathrm{i}=1}^{\mathrm{m}} \alpha_i y_i = 0$$

$$\Leftrightarrow \alpha_1 y_1 = -\sum_{\mathrm{i}=2}^{\mathrm{m}} \alpha_i y_i$$

$$\Leftrightarrow \alpha_1 = -y_1 \sum_{\mathrm{i}=2}^{\mathrm{m}} \alpha_i y_i (\because y_1 \in \{1,-1\}, y_1{}^2 = 1)$$

Outline of SMO

- The SMO (sequential minimal optimization) algorithm, therefore, instead of trying to change one α at a time, we will try to change two α at a time to keep satisfying the constraints.
- The term 'minimal' refers to the fact that we're choosing the smallest number of α

Repeat till convergence {

1. Select some pair α_i and α_j to update next (using a heuristic that tries to pick the two that will allow us to make the biggest progress towards the global maximum).
2. Reoptimize $W(\alpha)$ with respect to α_i and α_j, while holding all the other α_k's $(k \neq i, j)$ fixed.

}

Convergence of SMO

- *To* test for convergence of SMO algorithm, we can check whether the KKT conditions are satisfied to within some tolerance

$$a_i = 0 \Rightarrow y_i(w^T x_i + b) \geq 1$$
$$a_i = C \Rightarrow y_i(w^T x_i + b) \leq 1$$
$$0 < a_i < C \Rightarrow y_i(w^T x_i + b) = 1$$

- The key reason that SMO is an efficient algorithm is that updating α_i, α_j can be computed very efficiently.

Detail of SMO

- Suppose we've decide to hold $\alpha_3, \ldots, \alpha_m$, update α_1, α_2

$$\sum_{i=1}^{m} \alpha_i y_i = 0$$

$$\Leftrightarrow \alpha_1 y_1 + \alpha_2 y_2 = \sum_{i=3}^{m} \alpha_i y_i$$

$$\Leftrightarrow \alpha_1 y_1 + \alpha_2 y_2 = \zeta$$

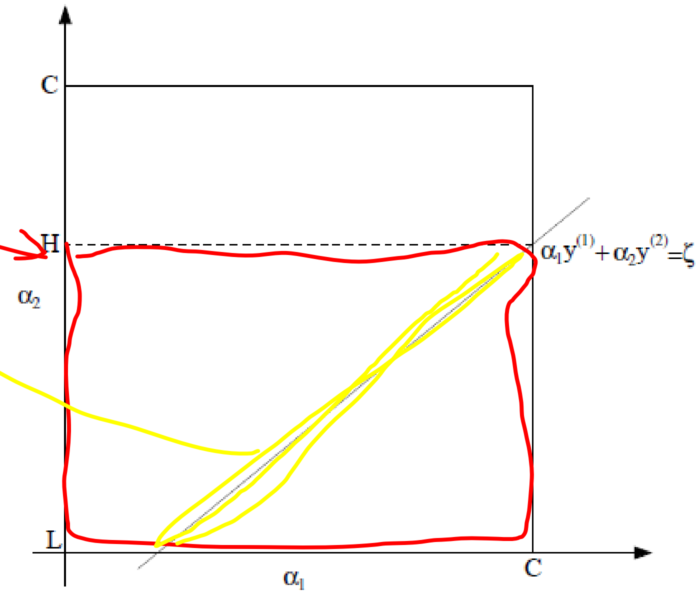
Detail of SMO

- We can thus picture the constraints on α_1, α_2 as follows
- α_1 and α_2 must lie within the box $[0,C]\times[0,C]$ shown.
- α_1 and α_2 must lie on line :

$$\alpha_1 y_1 + \alpha_2 y_2 = \zeta$$

- From these constraints, we know:

$$L \le \alpha_2 \le H$$



Detail of SMO

- From $\alpha_1 y_1 + \alpha_2 y_2 = \zeta$

$$\Leftrightarrow \alpha_1 y_1 = (\zeta - \alpha_2 y_2)$$

$$\Leftrightarrow \alpha_1 y_1^{\,2} = (\zeta - \alpha_2 y_2) y_1$$

$$\Leftrightarrow \alpha_1 = (\zeta - \alpha_2 y_2) y_1$$

- W(α) become

$$W(\alpha_1, \alpha_2, \ldots, \alpha_m) = W((\zeta - \alpha_2 y_2) y_1, \alpha_2, \ldots, \alpha_m)$$

$$= a\alpha_2^{\,2} + b\alpha_2 + c$$

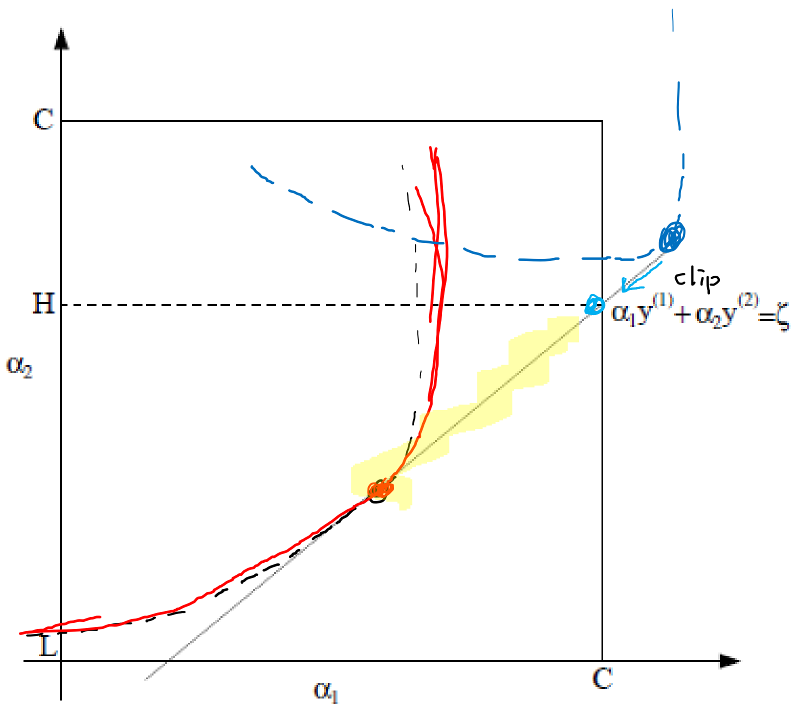
- This is a standard quadratic function.

Detail of SMO

- From

$$a\alpha_2^{\;2} + b\alpha_2 + c$$

- This is really easy to optimize by setting its derivative to zero.
- If you end up with a value outside, you must clip your solution.
- That'll give you the optimal soluti of this quadratic optimization problem subject to your solution satisfying this box constraint and lying on this straight line.



Detail of SMO

- In conclusion,

$$\alpha_2^{new} = \begin{cases} H & if \;\; \alpha_2^{new,unclipped} > H \\ \alpha_2^{new,unclipped} & if \;\; L \le \alpha_2^{new,unclipped} \le H \\ L & if \;\; \alpha_2^{new,unclipped} < L \end{cases}$$

$$\alpha_1 = (\zeta - \alpha_2 y_2) y_1$$

- We do this process very quickly, which makes the inner loop of the SMO algorithm very efficient.

Conclusion

- SMO

  - has scalability for large scale data.

  - makes SVM computationally inexpensive.

  - can be implemented easily because it has simple training structure (coordinate ascent).

Reference

- [1] Platt, John. Fast Training of Support Vector Machines using Sequential Minimal Optimization, in Advances in Kernel Methods – Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds., MIT Press (1998).
- [2] The Simplified SMO Algorithm , Machine Learning course @ Stanford http://www.stanford.edu/class/cs229/materials/smo.pdf