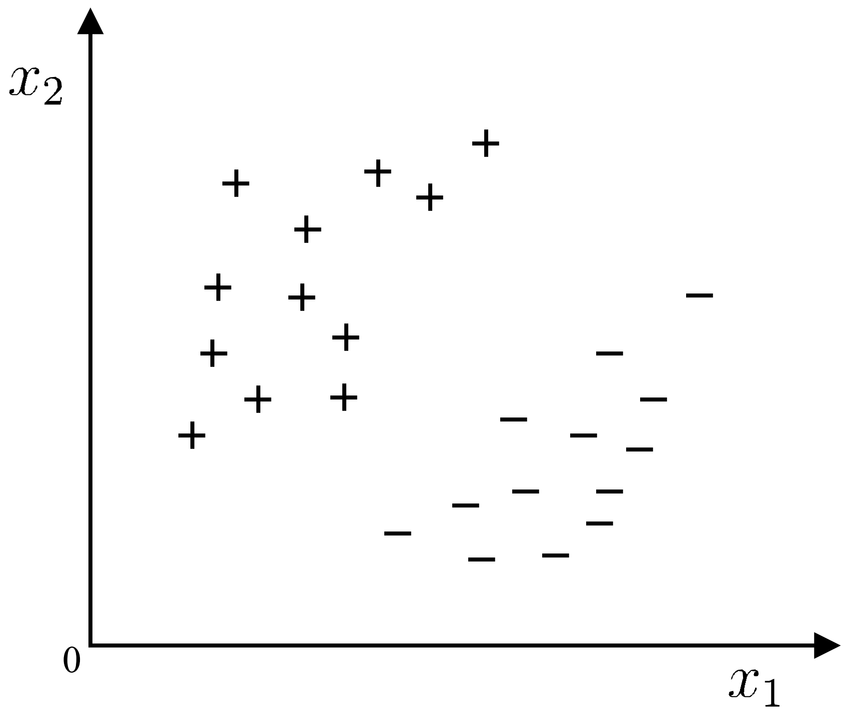
Machine Learning
机器学习

Lecture8：支持向量机

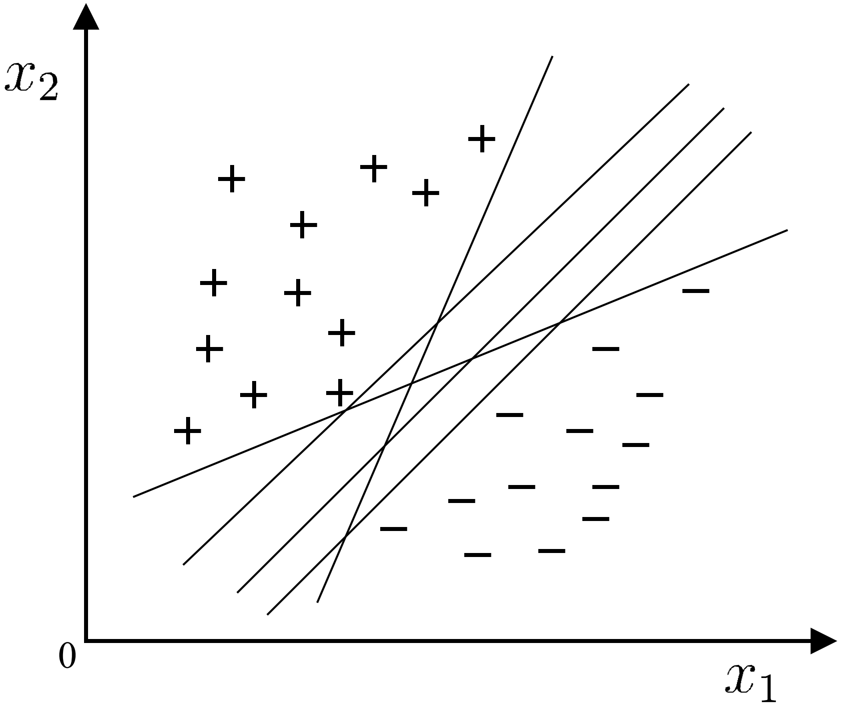
李洁
nijanice@163.com

二分类问题
Binary Classification



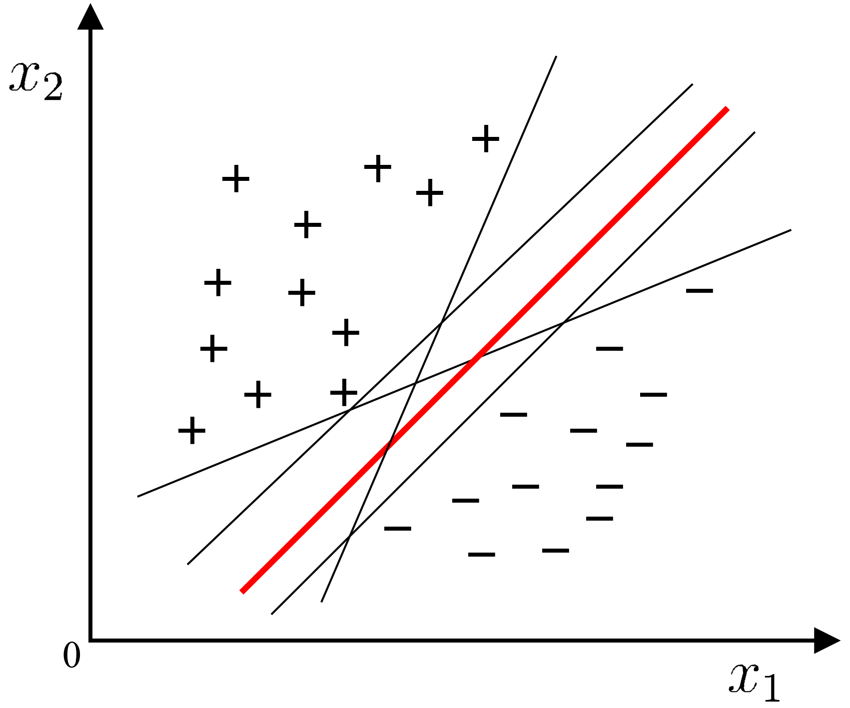
Question： Find a hyperplane in the sample space to separate samples of different categories.

二分类问题
Binary Classification



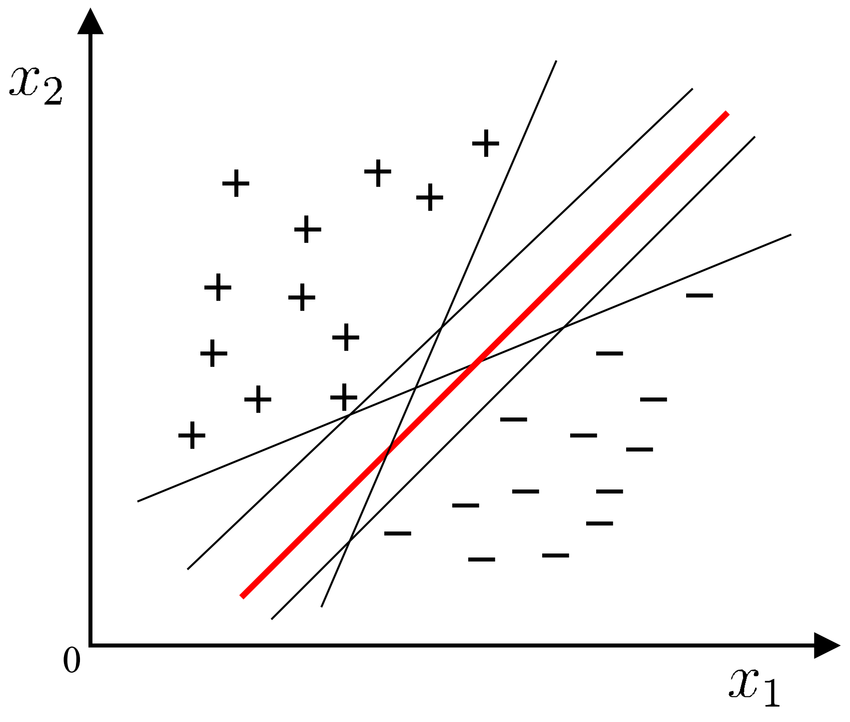
Question：Find a hyperplane in the sample space to separate samples of different categories.

二分类问题
Binary Classification



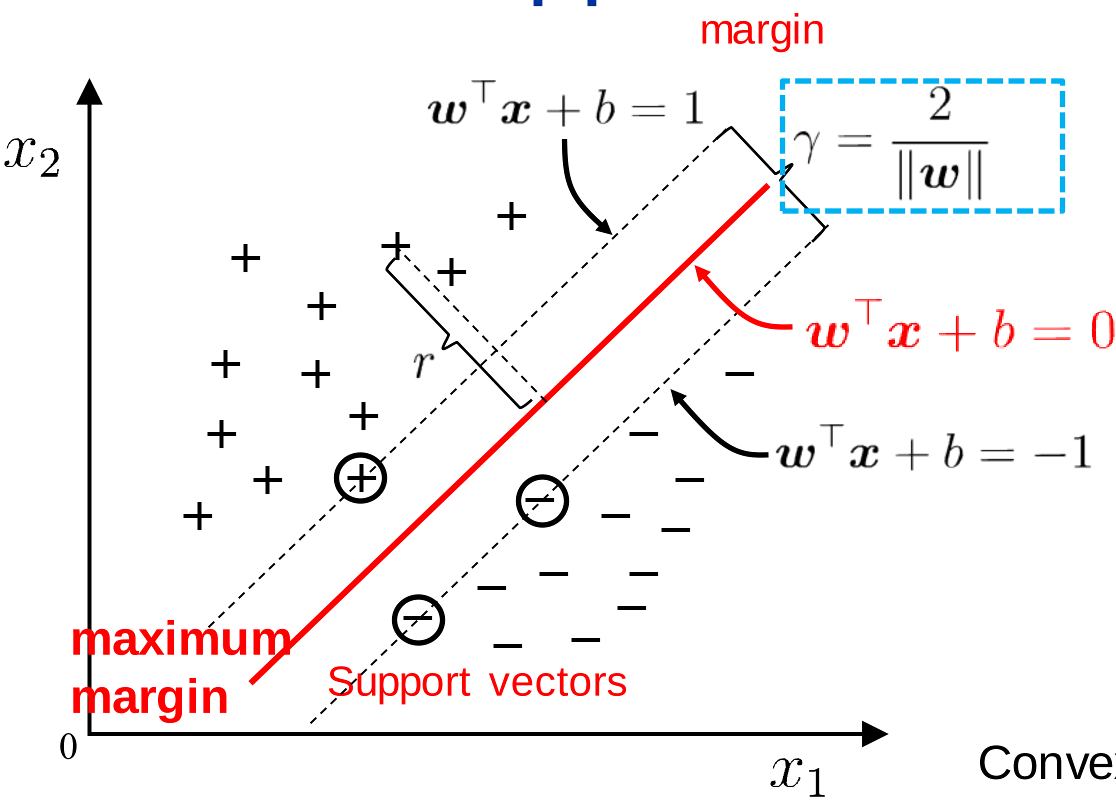
Question：Find a hyperplane in the sample space to separate samples of different categories.

二分类问题
Binary Classification



It should choose "right in the middle", with good tolerance, high robustness and the strongest generalization ability

支持向量机
Support Vector Machine



$$\gamma = \frac{2}{\|\boldsymbol{w}\|}$$

$$f(x) = w^T x + b$$

$$d = \frac{|w^{\mathrm{T}} x + b|}{||w||}$$

Convex Quadratic Programming

$$\arg\max_{\boldsymbol{w}, b} \frac{2}{\|\boldsymbol{w}\|} \Longrightarrow \arg\min_{\boldsymbol{w}, b} \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t. } y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \; i = 1, 2, \ldots, m.$$

对偶问题
Dual problem

A dual problem, in mathematics and optimization, is a counterpart to the primal optimization problem. It is constructed based on the constraints of the primal problem and aims to find either an upper bound (in case of minimization problems) or a lower bound (in case of maximization problems) on the optimal value of the primal problem.

For a minimization problem:

- Dual Problem: max g(λ,v) where g(λ,v)=infx L(x,λ,v)

For a maximization problem:

- Dual Problem: ming(λ,v) where g(λ,v)=supxL(x,λ,v)

Here:

- λ and v are dual variables (Lagrange Multipliers) corresponding to the constraints of the primal problem.
- L(x,λ,v) is the Lagrangian function, which is composed of the primal objective function and the constraints, typically in the form

L(x,λ,v)=objective function−λT·(inequality constraints)−vT·(equality constraints)

By solving the dual problem, we obtain either an upper bound or a lower bound on the optimal value of the primal problem.

拉格朗日乘数法的例子

- **问题**：假设你需要在平面上找到距离原点最近的点，但这个点必须位于直线 $y=2x+3$ 上。
- **解决方法**：

拉格朗日乘数法的例子

- **问题**：假设你需要在平面上找到距离原点最近的点，但这个点必须位于直线y=2x+3 上。
- **解决方法**：

1. **目标函数**：需要最小化的目标函数是到原点的距离的平方，$f(x,y)=x^2+y^2$。
2. **约束**：点必须位于y=2x+3 上，因此约束条件为$g(x,y)=y-2x-3=0$。
3. **构建拉格朗日函数**：构建拉格朗日函数$\mathrm{L}(x,y,\lambda)=x^2+y^2+\lambda(y-2x-3)$。
4. **求解**：通过$\mathrm{L}(x,y,\lambda)$关于x,y,λ的偏导数求解并置为零，可以得到最优解。

KKT（Karush-Kuhn-Tucker）条件的例子

- **问题**：假设你需要在一个盒子中放置最大体积的长方体，但其长、宽、高的和不能超过某个值，例如10
- **解决方法**：

KKT（Karush-Kuhn-Tucker）条件的例子

- **问题**：假设你需要在一个盒子中放置最大体积的长方体，但其长、宽、高的和不能超过某个值，例如10
- **解决方法**：

 1. **目标函数**：最大化长方体的体积，即 $f(x,y,z)=xyz$。
 2. **约束**：长、宽、高的和不超过10，即 $g(x,y,z)=x+y+z-10\leq 0$。这是一个不等式约束。
 3. **构建拉格朗日函数：拉格朗日函数 L(x,y,z,λ) 结合了目标函数和约束条件，通过引入拉格朗日乘子λ**：
 $L(x,y,\lambda)=xyz-\lambda(x+y+z-10)$

KKT（Karush-Kuhn-Tucker）条件的例子

$L(x,y,\lambda)=xyz-\lambda(x+y+z-10)$

4.应用KKT条件

- **梯度为零**：对 $L(x,y,z,\lambda)$ 对 x,y,z,λ 的偏导数应为零。
  - $\frac{\partial L}{\partial x}=yz-\lambda=0$ $\frac{\partial L}{\partial y}=xz-\lambda=0$ $\frac{\partial L}{\partial z}=xy-\lambda=0$
  - $\frac{\partial L}{\partial \lambda}=-x-y-z+10=0$
- **约束条件**：$x+y+z\leq 10$。
- **拉格朗日乘子非负**：$\lambda\geq 0$。
- **互补松弛性**：$\lambda(x+y+z-10)=0$。

5.求解：通过解上述方程组，我们可以找到满足约束的最优解。

最优超平面求解
Optimal hyperplane Solution

- Using Lagrangian Multiplier Method and KKT Condition to solve the Optimal Value

$$\arg\min_{\boldsymbol{w},b} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{s.t.} \quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \; i = 1, 2, \ldots, m.$$

- Integrated into: (Where $\alpha_i \geq 0$ is a Lagrangian multiplier)

$$L(w, b, a\ ) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i(w^T x_i + b))$$

- Let the partial derivative=0

$$\frac{\partial L(w, b, \alpha)}{\partial w} = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0 \qquad \frac{\partial L(w, b, \alpha)}{\partial b} = \sum_{i=1}^{m} \alpha_i y_i = 0$$

- then

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

最优超平面求解
Optimal hyperplane Solution

$$L(w,b,a) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \alpha_i(1 - y_i(w^T x_i + b))$$

$$= \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i y_i w^T x_i - \mathrm{b} \sum_{i=1}^{m} \alpha_i y_i + \sum_{i=1}^{m} \alpha_i$$

$$= \frac{1}{2} w^T \sum_{i=1}^{m} \alpha_i y_i x_i - \sum_{i=1}^{m} \alpha_i y_i w^T x_i + + \sum_{i=1}^{m} \alpha_i$$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \alpha_i y_i w^T x_i$$

Dual problem:

$$\max_{a} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j = \min_{a} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0, \alpha_i \geq 0, i = 1,2,\ldots,m$$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm

The optimal solution can be obtained

$$\boxed{f(x) = w^T x + b = \sum_{i=1}^{m} \alpha_i y_i x_i^T x + b}$$

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

$$y_i f(\boldsymbol{x}_i) > 1 \Rightarrow \alpha_i = 0$$

$$\alpha_i > 0 \Rightarrow y_i f(x_i) = 1$$

最优超平面求解
Optimal hyperplane Solution

$$L(w,b,a\ )=\frac{1}{2}\|w\|^2+\sum_{i=1}^{m}\alpha_i(1-y_i(w^Tx_i+b))$$

$$=\ \frac{1}{2}w^Tw-\sum_{i=1}^{m}\alpha_iy_iw^Tx_i-\mathrm{b}\sum_{i=1}^{m}\alpha_iy_i+\sum_{i=1}^{m}\alpha_i$$

$$=\frac{1}{2}w^T\sum_{i=1}^{m}\alpha_iy_ix_i-\sum_{i=1}^{m}\alpha_iy_iw^Tx_i++\sum_{i=1}^{m}\alpha_i$$

$$=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\alpha_iy_iw^Tx_i$$

Dual problem:

$$\max_a\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\,\alpha_jy_iy_jx_i^Tx_j=\min_a\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\,\alpha_jy_iy_jx_i^Tx_j-\sum_{i=1}^{m}\alpha_i$$

$$s.t.\sum_{i=1}^{m}\alpha_iy_i=0\,,\alpha_i\geq 0,i=1,2,\ldots,m$$

KKT Condition

$$\begin{cases}\alpha_i\geq 0,\\ y_if(\boldsymbol{x}_i)\geq 1,\\ \alpha_i(y_if(\boldsymbol{x}_i)-1)=0.\end{cases}$$

$y_if(\boldsymbol{x}_i)>1 \Rightarrow \alpha_i=0$

$\alpha_i>0 \Rightarrow y_if(x_i)=1$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm

The optimal solution can be obtained

$$\boxed{f(x)=w^Tx+b=\sum_{i=1}^{m}\alpha_iy_ix_i^Tx+b}$$

最优超平面求解
Optimal hyperplane Solution

$$L(w,b,a) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i(w^T x_i + b))$$

$$= \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i y_i w^T x_i - b \sum_{i=1}^{m} \alpha_i y_i + \sum_{i=1}^{m} \alpha_i$$

$$= \frac{1}{2} w^T \sum_{i=1}^{m} \alpha_i y_i x_i - \ldots$$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \alpha_i y \ldots$$

Dual problem:

$$\max_{a} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y \ldots$$

$$s.t. \sum_{i=1}^{m} \ldots$$

若α_i= 0，则该样本将不会在左式的求和中出现，也就不会对 f(x) 有任何影响;
若α_i > 0,则必有执 f(Xi) = 1，所对应的样本点位于最大间隔边界上，是一个支持向量。

这显示出支持向量机的一个重要性质:训练完成后，大部分的训练样本都不需保留，最终模型仅与支持向量有关。

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

$y_i f(\boldsymbol{x}_i) > 1 \Rightarrow \alpha_i = 0$

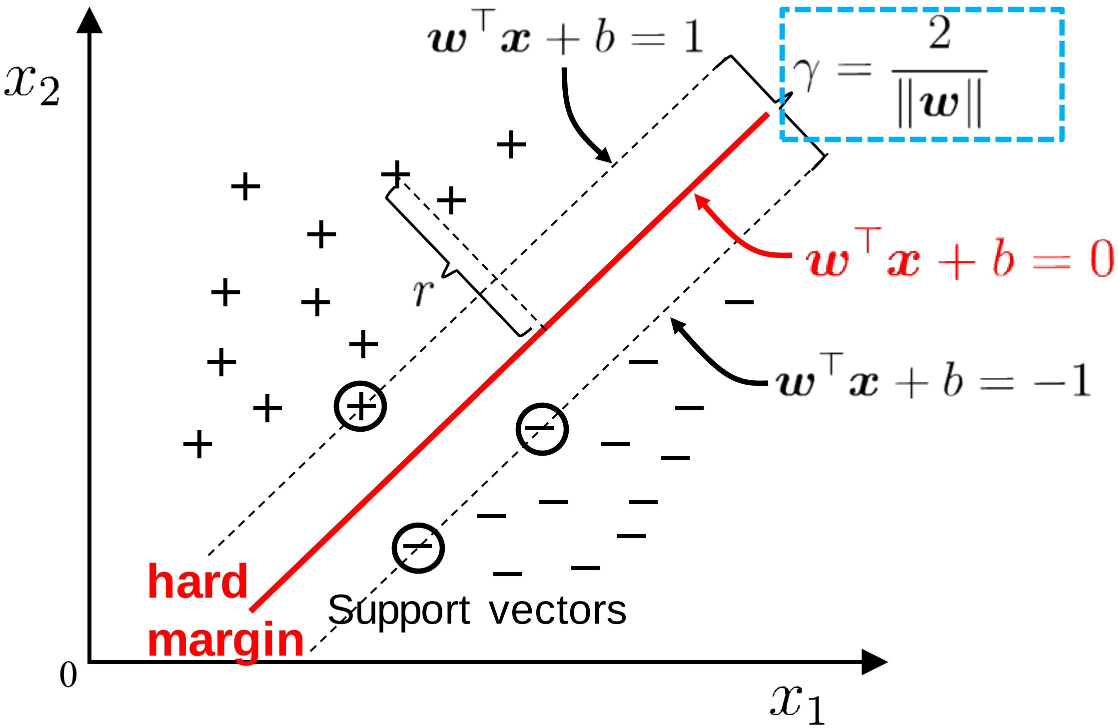
$\alpha_i > 0 \Rightarrow y_i f(x_i) = 1$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm

The optimal solution can be obtained

$$f(x) = w^T x + b = \sum_{i=1}^{m} \alpha_i y_i x_i^T x + b$$

支持向量机
Support Vector Machine



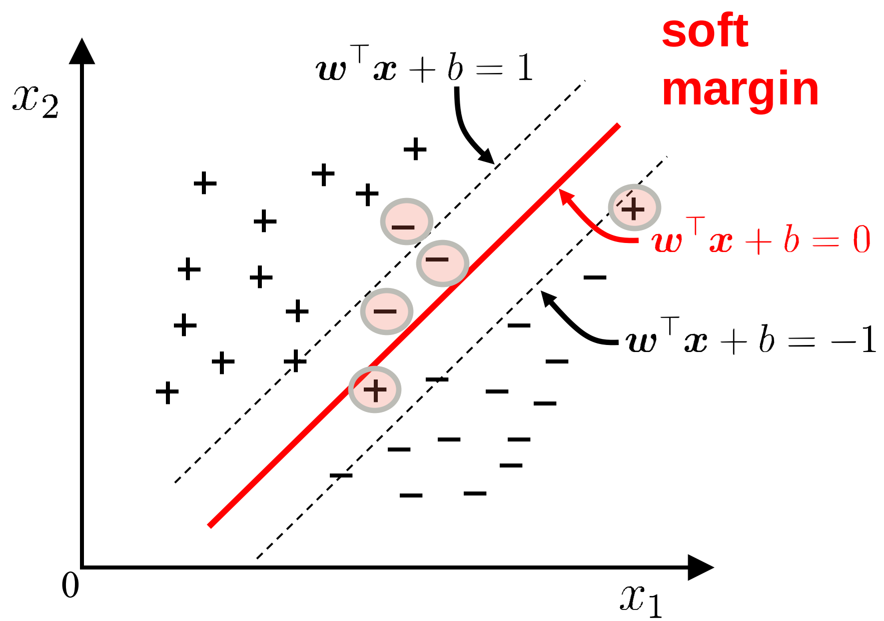
$$f(x) = w^T x + b$$

$$\arg\max_{\boldsymbol{w},b} \frac{2}{\|\boldsymbol{w}\|} \Rightarrow \arg\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t.} \quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \; i = 1, 2, \ldots, m.$$

软间隔
Soft Margin

-Q: It is difficult to determine a linearly separable hyperplane in the feature space; At the same time, it is difficult to determine whether a linearly separable result is caused by over fitting
-A: The concept of "soft margin" is introduced to allow the support vector machine to not meet the constraints on some samples



软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2}\|w\|_2 + C\sum_i \varepsilon_i \qquad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

$$\varepsilon_i \geq 0, \text{i=1,2,…m}$$

Integrated into: (Where α_i , ε_i are Lagrangian multipliers)

$$L(w,b,a,\varepsilon,\mu) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\varepsilon_i + \sum_{i=1}^{m}\alpha_i(1-\varepsilon_i - y_i(w^T x_i + b)) - \sum_{i=1}^{m}\mu_i\varepsilon_i$$

Let the partial derivative=0

$$\frac{\partial L(w,b,a,\varepsilon,\mu)}{\partial w} = w - \sum_{i=1}^{m}\alpha_i y_i x_i = 0 \qquad \frac{\partial L(w,b,a,\varepsilon,\mu)}{\partial b} = \sum_{i=1}^{m}\alpha_i y_i = 0$$

$$\frac{\partial L(w,b,a,\varepsilon,\mu)}{\partial \varepsilon_i} = C - \alpha_i - \mu_i = 0$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\min_{w,b}\frac{1}{2}\|w\|_2+C\sum_{i}\varepsilon_i \qquad \text{s.t. } y_i(w^Tx_i+b)\geq 1-\varepsilon_i$$

$$\varepsilon_i \geq 0, \text{ i=1,2,…m}$$

Integrated into: (Where α_i , ε_i are Lagrangian multipliers)

$$L(w,b,a,\varepsilon,\mu)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{m}\varepsilon_i+\sum_{i=1}^{m}\alpha_i(1-\varepsilon_i-y_i(w^Tx_i+b))-\sum_{i=1}^{m}\mu_i\varepsilon_i$$

Let the partial derivative=0

then

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

$$C=\alpha_i+\mu_i$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\min_{w,b} \frac{1}{2}\|w\|_2 + C\sum_i \varepsilon_i \qquad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

$$\varepsilon_i \geq 0, \text{ i=1,2,…m}$$

Integrated into: (Where α_i , ε_i are Lagrangian multipliers)

$$L(w,b,a,\varepsilon,\mu) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\varepsilon_i + \sum_{i=1}^{m}\alpha_i(1-\varepsilon_i - y_i(w^T x_i + b)) - \sum_{i=1}^{m}\mu_i\varepsilon_i$$

Dual problem:

$$\max_{a}\sum_{i=1}^{m}\alpha_i - \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j \quad s.t. \sum_{i=1}^{m}\alpha_i y_i = 0, \qquad C \geq \alpha_i \geq 0, i = 1,2,\ldots,m$$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm

The optimal solution can be obtained

$$\boxed{f(x) = w^T x + b = \sum_{i=1}^{m}\alpha_i y_i x_i^T x + b}$$

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \mu_i \geq 0 \\ y_i f(x_i)\text{-}1+\varepsilon_i \geq 0 \\ \alpha_i(y_i f(x_i)-1+\varepsilon_i) = 0 \\ \varepsilon_i \geq 0, \mu_i\varepsilon_i = 0 \end{cases}$$

软间隔支持向量机
Soft Margin Support Vector Machine

对任意训练样本，总有α_i =0或 $y_i f(x_i)=1-\varepsilon_i$

若α_i= 0，则该样本将不会在左式的求和中出现，也就不会对 f(x) 有任何影响;

若α_i > 0,则必有 $y_i f(x_i)=1-\varepsilon_i$，即该样本是一个支持向量。

若α_i < C，则μ_i > 0，进而有 ε_i = 0，即该样本恰在最大间隔边界上;

若 α_i =C，则有μ_i = 0，此时若 ε_i ≤1，则该样本落在最大间隔内部；若 ε_i > 1 则该样本被错误分类。

由此可看出软间隔支持向量机的最终模型仅与支持向量有关，仍保持了稀疏性。

$+ b) \geq 1 - \varepsilon_i$

i=1,2,…m

$(w^T x_i+b)) - \sum_{i=1}^{m} \mu_i \varepsilon_i$

$C \geq \alpha_i \geq 0, i = 1,2,\dots,m$

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \mu_i \geq 0 \\ y_i f(x_i)-1+\varepsilon_i \geq 0 \\ \alpha_i(y_i f(x_i)-1+\varepsilon_i) = 0 \\ \varepsilon_i \geq 0, \mu_i \varepsilon_i = 0 \end{cases}$$

$$f(x) = w^T x + b = \sum_{i=1} \alpha_i y_i x_i^T x + b$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\min_{w,b} \frac{1}{2}\|w\|_2 + C\sum_i \varepsilon_i \quad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

Dual Problem:

$$\max_a \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\, \alpha_j y_i y_j x_i^T x_j = \min_a \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\, \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0\,, C \geq \alpha_i \geq 0, i = 1,2, \dots, m$$

最优超平面求解
Optimal hyperplane Solution

$$L(w,b,a) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i(w^T x_i + b))$$

$$= \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i y_i w^T x_i - \mathrm{b} \sum_{i=1}^{m} \alpha_i y_i + \sum_{i=1}^{m} \alpha_i$$

$$= \frac{1}{2} w^T \sum_{i=1}^{m} \alpha_i y_i x_i - \sum_{i=1}^{m} \alpha_i y_i w^T x_i + + \sum_{i=1}^{m} \alpha_i$$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \alpha_i y_i w^T x_i$$

Dual problem:

$$\max_a \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j = \min_a \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0, \alpha_i \geq 0, i = 1,2,\dots,m$$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm

The optimal solution can be obtained

$$\boxed{f(x) = w^T x + b = \sum_{i=1}^{m} \alpha_i y_i x_i^T x + b}$$

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

$$y_i f(\boldsymbol{x}_i) > 1 \Rightarrow \alpha_i = 0$$

$$\alpha_i > 0 \Rightarrow y_i f(x_i) = 1$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\min_{w,b} \frac{1}{2} \|w\|_2 + C \sum_{i} \varepsilon_i \quad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

Dual Problem:

$$\max_{a} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j = \min_{a} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0 , \boxed{C \geq \alpha_i \geq 0,} i = 1,2, \dots , m$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced：

$$\min_{w,b} \frac{1}{2} \|w\|_2 + C \sum_{i} \varepsilon_i \quad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

在软间隔SVM中，引入正则化参数C控制了对分类错误的惩罚。

- 较小的C值会导致更大的间隔，容忍更多的分类错误，从而提高模型的容错性；
- 较大的C值会更强调正确分类，但可能导致对异常点更敏感。

$$\min_{a} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$\boxed{C \geq \alpha_i \geq 0}, i = 1,2,\ldots,m$$

软间隔支持向量机
Soft Margin Support Vector Machine

The slack variable ε can be introduced :

$$\min_{w,b} \frac{1}{2}\|w\|_2 + C\sum_{i}\varepsilon_i \quad \text{s.t. } y_i(w^T x_i + b) \geq 1 - \varepsilon_i$$

Dual Problem:

$$\max_{a} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j = \min_{a} \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

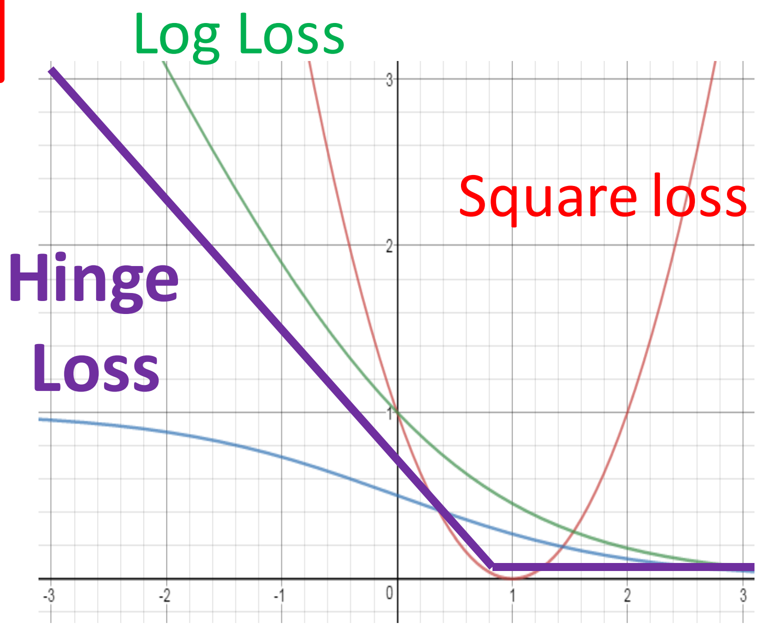
$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0 \,, C \geq \alpha_i \geq 0, i = 1,2, \dots, m$$

支持向量机
Support Vector Machine

The slack variable ε indicating the extent to which the sample does not meet the constraint.

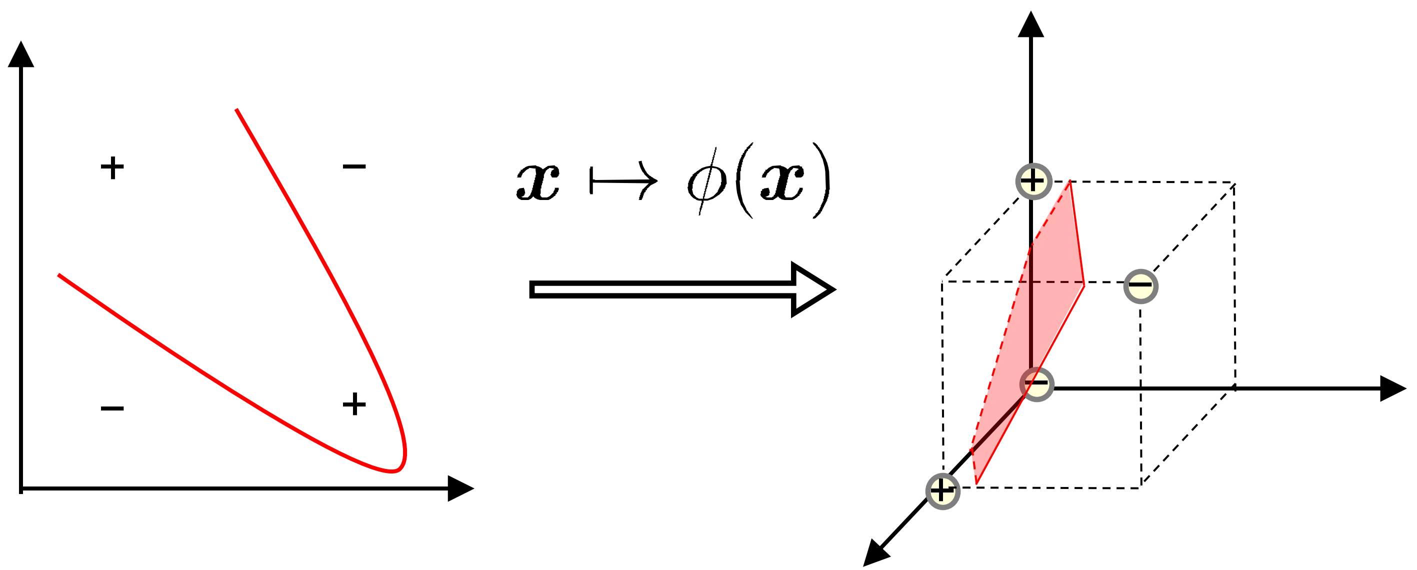
$$\boxed{\varepsilon^{\mathrm{i}} = max(0, 1 - y^{(i)} f(x^{(i)}))}$$

Hinge Loss function



线性不可分
linearly Inseparable Problem

-Q: What if there is no hyperplane that can correctly divide two types of samples?
-A: The samples are mapped from the original space to a higher dimensional feature space, making the samples linearly separable in this feature space



最优超平面求解
Optimal hyperplane Solution

$$L(w,b,a) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i (w^T x_i + b))$$

$$= \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i y_i w^T x_i - \mathrm{b} \sum_{i=1}^{m} \alpha_i y_i + \sum_{i=1}^{m} \alpha_i$$

$$= \frac{1}{2} w^T \sum_{i=1}^{m} \alpha_i y_i x_i - \sum_{i=1}^{m} \alpha_i y_i w^T x_i + + \sum_{i=1}^{m} \alpha_i$$

$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \alpha_i y_i w^T x_i$$

Dual problem:

$$\max_{a} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j = \min_{a} \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{m} \alpha_i$$

$$s.t. \sum_{i=1}^{m} \alpha_i y_i = 0, \alpha_i \geq 0, i = 1,2,\ldots,m$$

Obtain the optima α (SMO, Sequential Minimal Optimization) algorithm
The optimal solution can be obtained

$$\boxed{f(x) = w^T x + b = \sum_{i=1}^{m} \alpha_i y_i x_i^T x + b}$$

KKT Condition

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

$$y_i f(\boldsymbol{x}_i) > 1 \Longrightarrow \alpha_i = 0$$

$$\alpha_i > 0 \Longrightarrow y_i f(x_i) = 1$$

核支持向量机
Kernel SVM

sample $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$, then the hyperplane $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$

Original question:

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) \geq 1, \ i = 1, 2, \ldots, m. \end{aligned}$$

Dual problem:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j) - \sum_{i=1}^{m} \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \ \alpha_i \geq 0, \ i = 1, 2, \ldots, m. \end{aligned}$$

Optimal solution:

$$\boxed{f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b = \sum_{i=1}^{m} \alpha_i y_i \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}) + b}$$

核函数
Kernel Function

sample $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$, then the hyperplane $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$

Original question:

Kernel Function

$$\boxed{\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)}$$

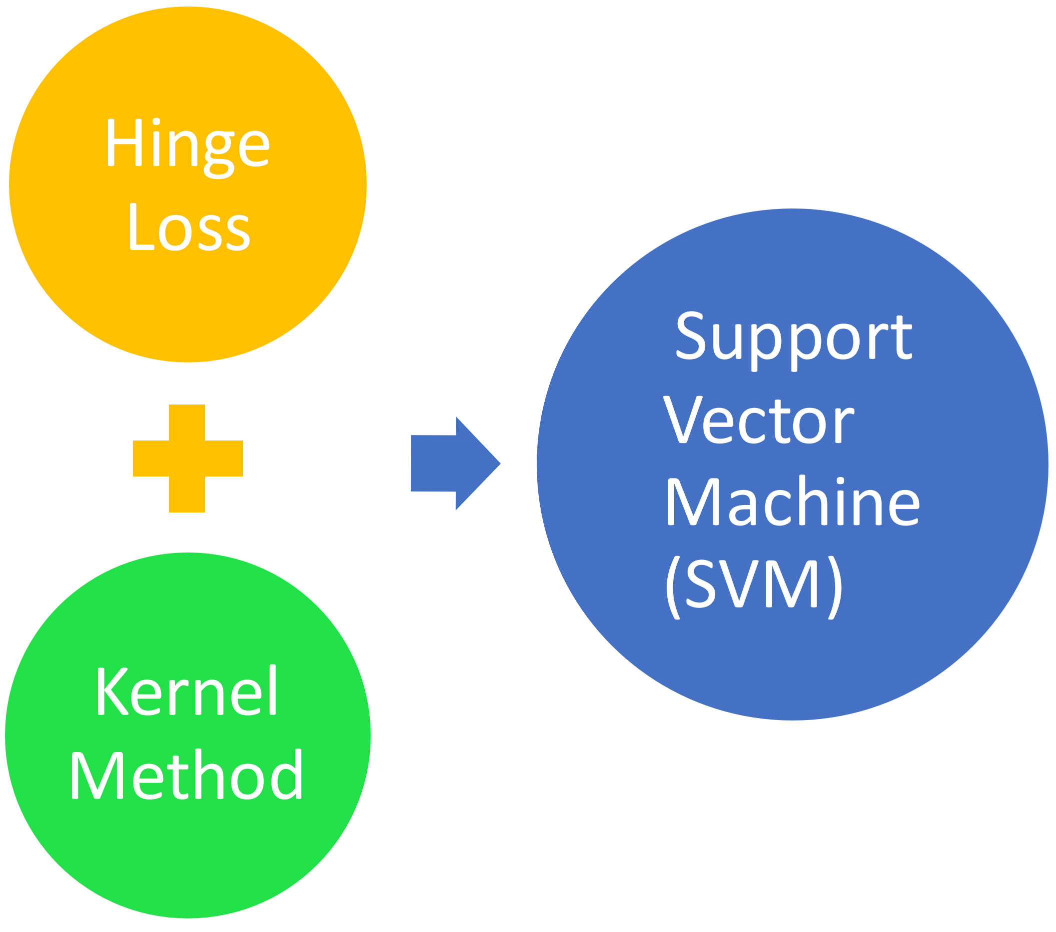
$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) \geq 1, \ i = 1, 2, \ldots, m. \end{aligned}$$

Dual problem:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \boxed{\phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)} - \sum_{i=1}^m \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \ \alpha_i \geq 0, \ i = 1, 2, \ldots, m. \end{aligned}$$

Optimal solution:

$$f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b = \sum_{i=1}^m \alpha_i y_i \boxed{\phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x})} + b$$


Hinge Loss
Kernel Method
Support Vector Machine (SVM)

监督学习
Supervised Learning

- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx f_\theta(x^{(i)})$$

- Function set $\{f_\theta(x^{(i)})\}$ is called hypothesis space
- Learning is referred to as updating the parameter θ to make the prediction closed to the corresponding label

监督学习
Supervised Learning

- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx f_{\theta}(x^{(i)})$$

- How to learn?

Update the parameter to make the prediction closed to the corresponding label

1. What is the learning objective?

2. How to update the parameters?

学习目标
Learning Objective

- Minimize the total loss

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y^{(i)}, f_\theta(x^{(i)}))$$

Loss function $L(y^{(i)}, f_\theta(x^{(i)}))$ measures the error between the label and prediction for single sample.

We have used

squared loss: $\frac{1}{2}(y^{(i)} - f_\theta(x^{(i)}))^2$

Log loss: $-y^{(i)} log\left((f_\theta(x)\right) - (1 - y^{(i)})\, log\left(1 - (f_\theta(x)\right)$

线性回归
Linear Regression

- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\ldots,N}$$

let the machine learn a function from data to label

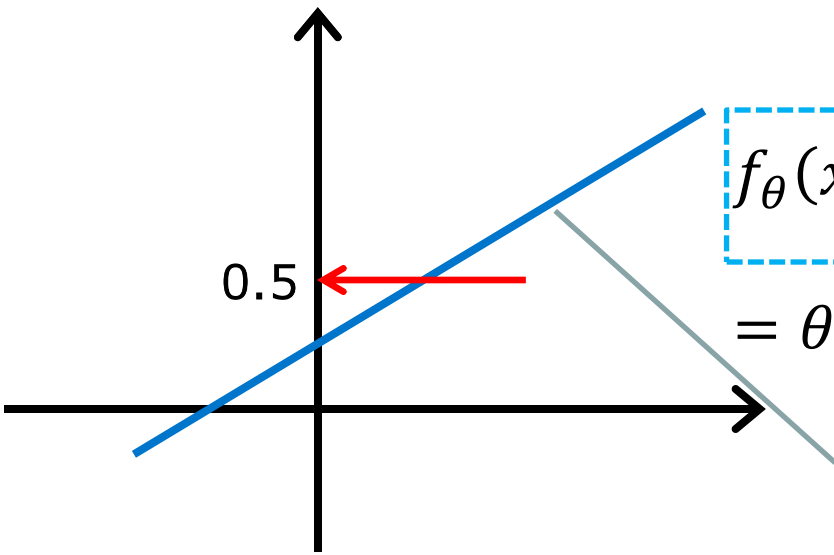
$$y^{(i)} \approx \boxed{f_\theta(x^{(i)})}$$

$$\boxed{f_\theta(x) = \theta_1\, x_1 + \theta_2\, x_2 + \cdots + \theta_n\, x_n + \theta_0}$$

- Function set $\{f_\theta(x^{(i)})\}$ is called hypothesis space
- Learning is referred to as updating the parameter θ to make the prediction closed to the corresponding label

如何用于分类
Classification task

$y \in \{0, 1\}$ 0: “Negative Class” (如，坏瓜)
1: “Positive Class” (如, 好瓜)



$$f_\theta(x) = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n + \theta_0$$

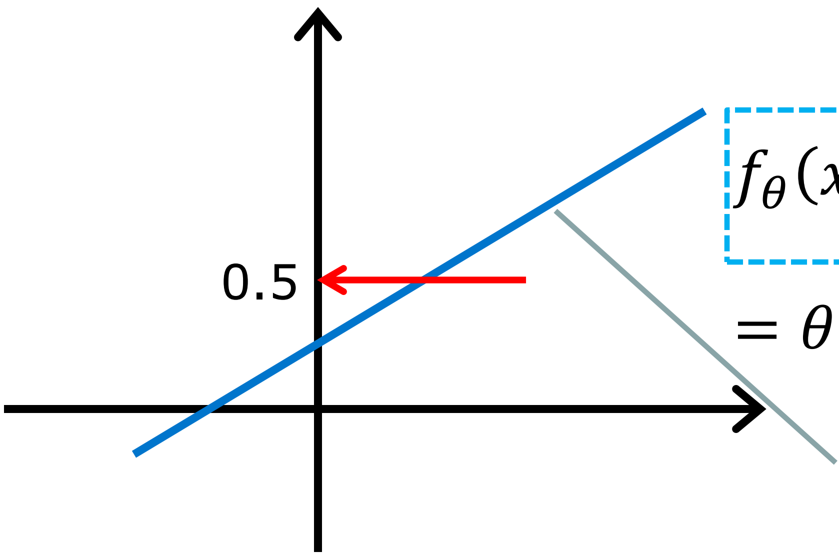
$$= \theta^T x$$

If $\theta^T x \geq 0.5$, predict “y = 1”

If $\theta^T x < 0.5$, predict “y = 0”

如何用于分类
Classification task

$y \in \{0, 1\}$ 0: "Negative Class" (如，坏瓜)
1: "Positive Class" (如, 好瓜)



$$f_\theta(x) = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n + \theta_0$$

$$= \theta^T x$$

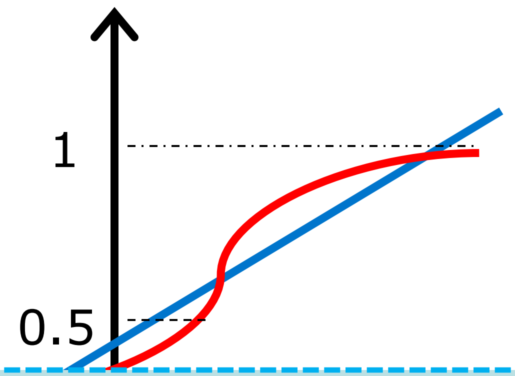
If $\theta^T x \geq 0.5$, predict "y = 1"

If $\theta^T x < 0.5$, predict "y = 0"

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ 0 & f(x) < 0 \end{cases}$$

逻辑斯蒂回归
Logistic regression

$y \in \{0, 1\}$ 0: "Negative Class" (如，坏瓜)
1: "Positive Class" (如, 好瓜)



$$f_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

$$P(y = 1/x) = \frac{1}{1 + e^{-\theta^T x}}$$

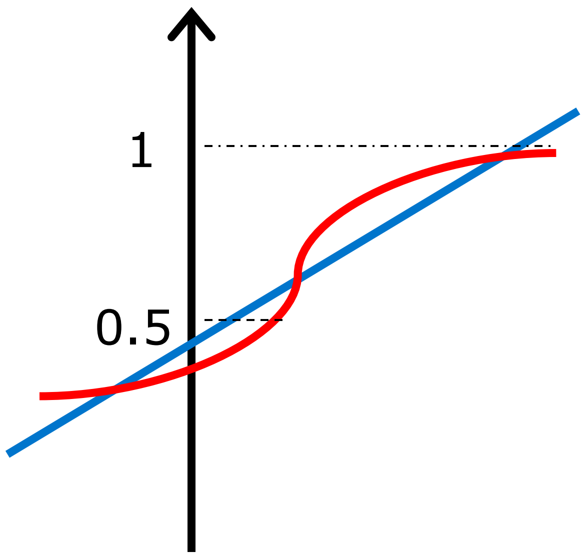
= estimated probability that y = 1, given x, parameterized by θ

$$P(y = 0/x) = \frac{e^{-\theta^T x}}{1 + e^{-\theta^T x}}$$

= estimated probability that y = 0, given x, parameterized by θ

逻辑斯蒂回归
Logistic regression

$y \in \{0, 1\}$ 0: "Negative Class" (如，坏瓜)
1: "Positive Class" (如, 好瓜)



$$f_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

= estimated probability that y = 1, given x, parameterized by θ

= estimated probability that y = 0, given x, parameterized by θ

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ 0 & f(x) < 0 \end{cases}$$

逻辑斯蒂回归
Logistic regression

- Given the training dataset of (data,label) pairs,

$$D = \{(x^{(i)}, y^{(i)})\}_{i=1.2,\dots,N}$$

let the machine learn a function from data to label

$$y^{(i)} \approx \boxed{f_\theta(x^{(i)})} \Rightarrow \boxed{f_\theta(x) = \frac{1}{1+e^{-\theta^T x}}}$$

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ 0 & f(x) < 0 \end{cases}$$

Cross-entropy loss：

$$-y^{(i)} log\left((f_\theta(x)\right) - (1-y^{(i)})\, log\left(1-(f_\theta(x)\right)$$

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ 0 & f(x) < 0 \end{cases}$$

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

- Step 2: Cost function

$$C(f) = \sum_n \delta\left(g\left(x^{(i)}\right) \neq y^{(i)}\right)$$

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

- Step 2: Cost function

$$C(f) = \sum_{n} \delta\left(g\left(x^{(i)}\right) \neq y^{(i)}\right))$$

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

- Step 2: Cost function

$$C(f) = \sum_{n} \delta\left(g\left(x^{(i)}\right) \neq y^{(i)}\right))$$

The number of times get incorrect results on training data.

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

- Step 2: Cost function

$$C(f) = \sum_{n} \delta\left(g\left(x^{(i)}\right) \neq y^{(i)}\right))$$

- Step 3： Training by gradient descent is difficult

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases}$$

- Step 2: Cost function

$$C(f) = \sum_n L\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

- Step 3： Training by gradient descent is ~~difficult~~

二分类
Binary Classification

- Step 1: Function (Model)

$$g(x) = \begin{cases} 1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases} \quad \Rightarrow \quad \begin{matrix} y^{(i)} f\left(x^{(i)}\right) \geq 0 \\ y^{(i)} f\left(x^{(i)}\right) < 0 \end{matrix}$$

- Step 2: Cost function

$$C(f) = \sum_{n} L\left(f\left(x^{(i)}\right), y^{(i)}\right))$$

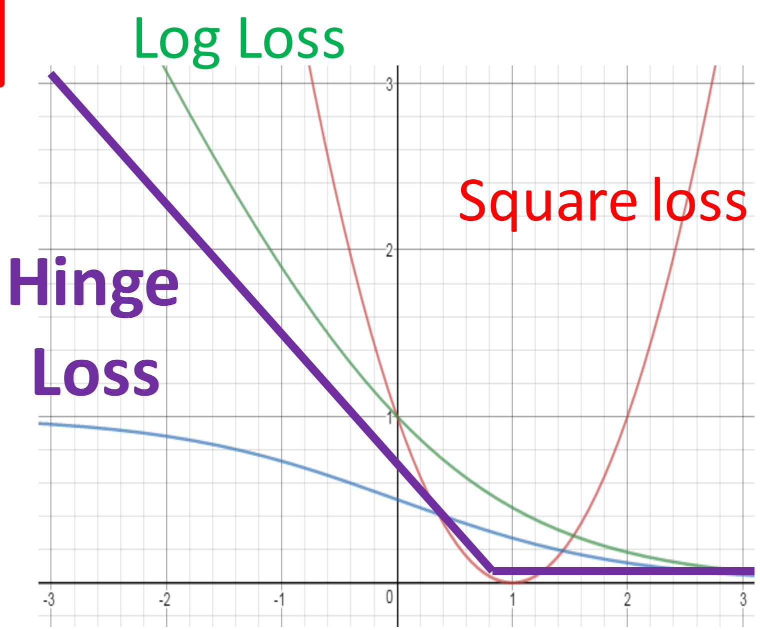
- Step 3： Training by gradient descent is ~~difficult~~

支持向量机
Support Vector Machine

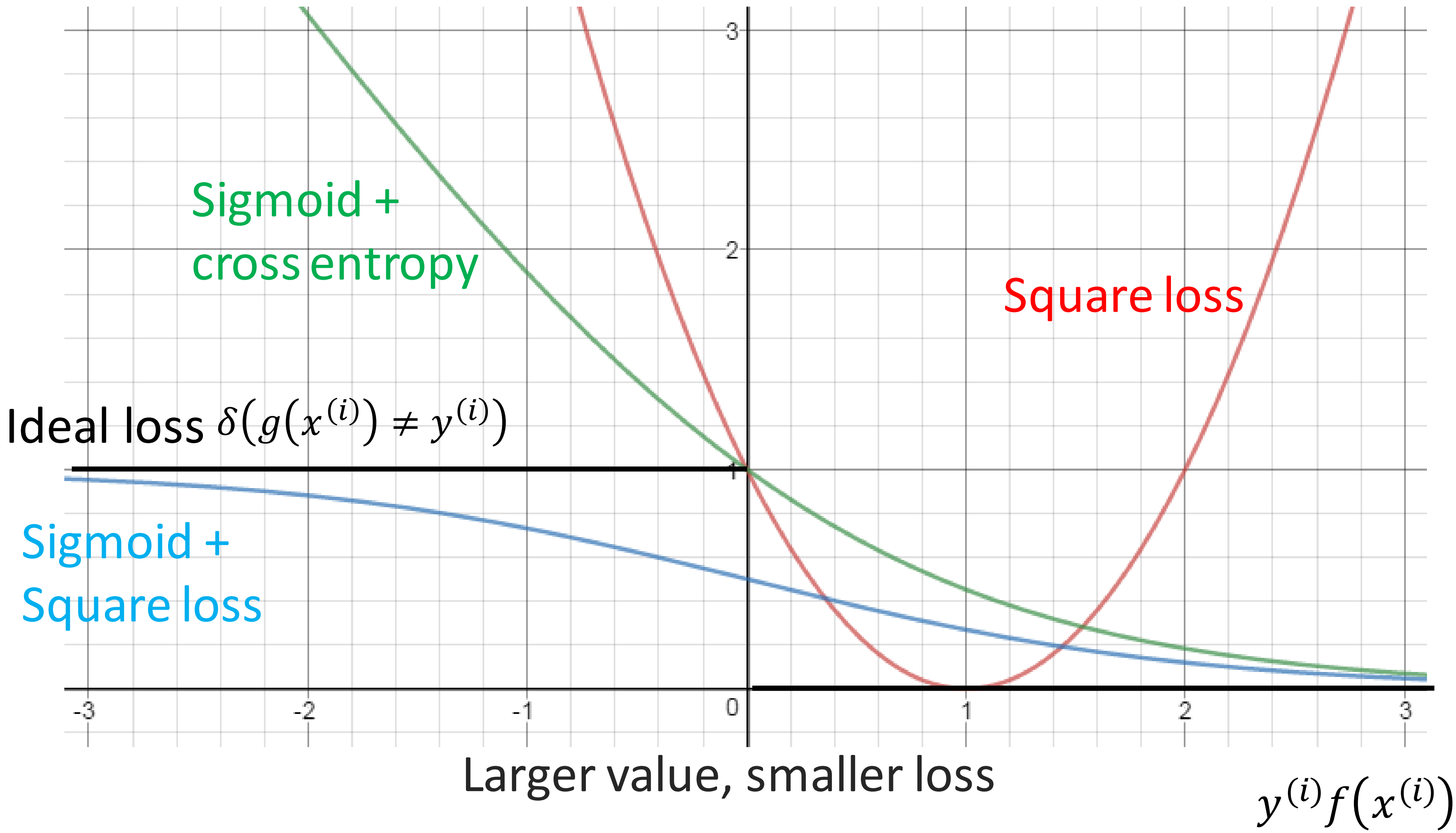
The slack variable ε indicating the extent to which the sample does not meet the constraint.

$$\boxed{\varepsilon^{\mathrm{i}} = max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))}$$

Hinge Loss function

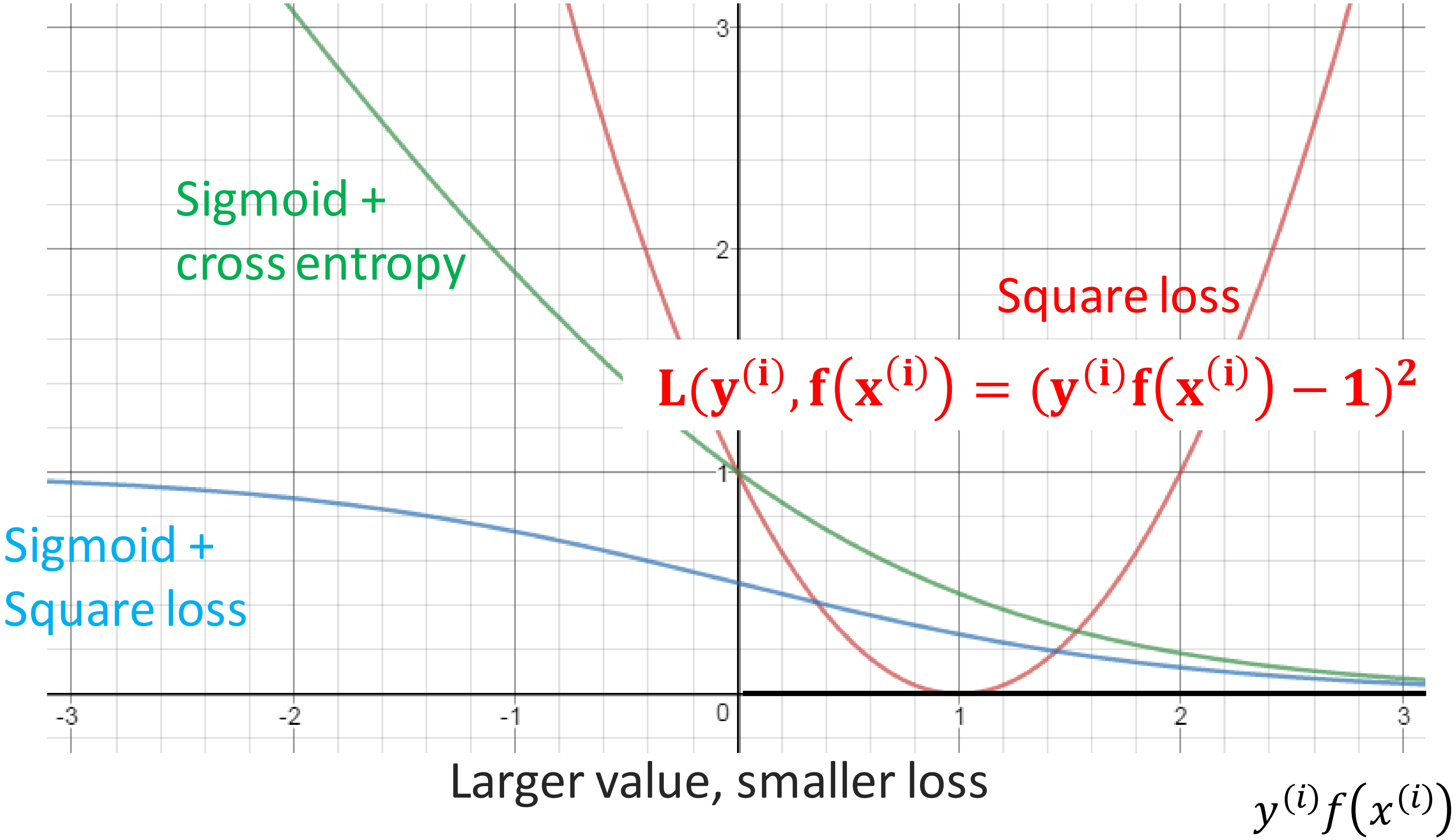


损失函数
Loss function



损失函数
Loss function

Square Loss:



损失函数
Loss function

Square Loss:

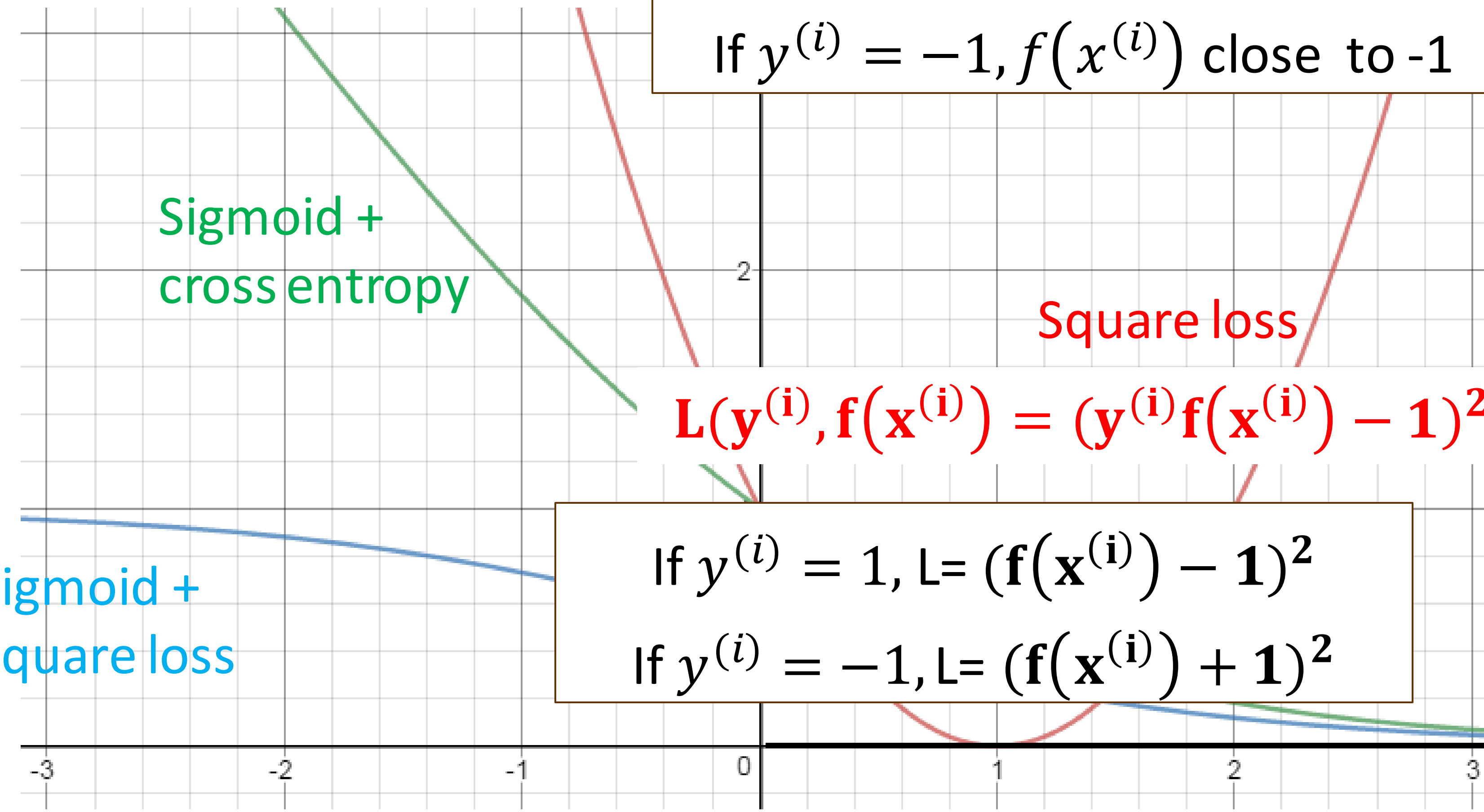
If $y^{(i)} = 1, f\left(x^{(i)}\right)$ close to 1

If $y^{(i)} = -1, f\left(x^{(i)}\right)$ close to -1

$$\mathbf{L}\left(\mathbf{y}^{(\mathbf{i})}, \mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right)\right) = \left(\mathbf{y}^{(\mathbf{i})}\mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right) - \mathbf{1}\right)^{\mathbf{2}}$$

If $y^{(i)} = 1$, L= $\left(\mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right) - \mathbf{1}\right)^{\mathbf{2}}$

If $y^{(i)} = -1$, L= $\left(\mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right) + \mathbf{1}\right)^{\mathbf{2}}$

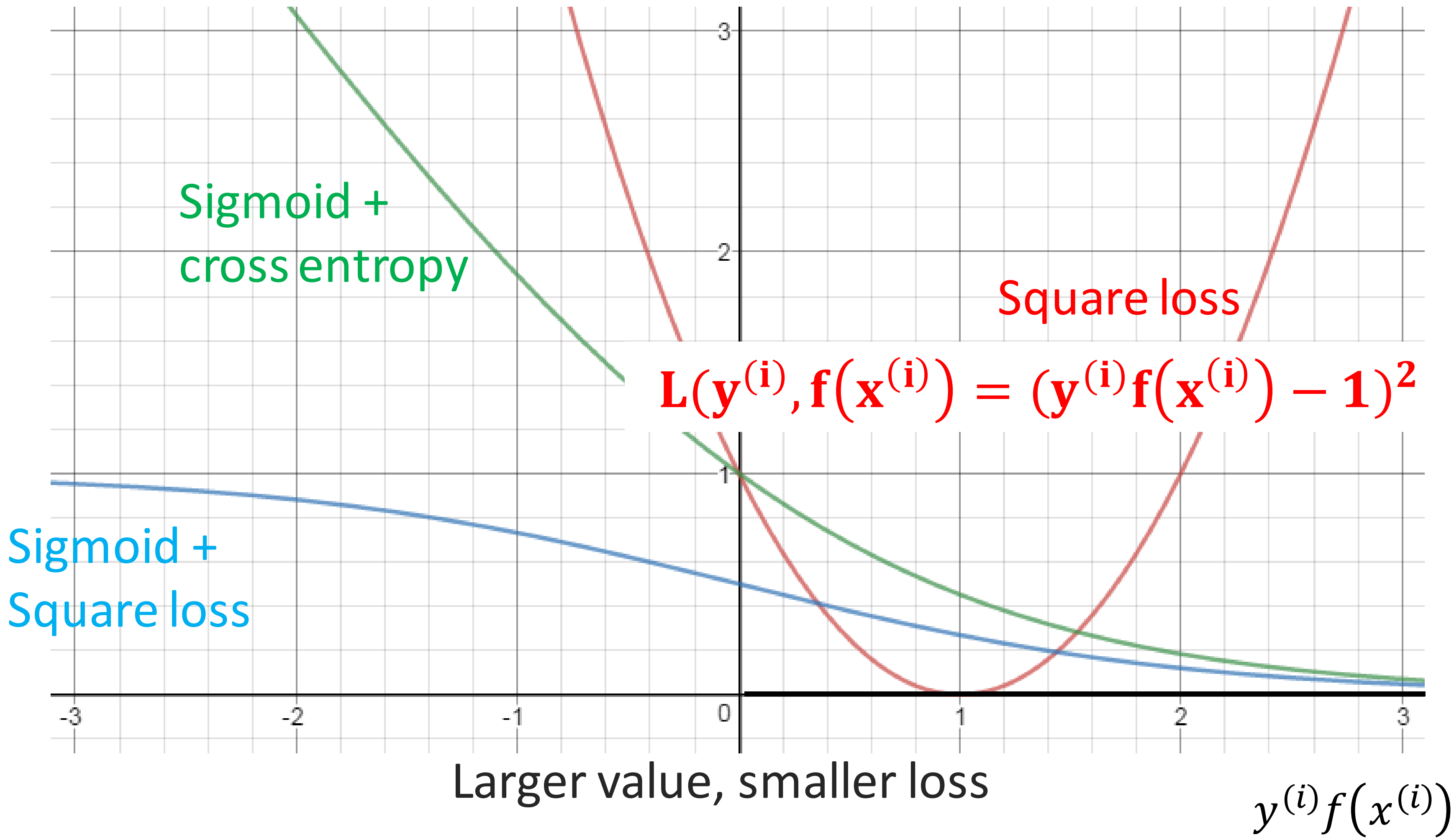


Larger value, smaller loss

$y^{(i)} f\left(x^{(i)}\right)$

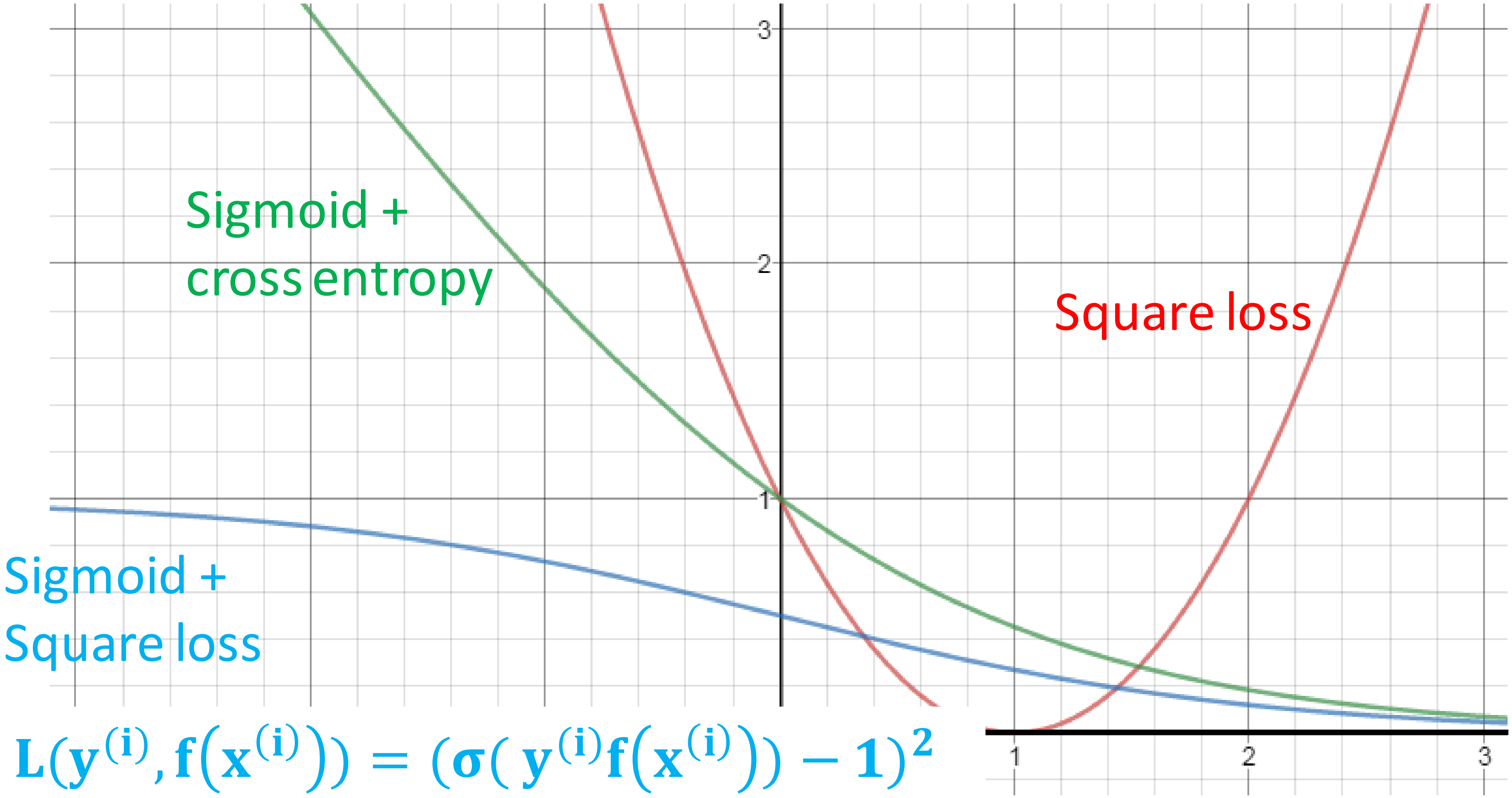
损失函数
Loss function

Square Loss:



损失函数
Loss function

Square Loss:



$$\mathbf{L(y^{(i)}, f(x^{(i)})) = (\sigma(y^{(i)} f(x^{(i)})) - 1)^2}$$

Larger value, smaller loss

$y^{(i)} f\left(x^{(i)}\right)$

损失函数
Loss function

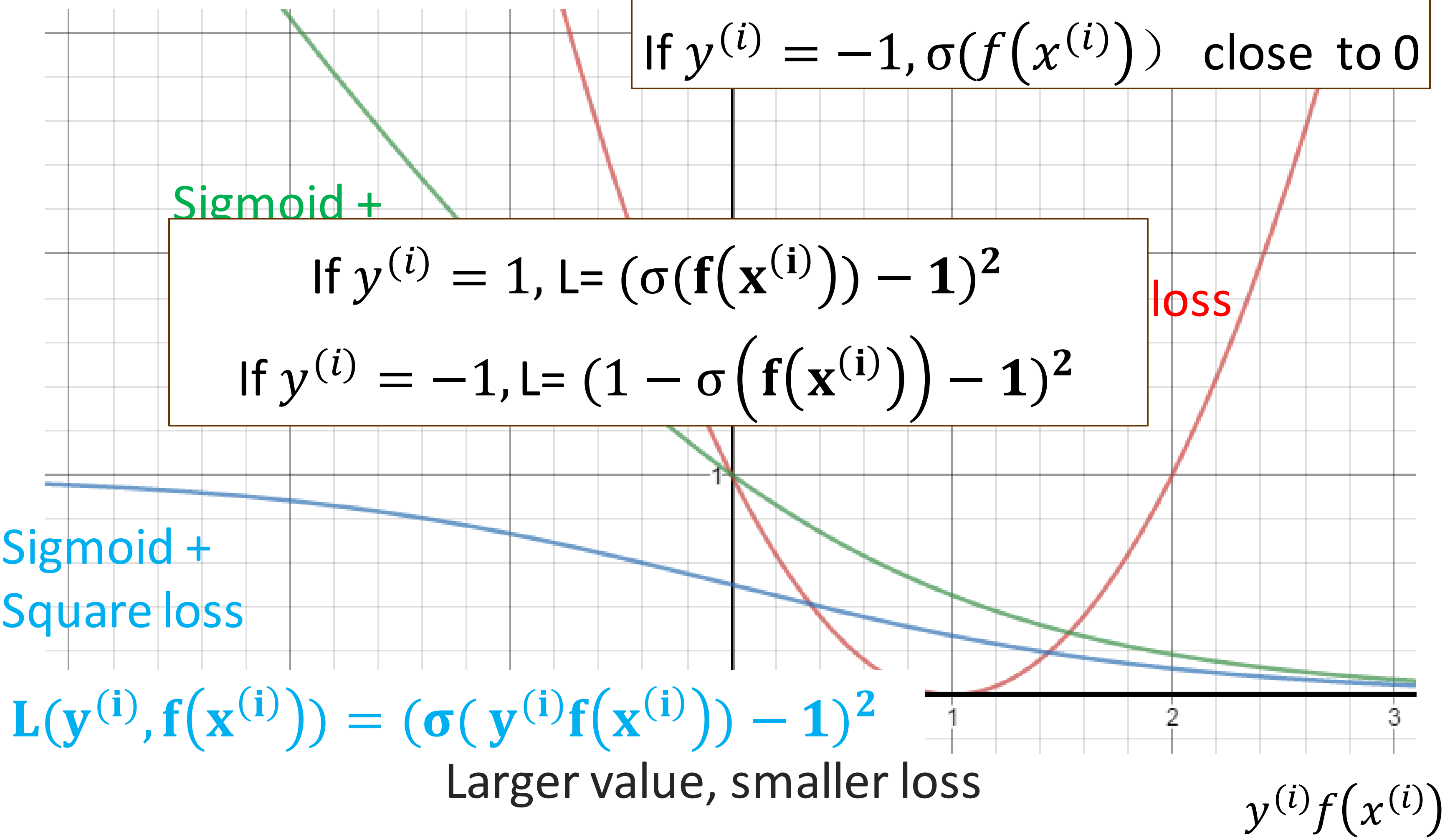
Square Loss:

If $y^{(i)}=1, \sigma(f(x^{(i)}))$ close to 1

If $y^{(i)}=-1, \sigma(f(x^{(i)}))$ close to 0

If $y^{(i)}=1$, L= $(\sigma(\mathbf{f}(\mathbf{x}^{(\mathbf{i})}))-\mathbf{1})^2$

If $y^{(i)}=-1$, L= $(1-\sigma\left(\mathbf{f}(\mathbf{x}^{(\mathbf{i})})\right)-\mathbf{1})^2$



$$\mathbf{L}(\mathbf{y}^{(\mathbf{i})},\mathbf{f}(\mathbf{x}^{(\mathbf{i})}))=(\boldsymbol{\sigma}(\mathbf{y}^{(\mathbf{i})}\mathbf{f}(\mathbf{x}^{(\mathbf{i})}))-\mathbf{1})^2$$

Larger value, smaller loss

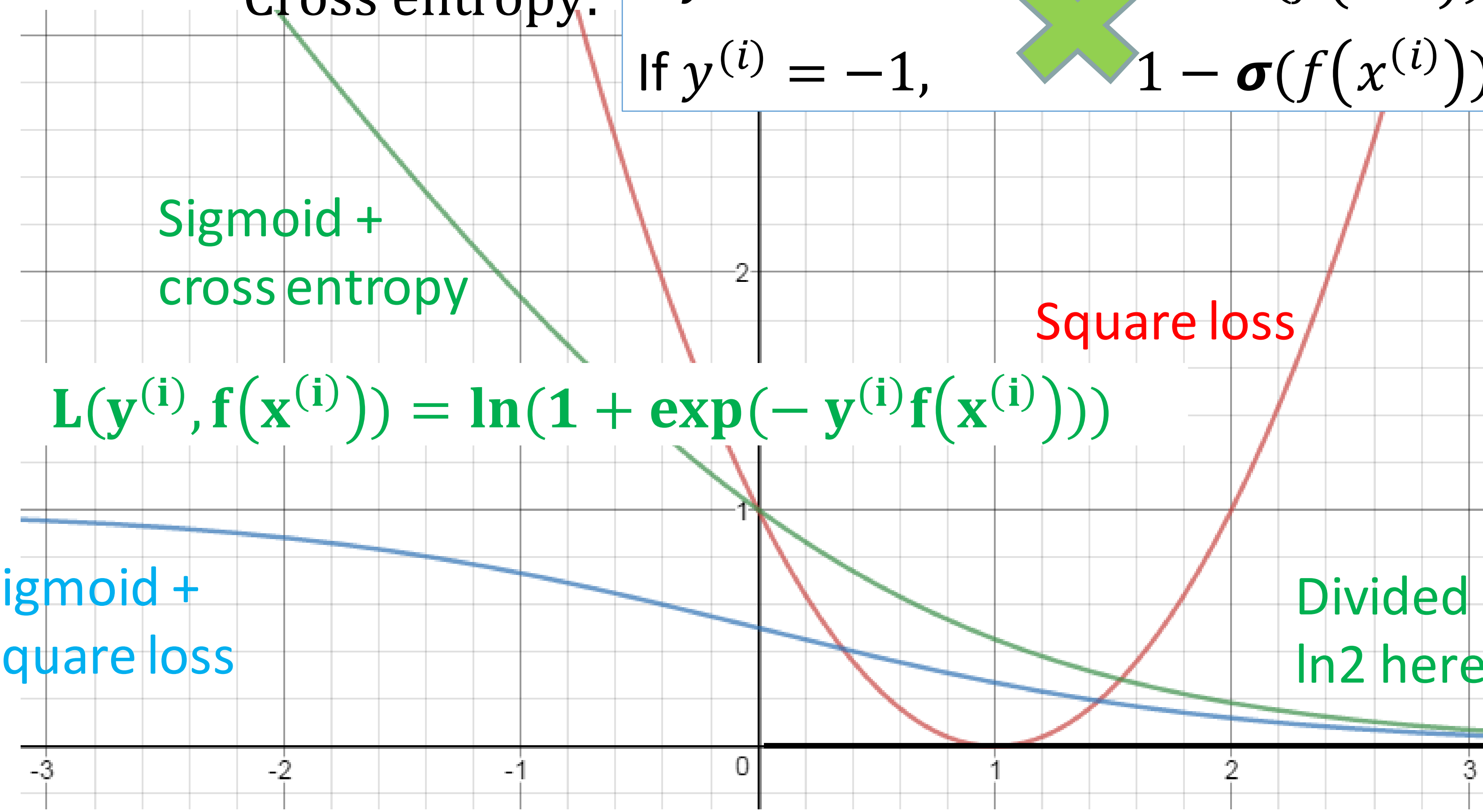
损失函数
Loss function

Cross entropy:

If $y^{(i)} = 1$, entropy ✖ $\boldsymbol{\sigma}\left(f\left(x^{(i)}\right)\right)$

If $y^{(i)} = -1$, $1 - \boldsymbol{\sigma}\left(f\left(x^{(i)}\right)\right)$

$$\mathbf{L}\left(\mathbf{y}^{(\mathbf{i})}, \mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right)\right) = \mathbf{ln}\left(\mathbf{1} + \mathbf{exp}\left(-\,\mathbf{y}^{(\mathbf{i})}\mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right)\right)\right)$$



Larger value, smaller loss

$y^{(i)}\left(f\left(x^{(i)}\right)\right)$

损失函数
Loss function

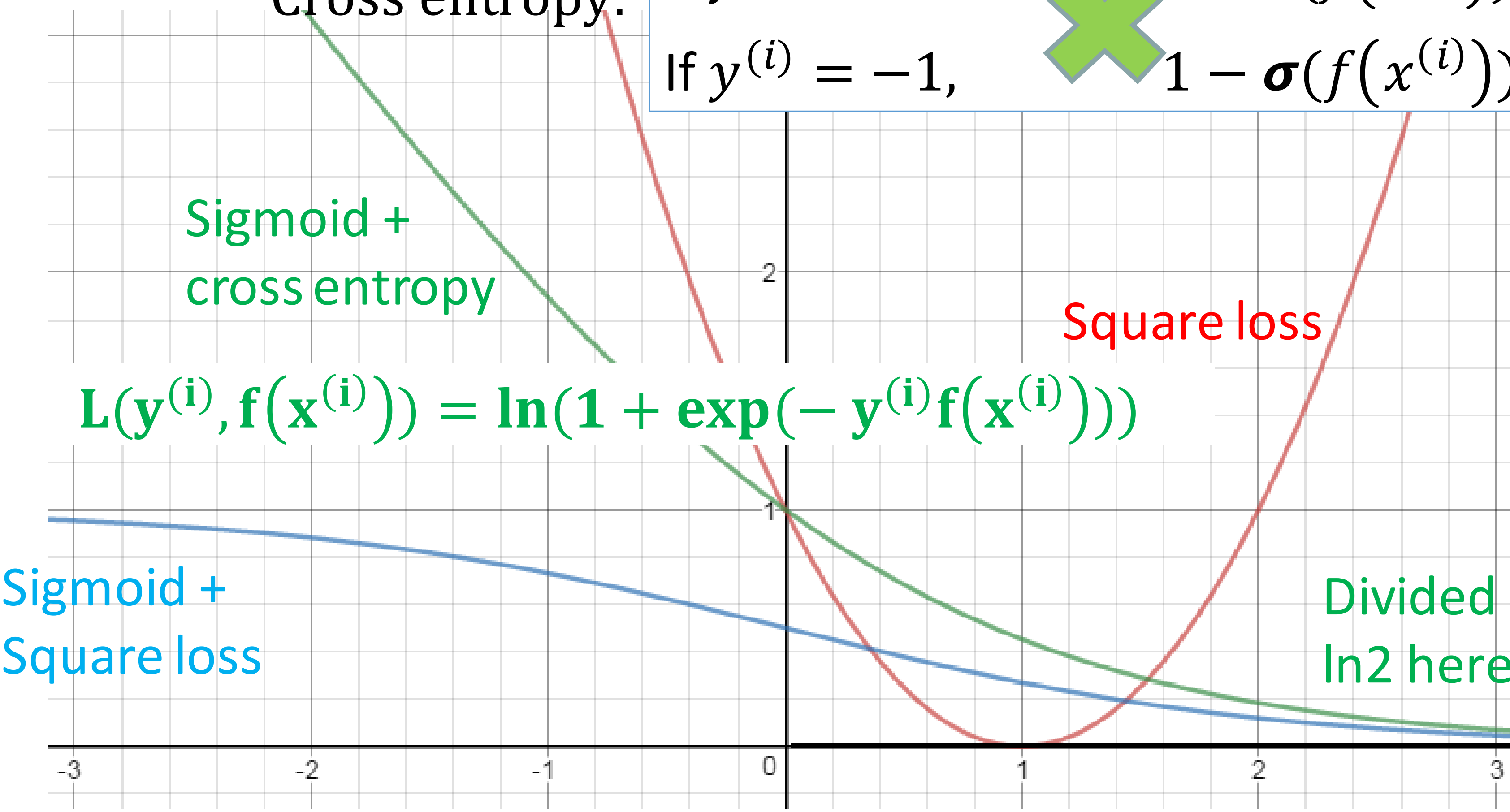
Cross entropy:

If $y^{(i)} = 1$, $\boldsymbol{\sigma}\left(f\left(x^{(i)}\right)\right)$

If $y^{(i)} = -1$, $1 - \boldsymbol{\sigma}\left(f\left(x^{(i)}\right)\right)$

entropy

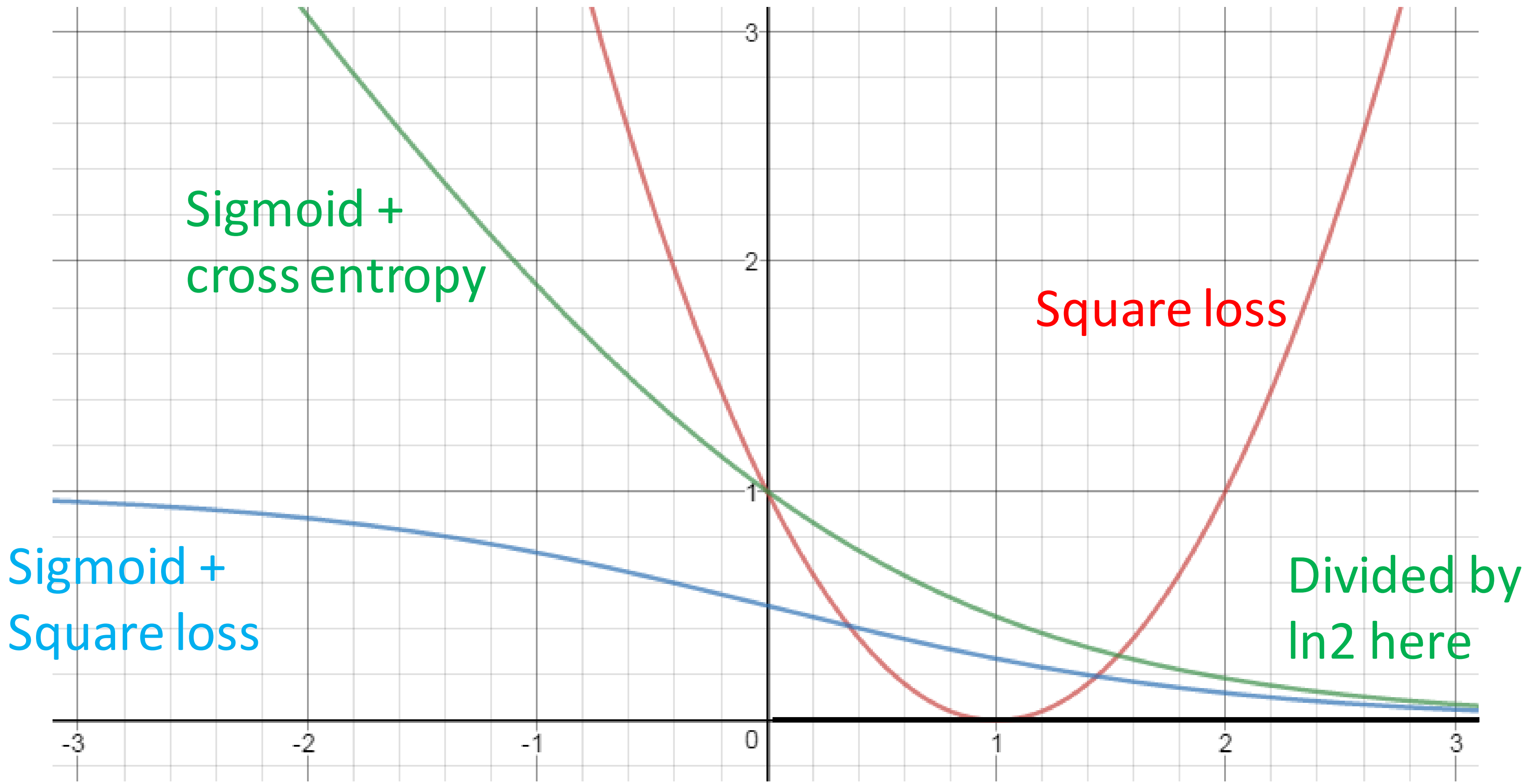
$$\mathbf{L}\left(\mathbf{y}^{(\mathbf{i})}, \mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right)\right) = \mathbf{ln}\left(\mathbf{1} + \mathbf{exp}\left(-\,\mathbf{y}^{(\mathbf{i})}\mathbf{f}\left(\mathbf{x}^{(\mathbf{i})}\right)\right)\right)$$



Larger value, smaller loss

$y^{(i)}\left(f\left(x^{(i)}\right)\right)$

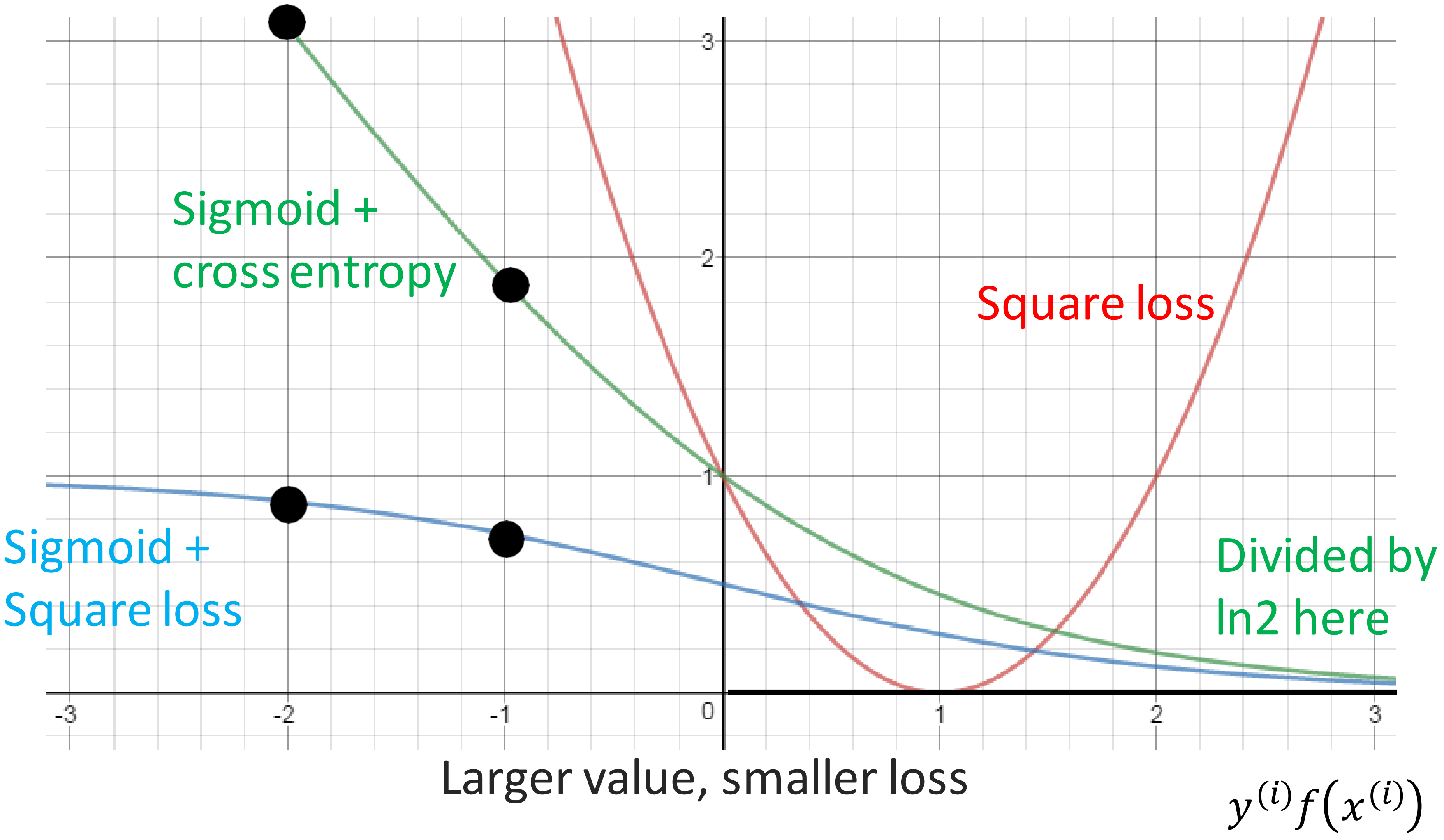
损失函数
Loss function



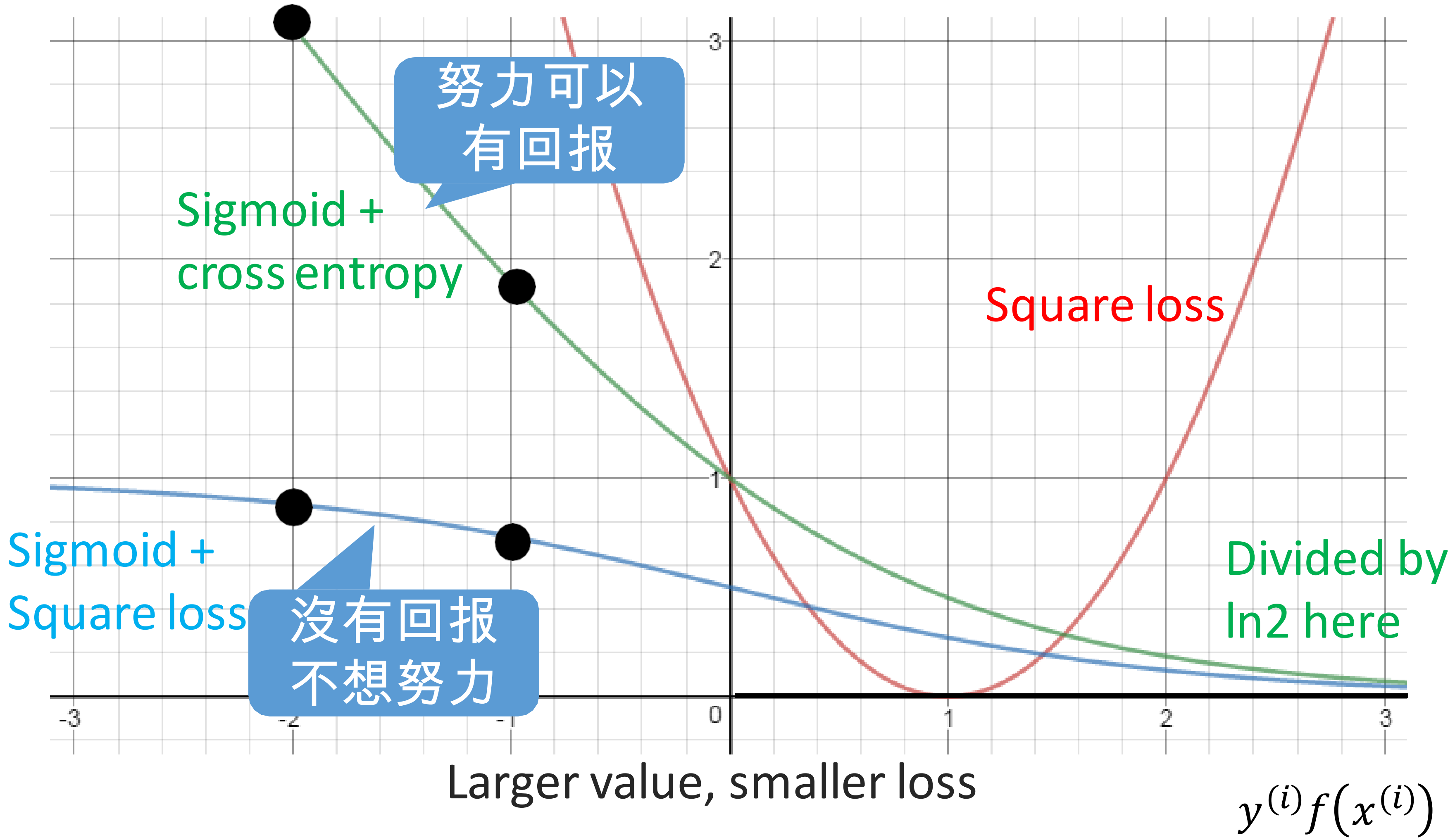
Larger value, smaller loss

$y^{(i)} f\left(x^{(i)}\right)$

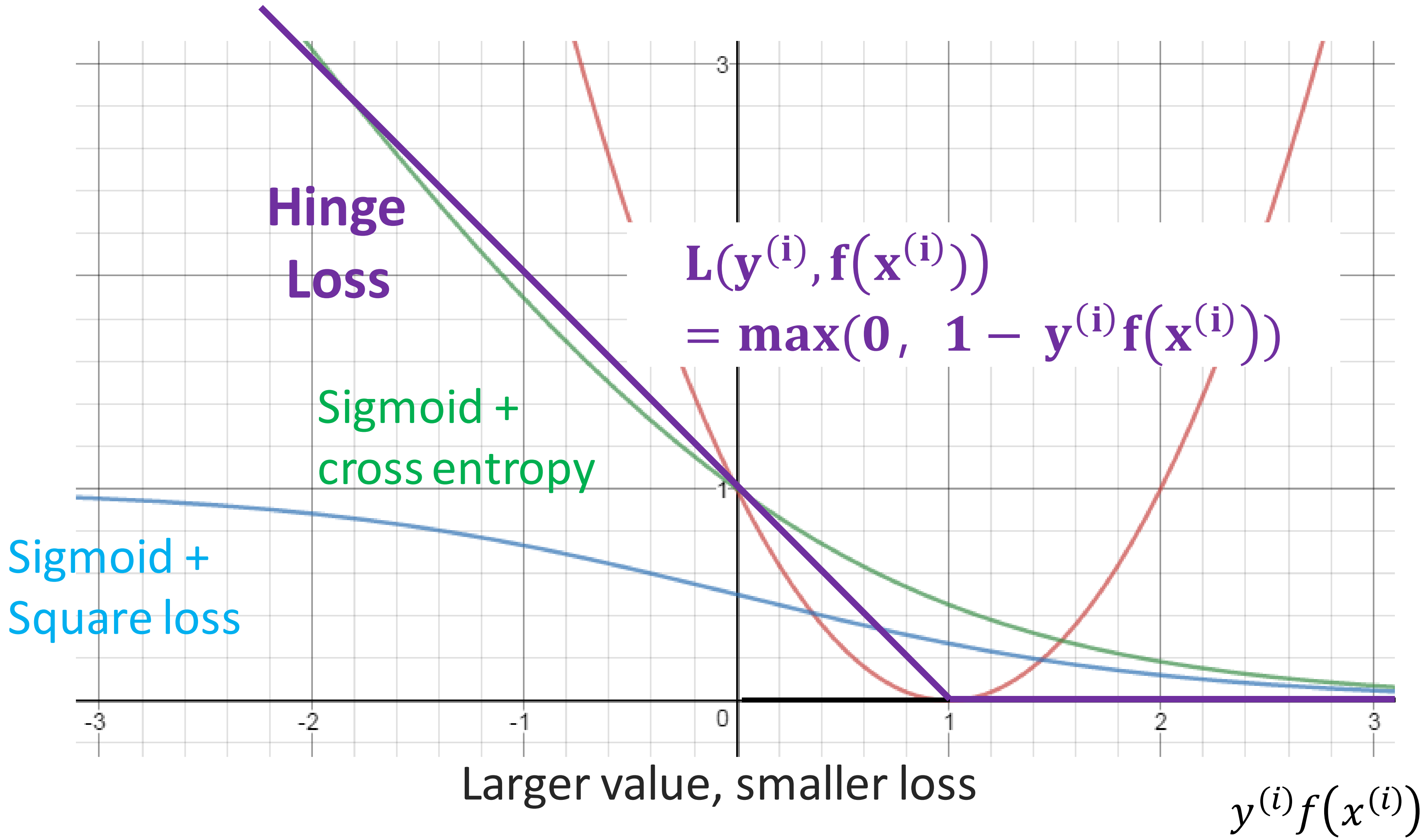
损失函数
Loss function



损失函数
Loss function



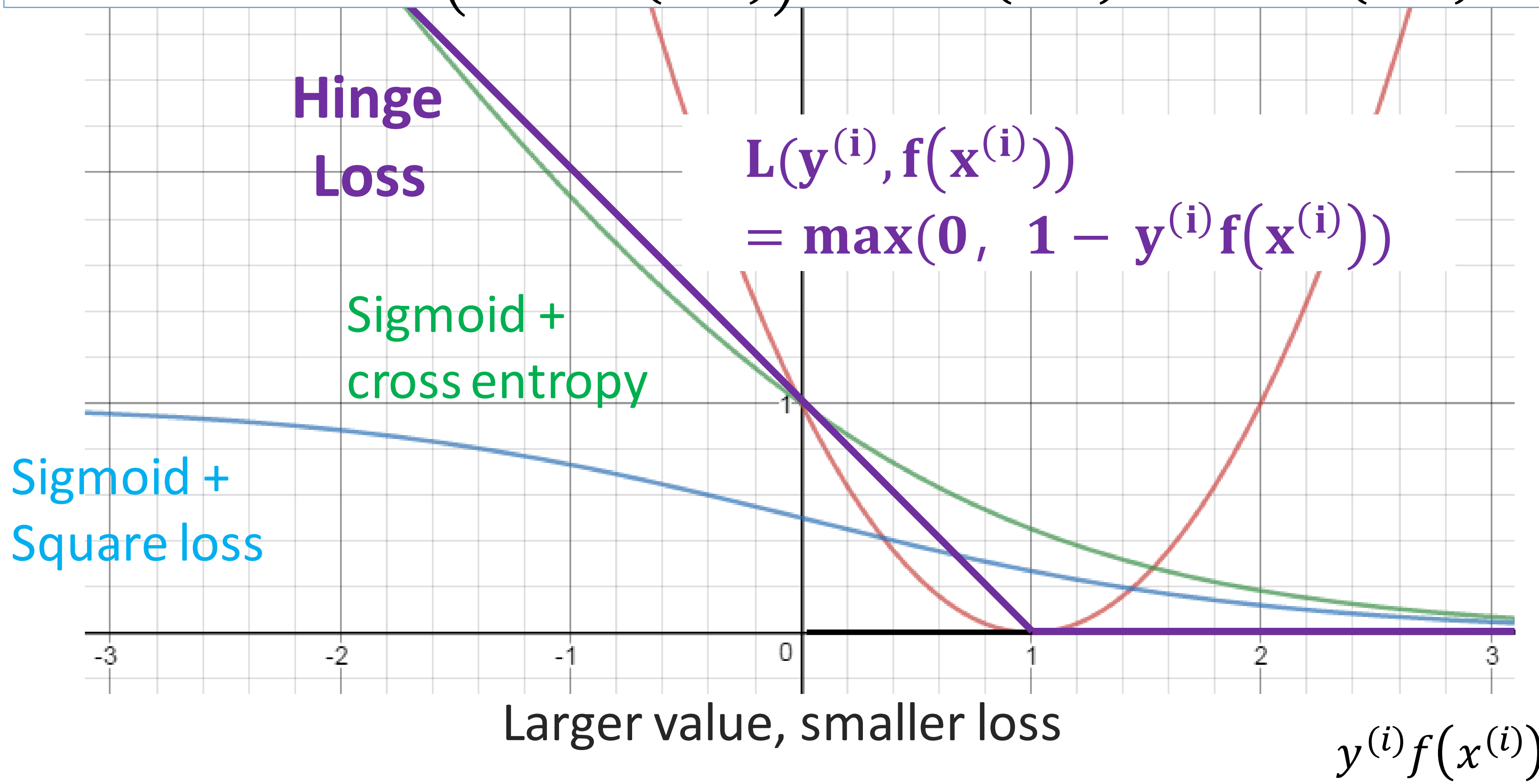
损失函数
Loss function



损失函数
Loss function

If $y^{(i)} = 1, \ \max\left(0, 1 - f\left(x^{(i)}\right)\right) \quad 1 - f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) > 1$

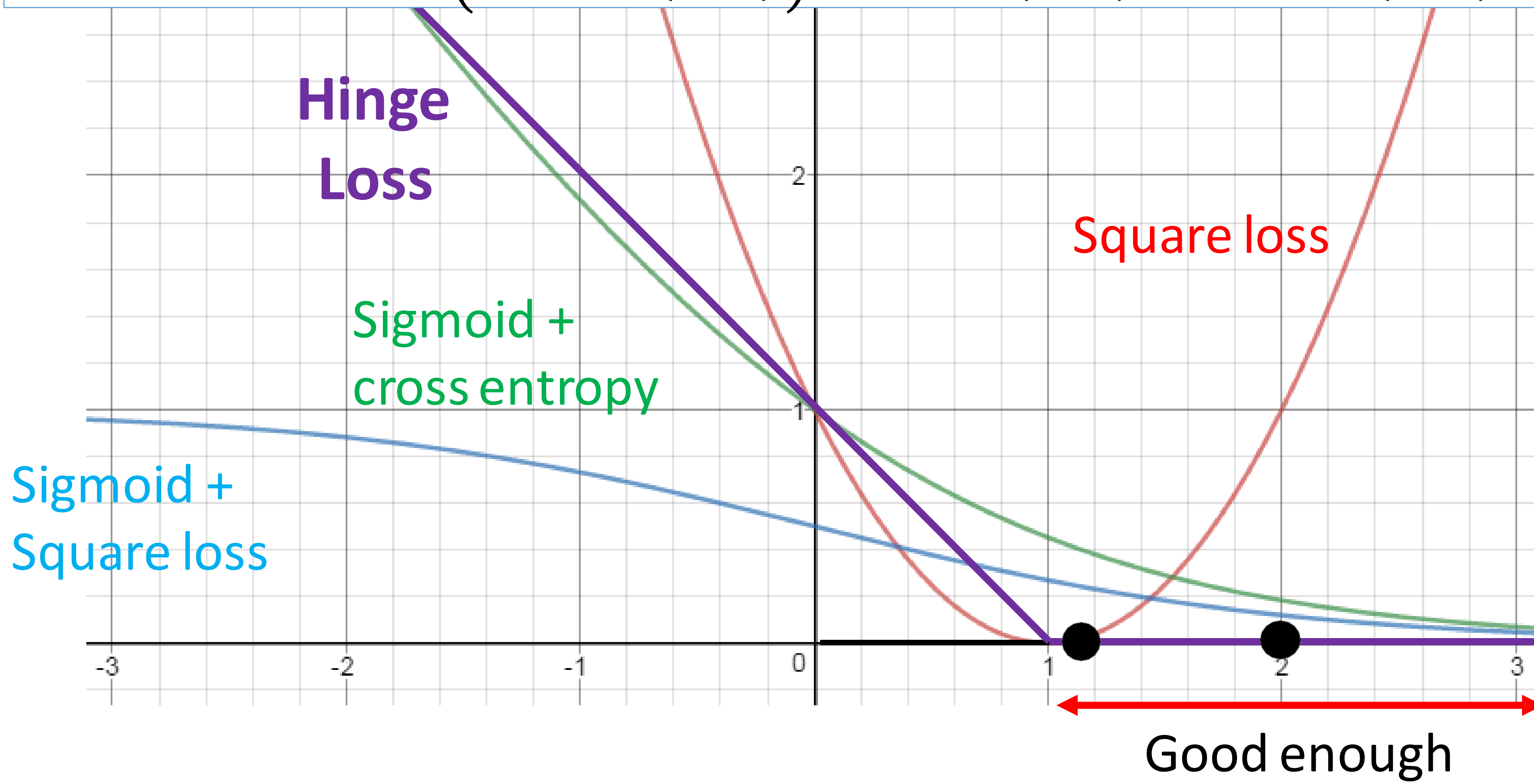
If $y^{(i)} = -1, \max\left(0, 1 + f\left(x^{(i)}\right)\right) \quad 1 + f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) < -1$



损失函数
Loss function

If $y^{(i)} = 1, \ \max\left(0, 1 - f\left(x^{(i)}\right)\right)$ $\quad 1 - f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) > 1$

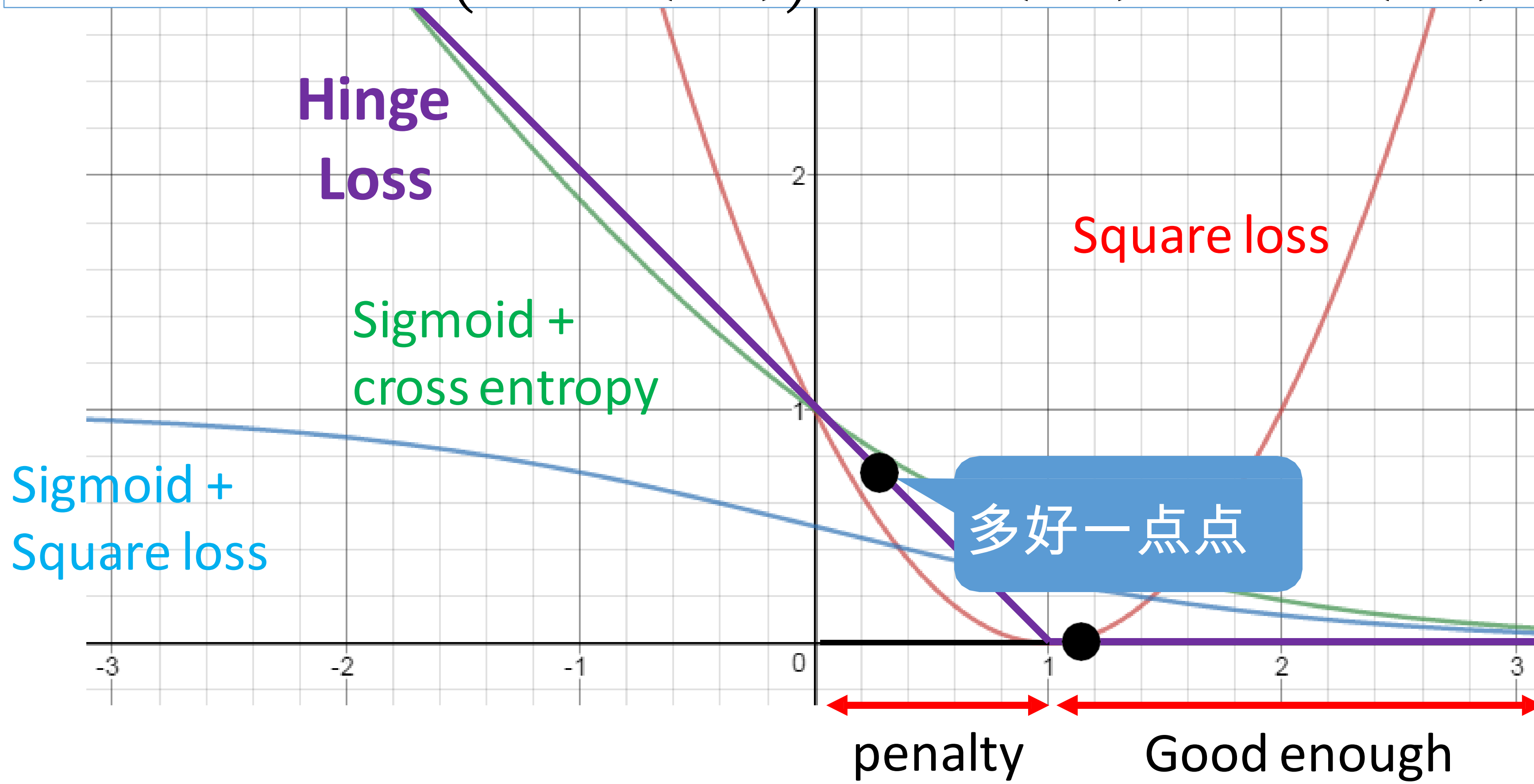
If $y^{(i)} = -1, \max\left(0, 1 + f\left(x^{(i)}\right)\right)$ $\quad 1 + f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) < -1$



损失函数
Loss function

If $y^{(i)} = 1,\ \ \max\left(0, 1 - f\left(x^{(i)}\right)\right)$ $\quad 1 - f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) > 1$

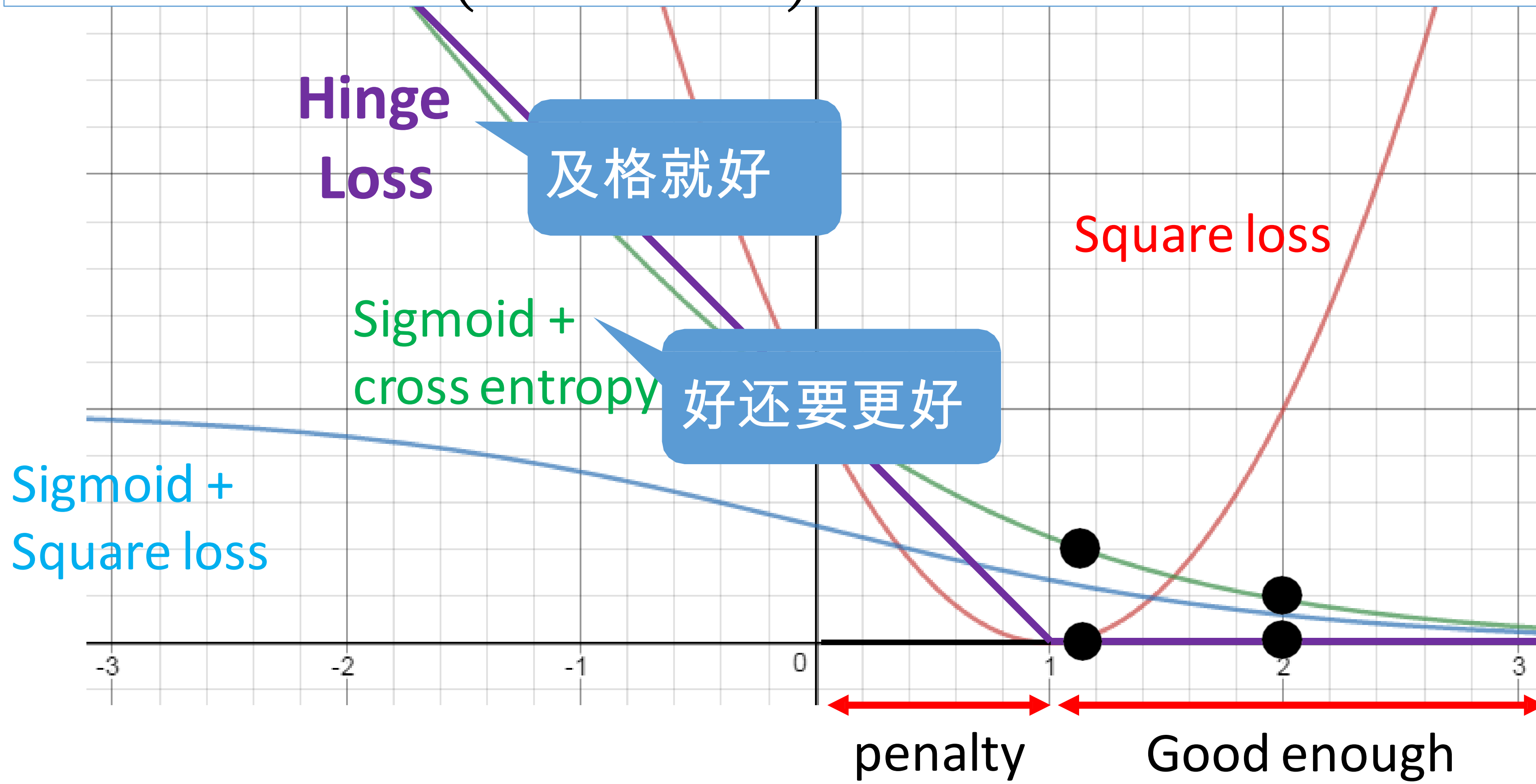
If $y^{(i)} = -1, \max\left(0, 1 + f\left(x^{(i)}\right)\right)$ $\quad 1 + f\left(x^{(i)}\right) < 0 \quad f\left(x^{(i)}\right) < -1$



损失函数
Loss function

If $y^{(i)} = 1, \;\; \max\left(0, 1 - f\left(x^{(i)}\right)\right) \quad 1 - f\left(x^{(i)}\right) < 0 \qquad f\left(x^{(i)}\right) > 1$

If $y^{(i)} = -1, \max\left(0, 1 + f\left(x^{(i)}\right)\right) \quad 1 + f\left(x^{(i)}\right) < 0 \qquad f\left(x^{(i)}\right) < -1$



二分类
Binary Classification

- Step 1: Function (Model)

$$f(x) = \sum_j w_j x_j + b$$

- Step 2: Cost function

$$C(f) = \sum_n L\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

线性SVM
Linear SVM

- Step 1: Function (Model)

New w

$$f(x) = \sum_j w_j x_j + b = \boxed{\begin{bmatrix} w \\ b \end{bmatrix}} \cdot \boxed{\begin{bmatrix} x \\ 1 \end{bmatrix}} = w^T x$$

New x

- Step 2: Cost function

$$= \sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right) + \lambda \|w\|_2$$

$$L(y^{(i)}, f\left(x^{(i)}\right) = max(0,1 - \ y^{(i)} f\left(x^{(i)}\right))$$

Hinge loss

线性SVM
Linear SVM

- Step 1: Function (Model)

$$f(x) = \sum_j w_j x_j + b = \boxed{\begin{bmatrix} w \\ b \end{bmatrix}} \cdot \boxed{\begin{bmatrix} x \\ 1 \end{bmatrix}}$$

- Step 2: Cost function

$$= \sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right) + \lambda \|w\|_2$$

convex

$$L(y^{(i)}, f\left(x^{(i)}\right) = max(0,1 - \ y^{(i)} f\left(x^{(i)}\right))$$

- Step 3: gradient descent?

线性SVM
Linear SVM

- Step 1: Function (Model)

$$f(x) = \sum_j w_j\, x_j + b \;= \boxed{\begin{bmatrix} w \\ b \end{bmatrix}} \cdot \boxed{\begin{bmatrix} x \\ 1 \end{bmatrix}}$$

- Step 2: Cost function

$$= \sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right) + \lambda \|w\|_2$$

$$L(y^{(i)}, f\left(x^{(i)}\right) = max(0, 1 - \; y^{(i)} f\left(x^{(i)}\right))$$

Compared with logistic regression,
linear SVM has different _(__________)_?

线性SVM
Linear SVM

- Step 1: Function (Model)

$$f(x) = \sum_j w_j\, x_j + b \; = \boxed{\begin{bmatrix} w \\ b \end{bmatrix}} \cdot \boxed{\begin{bmatrix} x \\ 1 \end{bmatrix}}$$

- Step 2: Cost function

$$= \sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right) + \lambda \|w\|_2$$

$$L(y^{(i)}, f\left(x^{(i)}\right) = max(0, 1 - \; y^{(i)} f\left(x^{(i)}\right))$$

Compared with logistic regression,
linear SVM has different ( ) ?

线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)}$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j} =}$$

线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)}$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right),y^{(i)}\right)}{\partial w_j} =} \frac{\partial L\left(f\left(x^{(i)}\right),y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \frac{\partial f\left(x^{(i)}\right)}{\partial w_j}$$

线性SVM
Linear SVM

Ignore regularization for simplicity

$$\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

$$L(y^{(i)}, f\left(x^{(i)}\right) = max(0, 1 - y^{(i)} f\left(x^{(i)}\right))$$

$$\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j} = \frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \underbrace{\frac{\partial f\left(x^{(i)}\right)}{\partial w_j}}_{x_j^i}$$

$$f\left(x^n\right) = w^T \cdot x^n$$

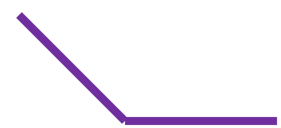
线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)} \qquad L(y^{(i)}, f\left(x^{(i)}\right) = max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j} =} \frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \underbrace{\frac{\partial f\left(x^{(i)}\right)}{\partial w_j}}_{x_j^i} \qquad \boxed{f\left(x^n\right) = w^T \cdot x^n}$$

$$\frac{\partial max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))}{\partial f\left(x^{(i)}\right)} = \begin{cases} -y^{(i)} & \text{If } \ y^{(i)} f\left(x^{(i)}\right) < 1 \\ 0 & \text{otherwise} \end{cases}$$



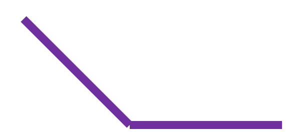
线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)} \qquad L\left(y^{(i)}, f\left(x^{(i)}\right)\right) = max\left(0, 1 - \ y^{(i)} f\left(x^{(i)}\right)\right)$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j}} = \frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \underbrace{\frac{\partial f\left(x^{(i)}\right)}{\partial w_j}}_{x_j^i} \qquad \boxed{f(x^n) = w^T \cdot x^n}$$

$$\frac{\partial max\left(0, 1 - \ y^{(i)} f\left(x^{(i)}\right)\right)}{\partial f\left(x^{(i)}\right)} = \begin{cases} -y^{(i)} & \text{If } \ y^{(i)} f\left(x^{(i)}\right) < 1 \\ 0 & \text{otherwise} \end{cases}$$



$$\frac{\partial \sum_{\mathrm{i}} L\left(y^{(i)}, f\left(x^{(i)}\right)\right)}{\partial w_j} = \sum_{\mathrm{i}} -\delta\left(y^{(i)} f\left(x^{(i)}\right) < 1\right) y^{(i)} \ x_j^i$$

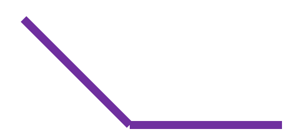
线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)} \qquad L\left(y^{(i)}, f\left(x^{(i)}\right)\right) = max\left(0, 1 - y^{(i)} f\left(x^{(i)}\right)\right)$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j} =} \frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \underbrace{\frac{\partial f\left(x^{(i)}\right)}{\partial w_j}}_{x_j^i} \qquad \boxed{\begin{array}{c} f(x^n) \\ = w^T \cdot x^n \end{array}}$$

$$\frac{\partial max\left(0, 1 - y^{(i)} f\left(x^{(i)}\right)\right)}{\partial f\left(x^{(i)}\right)} = \begin{cases} -y^{(i)} & \text{If } y^{(i)} f\left(x^{(i)}\right) < 1 \\ 0 & \text{otherwise} \end{cases}$$



$$\frac{\partial \sum_i L\left(y^{(i)}, f\left(x^{(i)}\right)\right)}{\partial w_j} = \sum_i \underbrace{-\delta\left(y^{(i)} f\left(x^{(i)}\right) < 1\right) y^{(i)}}_{c^n(W)} x_j^i$$

$$\boxed{w_j \leftarrow w_j - \eta \sum_i c^n(W)\, x_j^i}$$

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_{i} （max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))） \ + \lambda \|w\|_2$$

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_{i} （max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))） + \lambda \|w\|_2$$

$$\boxed{\varepsilon^{\mathrm{i}} = max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))}$$

ε^{i}: slack variable

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_i \varepsilon^{\mathrm{i}} + \lambda \|w\|_2$$

$$\boxed{\varepsilon^{\mathrm{i}} = max(0, 1 - \ y^{(i)} f\left(x^{(i)}\right))}$$

ε^{i}: slack variable

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_{i} \varepsilon^{i} + \lambda \|w\|_2$$

$$\boxed{\varepsilon^{i} = max(0, 1 - y^{(i)} f(x^{(i)}))}$$

ε^{i}: slack variable

$$\boxed{\begin{array}{c} \varepsilon^{i} \geq 0 \\ \varepsilon^{i} \geq 1 - y^{(i)} f(x^{(i)}) \end{array}}$$

$$\Rightarrow \quad y^{(i)} f(x^{(i)}) \geq 1 - \varepsilon^{i}$$

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_i \varepsilon^{\mathrm{i}} + \lambda \|w\|_2$$

$$\boxed{\varepsilon^{\mathrm{i}} = max(0, 1 - y^{(i)} f(x^{(i)}))}$$

$$y^{(i)}(w^T x_j + b) \geq 1 - \varepsilon^{\mathrm{i}}$$

ε^{i}: slack variable

$$\boxed{\begin{gathered} \varepsilon^{\mathrm{i}} \geq 0 \\ \varepsilon^{\mathrm{i}} \geq 1 - y^{(i)} f(x^{(i)}) \end{gathered}}$$

$$\Rightarrow \quad y^{(i)} f(x^{(i)}) \geq 1 - \varepsilon^{\mathrm{i}}$$

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_{i} \varepsilon^{\mathrm{i}} + \lambda \|w\|_2$$

$$s.t.\ y^{(i)}(w^T x_j + b) \geq 1 - \varepsilon^{\mathrm{i}}$$

ε^{i}: slack variable

线性SVM
Linear SVM

Minimizing total loss function L:

$$min \sum_i \varepsilon^{\mathrm{i}} + \lambda\|w\|_2$$

$$s.t.\ y^{(i)}(w^T x_j + b) \geq 1 - \varepsilon^{\mathrm{i}}$$

ε^{i}: slack variable

$$min \frac{1}{2}\|w\|_2 + C \sum_i \varepsilon^{\mathrm{i}}$$

$$s.t.\ y^{(i)}(w^T x_j + b) \geq 1 - \varepsilon^{\mathrm{i}}$$

支持向量
Support vectors

$$w^{(*)} = \sum_{\mathrm{n}} a_n^* x^n$$

$a^{(*)}$ may be sparse

➡

Linear combination of data points

$x^{(i)}$ with non-zero $a^{(*)}$ are support vectors

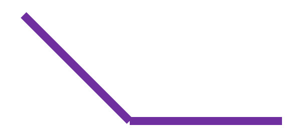
线性SVM
Linear SVM

Ignore regularization for simplicity

$$\boxed{\sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)} \qquad L\left(y^{(i)}, f\left(x^{(i)}\right)\right) = max\left(0, 1 - \ y^{(i)} f\left(x^{(i)}\right)\right)$$

$$\boxed{\frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial w_j} =} \frac{\partial L\left(f\left(x^{(i)}\right), y^{(i)}\right)}{\partial f\left(x^{(i)}\right)} \underbrace{\frac{\partial f\left(x^{(i)}\right)}{\partial w_j}}_{x_j^i} \qquad \boxed{\begin{aligned} &f\left(x^n\right) \\ &= w^T \cdot x^n \end{aligned}}$$

$$\frac{\partial max\left(0, 1 - \ y^{(i)} f\left(x^{(i)}\right)\right)}{\partial f\left(x^{(i)}\right)} = \begin{cases} -y^{(i)} & \text{If } \ y^{(i)} f\left(x^{(i)}\right) < 1 \\ 0 & \text{otherwise} \end{cases}$$



$$\frac{\partial \sum_i L\left(y^{(i)}, f\left(x^{(i)}\right)\right)}{\partial w_j} = \sum_i \underbrace{-\delta\left(\ y^{(i)} f\left(x^{(i)}\right) < 1\right) y^{(i)}}_{c^n(W)} \ x_j^i$$

$$\boxed{w_j \leftarrow w_j - \eta \sum_i c^n(W) \ x_j^i}$$

支持向量
Support vectors

$$w^{(*)} = \sum_{n} a_n^* x^n$$

$a^{(*)}$ may be sparse

➡

Linear combination of data points

$x^{(i)}$ with non-zero $a^{(*)}$ are support vectors

$$w_1 = w_1 - \textstyle\sum_i c^n(w)\, x_1^n$$

$$w_i = w_i - \textstyle\sum_i c^n(w)\, x_i^n$$

$$w_k = w_k - \sum_i c^n(w)\, x_k^n$$

线性SVM
Linear SVM

$$w^{(*)} = \sum_{\mathrm{n}} a_n^* x^n$$

$a^{(*)}$ may be sparse

Linear combination of data points

$x^{(i)}$ with non-zero $a^{(*)}$ are support vectors

$$w_1 = w_1 - \sum_i c^n(w)\, x_1^n$$

$$w_i = w_i - \sum_i c^n(w)\, x_i^n$$

$$w_k = w_k - \sum_i c^n(w)\, x_k^n$$

$$c^n(w) = \frac{\partial\, L(y^{(i)} f(x^{(i)}))}{\partial f(x^{(i)})}$$

Hinge loss: usually zero

线性SVM
Linear SVM

$$w^{(*)} = \sum_{\text{n}} a_n^* x^n$$

$a^{(*)}$ may be sparse

Linear combination of data points

$x^{(i)}$ with non-zero $a^{(*)}$ are support vectors

$$w_1 = w_1 - \sum_i c^n(w)\, x_1^n$$

$$w_i = w_i - \sum_i c^n(w)\, x_i^n$$

$$w_k = w_k - \sum_i c^n(w)\, x_k^n$$

If w initialized as 0

$$c^n(w) = \frac{\partial\, L(y^{(i)} f(x^{(i)}))}{\partial f(x^{(i)})}$$

Hinge loss:
usually zero

线性SVM
Linear SVM

$$w^{(*)} = \sum_{n} a_n^* x^n$$

$a^{(*)}$ may be sparse

Linear combination of data points

$x^{(i)}$ with non-zero $a^{(*)}$ are support vectors

$$w_1 = w_1 - \sum_i c^n(w)\, x_1^n$$

$$w_i = w_i - \sum_i c^n(w)\, x_i^n$$

$$w_k = w_k - \sum_i c^n(w)\, x_k^n$$

If w initialized as 0

c.f. for logistic regression,
it is always non-zero

$$c^n(w) = \frac{\partial\, L(y^{(i)} f(x^{(i)}))}{\partial f(x^{(i)})}$$

Hinge loss:
usually zero

逻辑斯蒂回归求解
Logistic regression solution

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{c} -y^{(i)} log\left((f_\theta(x^{(i)})\right) \\ -(1-y^{(i)})\ log\left(1-(f_\theta(x^{(i)})\right) \end{array}\right]$$

Want $\{ \min_\theta J(\theta)$:

Repeat

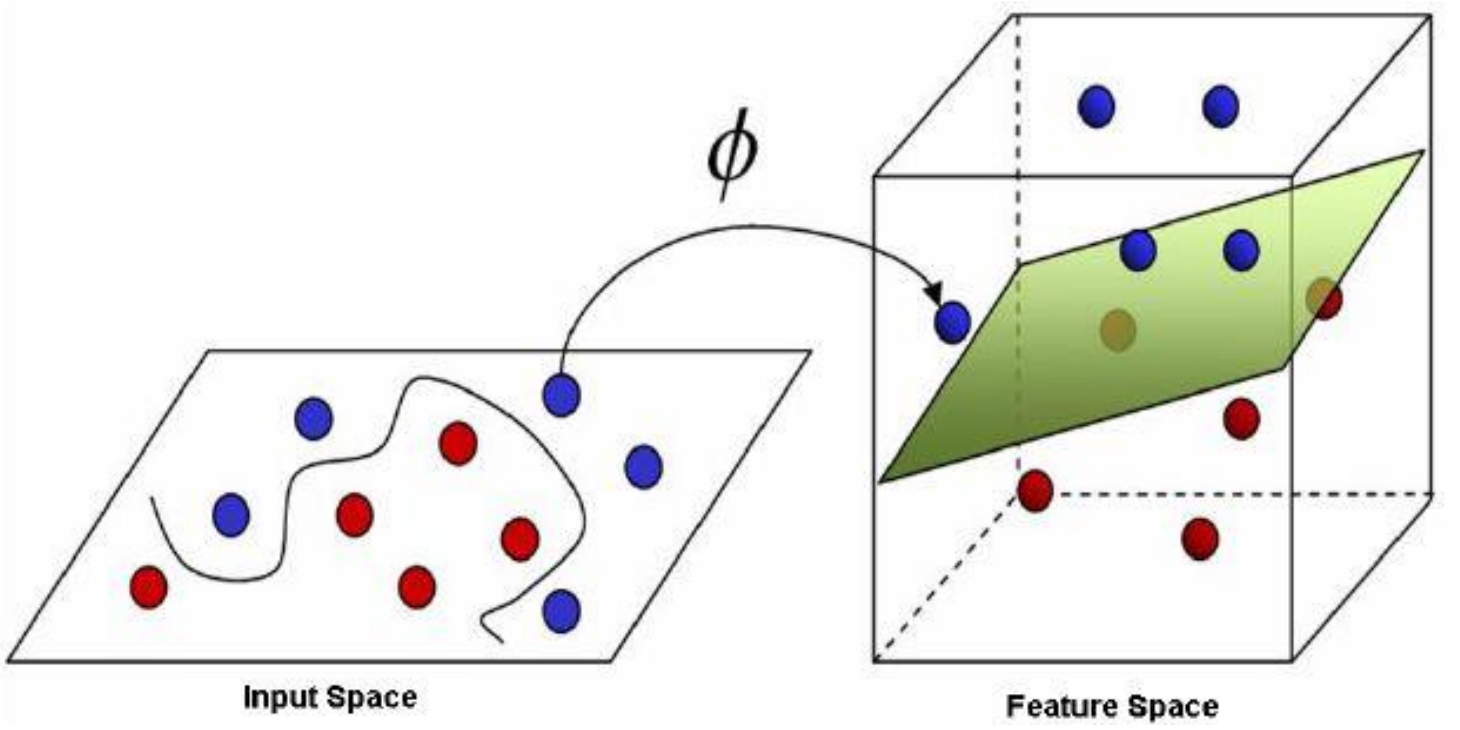
$$\theta_j := \theta_j - a\frac{1}{N}\sum_{i=1}^{N}(f_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

(simultaneously update all θ_j)

$\}$

Algorithm looks identical to linear regression!

核技巧
Kernel Trick



核技巧
Kernel Trick

$$w = \sum_n a_n x^n = Xa$$

$$X = \begin{bmatrix} x^1 & x^2 & \cdots\cdots & x^N \end{bmatrix}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$

核技巧
Kernel Trick

$$w = \sum_n a_n x^n = Xa$$

$$X = \begin{bmatrix} x^1 & x^2 & \cdots\cdots & x^N \end{bmatrix}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$

Step 1: $f(x) = w^T x$

核技巧
Kernel Trick

$$w = \sum_n a_n x^n = Xa$$

$$X = \begin{bmatrix} x^1 & x^2 & \cdots\cdots & x^N \end{bmatrix} \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$

Step 1: $f(x) = w^T x \xrightarrow{\quad w = X\boldsymbol{\alpha} \quad} f(x) = \boldsymbol{\alpha}^T X^T x$

核技巧
Kernel Trick

$$w = \sum_n a_n x^n = Xa$$

$$X = \boxed{\begin{matrix} x^1 & x^2 & \cdots\cdots & x^N \end{matrix}} \qquad \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix}$$

Step 1: $f(x) = w^T x \xrightarrow{\;w = X\boldsymbol{\alpha}\;} f(x) = \boldsymbol{\alpha}^T X^T x$

$$\begin{bmatrix} \alpha_1 & \cdots & \alpha_N \end{bmatrix} \qquad \boxed{\begin{matrix} x^1 \\ x^2 \\ \vdots \\ x^N \end{matrix}} \qquad x$$

$$\begin{bmatrix} x^1 \cdot x \\ x^2 \cdot x \\ \vdots \\ x^N \cdot x \end{bmatrix}$$

$$f(x) = \sum_n a_n (x^n . x)$$

$$= \sum_n a_n K(x^n . x)$$

核技巧
Kernel Trick

Step 1: $f(x) = \sum_n a_n K(x^n . x)$ Find $a_1^*, a_2^*, \ldots a_n^*,$

Step 2, 3: Find $a_1^*, \ldots, a_n^*, \ldots, a_N^*$, minimizing loss function L

$$L(f) = \sum_i L\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

$$= \sum_i L\left(\sum_n a_n K(x^n . x) \ , y^{(i)}\right)$$

We only need to know the inner project between a pair of vectors x and z

Kernel Trick

核技巧
Kernel Trick

Directly computing $K(x, z)$ can be faster than "feature transformation + inner product" sometimes.

Kernel trick is useful when we transform all x to $\phi(x)$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\phi(x) = \begin{bmatrix} {x_1}^2 \\ \sqrt{2}x_1x_2 \\ {x_2}^2 \end{bmatrix}$$

$$K(x,z) = \phi(x) \cdot \phi(z) = \begin{bmatrix} {x_1}^2 \\ \sqrt{2}x_1x_2 \\ {x_2}^2 \end{bmatrix} \cdot \begin{bmatrix} {z_1}^2 \\ \sqrt{2}z_1z_2 \\ {z_2}^2 \end{bmatrix}$$

$$= {x_1}^2{z_1}^2 + 2x_1x_2z_1z_2 + {x_2}^2{z_2}^2$$

$$= (x_1z_1 + x_2z_2)^2 = \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}\right)^2$$

$$= (x \cdot z)^2$$

核技巧
Kernel Trick

$$K(x,z) = (x \cdot z)^2$$

$$= (x_1 z_1 + x_2 z_2 + \cdots + x_k z_k)^2$$

$$= {x_1}^2 {z_1}^2 + {x_2}^2 {z_2}^2 + \cdots + {x_k}^2 {z_k}^2$$

$$+2 x_1 x_2 z_1 z_2 + 2 x_1 x_3 z_1 z_3 + \cdots$$

$$+2 x_2 x_3 z_2 z_3 + 2 x_2 x_4 z_2 z_4 + \cdots$$

$$= \phi(x) \cdot \phi(z)$$

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_k \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ \vdots \\ z_k \end{bmatrix}$$

$$\phi(x) = \begin{bmatrix} {x_1}^2 \\ \vdots \\ {x_k}^2 \\ \sqrt{2} x_1 x_2 \\ \sqrt{2} x_1 x_3 \\ \vdots \\ \sqrt{2} x_2 x_3 \\ \vdots \end{bmatrix}$$

常用核函数
Kernel Function

- Liner kernel

$$K(x, z) = x^T z$$

- Polynomial kernel

$$K(x, z) = (x^T z)^d$$

- Gaussian kernel / Radial Basis Function Kernel

$$K(x, z) = \exp(-\frac{\|x - z\|^2}{2\sigma^2})$$

- Sigmoid kernel

$$K(x, z) = \tanh(\beta x^T z + \theta)$$

RBF核
Radial Basis Function Kernel

$$K(x,z) = exp\left(-\frac{1}{2}\|x-z\|_2\right) = \phi(x)\cdot\phi(z)?$$

$\phi(*)$ has inf dim!!!

$$= exp\left(-\frac{1}{2}\|x\|_2 - \frac{1}{2}\|z\|_2 + x\cdot z\right)$$

$$= exp\left(-\frac{1}{2}\|x\|_2\right) exp\left(-\frac{1}{2}\|z\|_2\right) exp(x\cdot z) \quad = C_x C_z exp(x\cdot z)$$

$$= C_x C_z \sum_{i=0}^{\infty} \frac{(x\cdot z)^i}{i!} = C_x C_z + C_x C_z (x\cdot z) + C_x C_z \frac{1}{2}(x\cdot z)^2 \cdots$$

$$[C_x]\cdot[C_z] \qquad \begin{bmatrix} C_x x_1 \\ C_x x_2 \\ \vdots \end{bmatrix}\cdot\begin{bmatrix} C_z z_1 \\ C_z z_2 \\ \vdots \end{bmatrix} \qquad \frac{1}{\sqrt{2}}\begin{bmatrix} C_x {x_1}^2 \\ \vdots \\ \sqrt{2} C_x x_1 x_2 \\ \vdots \end{bmatrix}\cdot\frac{1}{\sqrt{2}}\begin{bmatrix} C_z {z_1}^2 \\ \vdots \\ \sqrt{2} C_z z_1 z_2 \\ \vdots \end{bmatrix}$$

Sigmoid核
Sigmoid Kernel

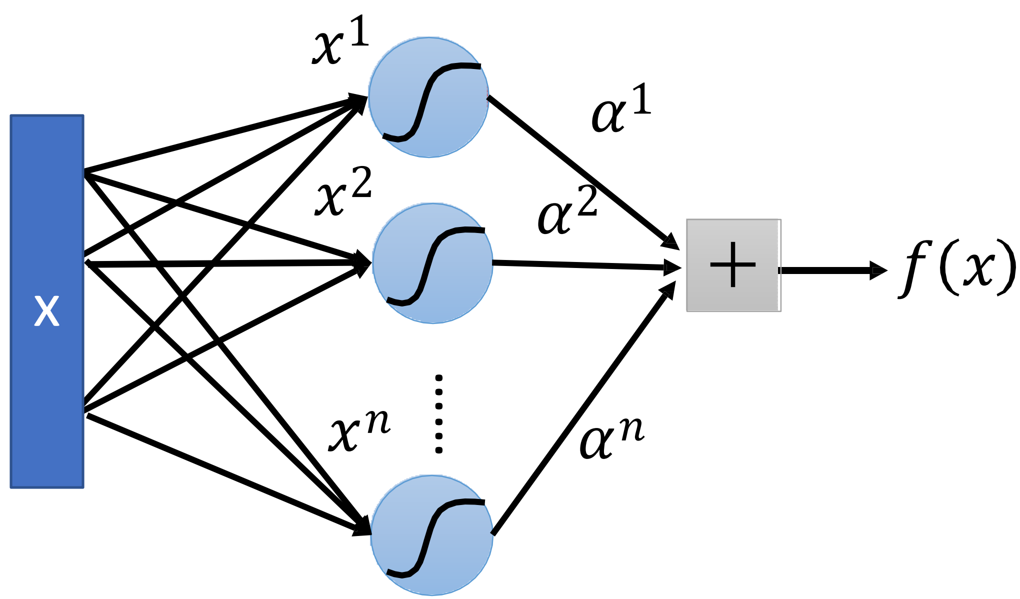
$$K(x,z) = tanh(x \cdot z)$$

- When using sigmoid kernel, we have a 1 hidden layer network.

$$f(x) = \sum_n a_n K(x^n . x) = \sum_n a_n tanh(x^n . x)$$

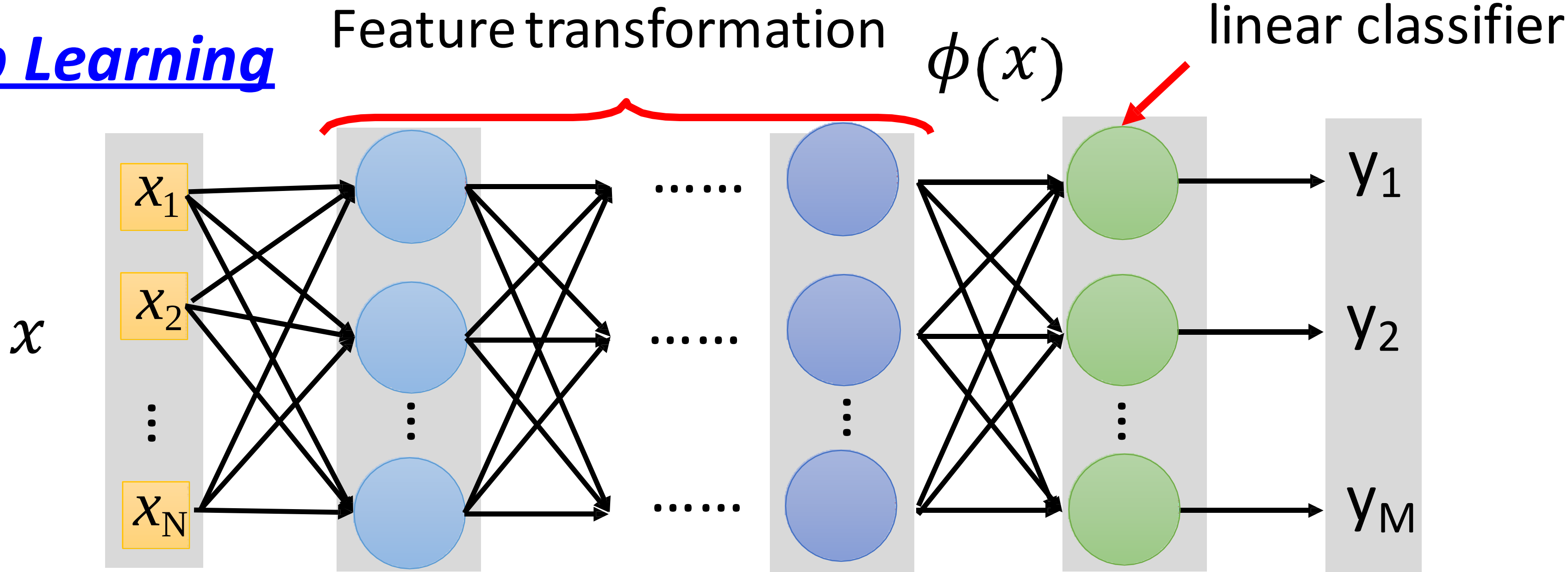
The weight of each neuron is a data point

The number of support vectors is the number of neurons.



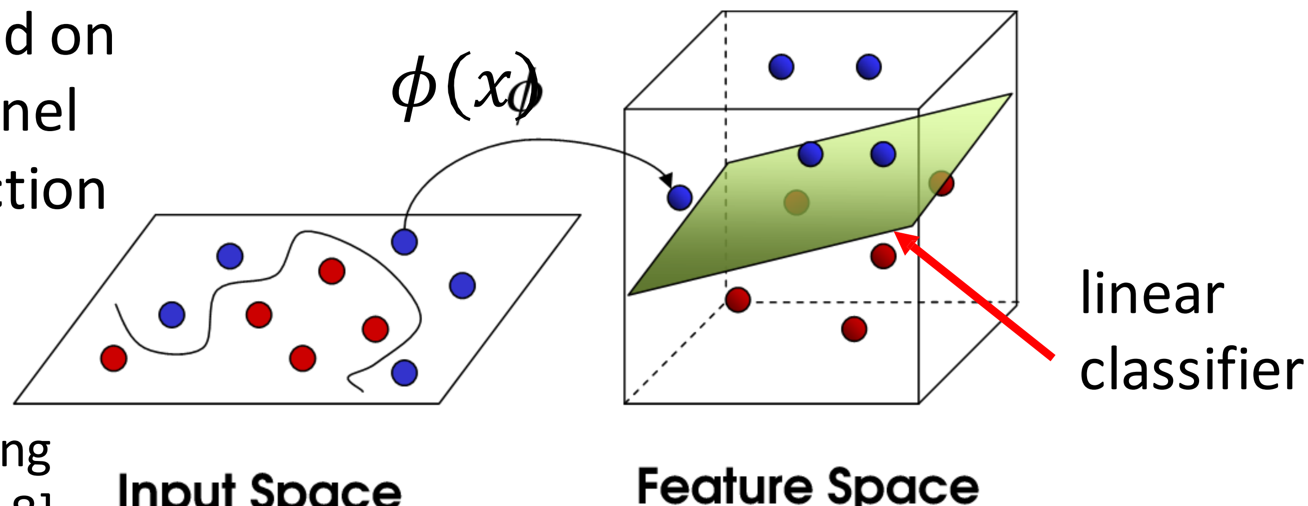
深度学习和支持向量机
Deep learning VS SVM

Deep Learning



SVM

Based on kernel function



Multiple Kernel learning
[Alpaydin, Chapter 13.8]

SVM 软件包

- LIBSVM
  http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LIBLINEAR
  http://www.csie.ntu.edu.tw/~cjlin/liblinear/
- $\text{SVM}^{\text{light}}$、$\text{SVM}^{\text{perf}}$、$\text{SVM}^{\text{struct}}$
  http://svmlight.joachims.org/svm_struct.html
- Pegasos
  http://www.cs.huji.ac.il/~shais/code/index.html

- Demo
  - Support Vector Machine (dash.gallery)

SVM 方法
SVM related methods

- Support Vector Regression (SVR)
  - [Bishop chapter 7.1.4]
- Ranking SVM
  - [Alpaydin, Chapter 13.11]
- One-class SVM
  - [Alpaydin, Chapter 13.11]

  - Support Vector Machine (dash.gallery)

简述一下本节课介绍的SVM的两个最重要的特点以及带来的好处?