人工智能
——逻辑回归模型

Yanghui Rao
Assistant Prof., Ph.D
School of Data and Computer Science,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

逻辑回归模型

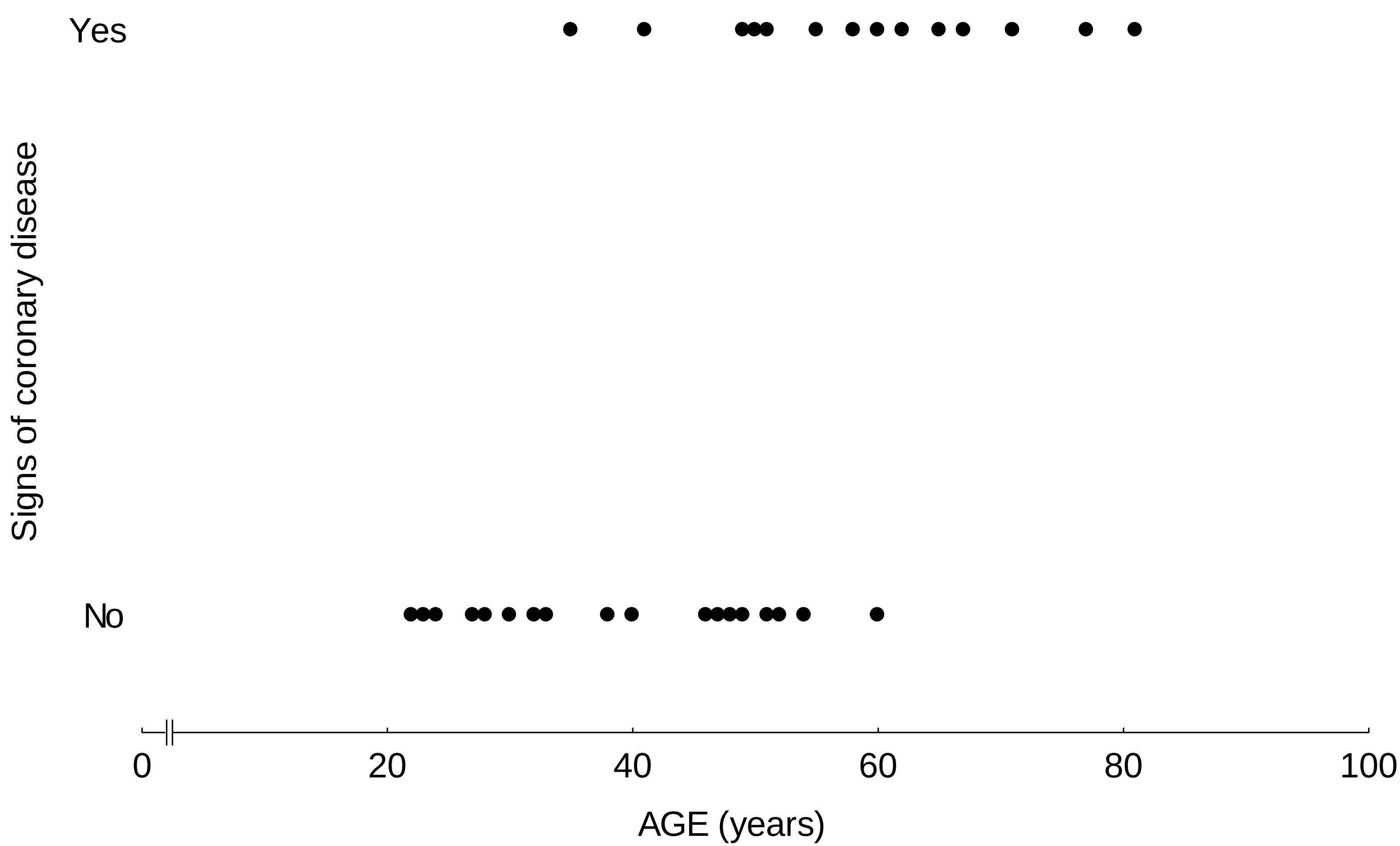
- 如果使用最小二乘法的回归模型来做二分类任务：

$$y = w_0 + \sum_{j=1}^{d} w_j x_j + u$$

$$= \tilde{\mathbf{W}}^{\mathrm{T}} \tilde{\mathbf{X}}$$

- 基于上述模型预测的y值，即样本属于某个类的概率，会超出0到1的范围。

逻辑回归模型



逻辑回归模型

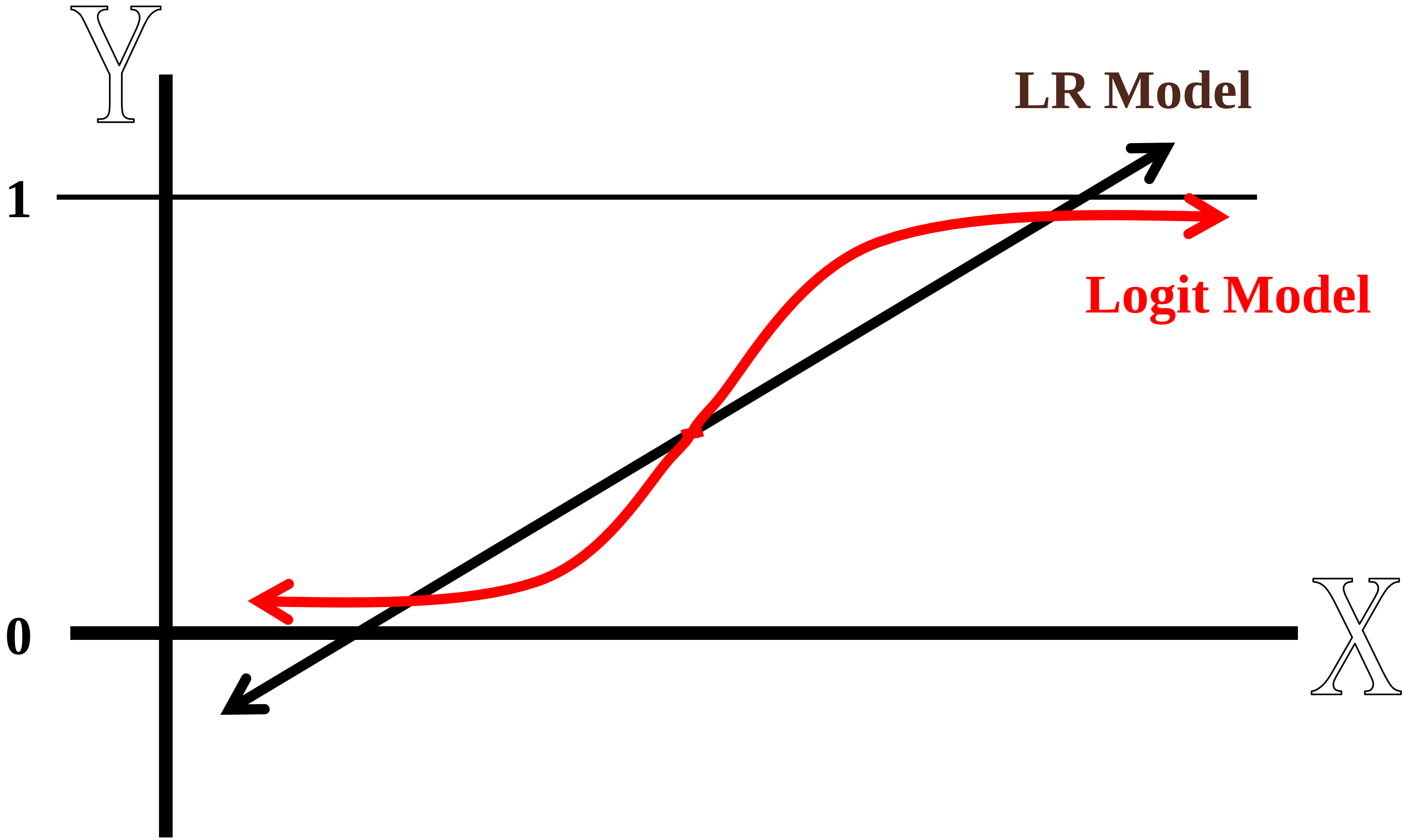
- “logit” 变换可以解决上述问题:

$$\log\left(\frac{p}{1-p}\right) = w_0 + \sum_{j=1}^{d} w_j x_j + u$$

$$= \tilde{\mathbf{W}}^{\mathrm{T}} \tilde{\mathbf{X}}$$

- p 是事件y发生的概率，比如：$p=p(y=1|\mathbf{X})$
- $p/(1-p)$ 称为机率比或优势比 (odds ratio)
- $\log[p/(1-p)]$ 是机率比的对数, 或称为 "logit"

逻辑回归模型



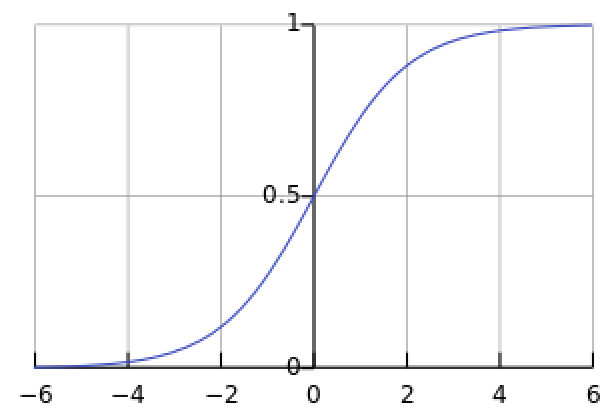
逻辑回归模型

- logistic 函数使得输出的概率值在0到1的范围内.

- 样本 **X** 标签为正的概率 $p(y=1|\mathbf{X})$ 是:

$$p = \frac{1}{1+e^{-w_0-\sum_{j=1}^{d} w_j x_j}} = \frac{e^{w_0+\sum_{j=1}^{d} w_j x_j}}{1+e^{w_0+\sum_{j=1}^{d} w_j x_j}}$$

$$= \frac{1}{1+e^{-\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}} = \frac{e^{\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}}{1+e^{\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}}$$



- 如果 $w_0+\sum_{j=1}^{d} w_j x_j=0$, 那么 $p=0.5$
- 当 $w_0+\sum_{j=1}^{d} w_j x_j$ 很大时, p 趋近于 1
- 当 $w_0+\sum_{j=1}^{d} w_j x_j$ 很小时, p 趋近于 0

PLA ?

逻辑回归模型

- 最小二乘法回归模型，使用了最小二乘的公式，直接得到了最终的模型。
- 对于逻辑回归，可以使用极大似然估计，配合以一种迭代式的方法，计算出最终的模型。
- 算法:
  - 首先，随机初始化权重，并对某个样本进行预测；
  - 接着，计算这个模型在这次预测上的误差，改变权重，以提高模型在这个样本上的似然度；
  - 重复这个过程，直到模型收敛，即当前模型和上一步的模型的表现相差无几。
- 这个想法的本质是：找到一个最有可能产生你观察到的数据的参数。

逻辑回归模型

- 似然函数：$\prod_{i=1}^{n}(p_i)^{y_i}(1-p_i)^{1-y_i}$
- 极大似然法:

$$L(\tilde{\mathbf{W}})=\sum_{i=1}^{n}\left(y_i\log p_i+(1-y_i)\log(1-p_i)\right)$$

$$=\sum_{i=1}^{n}\left(y_i\log\frac{p_i}{1-p_i}+\log(1-p_i)\right)$$

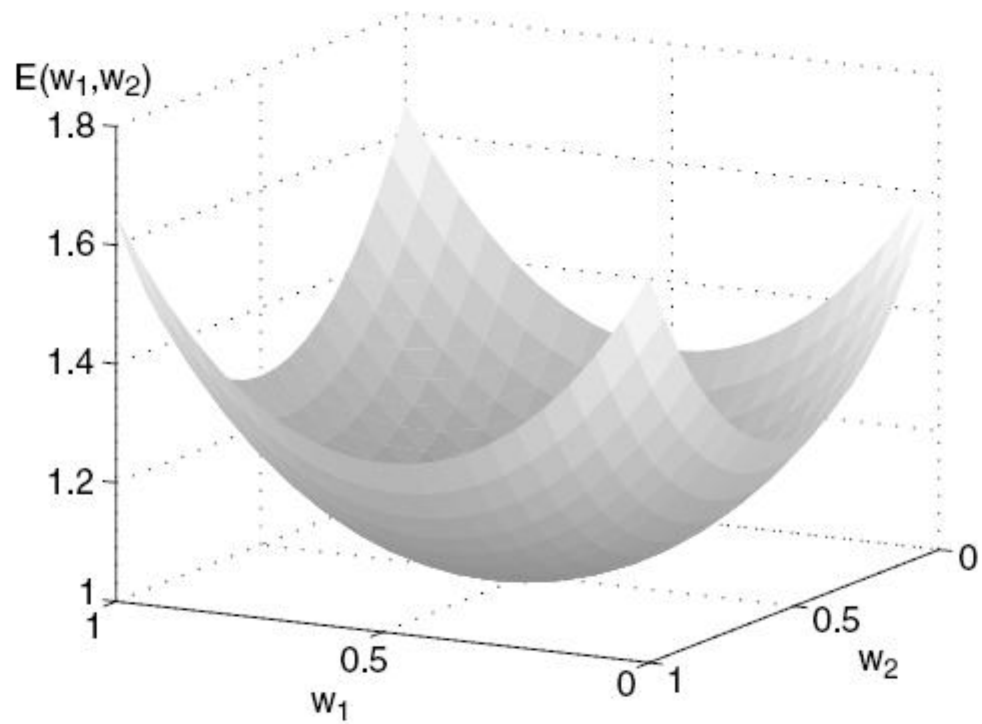
$$=\sum_{i=1}^{n}\left(y_i\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i-\log(1+e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i})\right)$$

$$\frac{\partial L(\tilde{\mathbf{W}})}{\partial\tilde{\mathbf{W}}}=\sum_{i=1}^{n}\left[\left(y_i-\frac{e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}{1+e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}\right)\tilde{\mathbf{X}}_i\right]$$

- 等价于最小化代价函数：

$$C(\tilde{\mathbf{W}})=-L(\tilde{\mathbf{W}})=-\sum_{i=1}^{n}\left(y_i\log p_i+(1-y_i)\log(1-p_i)\right)$$ 交叉熵

梯度下降



逻辑回归模型

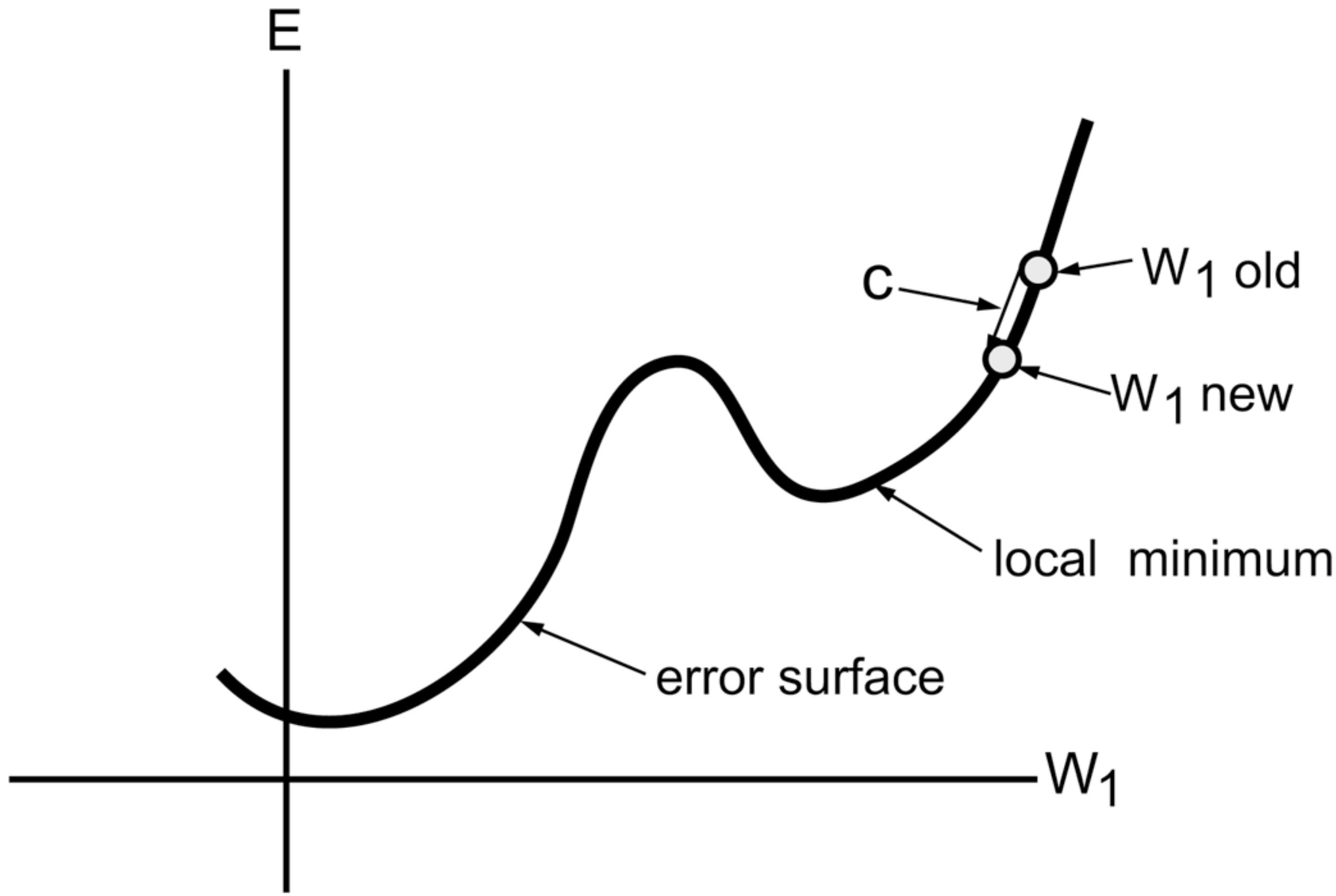
- 梯度下降
  - 计算梯度向量
  - 每次用梯度向量的反方向来更新权重
- 重复:

$$\tilde{\mathbf{W}}_{new}^{(j)} = \tilde{\mathbf{W}}^{(j)} - \eta \frac{\partial C(\tilde{\mathbf{W}})}{\partial \tilde{\mathbf{W}}^{(j)}}$$

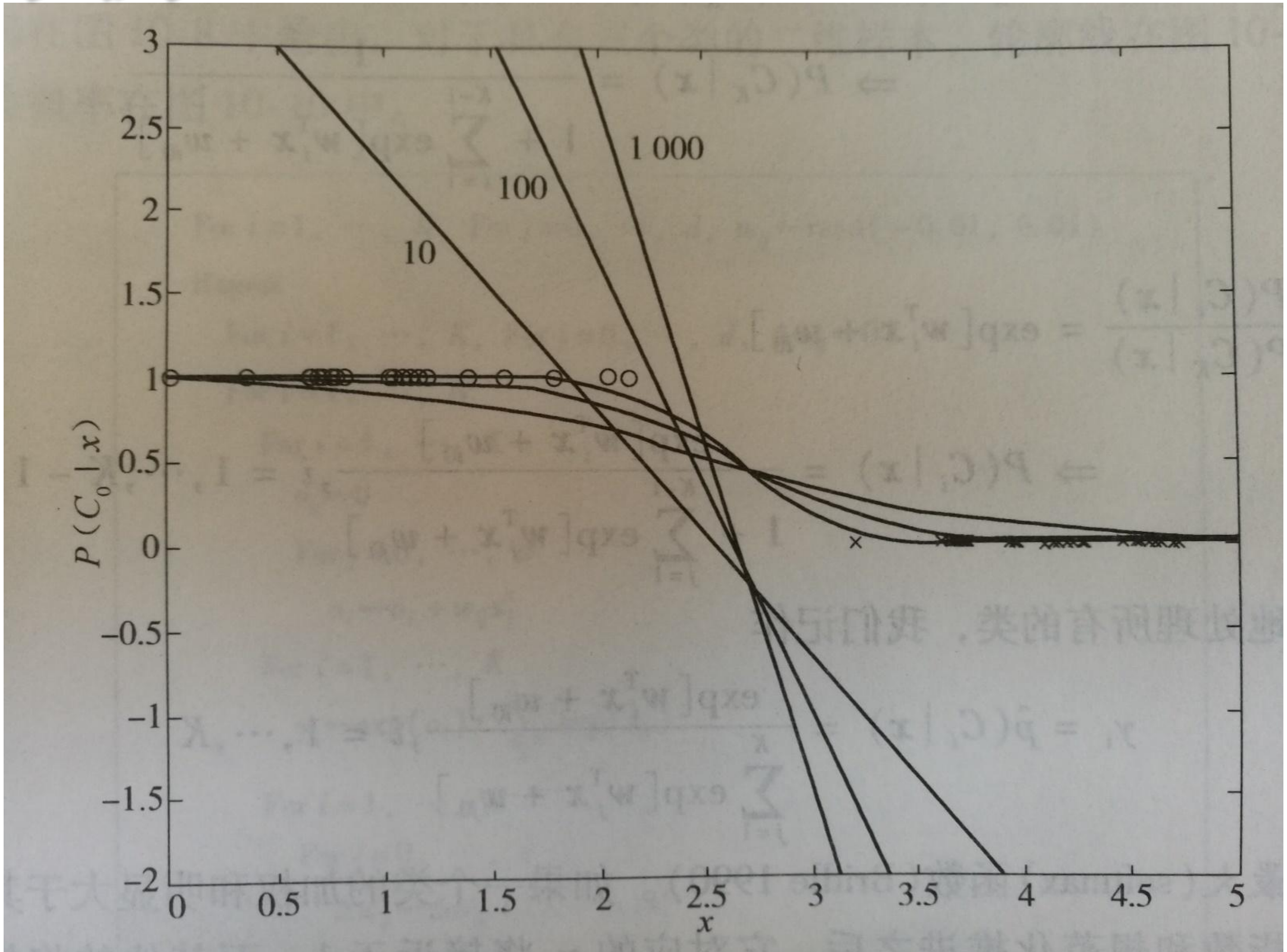
$$= \tilde{\mathbf{W}}^{(j)} - \eta \sum_{i=1}^{n} \left[\left(\frac{e^{\tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}_i}}{1 + e^{\tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}_i}} - y_i \right) \tilde{\mathbf{X}}_i^{(j)} \right]$$

- 直至收敛。

梯度下降



示例



对于一元两类问题，训练样本上迭代10次、100次和1000次之后，直线 w_0+wx 和 S形（Sigmoid）函数输出的演变