Artificial Intelligence
——Logistic Regression Model

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

Logistic Regression Model

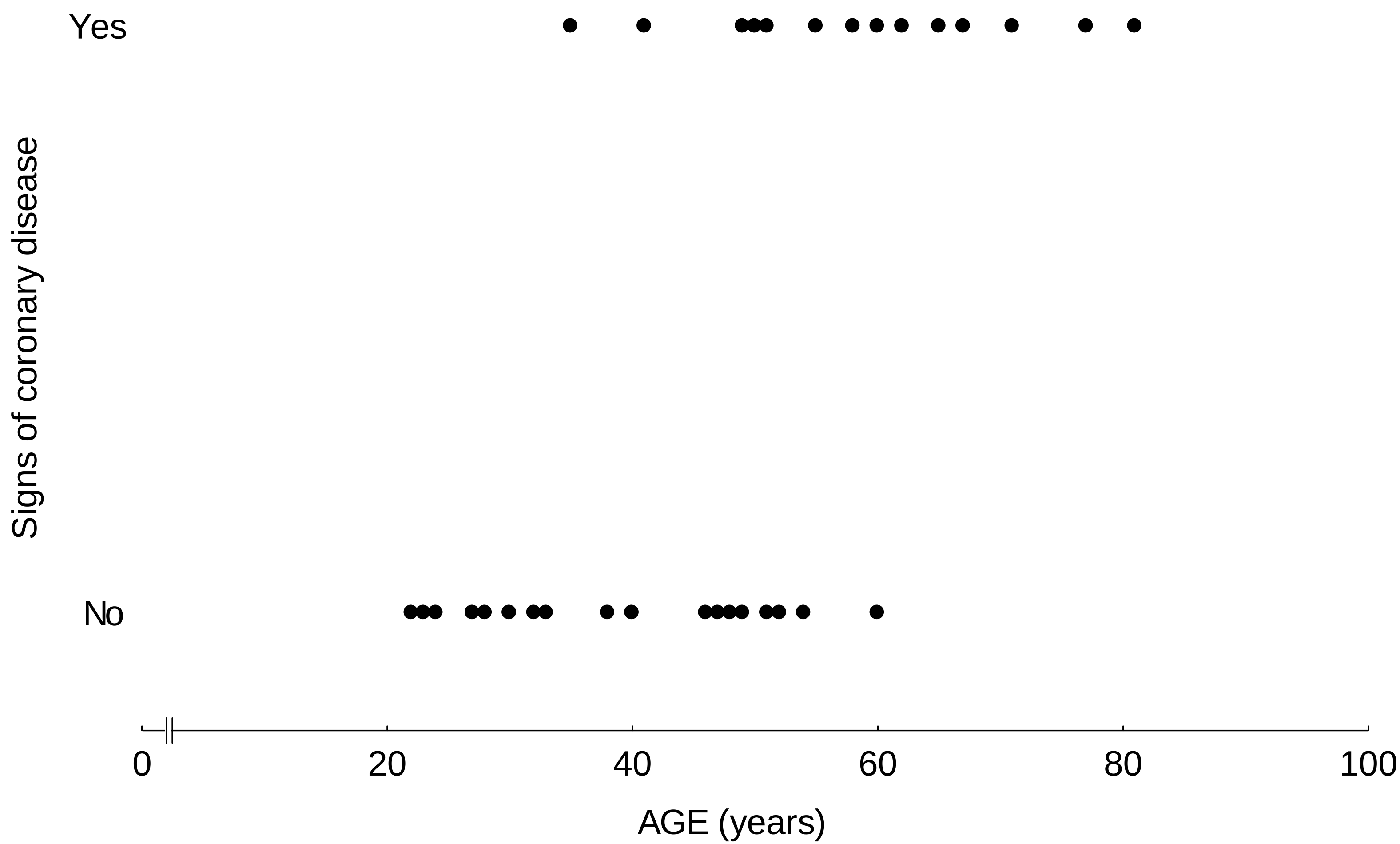
- If using the ordinary least squares (OLS) regression model for binary classification

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

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

  - The error terms u are heteroscedastic (异方差)
  - u is not normally distributed because y takes on only two values
  - The predicted probabilities can be greater than 1 or less than 0

Logistic Regression Model



Logistic Regression Model

- The "logit" model solves these problems:

如果p是-00到+00.分对数变换，然后就可以用PLA

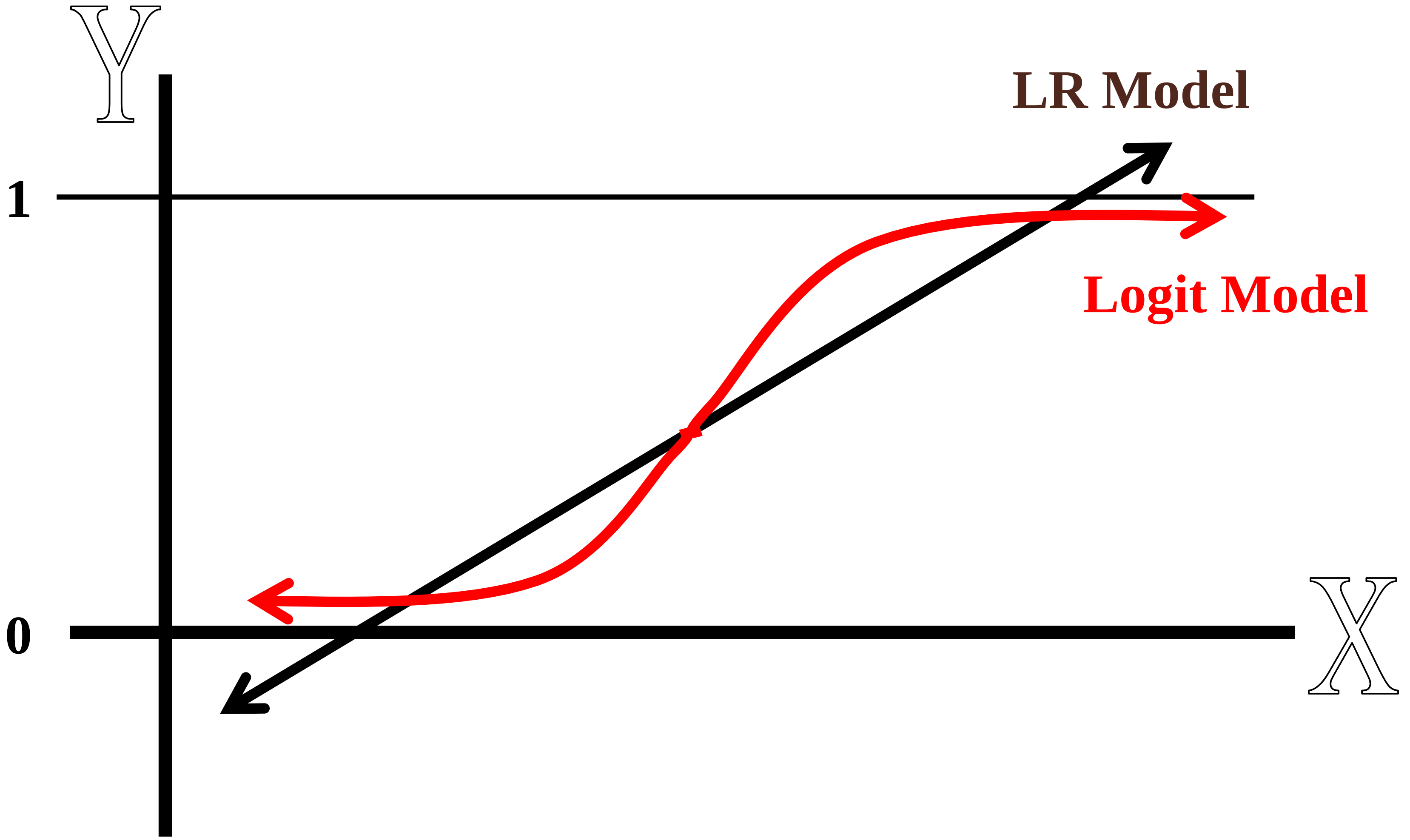
$$\log\left(\frac{p}{1-p}\right) = w_0 + \sum_{j=1}^{d} w_j x_j + u$$

几率

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

- p is the probability that the event y occurs, $p(y=1 \mid \mathbf{X})$
- $p/(1-p)$ is the odds ratio (*e.g.*, odds of disease)
- $\log[p/(1-p)]$ is the log odds ratio, or "logit"

Logistic Regression Model



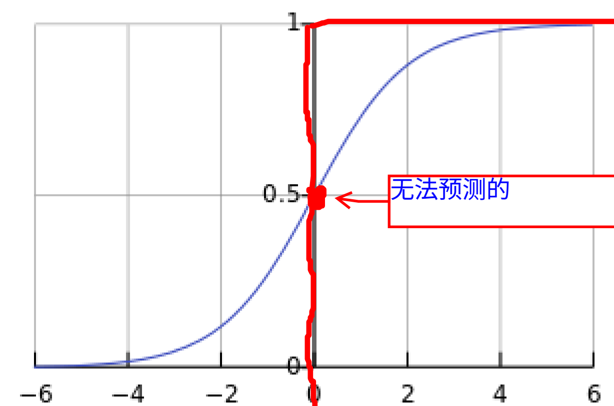
Logistic Regression Model

- The logistic distribution constrains the estimated probabilities to lie between 0 and 1.
- The estimated probability $p(y=1|\mathbf{X})$ is:

可以从概率进行解释

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

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



无法预测的

- if you let $w_0+\sum_{j=1}^{d}w_jx_j=0$, then $p=0.5$
- as $w_0+\sum_{j=1}^{d}w_jx_j$ gets really big, p approaches 1
- as $w_0+\sum_{j=1}^{d}w_jx_j$ gets really small, p approaches 0

PLA

PLA ?

Logistic Regression Model

- Recall that OLS Regression could utilized an “ordinary least squares” formula to create the “linear model” we used.
- The Logistic Regression model will be solved by an **iterative maximum likelihood** procedure.
- This is a computer dependent program that:
  - starts with arbitrary values of the regression coefficients and constructs an initial model for predicting the observed data.
  - then evaluates errors in such prediction and changes the regression coefficients so as make the likelihood of the observed data greater under the new model.
  - repeats until the model converges, meaning the differences between the newest model and the previous model are trivial.
- The idea is that you “find and report as statistics” the parameters that are most likely to have produced your data.

Logistic Regression Model

- The likelihood function is $\prod_{i=1}^{n}(p_i)^{y_i}(1-p_i)^{1-y_i}$
- We want to maximize the log likelihood:

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

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

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

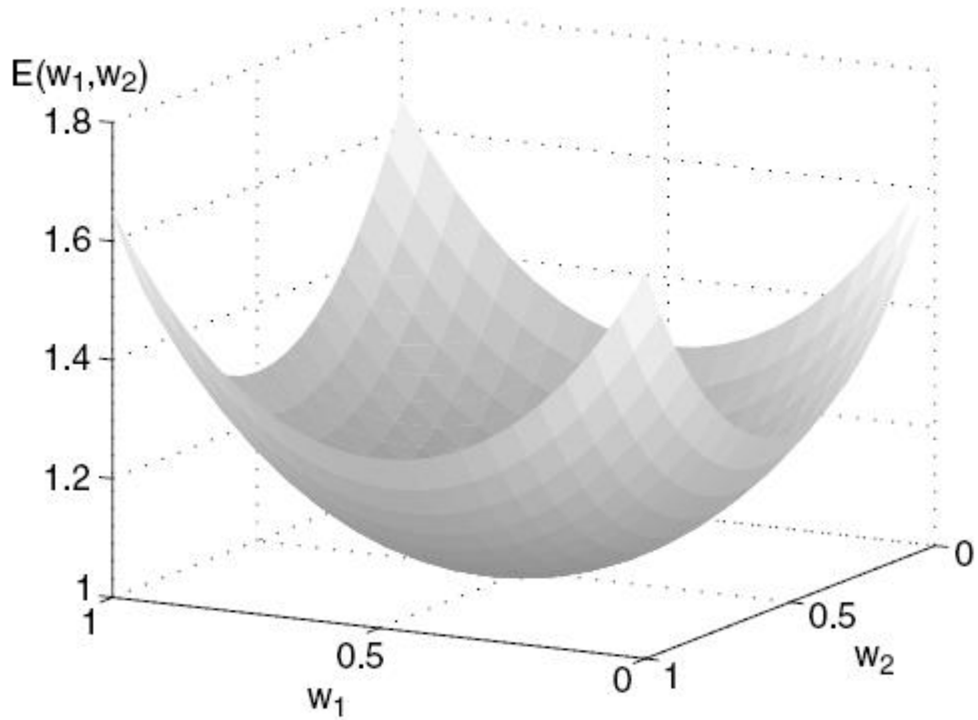
$$\frac{\partial L(\tilde{\mathbf{W}})}{\partial \tilde{\mathbf{W}}} = \sum_{i=1}^{n}\left[\left(y_i - \frac{e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}{1+e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}\right)\tilde{\mathbf{X}}_i\right]$$

- It is equal to minimize the cost function

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

Cross-entropy

Gradient Decent (梯度下降)



Logistic Regression Model

- Gradient Decent (梯度下降)
  - Calculate the gradient vector
  - Update the weighting in the opposite direction of the gradient vector at each surface point

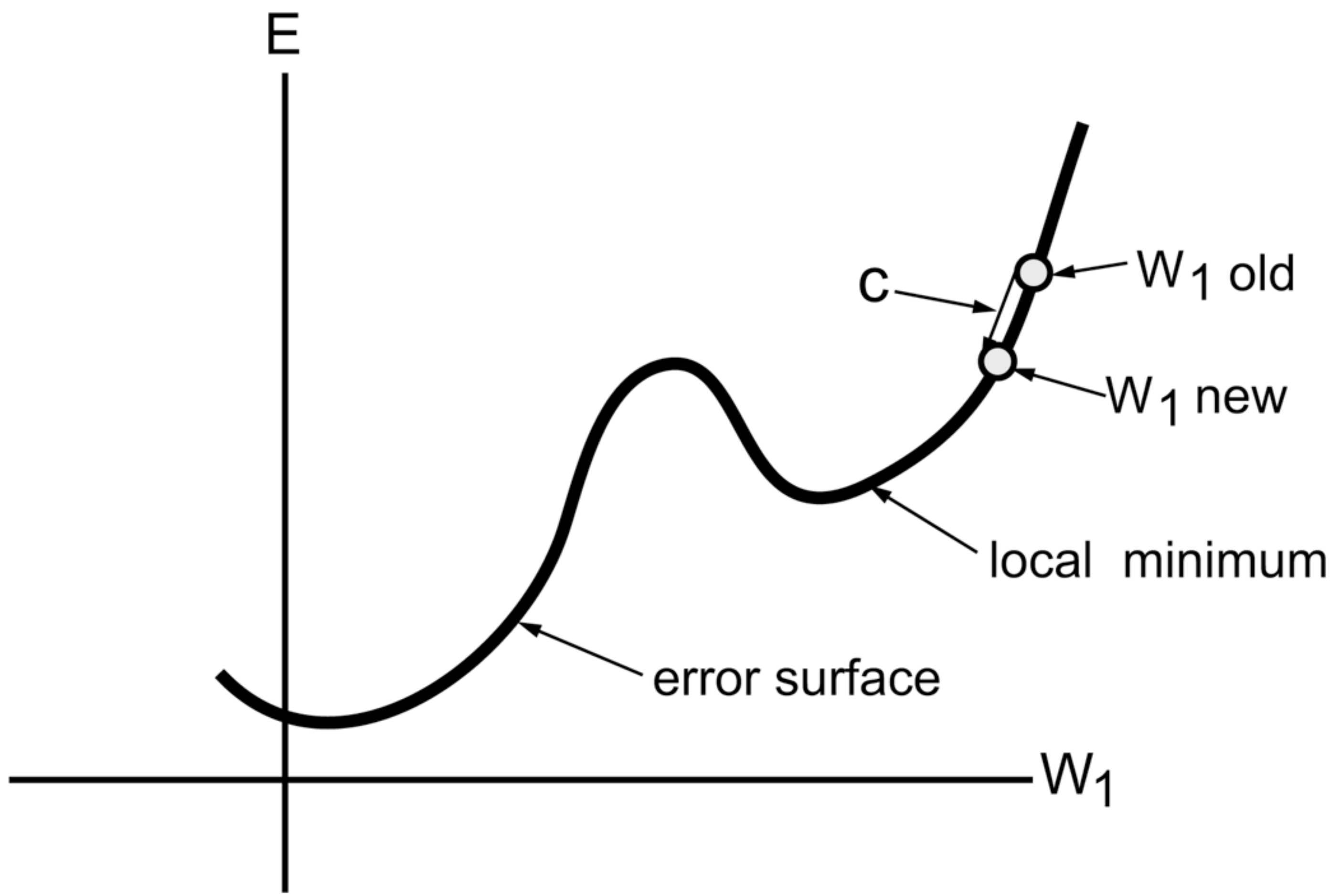
- Repeat: $\tilde{\mathbf{w}}_{new}^{(j)} = \tilde{\mathbf{w}}^{(j)} - \eta \dfrac{\partial C(\tilde{\mathbf{w}})}{\partial \tilde{\mathbf{w}}^{(j)}}$

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

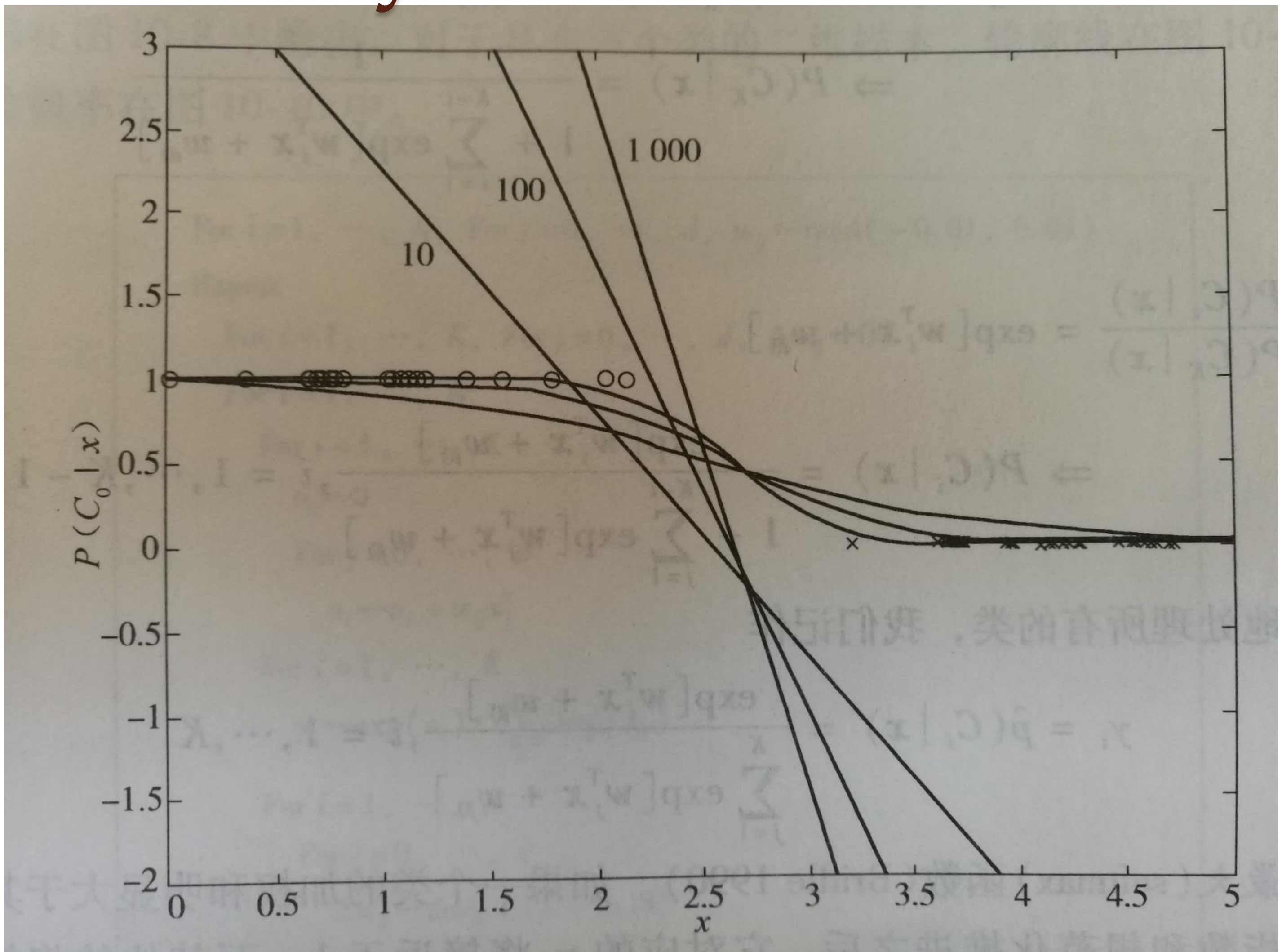
步长

- Until convergence

Gradient Decent (梯度下降)



Summary



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