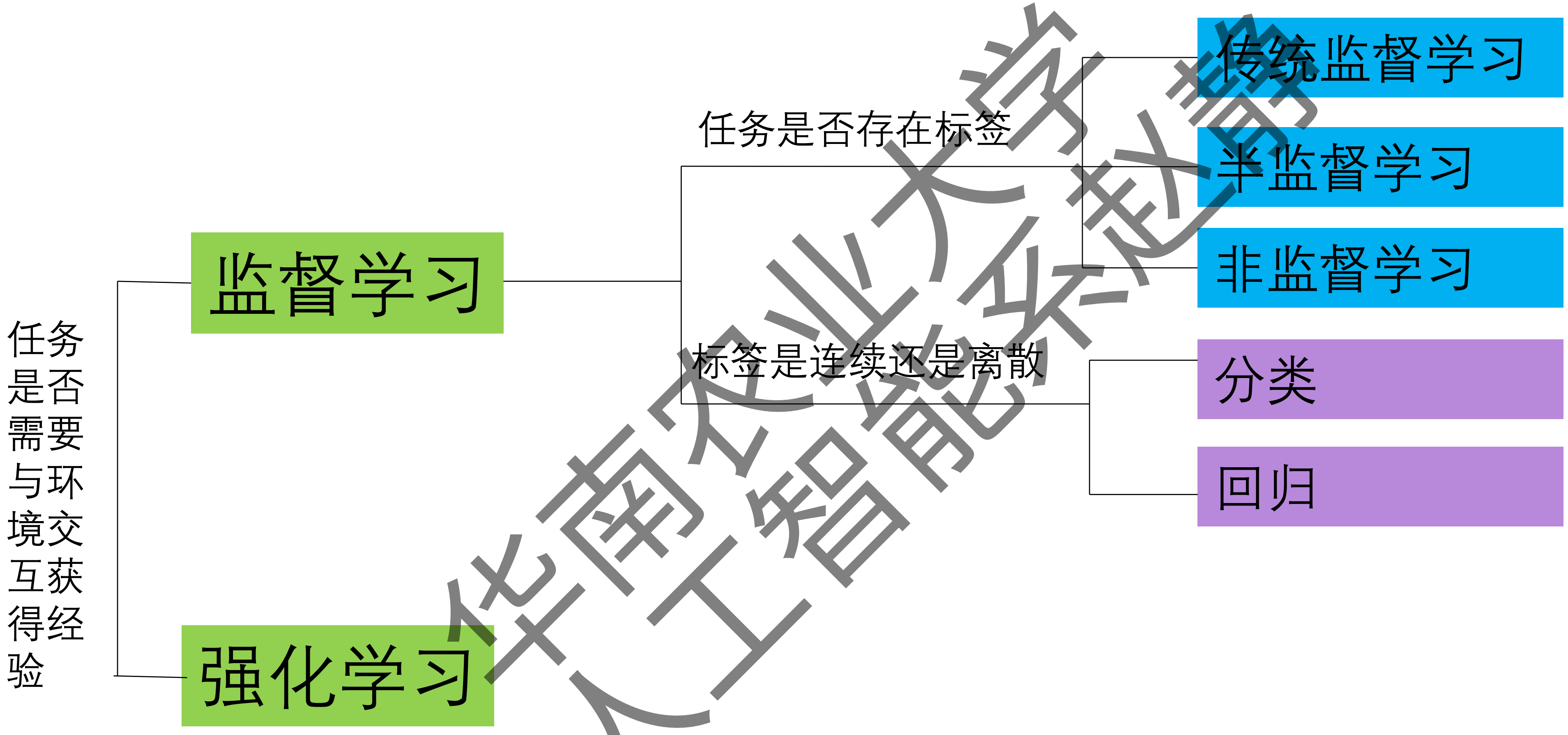
明理 精工
笃学 致远

第二章 模型评估与选择

电子工程学院、人工智能学院
college of Electronic Engineering , college of Artificial Intelligence

- 机器学习分类
- 机器学习的一般步骤
- 建模过程
- 模型选择
- 模型评估

一. 分类


任务是否需要与环境交互获得经验
监督学习
强化学习
任务是否存在标签
标签是连续还是离散
传统监督学习
半监督学习
非监督学习
分类
回归

监督学习

第i个训练样本的输入特征

第i个训练样本的标签

$$\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$$

监督学习

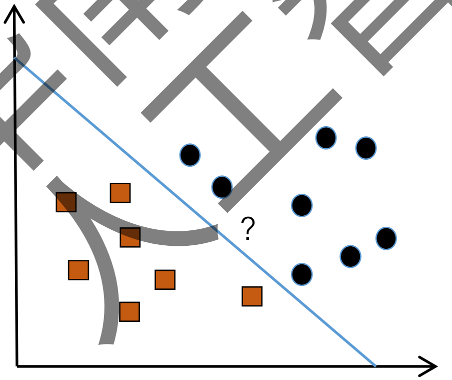
分类
(Classification)

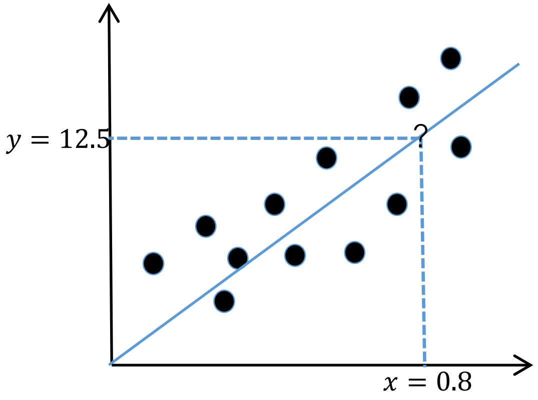
$y_i \in \{1,2,\dots,C\}$

回归
(Regression)

$y_i \in \mathbb{R}$

$$\hat{y} = f(\boldsymbol{x})$$

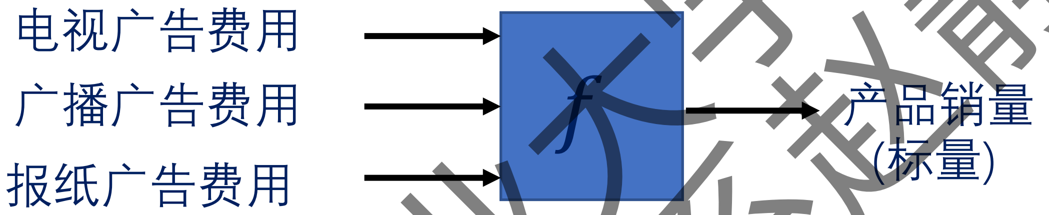




回归

函数 f 的输出为标量 $\hat{y} \in \mathbb{R}$

例：产品销量预测



训练数据:

输入

电视广告费用	广播广告费用	报纸广告费用	产品销量
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
151.5	41.3	58.5	18.5
180.8	10.8	58.4	12.9

输出

分类

函数 f 的输出为离散值

- 两类分类

Yes， No

函数f

输入

- 多类分类

类别1,类别2, ⋯类别C

函数f

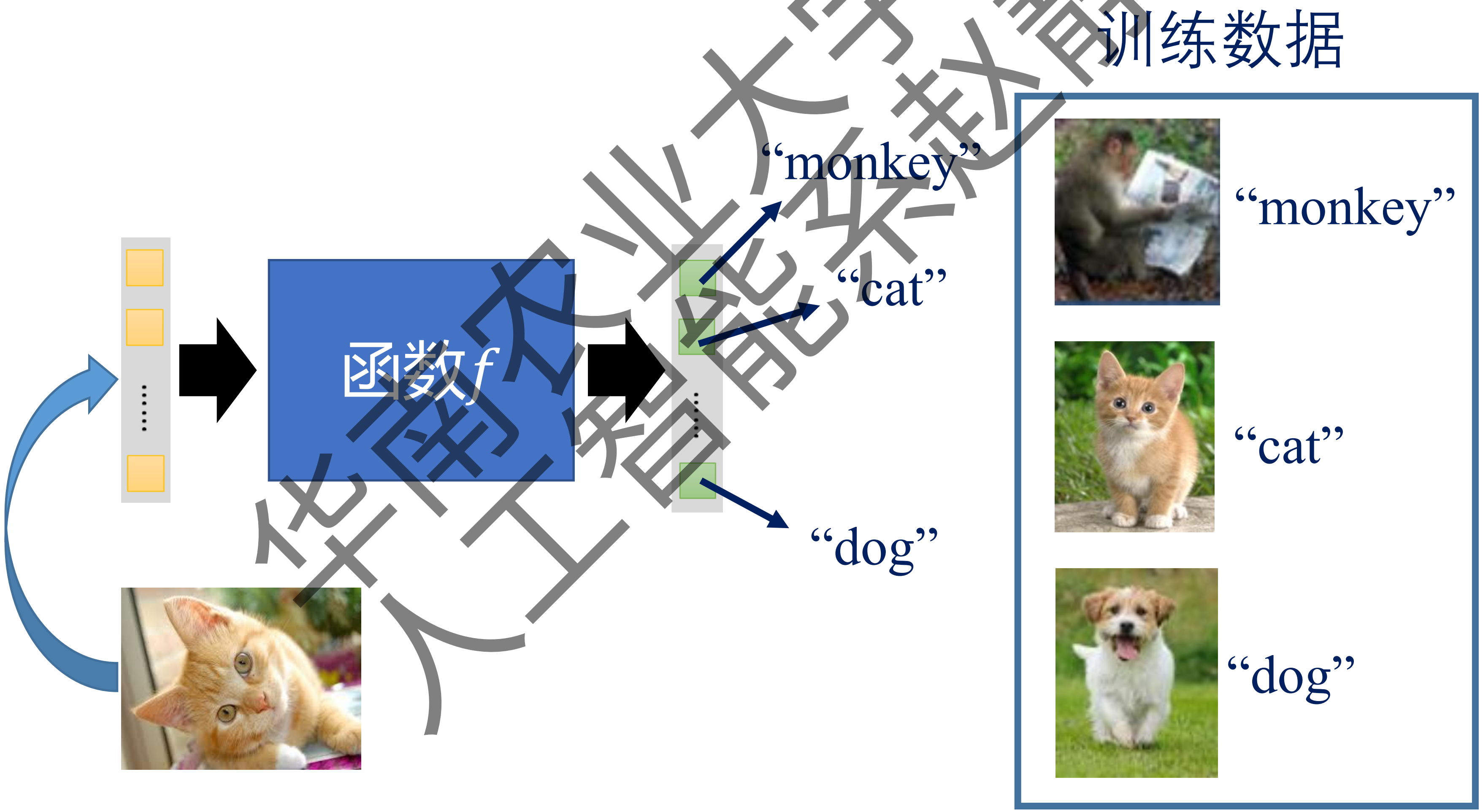
输入

例：两类分类

垃圾邮件过滤

(http://spam-filter-review.toptenreviews.com/)

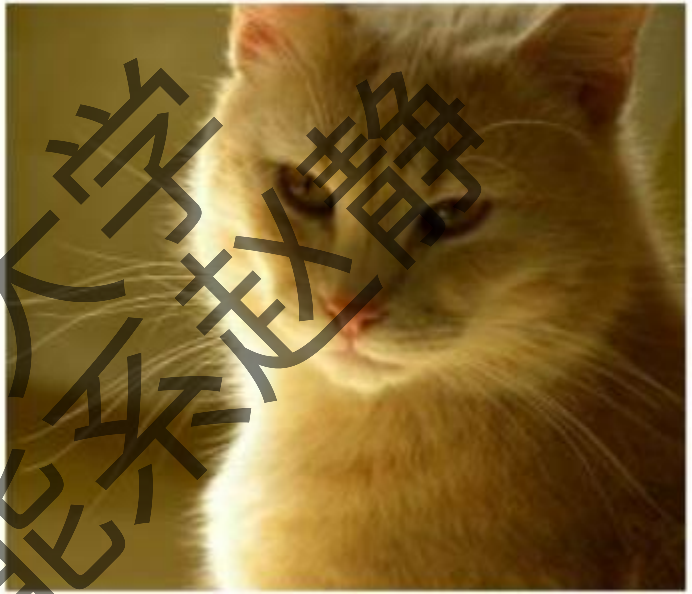
例：多类分类



图像识别

图像识别





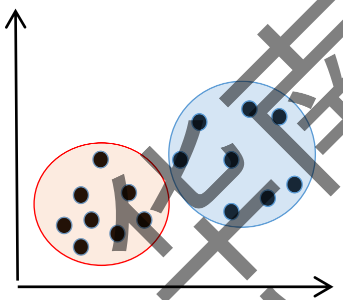


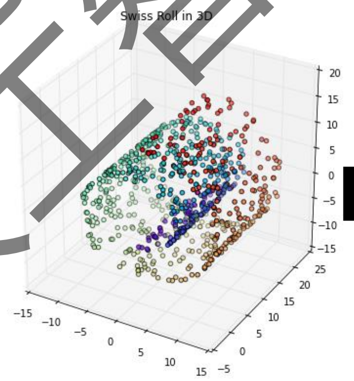
无监督学习

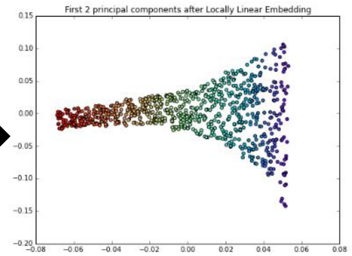
发现数据中的“有意义的模式”，亦被称为知识发现。

无监督学习 $\mathcal{D}=\{\boldsymbol{x}_i\}_{i=1}^{N}$

- 聚类 (Clustering)
- 降维 (Dimension Reduction)
- 密度估计/生成模型







$$z=f(\boldsymbol{x})$$

聚类

Clustering

- K-means

强化学习

■监督学习

下一次移动:
“5-5”

下一次移动:
“3-3”

■强化学习

第一次移动 …… 移动…… 赢!

- 从行为的反馈（奖励或惩罚）中学习
  - 设计一个奖励函数
  - 增强学习的任务：找到一条回报值最大的路径

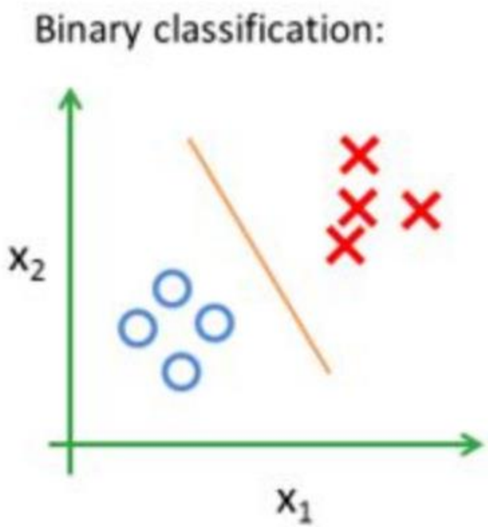
AlphaGo：监督学习 + 强化学习

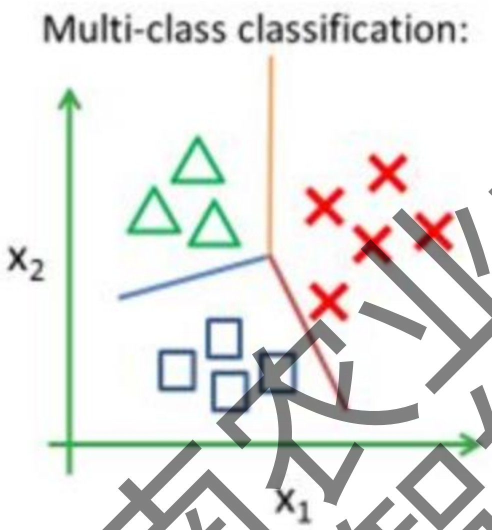
对比总结

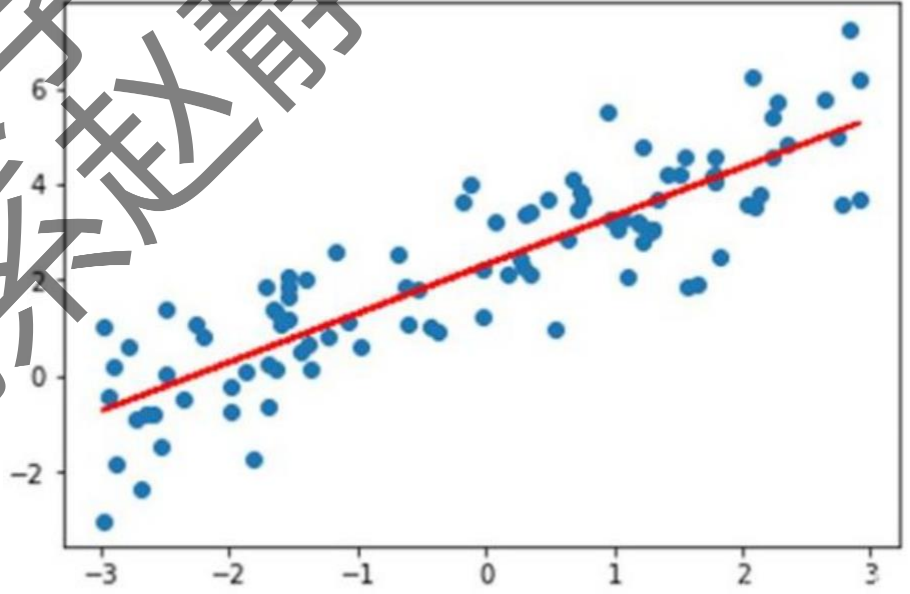
- 传统监督学习（supervised learning）**VS** 非监督学习（unsupervised learning）

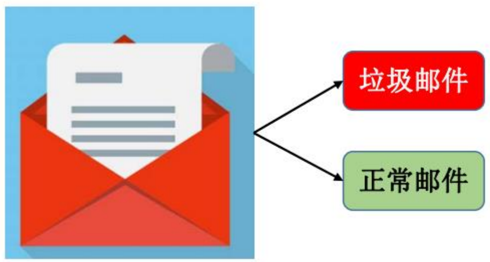
- 半监督学习（semi-supervised learning）

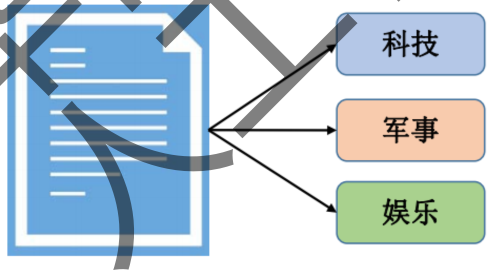
- 分类（Classification ）VS 回归（Regression）











- 监督学习（supervised learning）**VS** 强化学习（reinforcement learning）

华智冰

常见的监督学习算法

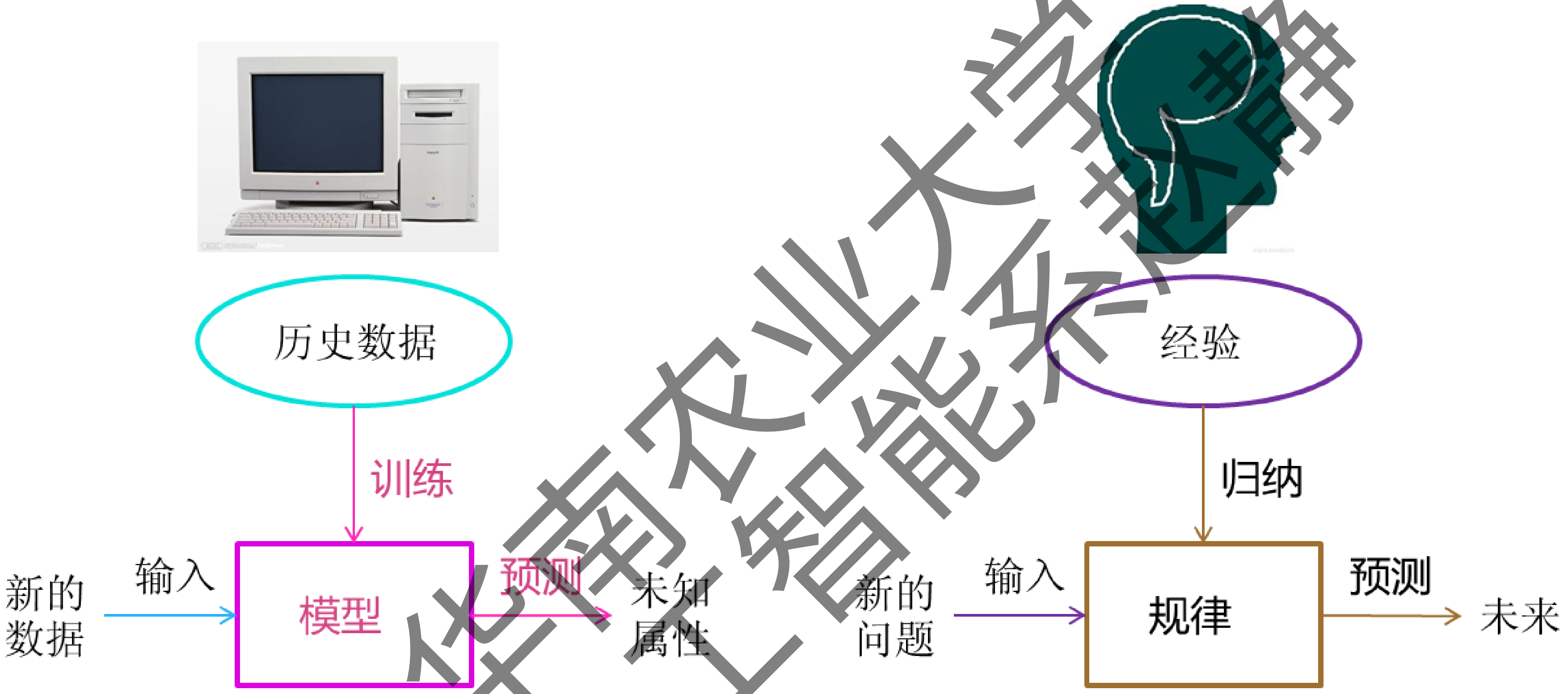
k-Nearest Neighbors （K近邻算法，分类）
Logistic Regression （逻辑回归，分类）
Naive Bayesian Mode（朴素贝叶斯算法，分类）
Support Vector Machines (SVMs) （支持向量机，分类）
Decision Trees and Random Forests （决策树和随机深林，分类）
Neural networks （神经网络，分类）

Linear Regression （线性回归，回归）
Partial Least Square Regression（PLS，偏最小二乘，回归）

常见的无监督学习算法

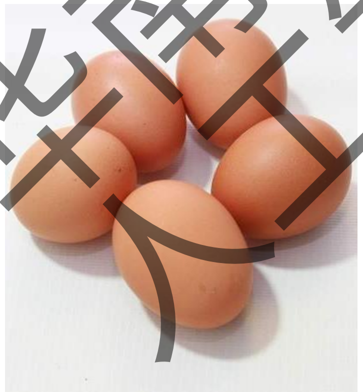
Clustering （聚类）
k-Means （K均值）
Principal Component Analysis (PCA，主成分分析)
t-distributed Stochastic Neighbor Embedding (t-SNE)

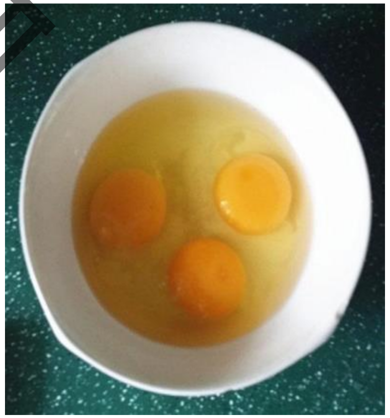
二. 机器学习的一般步骤



数据搜集 → 数据清洗 → 特征工程 → 数据建模






机器学习的一般步骤

数据搜集

- 网络下载
- 网络爬虫
- 数据库读取
- 开放数据
- ……

数据清洗

- 数据清理和格式化
- 探索性数据分析(EDA)

特征工程

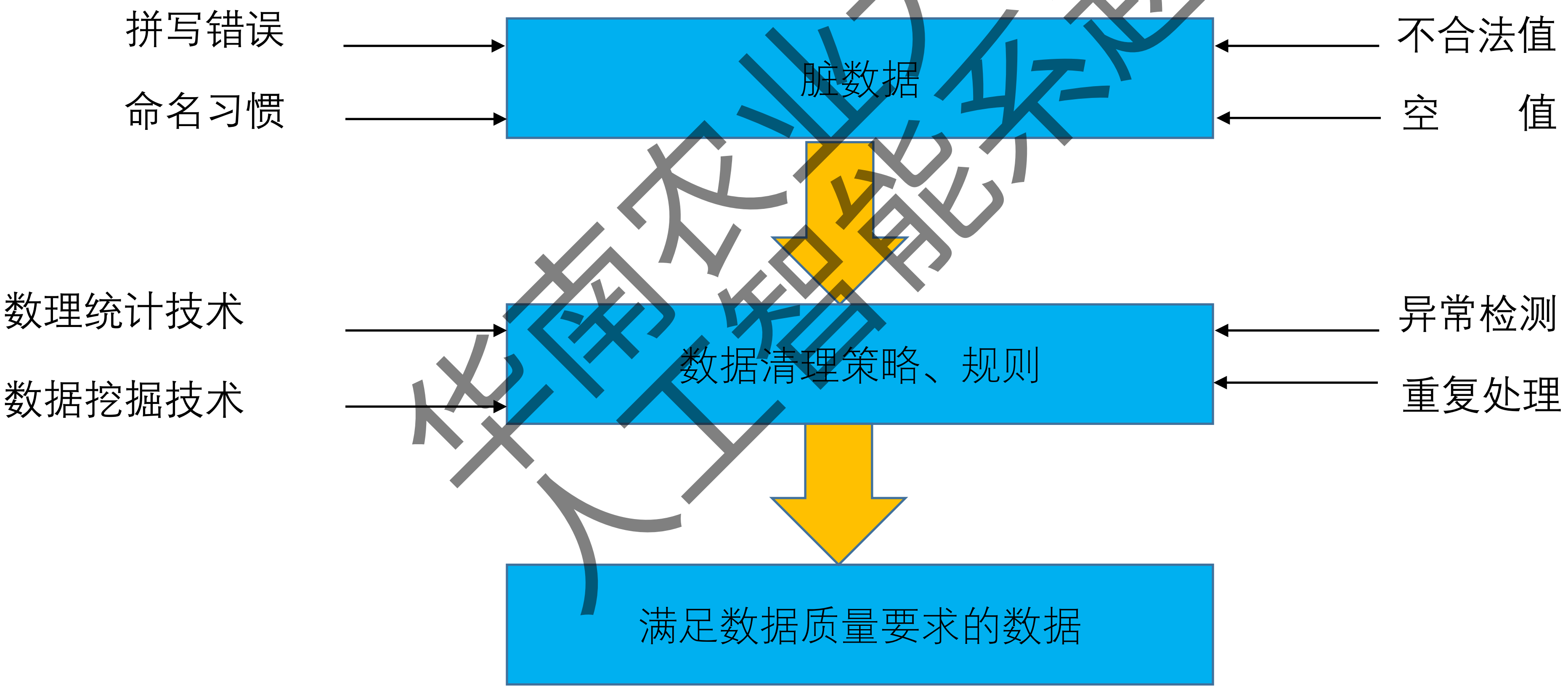
- 特征工程
- 特征选择

数据建模

- 基于性能指标比较几种机器学习模型
- 对最佳模型执行超参数调整
- 在测试集上评估最佳模型
- 解释模型结果
- 得出结论

数据清洗&特征工程

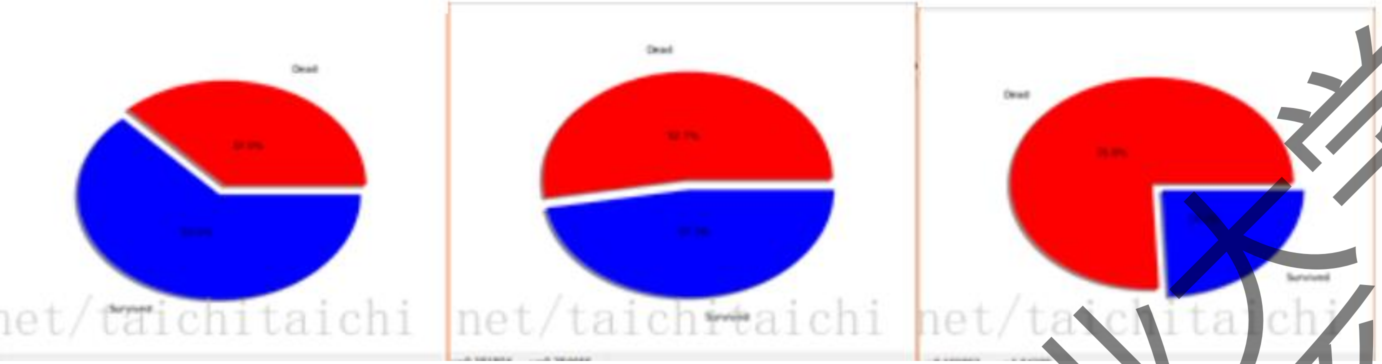
数据清洗原理



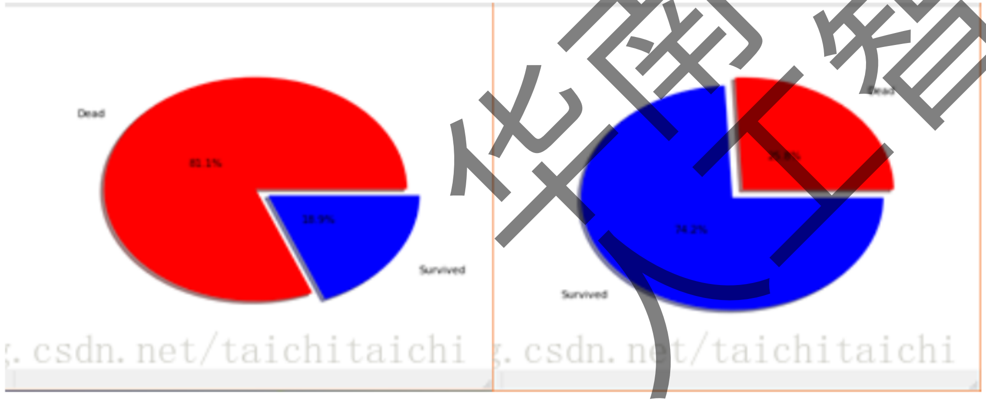
kaggle泰坦尼克号生存预测数据集

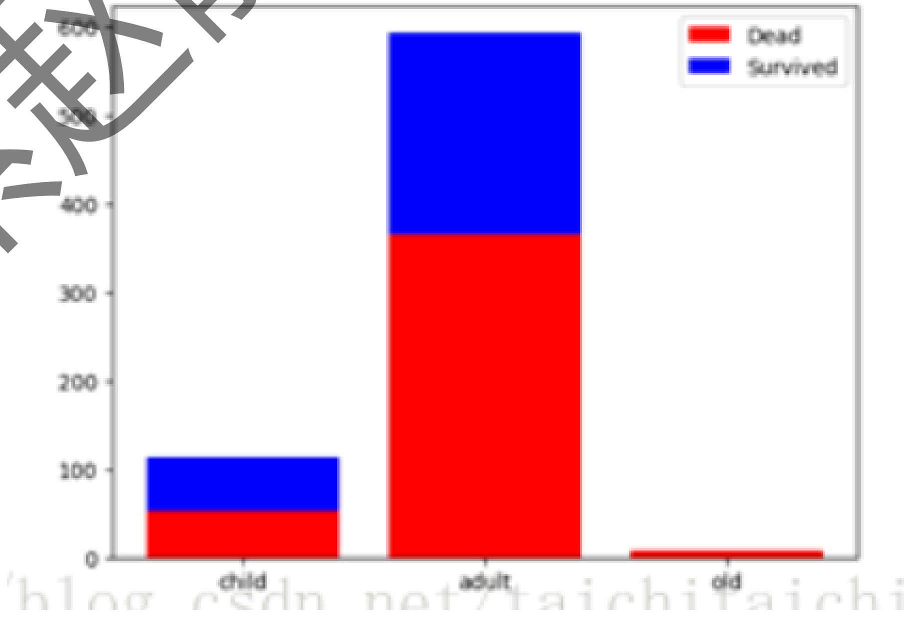
PassengerI	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, M	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen,	female	26	0	0	STON/O2. 3	7.925		S
4	1	1	Futrelle,	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr.	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr.	male		0	0	330877	8.4583		Q
7	0	1	McCarthy,	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, M	male	2	3	1	349909	21.075		S
9	1	3	Johnson, M	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mr	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom,	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, M	female	58	0	0	113783	26.55	C103	S
13	0	3	Saundercoc	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson,	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, M	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, M	female	55	0	0	248706	16		S
17	0	3	Rice, Mast	male	2	4	1	382652	29.125		Q
18	1	2	Williams,	male		0	0	244373	13		S
19	0	3	Vander Pla	female	31	1	0	345763	18		S
20	1	3	Masselmani	female		0	0	2649	7.225		C
21	0	2	Fynney, Mr	male	35	0	0	239865	26		S

Pclass=1 Pclass=2 Pclass=3



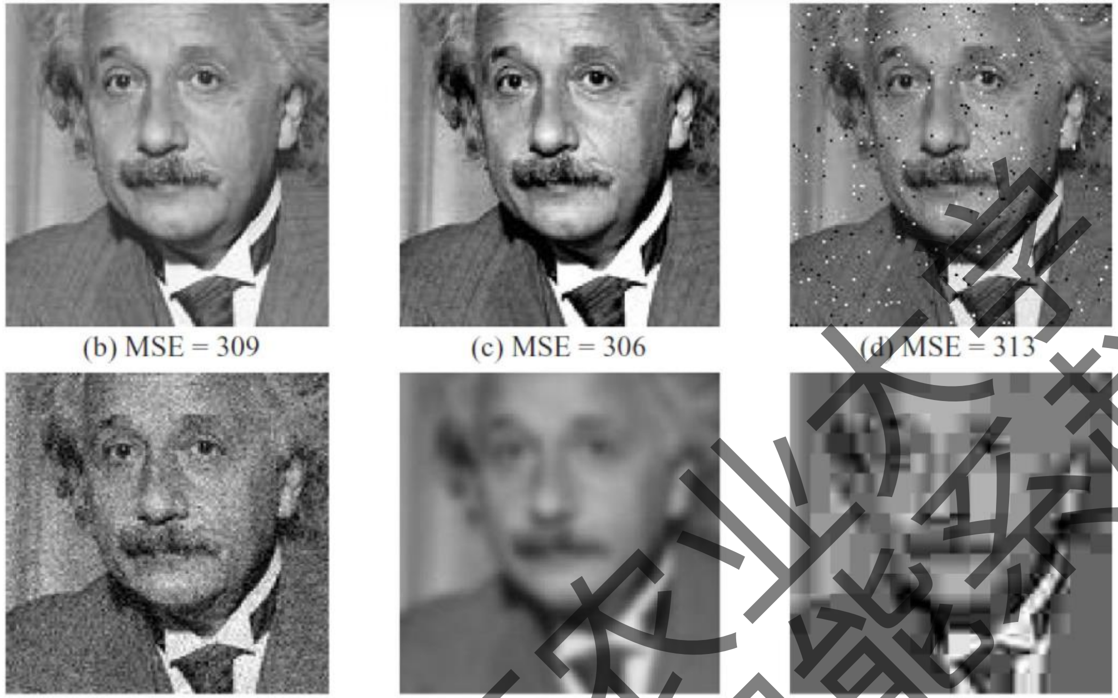
男性 女性





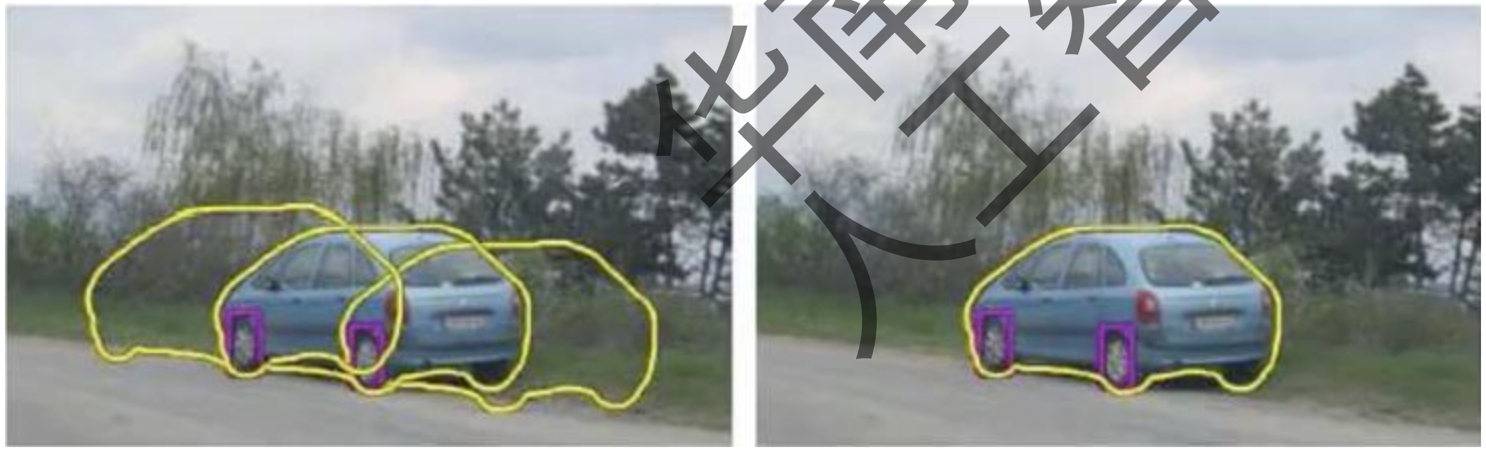
数据相关性分析

图像数据



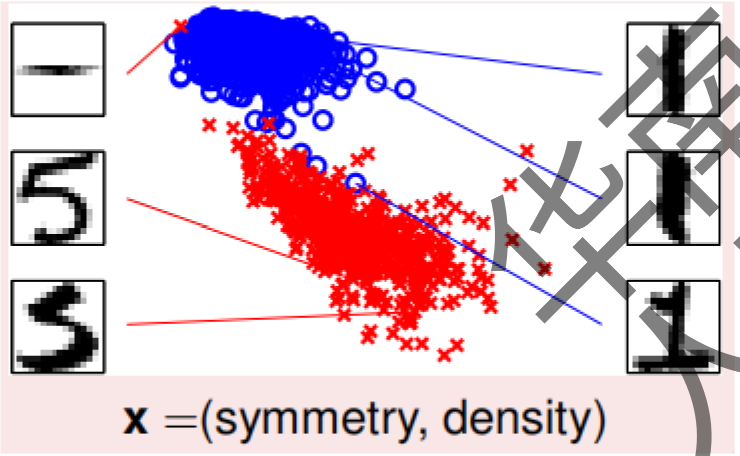
(b) MSE = 309 (c) MSE = 306 (d) MSE = 313

(e) MSE = 309 (f) MSE = 308 (g) MSE = 309



MNIST手写数据集

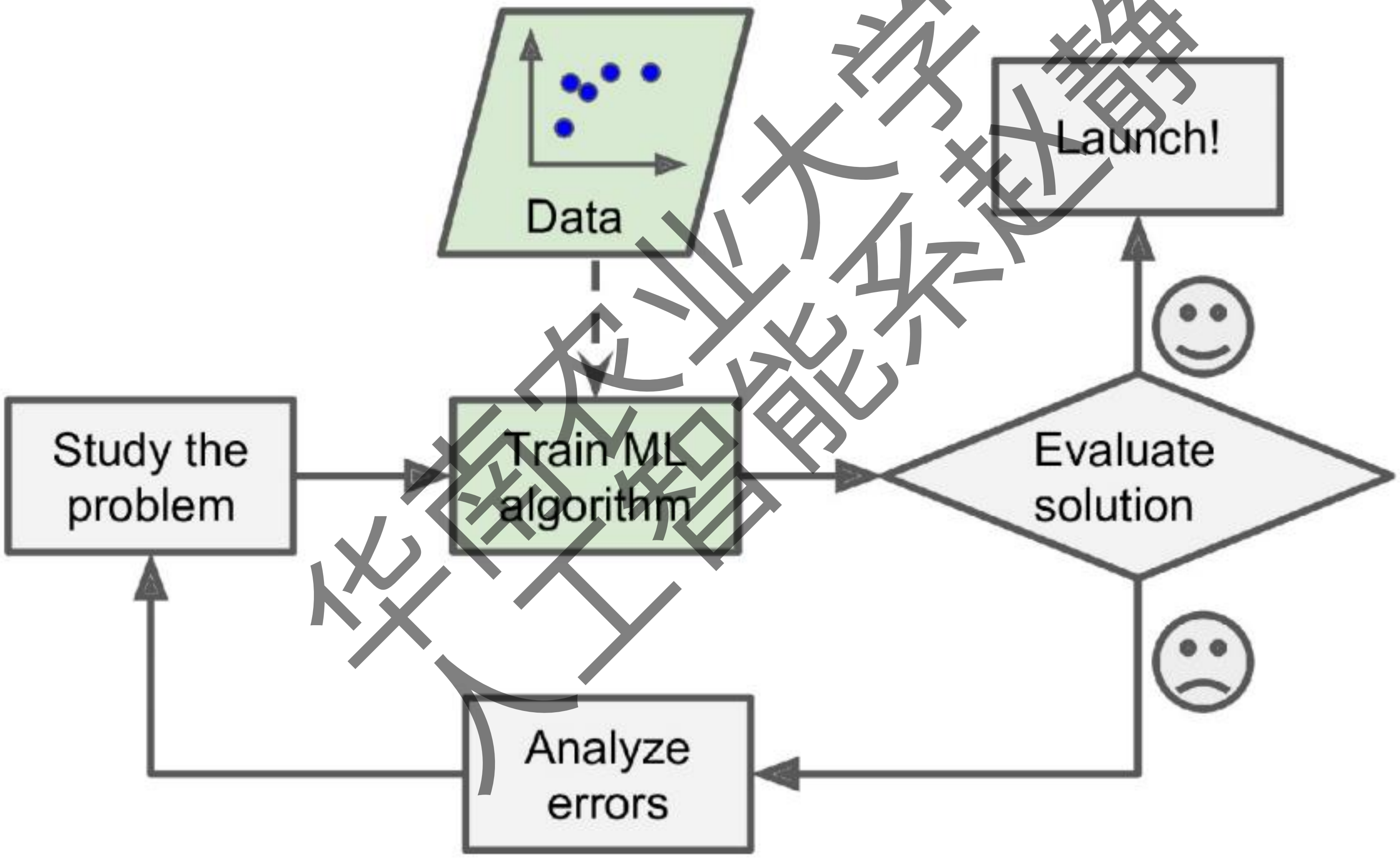
28*28=784像素，1*784的向量



具体特征

原始特征

三. 建模过程


Data
Study the problem
Train ML algorithm
Evaluate solution
Launch!
Analyze errors

ML≈ 寻找函数

- 股票预测

$f($  $)=$ 沪指涨跌

- 自动驾驶

$f($  $)=$ 方向盘角度

- 图像识别

$f($  $)=$ “猫”

机器学习的三要素

- 机器学习 ≈ 找到一个函数f

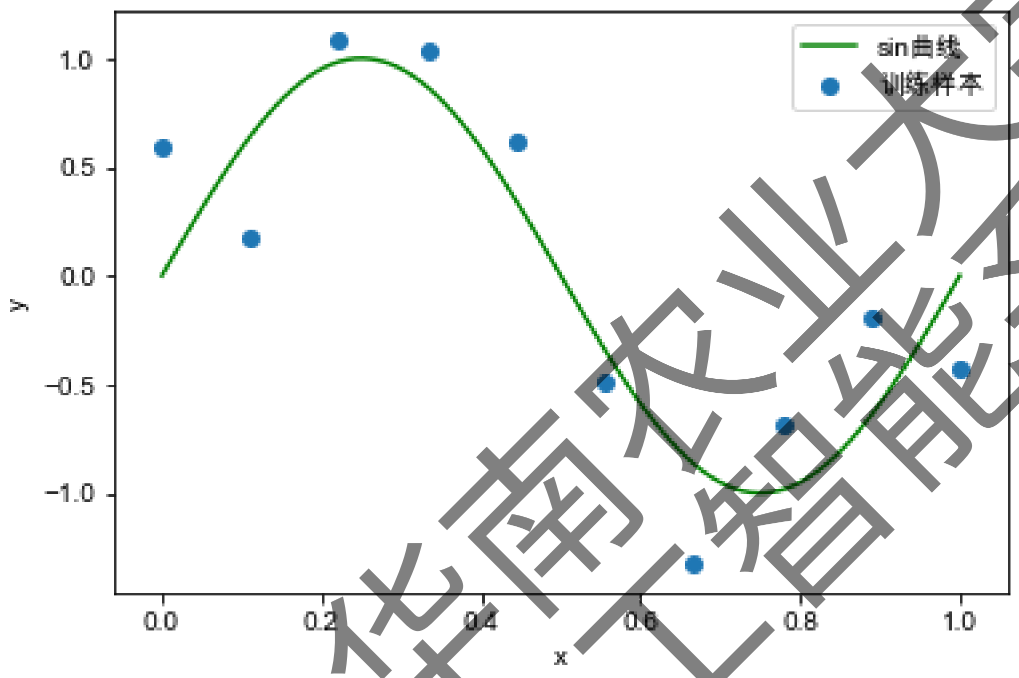
$f($  $) =$ “猫”

- 机器学习的三要素
  - 1. 函数集合$\{f_1, f_2, \ldots\}$
  - 2. 目标函数$J(f)$：函数的好坏
  - 3. 优化算法：找到最佳函数

$$f^* = \underset{f}{\operatorname{argmin}}\, J(f)$$

例：sin曲线拟合

- 训练数据： $y = \sin(2\pi x) + \varepsilon$



sin_fitting.ipynb

x	y
0	0.59752366
0.11111111,	0.17622454
0.22222222	1.07945343
0.33333333	1.03683804
0.44444444	0.62284356
0.55555556	-0.48839896
0.66666667	-1.32192983
0.77777778,	-0.68512136
0.88888889	-0.19685902
1	-0.42567777

1. 函数集合

- 1. 函数集合：3次多项式

$$\hat{y} = f_3(x, \boldsymbol{w}) = \sum_{j=1}^{3} w_j x^j + w_0 = \sum_{j=0}^{3} w_j x^j$$

2.目标函数：函数的好坏

- 1. 函数集合： 3次多项式

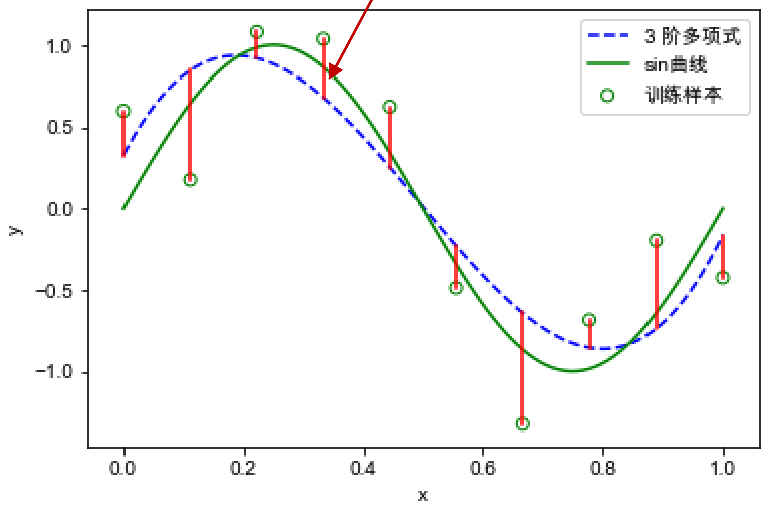
$$\hat{y} = f_3(x, \boldsymbol{w}) = \sum_{j=0}^{\mathbf{3}} w_j x^j$$

- 2. 目标函数$J(f)$： 函数的好坏
  - 损失函数$L(y, \hat{y}) = (y - \hat{y})^2$；

$$J(f) = L(y_i, \hat{y}_i) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \sum_{j=0}^{\mathbf{3}} w_j x^j\right)^2$$

预测残差$r = y - \hat{y}$



3. 优化算法：挑选最佳的函数

- 1. 函数集合：1个特征（x_1）的线性模型

$$\hat{y} = f_3(x, \boldsymbol{w}) = \sum_{j=0}^{\mathbf{3}} w_j x^j$$

- 2. 目标函数：函数的好坏

$$J(f) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- 3. 优化算法：挑选最佳的函数 $f^* = \underset{f}{\operatorname{argmin}} J(f)$

$$\boldsymbol{w}^* = \underset{f}{\operatorname{argmin}} J(f) = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \sum_{j=0}^{\mathbf{3}} w_j x^j\right)^2$$

最小二乘：
解析求解

最小二乘解析求解

- 目标函数为：$J(f) = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^{\mathrm{T}}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})$

$$\frac{\partial J}{\partial \boldsymbol{w}} = -2\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} + 2\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{w} = \boldsymbol{0}$$

- 得到：

$$\boxed{\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{w} = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}} \rightarrow \hat{\boldsymbol{w}}_{OLS} = \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$$

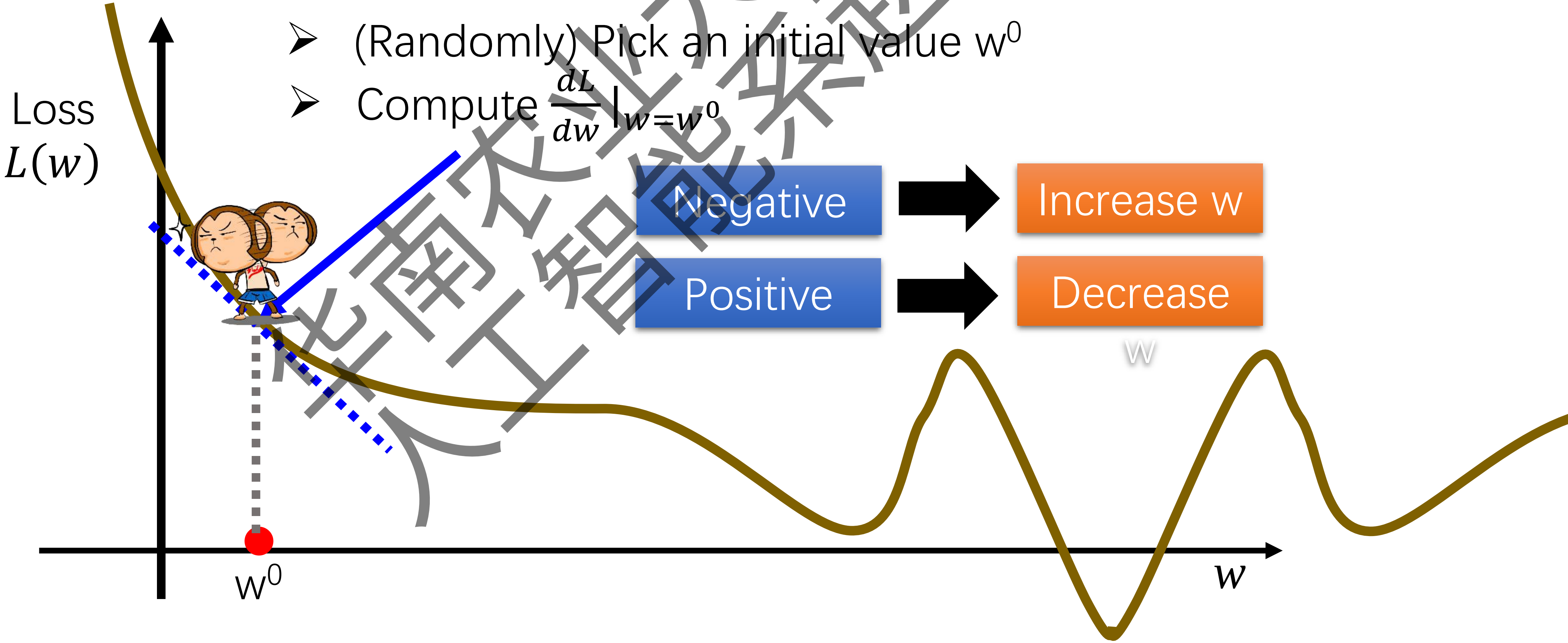
$$\left[\frac{\partial\left(\boldsymbol{y}^{\mathrm{T}}\boldsymbol{b}\right)}{\partial \boldsymbol{y}} = \boldsymbol{b}\right]$$

$$\left[\frac{\partial\left(\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{y}\right)}{\partial \boldsymbol{y}} = \left(A^{\mathrm{T}} + \boldsymbol{A}\right)\boldsymbol{y}\right]$$

梯度下降Gradient Descent

$$w^* = arg\min_{w} L(w)$$

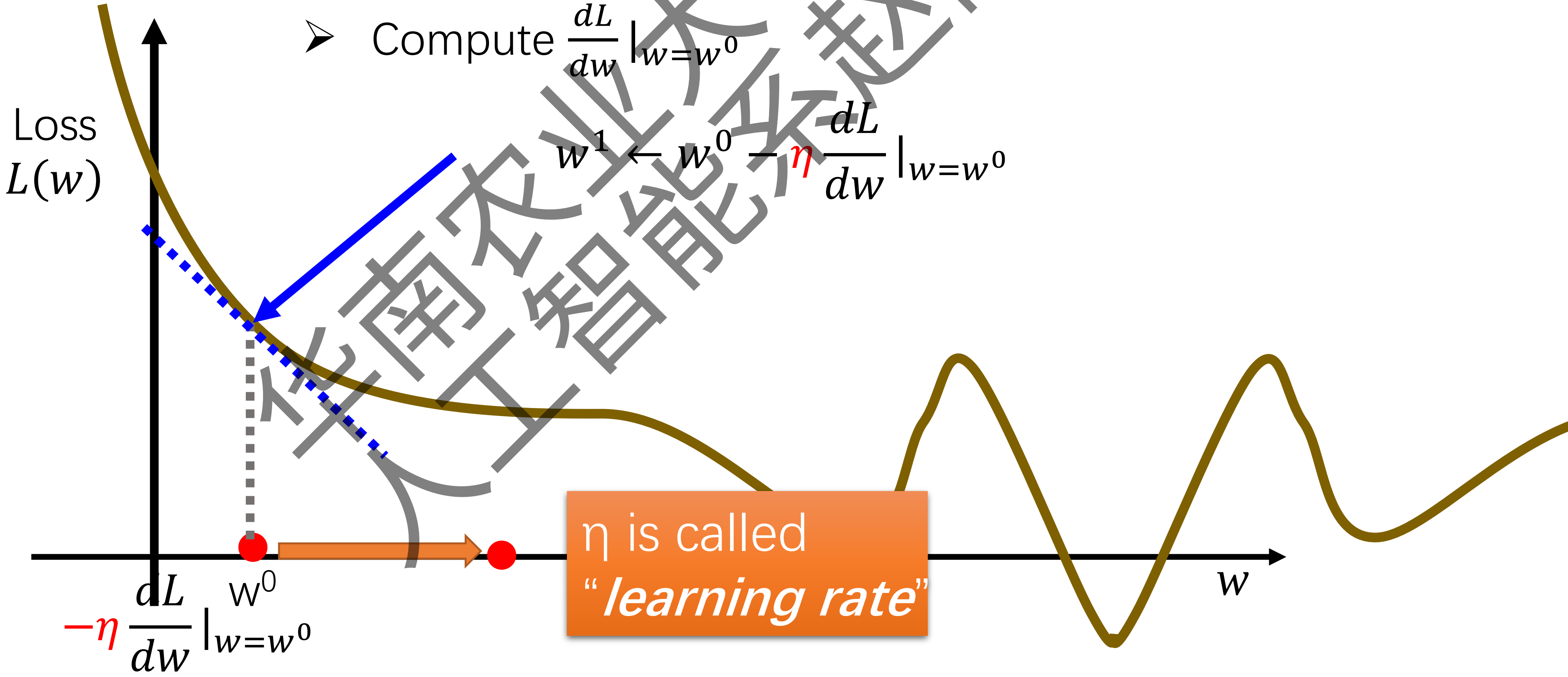
- (Randomly) Pick an initial value w^0
- Compute $\frac{dL}{dw}|_{w=w^0}$



$$w^* = arg\min_{w} L(w)$$

- Consider loss function $L(w)$ with one parameter w:
  - ➢ (Randomly) Pick an initial value w^0
  - ➢ Compute $\frac{dL}{dw}|_{w=w^0}$

$$w^1 \leftarrow w^0 - \eta \frac{dL}{dw}|_{w=w^0}$$



$$w^* = arg\min_{w} L(w)$$

- Consider loss function $L(w)$ with one parameter w:

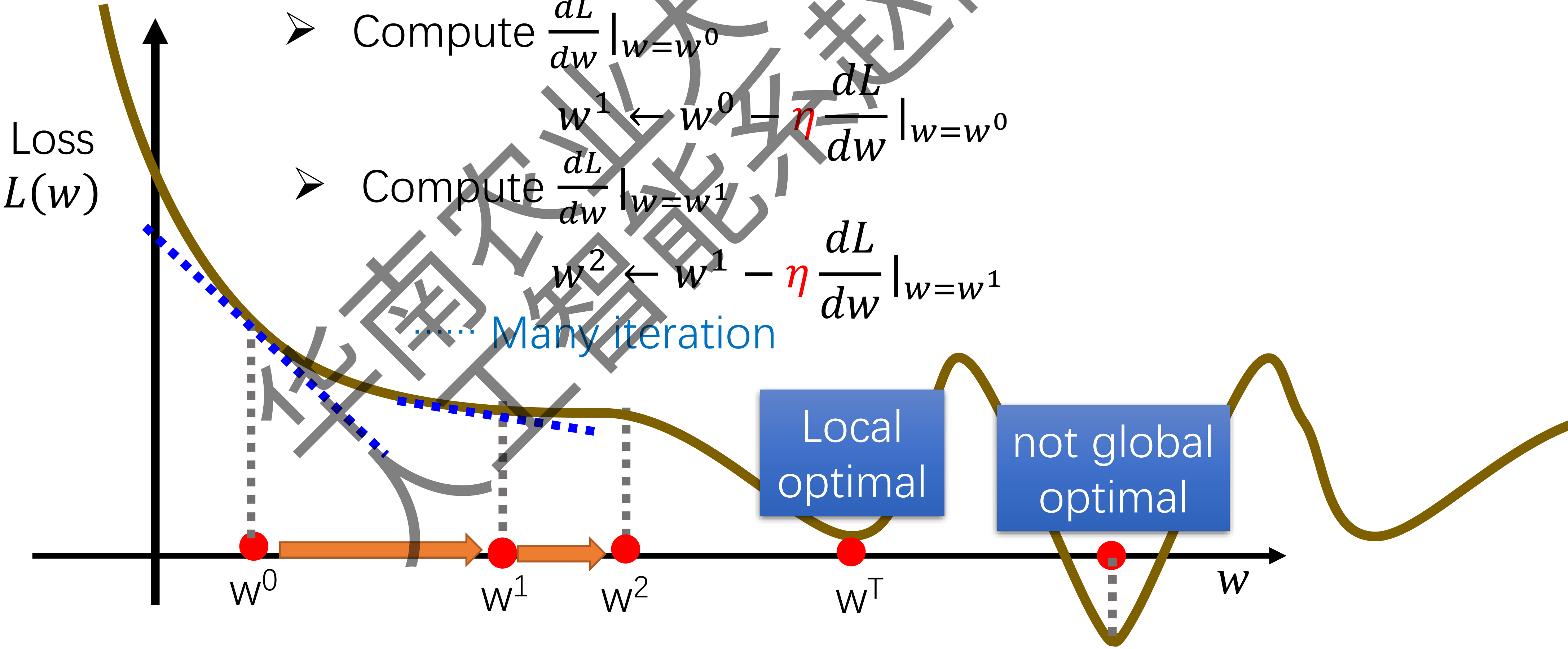
  - (Randomly) Pick an initial value w^0
  - Compute $\frac{dL}{dw}|_{w=w^0}$

$$w^1 \leftarrow w^0 - \eta \frac{dL}{dw}|_{w=w^0}$$

  - Compute $\frac{dL}{dw}|_{w=w^1}$

$$w^2 \leftarrow w^1 - \eta \frac{dL}{dw}|_{w=w^1}$$

…… Many iteration



- How about two parameters?

$$w^*, b^* = arg \min_{w,b} L(w, b)$$

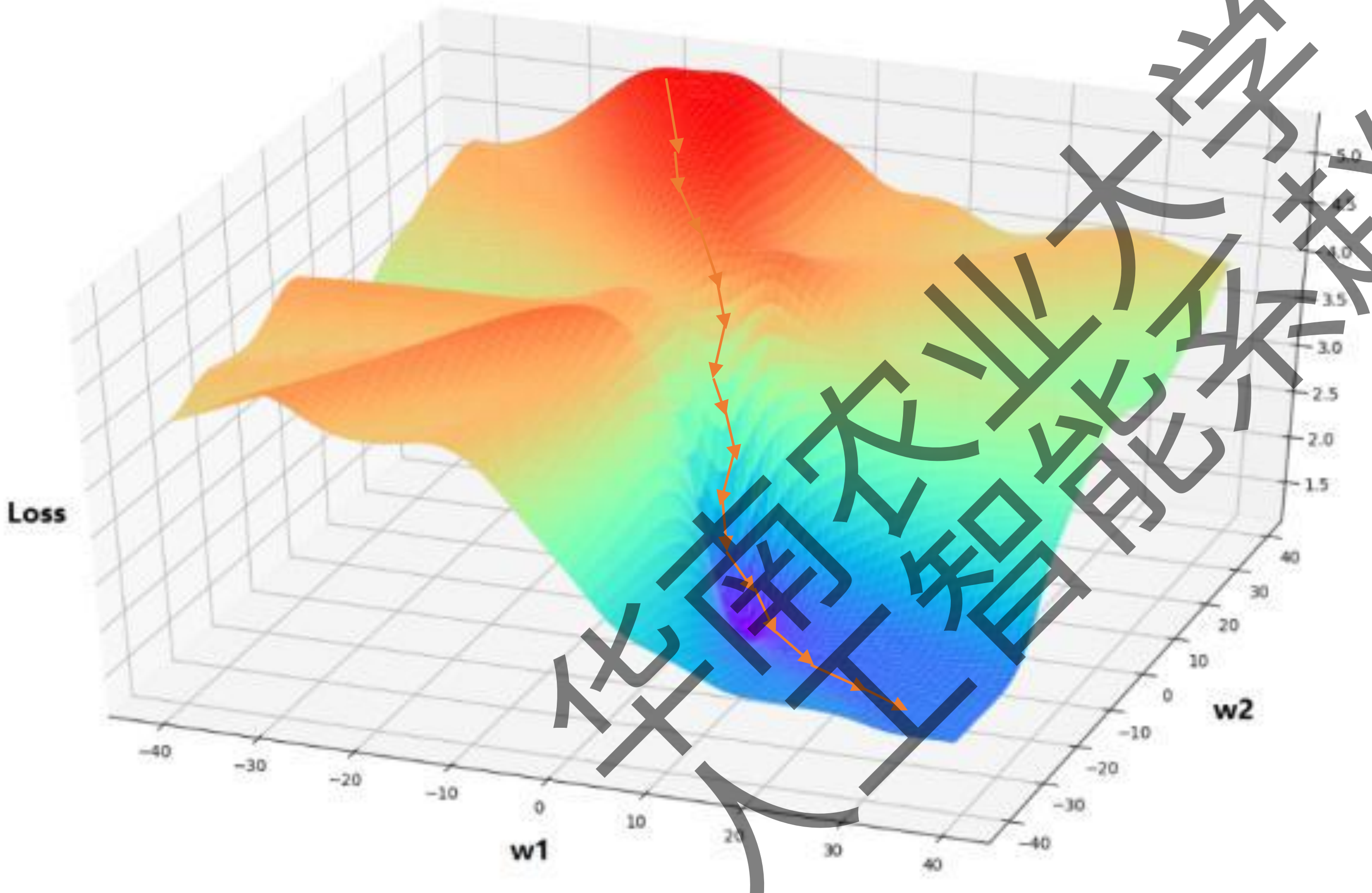
$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w} \\ \frac{\partial L}{\partial b} \end{bmatrix} \text{gradient}$$

- ➢ (Randomly) Pick an initial value w^0, b^0
- ➢ Compute $\frac{\partial L}{\partial w}|_{w=w^0,b=b^0}, \frac{\partial L}{\partial b}|_{w=w^0,b=b^0}$

$$w^1 \leftarrow w^0 - \eta \frac{\partial L}{\partial w}|_{w=w^0,b=b^0} \qquad b^1 \leftarrow b^0 - \eta \frac{\partial L}{\partial b}|_{w=w^0,b=b^0}$$

- ➢ Compute $\frac{\partial L}{\partial w}|_{w=w^1,b=b^1}, \frac{\partial L}{\partial b}|_{w=w^1,b=b^1}$

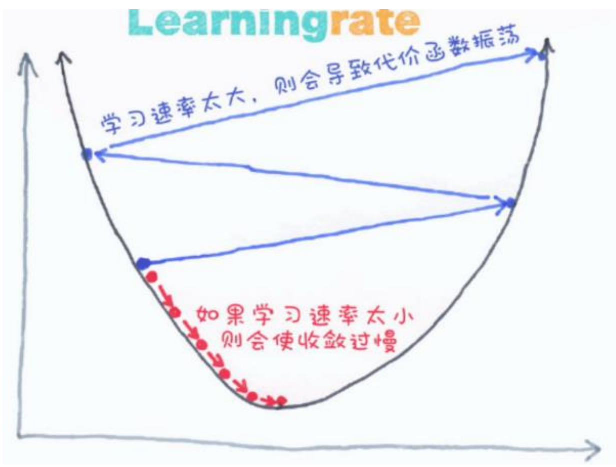
$$w^2 \leftarrow w^1 - \eta \frac{\partial L}{\partial w}|_{w=w^1,b=b^1} \qquad b^2 \leftarrow b^1 - \eta \frac{\partial L}{\partial b}|_{w=w^1,b=b^1}$$

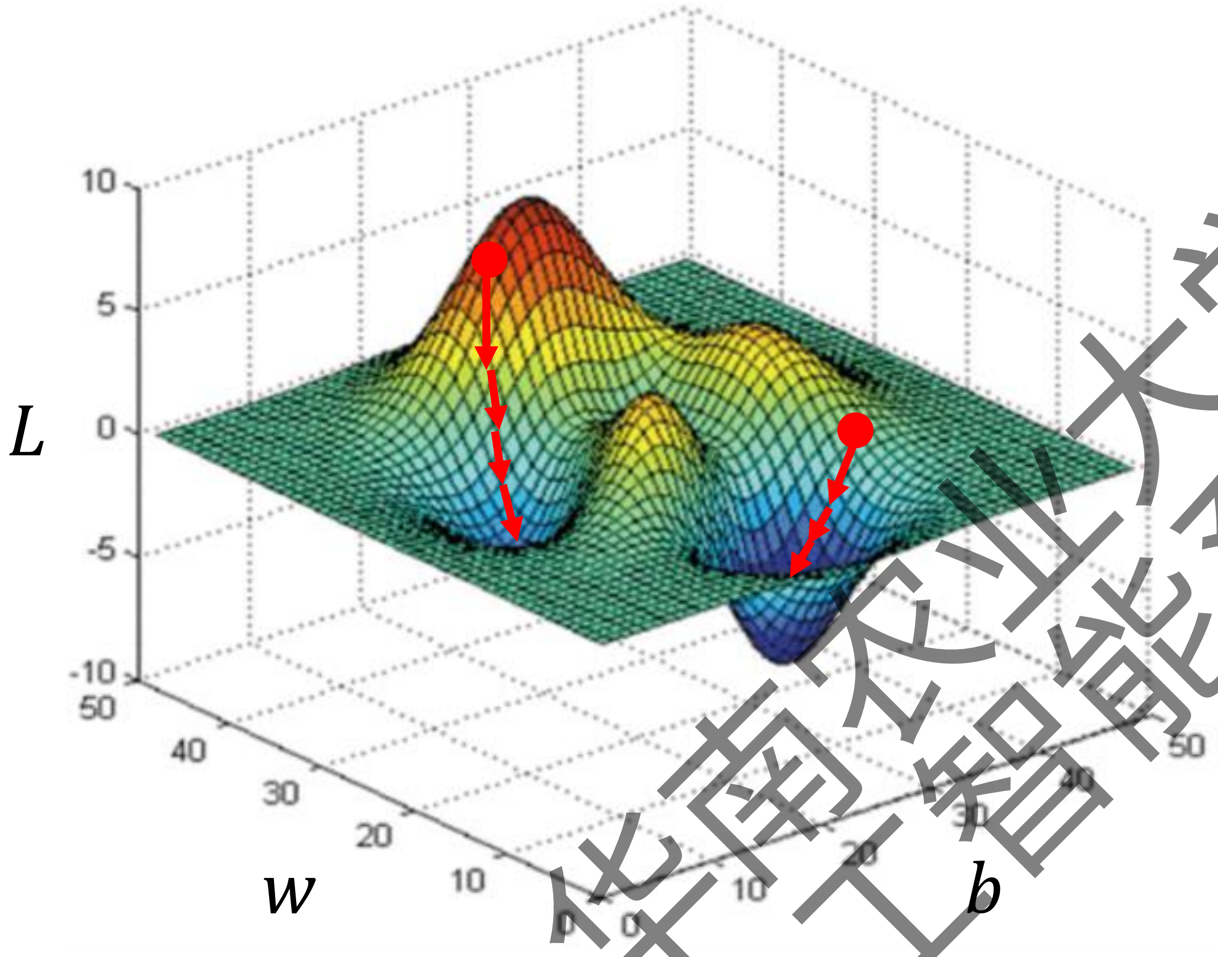

Loss
w1
w2

学习率

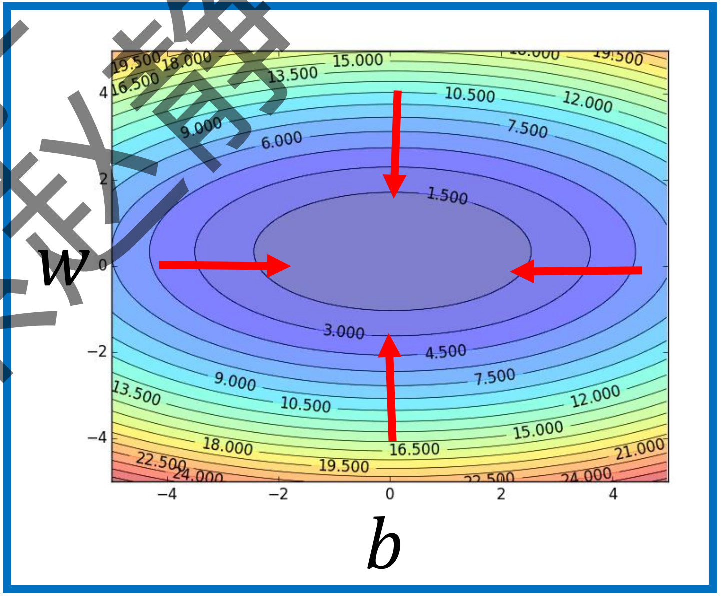
η

步长


Learningrate
学习速率太大，则会导致代价函数振荡
如果学习速率太小
则会使收敛过慢



L
w
b
10
5
0
-5
-10
50
40
30
20
10
0
0
10
20
30
40
50



w
b
15.000
13.500
10.500
12.000
9.000
7.500
6.000
1.500
3.000
4.500
16.500
18.000
19.500
21.000
22.500
24.000
−4
−2
0
2
4

解决凸优化问题

注：“调参”与最终模型

算法的参数：一般由人工设定，亦称“超参数”

模型的参数：一般由学习确定

参数调得好不好往往对

模型最终性能有关键影响

梯度下降的三种形式

批量梯度下降（Batch Gradient Descent,BGD）

梯度下降的每一步中，都用到了**所有**的训练样本

随机梯度下降（Stochastic Gradient Descent,SGD）

梯度下降的每一步中，用到**一个**样本，在每一次计算之后便更新参数，而不需要首先将所有的训练集求和

小批量梯度下降（Mini-Batch Gradient Descent,MBGD）

梯度下降的每一步中，用到了**一定批量**的训练样本

梯度下降与最小二乘法比较

梯度下降：需要选择学习率η，需要多次迭代，当特征数量n大时也能较好适用，适用于各种类型的模型。

最小二乘法：不需要选择学习率η，一次计算得出，需要计算$(X^TX)^{-1}$，如果特征数量n较大则运算代价大，因为矩阵逆的计算时间复杂度为$O(n^3)$，通常来说当n小于10000时还是可以接受的，只适用于线性模型，不适合逻辑回归模型等其他模型。

Fun Time

监督学习中，分类和回归最本质的不同在于：

1. 特征数据
2. 标签数据
3. 目标函数
4. 优化算法

结果

- 最佳函数（模型）：

$$\hat{y} = 15.64284771x^3 - 23.18415538x^2 + 7.05019844x + 0$$

训练集上的平均损失:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= 0.172869$$

测试集上的平均损失:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= 0.138302$$

还能更好?

回到第1步：重新设计函数集合

- 函数集合：5阶多项式

$$\hat{y} = \sum_{j=0}^{5} w_j x^j$$

训练集上的平均损失：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= 0.013075$$

测试集（200个样本）上的平均损失：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= 0.1930624$$

回到第1步：重新设计函数集合

- 函数集合：9阶多项式

$$\hat{y} = \sum_{j=0}^{9} w_j x^j$$

训练集上的平均损失：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = 0$$

测试集上的平均损失：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$= 0.274700$$

四. 模型选择

➢ 过拟合vs欠拟合

1. $$\hat{y} = \sum_{j=0}^{3} w_j x^j$$

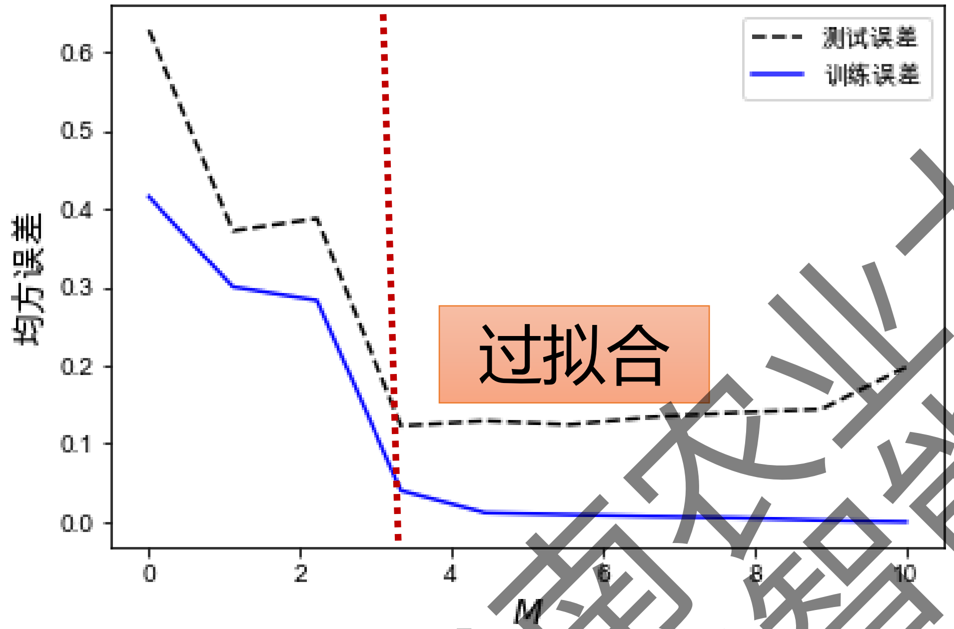
2. $$\hat{y} = \sum_{j=0}^{5} w_j x^j$$

3. $$\hat{y} = \sum_{j=0}^{9} w_j x^j$$

	Training	Testing
1	0.13	0.17
2	0.013	0.193
3	0	0.274

复杂模型能在训练集上得到更小的损失。

模型选择



	Training	Testing
1	0.13	0.17
2	0.013	0.193
3	0	0.274

复杂模型并不能在测试集上得到更好的性能。

过拟合 → 选择合适的模型

泛化误差（Generalization error） vs. 经验误差（empirical error）

泛化误差：在“未来”样本上的误差

经验误差：在训练集上的误差，亦称“训练误差”

- 泛化误差越小越好
- 经验误差是否越小越好?

NO! 因为会出现“**过拟合**”(overfitting)

过拟合 (overfitting) vs. 欠拟合 (underfitting)

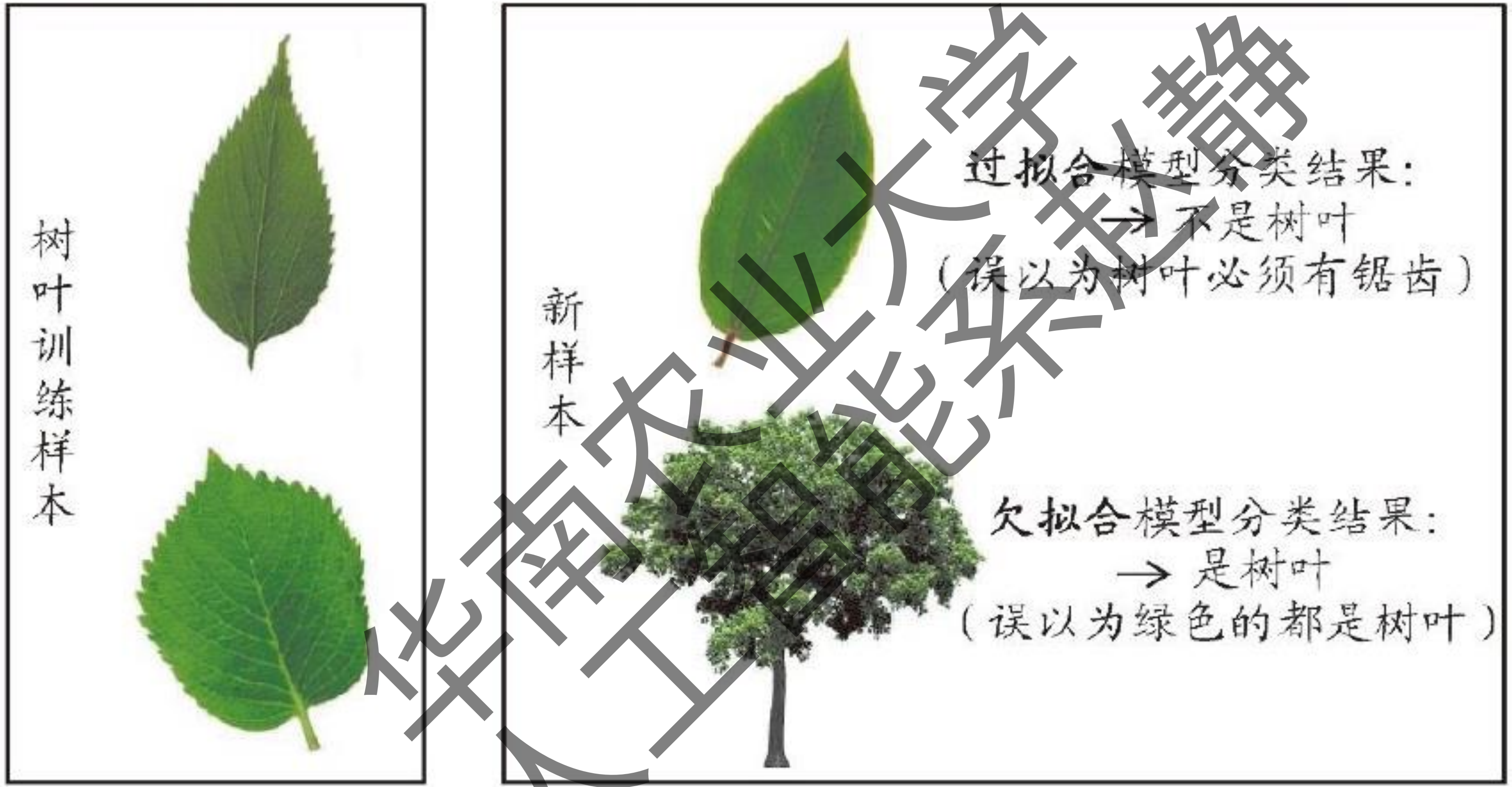


图 2.1 过拟合、欠拟合的直观类比

#周志华

回到第2步：重新设计目标函数

- 函数集合：$\hat{y} = \sum_{j=0}^{M} w_j x^j$
- 目标函数：度量函数的好坏

$$J(\mathbf{w}, \lambda) = \sum_{i=1}^{N} L(y_i, \hat{y}_i) + \lambda R(\mathbf{w})$$

- 损失函数

$$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

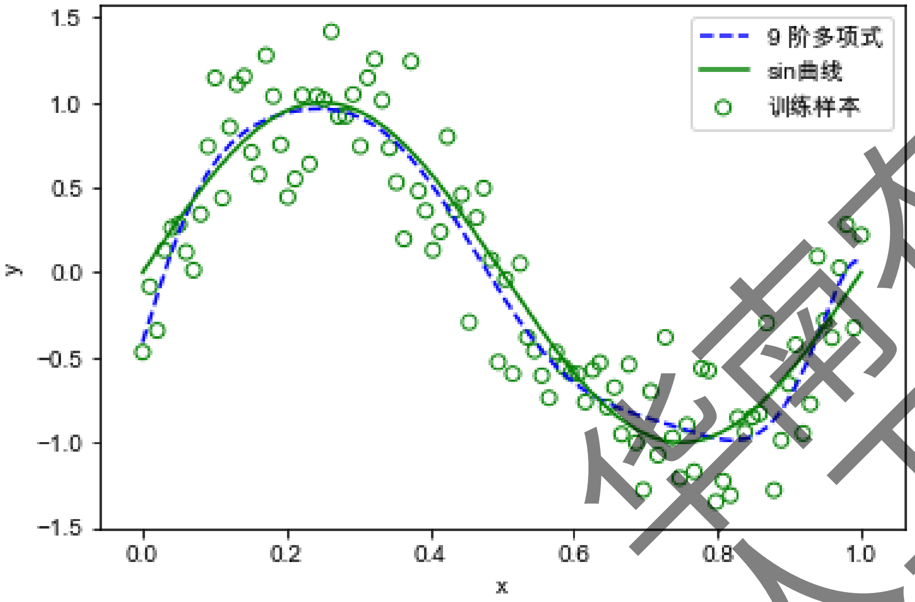
- 正则项：
  - L2正则：$R(\mathbf{w}) = \|\mathbf{w}\|_2^2 = \sum_{j=1}^{M} w_j^2$
  - L1正则：$R(\mathbf{w}) = \|\mathbf{w}\|_1 = \sum_{j=1}^{M} |w_j|$

更小的w_j 对应的函数更好

训练样本数目

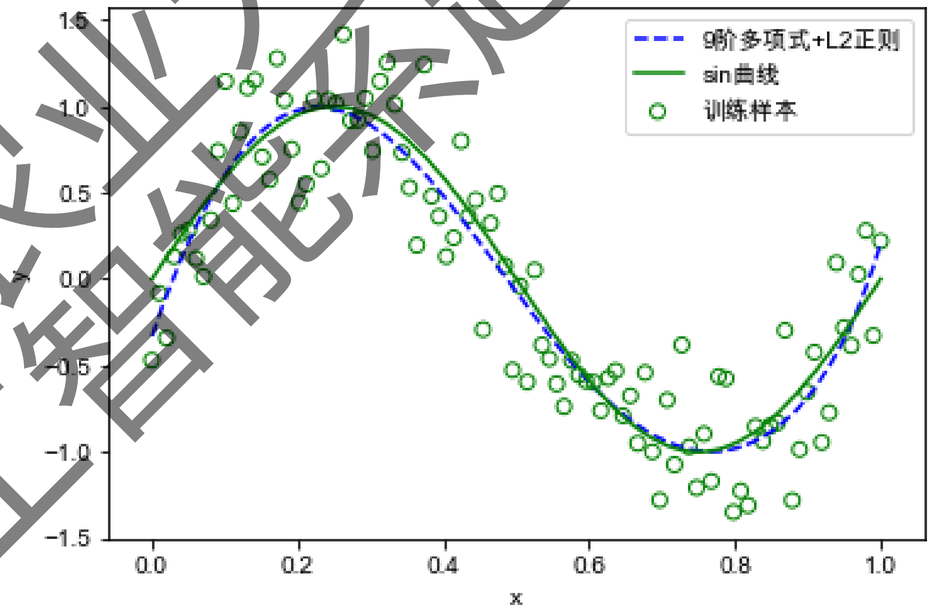
- 训练样本数目$N = 100$（增大）

无正则



$$MSE = 0.091891$$

L2正则

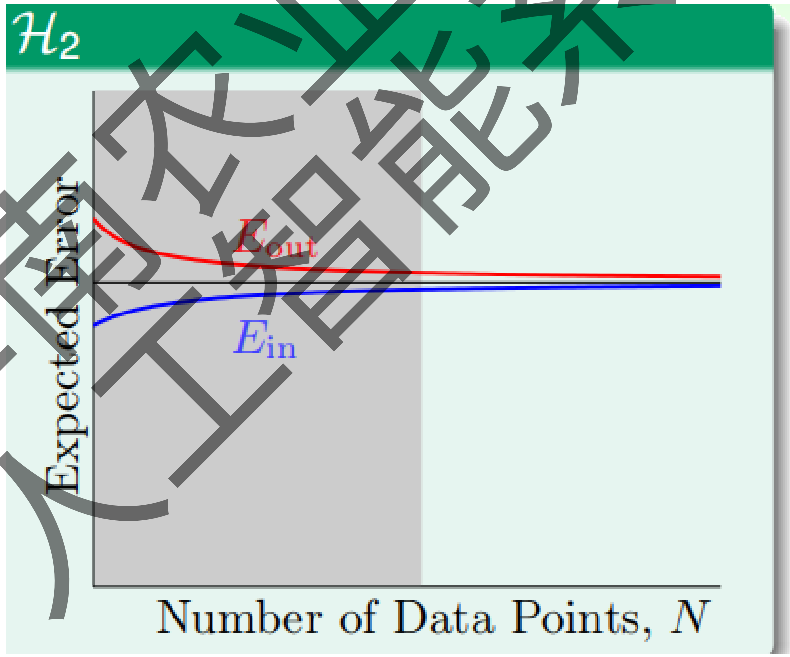


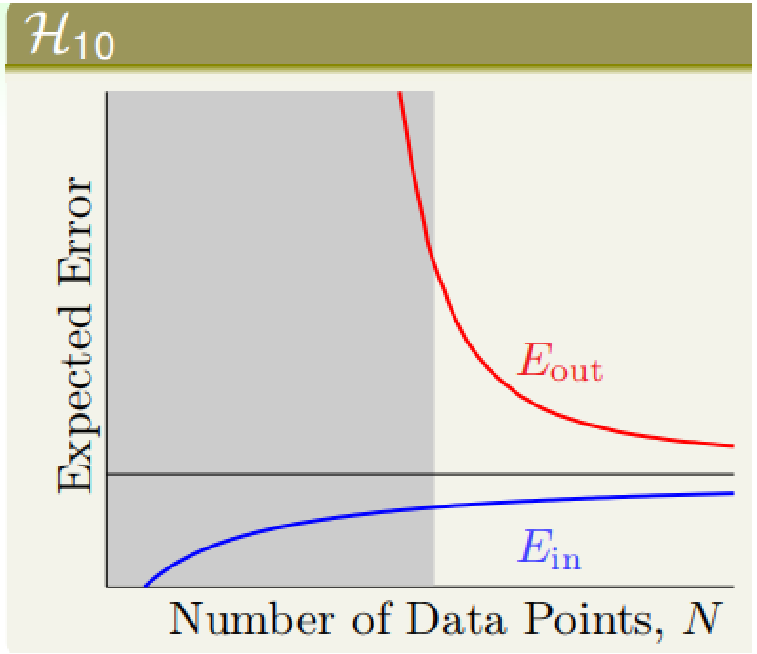
$$MSE = 0.091645$$

当训练样本数目无穷大时，带正则和无正则最优模型相同。

总结:引起过拟合的原因

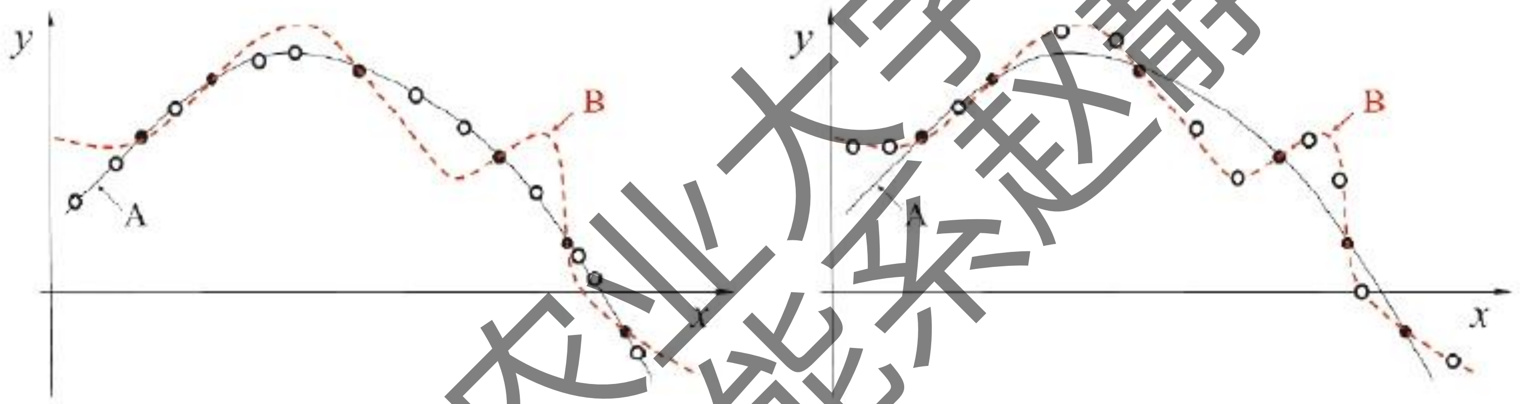
1. 更高阶的拟合函数;
2. 训练样本N太少;
3. 数据噪声。





➢ NFL定理（No Free Lunch Theorem）

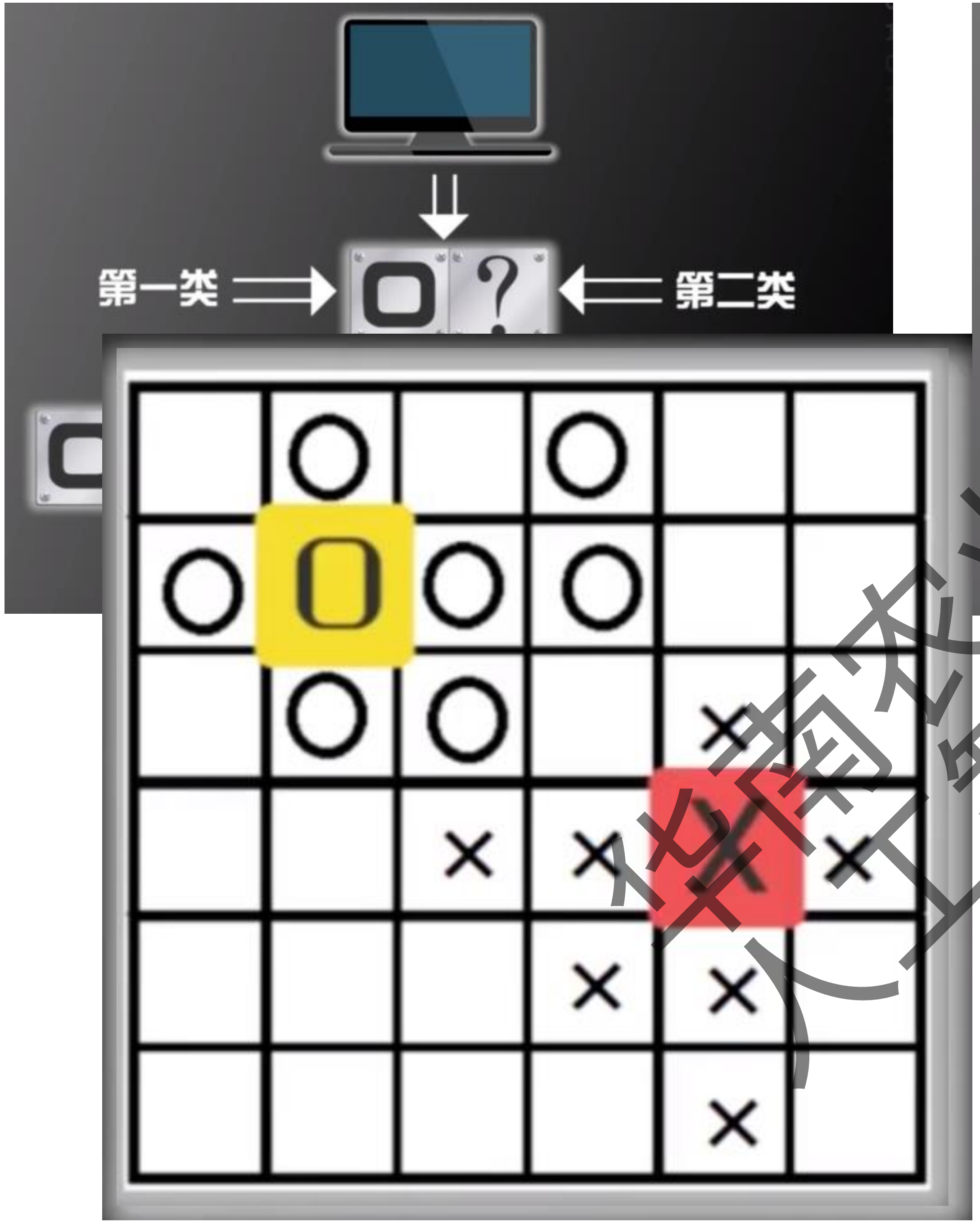
哪个算法更好

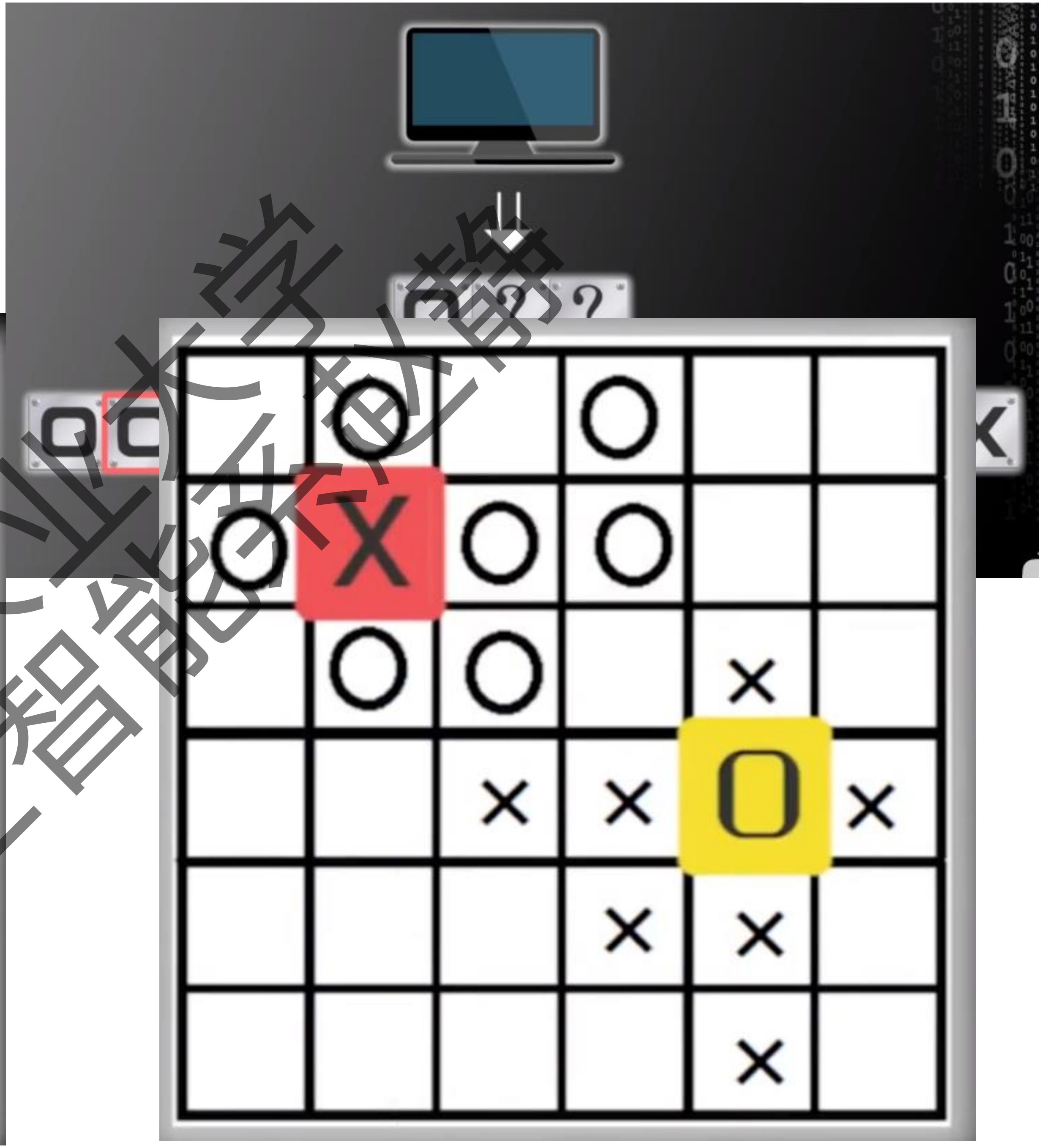


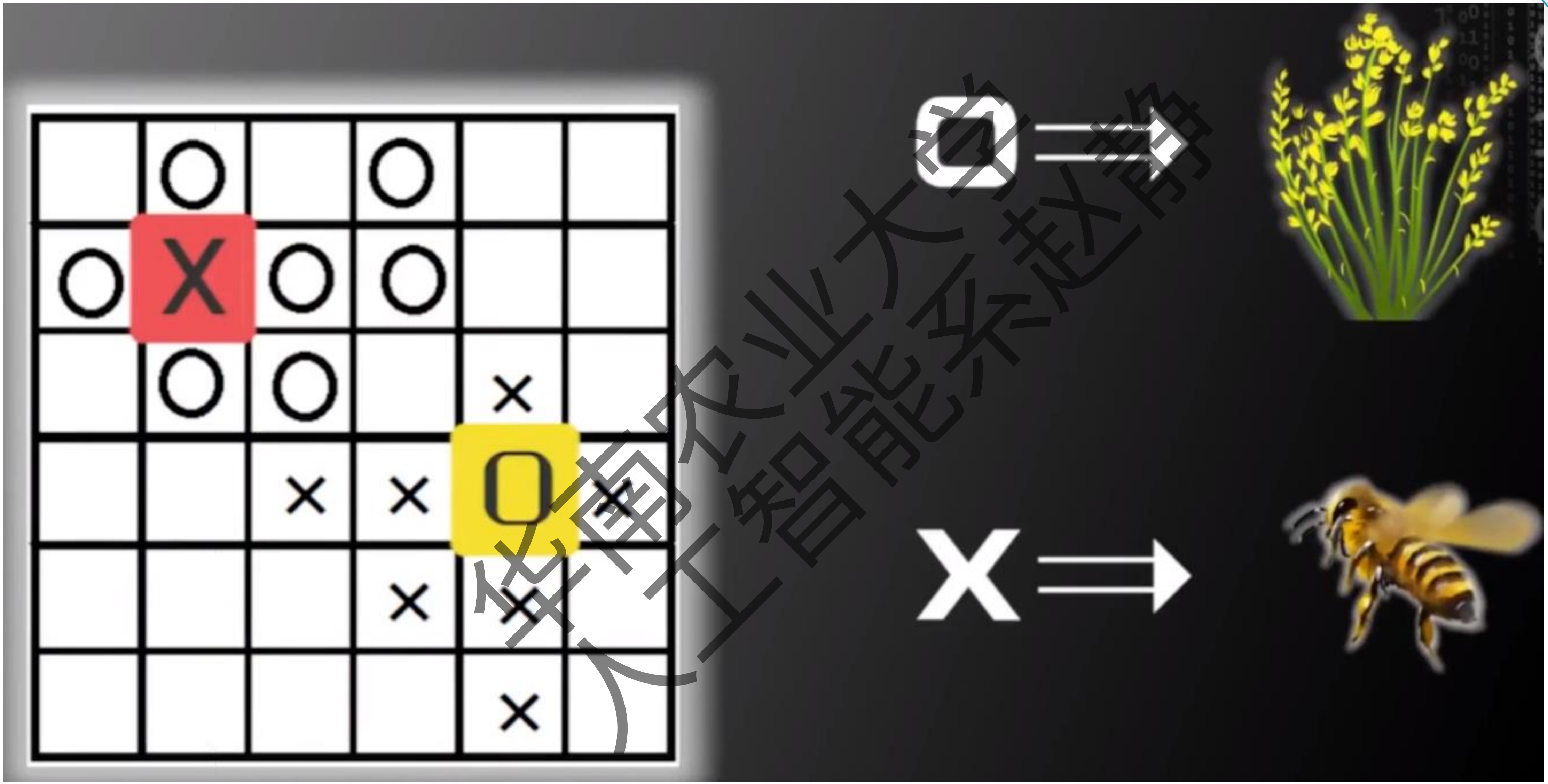
(a) A 优于 B　　　　(b) B 优于 A

(黑点：训练样本；白点：测试样本)

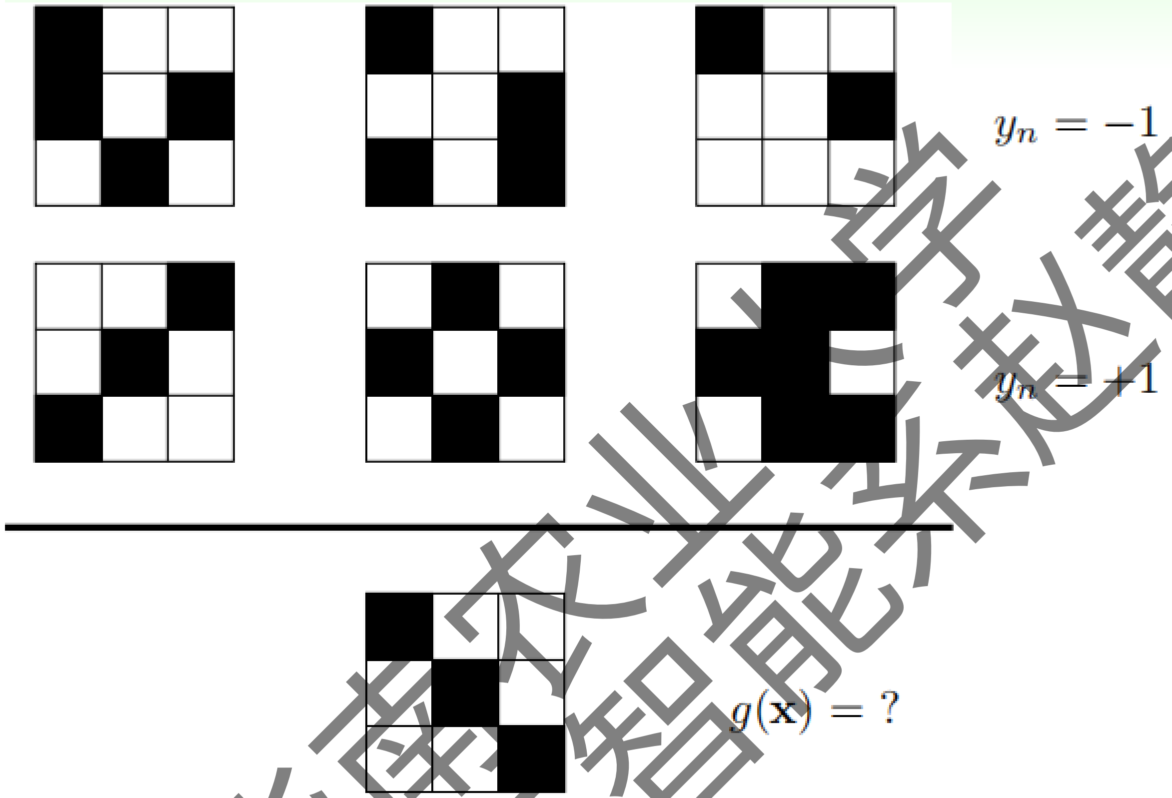
没有免费的午餐！


第一类
第二类





#胡浩基


$y_n = -1$
$y_n = +1$
$g(\mathbf{x}) = ?$

没有免费午餐定理（No Free Lunch Theorem）

NFL定理：一个算法 $\mathfrak{L}_a$ 若在某些问题上比另一个算法 $\mathfrak{L}_b$ 好，必存在另一些问题，$\mathfrak{L}_b$ 比 $\mathfrak{L}_a$ 好。

脱离具体问题，空泛地谈论

“什么学习算法更好”毫无意义

总误差与学习算法无关

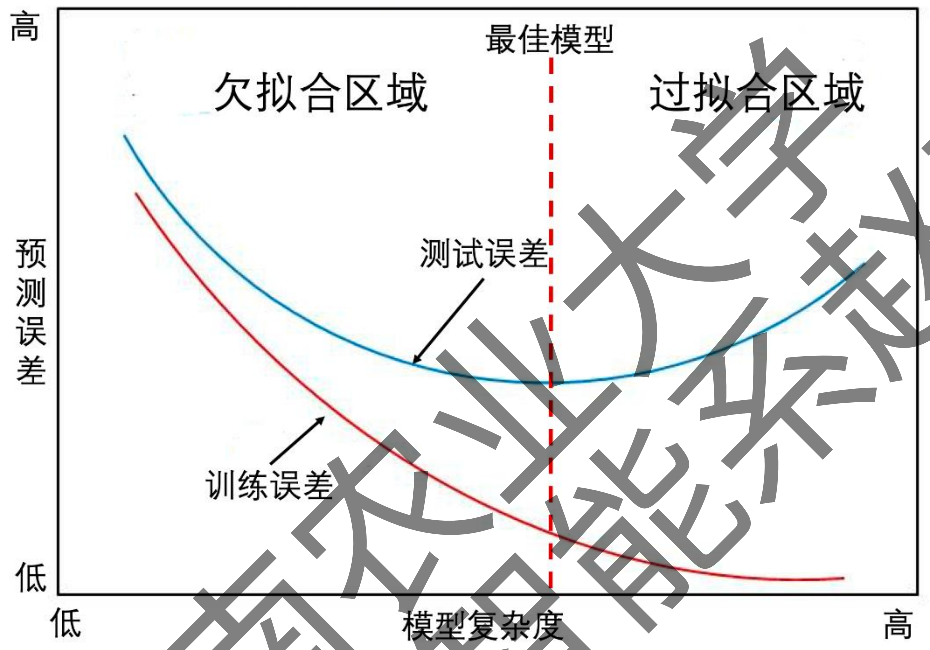
⇩

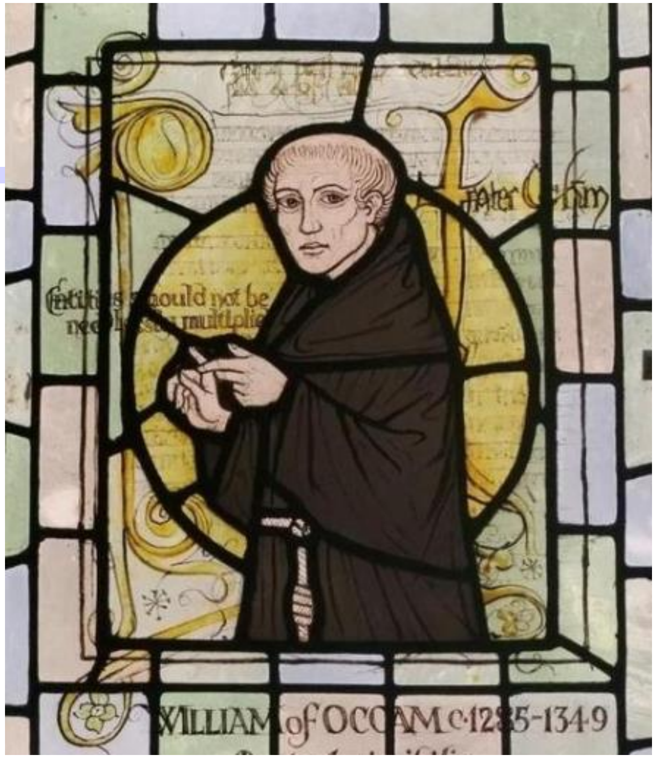
所有算法一样

➢ 奥卡姆剃刀原理(Ocam’s razor)

归纳偏好

(inductive bias)





任何一个有效的机器学习算法必有其偏好

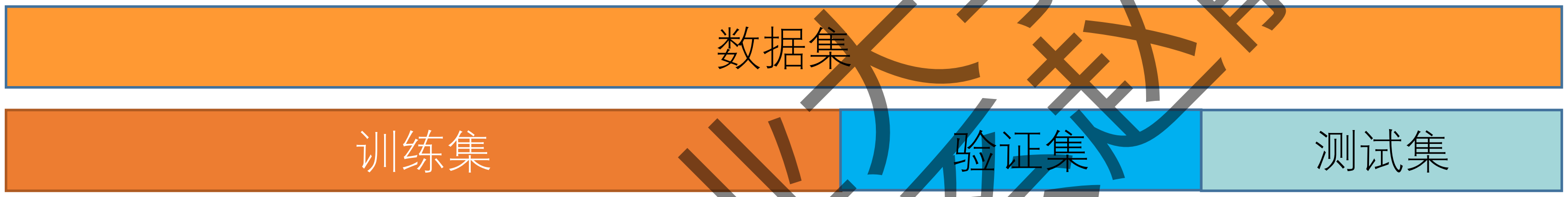
学习算法的归纳偏好是否与问题本身匹配，
大多数时候直接决定了算法能否取得好的性能！

总结：机器学习建模

- 给定任务，分析可能的相关特征，收集训练数据
- 特征工程
- 确定函数集合
- 确定目标函数
  - 损失函数
  - 正则项
- 模型训练，确定优化算法
  - 根据训练数据得到模型参数
- 模型选择：在验证集上评估模型预测性能

机器学习任务是一个迭代的过程。

五. 模型评估

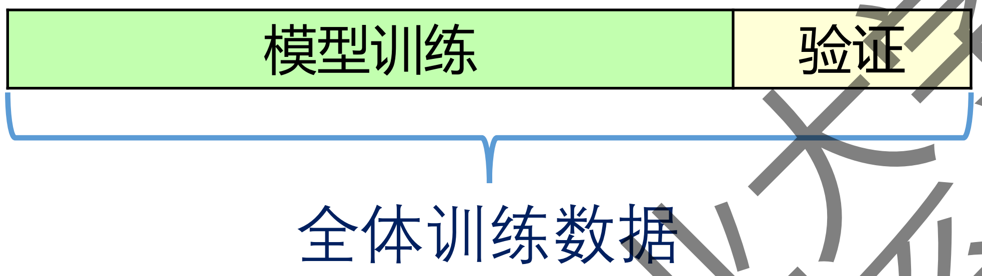


数据划分

测试集应该与训练集“互斥”

➢ 数据划分

划分方法——留出法



留出法：当训练数据很多，留出一部分数据做验证集后，训练集样本还足够时，可以采用上述方式得到验证集。

注意：

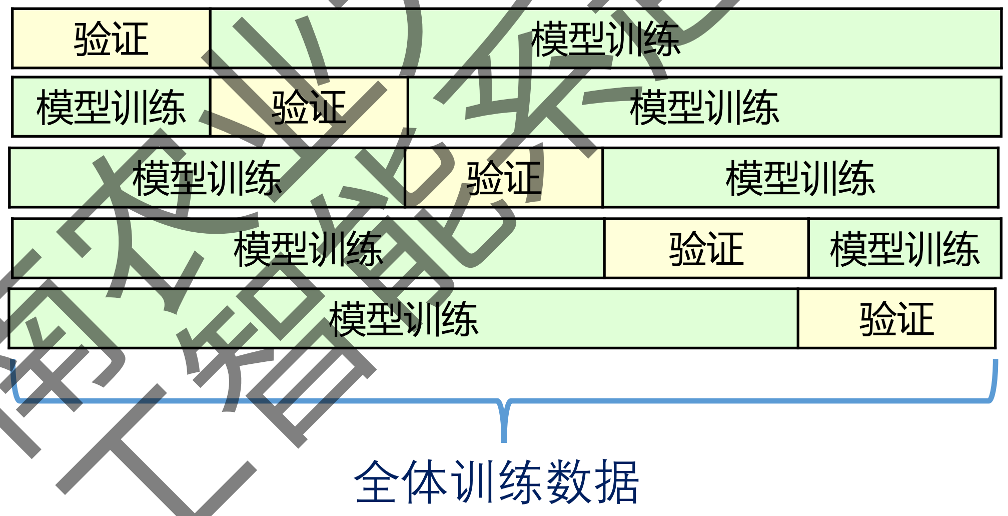
- 保持数据分布一致性 （例如: 分层采样)
- 多次重复划分 (例如: 100次随机划分)
- 测试集不能太大、不能太小 (例如：1/5~1/3)

划分方法——交叉验证

- 从训练集中分离一部分数据作为验证集，会使得模型训练能使用的数据减少（机器学习中训练样本越多越好）。有更好的办法?

K折交叉验证

KFold



相比留出法，
交叉验证计算代价高，
但不会浪费太多的数据，
当样本数据集较少时有优势。

- 模型选择好后，再用所有的训练数据训练模型参数。

➢ 性能度量 ——衡量模型泛化能力的评价标准，反映任务需求

✓ 回归(regression) 任务常用均方误差：

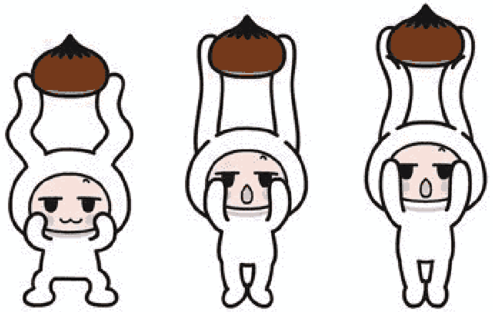
$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}\left(f(\boldsymbol{x}_i)-y_i\right)^2$$

✓ 分类任务常用错误率：

$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}\mathbb{I}\left(f(\boldsymbol{x}_i)\neq y_i\right)$$

✓ 精度：

$$\begin{aligned}\operatorname{acc}(f;D) &= \frac{1}{m}\sum_{i=1}^{m}\mathbb{I}\left(f\left(\boldsymbol{x}_i\right)=y_i\right)\\ &= 1-E(f;D)\ .\end{aligned}$$



$$P(y=1|\mathbf{x}) = 0.2, P(y=2|\mathbf{x}) = 0.7, P(y=3|\mathbf{x}) = 0.1$$

$$\mathrm{err}(\tilde{y}, y) = [\![\tilde{y} \neq y]\!]$$

$$\tilde{y} = \begin{cases} 1 & \text{err } 0.8 \\ 2 & \text{err } 0.3 \\ 3 & \text{err } 0.9 \\ 1.9 & \text{err } 1.0 \end{cases}$$

$$\mathrm{err}(\tilde{y}, y) = (\tilde{y} - y)^2$$

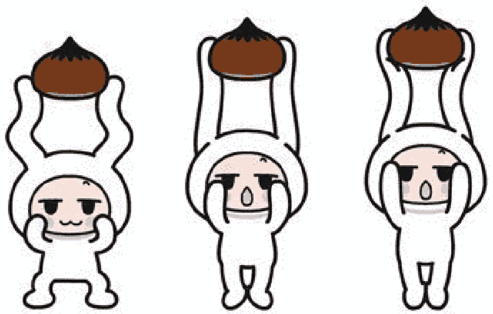
$$\tilde{y} = \begin{cases} 1 & \text{err } 1.1 \\ 2 & \text{err } 0.3 \\ 3 & \text{err } 1.5 \\ 1.9 & \text{err } 0.29 \end{cases}$$

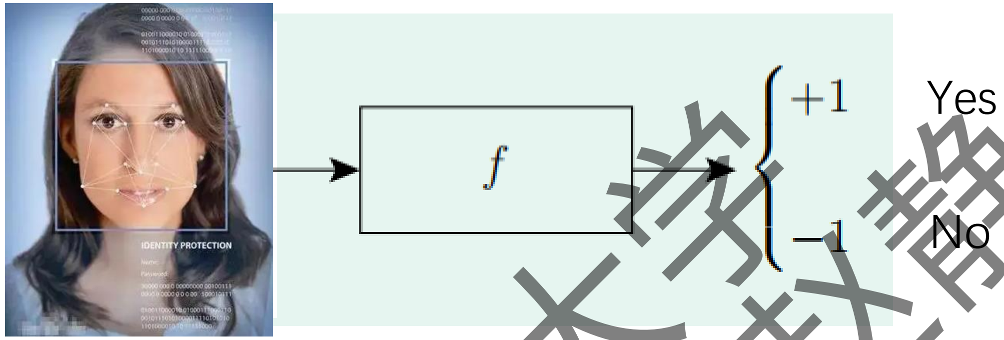
✓ 查准率（precision） vs. 查全率（recall）

表 2.1 分类结果混淆矩阵

真实情况	预测结果	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

- 查准率： $$P = \frac{TP}{TP + FP}$$
- 查全率： $$R = \frac{TP}{TP + FN}$$





- 超市VIP的折扣

查全率！

		g	
		+1	-1
f	+1	0	10
	-1	1	0

真实情况f	预测结果g	
	正例+1	反例-1
正例+1	√（TP）	错误的拒绝（FN）
反例-1	错误的接受（FP）	√（TN）

- 国家安全局资料库

查准率！

		g	
		+1	-1
f	+1	0	1
	-1	1000	0

混淆矩阵

真值	$\hat{y}=1$	$\hat{y}=0$	Σ
$y=1$	TP	FN	N_+
$y=0$	FP	TN	N_-
Σ	$\hat{N}_+$	$\hat{N}_-$	

预测值

PR曲线

查准率、精度、准确率：预测结果为真的样本中真正为真的比例

$$\text{Precison} = \frac{TP}{\hat{N}_+}$$

查全率、召回率：预测结果召回了多少真正的真样本；真阳率：有多少真正的正样本被预测为真

$$\text{TPR} = \text{Recall} = \frac{TP}{N_+}$$

假阳率：预测结果将多少假的样本预测预测成了真

$$\text{FPR} = \frac{FP}{N_-}$$

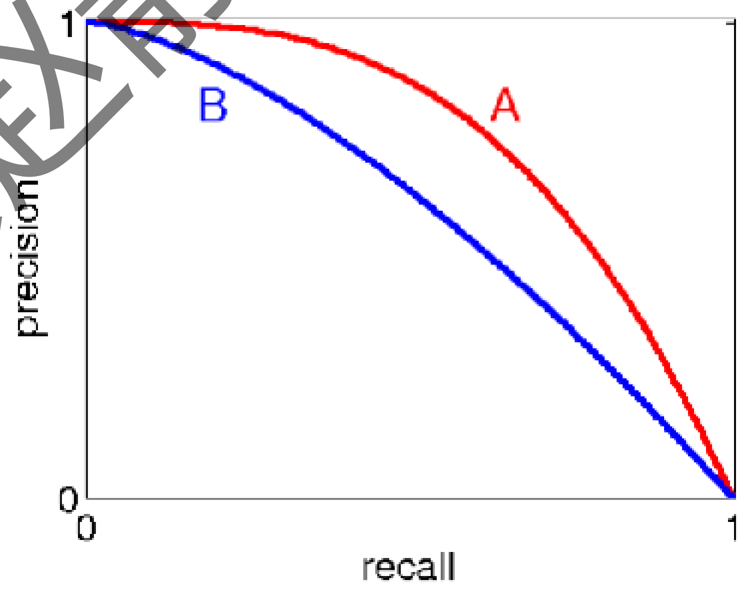
ROC曲线

✓ Precision and Recall (PR)曲线

——阈值变化时的Precision和Recall

$$\text{Precison} = \frac{TP}{\widehat{N}_+}$$：检测结果真正为正的比例

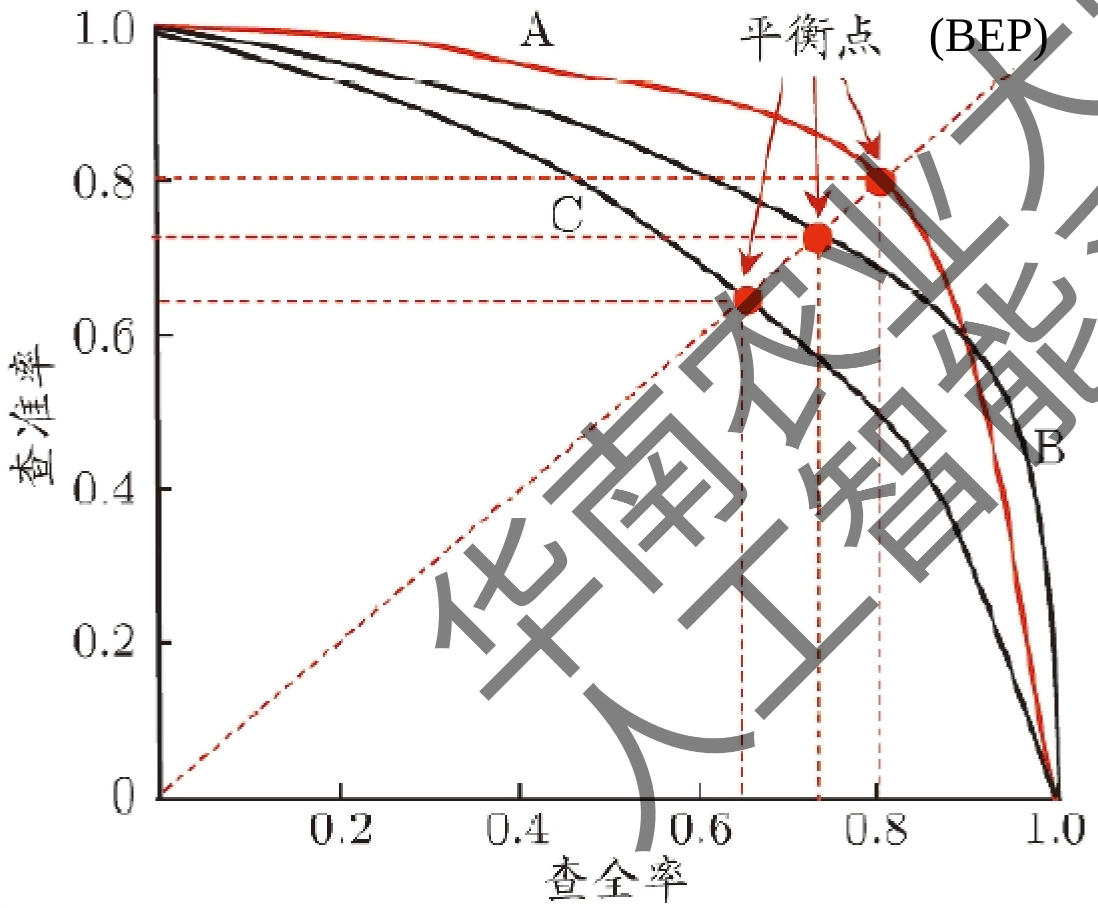
$$\text{Recall} = \frac{TP}{N_+}$$：被正确检测到的正样本的比例



F1分数：PR合一：

$$F1 = \frac{2(\text{Precison} \times \text{Recall})}{(\text{Precison} + \text{Recall})}$$

注意：准确率和召回率是互相影响的。一般情况下准确率高、召回率就低；召回率低、准确率高。



PR图:

- 学习器 A 优于 学习器 C
- 学习器 B 优于 学习器 C
- 学习器 A ?? 学习器 B

BEP:

- 学习器 A 优于 学习器 B
- 学习器 A 优于 学习器 C
- 学习器 B 优于 学习器 C

- AP：Average Precision

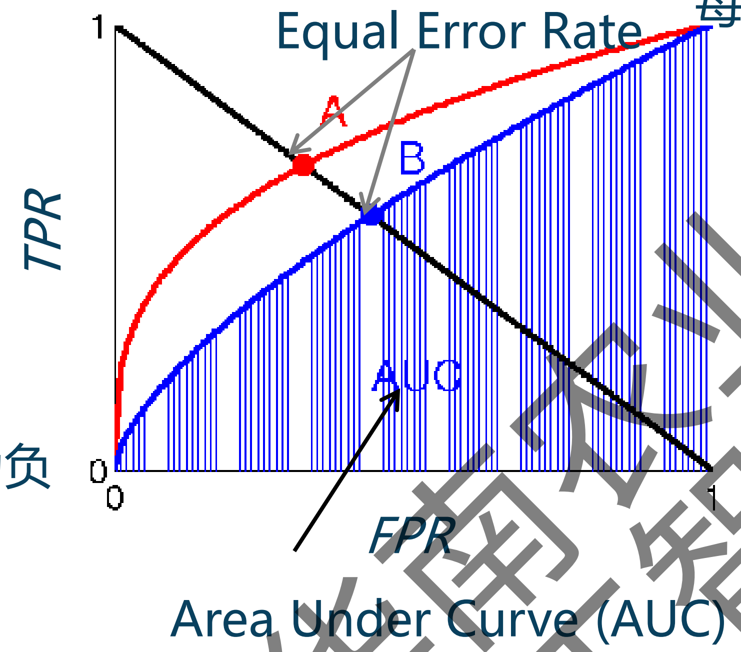
  对不同召回率点上的精度进行平均,即PR曲线下的面积

$$AP = \int_0^1 P(R)dR = \sum_{k=0}^{n} P(k)\Delta R(k)$$

- 平均AP（Mean Average Percision, mAP）

  多个物体类别的AP的平均

✓ Receiver Operating Characteristic (ROC)曲线



$$TPR = \frac{TP}{N_+}$$

$$FPR = \frac{FP}{N_-}$$

预测值

		$\hat{y}=1$	$\hat{y}=0$	Σ
真值	$y=1$	TP	FN	N_+
	$y=0$	FP	TN	N_-
	Σ	$\hat{N}_+$	$\hat{N}_-$	

- 一般步骤
- 建模过程
- 模型选择与评估

Fun Time

在查准率更重要的任务中，如视频推荐，下列哪项最适合做损失函数？

		g: +1	g: -1
y	+1	0	1
	-1	1000	0

① $\frac{1}{N}\sum_{n=1}^{N}[\![y_n \neq g(\mathbf{x}_n)]\!]$

② $\frac{1}{N}\left(\sum_{y_n=+1}[\![y_n \neq g(\mathbf{x}_n)]\!] + 1000\sum_{y_n=-1}[\![y_n \neq g(\mathbf{x}_n)]\!]\right)$

③ $\frac{1}{N}\left(\sum_{y_n=+1}[\![y_n \neq g(\mathbf{x}_n)]\!] - 1000\sum_{y_n=-1}[\![y_n \neq g(\mathbf{x}_n)]\!]\right)$

④ $\frac{1}{N}\left(1000\sum_{y_n=+1}[\![y_n \neq g(\mathbf{x}_n)]\!] + \sum_{y_n=-1}[\![y_n \neq g(\mathbf{x}_n)]\!]\right)$