机器学习基础

内容大纲

- 数学基础
- 什么是机器学习
- 机器如何学习
- 如何学习得更好
- 总结
- 机器学习实例

Prerequisite

- 微积分
  - 导数、梯度、偏导数、链式法则
- 线性代数
  - 向量、矩阵、张量、矩阵运算
- 概率论
  - 均值、方差、概率密度函数
- Python
  - 类、函数、列表、循环、条件分支、numpy、matplotlib等

数学基础

向量

- 标量（scalar）：一个标量就是一个单独的数，如$s \in R$
- 向量（vector）：是指由n个实数组成的有序数组，称为n纬向量，无特殊说明一般将其表示为一条列向量。如：

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

其中，x_n为向量的第n纬元素。

向量范数

- 范数（norm）是具有“长度”概念的函数。在线性代数、泛函分析及相关的数学领域，范数是一个函数，其为向量空间内的所有向量赋予非0的正长度或大小。较常用的向量范数有：
- L1-范数： $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n}|x_i|$，即向量元素绝对值之和
- L2-范数： $\|\boldsymbol{x}\|_2 = \left(\sum_{i=1}^{n}|x_i|^2\right)^{\frac{1}{2}}$，即欧几里得范数
- ∞-范数： $\|\boldsymbol{x}\|_\infty = \max_i|x_i|$，即所有向量元素绝对值的最大值
- P-范数： $\|\boldsymbol{x}\|_p = (\sum_{i=1}^{n}|x_i|^p)^{\frac{1}{p}}, (p \geq 1)$，即向量元素绝对值的$p$次方和$\frac{1}{p}$次幂，一般形式

向量运算

- 假设有向量$\boldsymbol{x} = (x_1, x_2, \dots, x_n)^T$和$\mathbf{y} = (y_1, y_2, \dots, y_n)^T$
- 向量加法： $\boldsymbol{x} + \boldsymbol{y} = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n)^T$
- 向量减法： $\boldsymbol{x} - \boldsymbol{y} = (x_1 - y_1, x_2 - y_2, \dots, x_n - y_n)^T$
- 向量数乘： $\lambda\boldsymbol{x} = (\lambda x_1, \lambda x_2, \dots, \lambda x_n)^T$，其中$\lambda$为标量
- 向量点积（dot product），又称向量内积（inner product）： $\boldsymbol{x} \cdot \boldsymbol{y} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n = \boldsymbol{x}^T \boldsymbol{y}$

矩阵

- 数学上，一个$m \times n$的矩阵（matrix）是一个由m行、n列元素排列成的矩形阵列。其中，从左上角数起第i行第j列上的元素称为矩阵第(i,j)项，通常记作$\boldsymbol{A}_{ij}$。

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

矩阵的范数

- 类似向量的范数，常用的矩阵的范数如下：
- F-范数： $\|\boldsymbol{A}\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}\left|a_{ij}\right|^2\right)^{\frac{1}{2}}$
- P-范数： $\|\boldsymbol{A}\|_p = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}\left|a_{ij}\right|^p\right)^{\frac{1}{p}}, (p \geq 1)$

矩阵运算

- 假设$\boldsymbol{A}$和$\boldsymbol{B}$都是$m \times n$的矩阵，则：
- 矩阵加法：$(\boldsymbol{A}+\boldsymbol{B})_{ij} = \boldsymbol{A}_{ij} + \boldsymbol{B}_{ij}$
- 矩阵减法：$(\boldsymbol{A}-\boldsymbol{B})_{ij} = \boldsymbol{A}_{ij} - \boldsymbol{B}_{ij}$
- 矩阵数乘：$(\lambda\boldsymbol{A})_{ij} = \lambda\boldsymbol{A}_{ij}$
- 矩阵转置：$\left(\boldsymbol{A}^{T}\right)_{ij} = \boldsymbol{A}_{ji}$
- 矩阵乘法：矩阵乘法中$\boldsymbol{A}$的列必须和$\boldsymbol{B}$的行数相同，即$\boldsymbol{A}$是$m \times n$的矩阵，$\boldsymbol{B}$是$n \times p$的矩阵，那么$\boldsymbol{C} = \boldsymbol{AB}$是$m \times p$的矩阵，其中$\boldsymbol{C}_{ij} = \sum_{k=1}^{n} \boldsymbol{A}_{ik}\,\boldsymbol{B}_{kj}$

单位矩阵和逆矩阵

- 主对角线上的元素都为1，其余元素全为0的n阶矩阵称为n阶单位矩阵

$$\boldsymbol{I}_n = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

$$\boldsymbol{I}_n\boldsymbol{A} = \boldsymbol{A}$$

- 矩阵$\boldsymbol{A}$的逆矩阵被记作$\boldsymbol{A}^{-1}$，被定义为如下形式：

$$\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}_n$$

张量

- 张量（tensor）：某些情况下，我们会讨论不止二维坐标的数组。如果一组数组中的元素分布在若干维坐标的规则网络中，就将其称为张量。
- 一幅图像，有3个通道（RGB），是三维张量；更进一步，一个batch（比如128）图像，是四维张量。
- Tensorflow、PyTorch等都是围绕张量构建的计算库

概率基础

- 随机变量：在概率论中，随机变量扮演了重要的角色。随机变量可以随机地取不同值的变量。我们通常用小写字母（x）来表示随机变量本身，而用带数字下标的小写字母（x_1）来表示随机变量能够取到的值。随机变量可以是离散的或者连续的。
- 概率分布：用来描述随机变量在每一个可能取到的状态的可能性大小。$P(x = x_1)$

条件概率

- 在很多情况下，我们感兴趣的是某个事件在给定其它事件发生时出现的概率，这种概率叫作条件概率：其记号为$P(A|B)$，表示在给定条件B下A事件发生的概率。
- 联合概率：联合概率为多个事件同时发生的概率，计为$P(A,B)$，$P(A,B) = P(A)P(B|A) = P(B)P(A|B)$
- 独立性：$P(A,B) = P(A)P(B)$

期望、方差和协方差

- 期望：是试验中每次可能结果的概率乘以其结果的总和。它是最基本的数学特征之一，反映随机变量平均值的大小

$$E(X) = \sum_{k=1}^{n} x_k P(x_k)$$

- 方差：用来衡量随机变量与其数学期望之间的偏离程度；统计中的方差为样本方差，是各个样本数据分别与其平均数之差的平方和的平均数

$$Var(X) = E(X - E(X))^2$$

- 协方差：用于衡量两个随机变量X和Y之间的总体误差，在某种意义上给出了两个变量线性相关性的强度

$$Cov(X, Y) = E\big((X - E(X))(Y - E(Y))\big)$$

贝叶斯规则

- 我们经常需要在已知$P(A|B)$时计算$P(B|A)$，幸运的是，如果还知道$P(B)$，我们可以用贝叶斯规则来实现这一目的：

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

贝叶斯规则可以从条件概率的定义直接推导。

什么是机器学习？

为什么我们需要机器学习

- 编程显示的“定义”出树：很难

基本字义

树（樹） shù（ㄕㄨˋ）

1、木本植物的通称：树木。树林。树大根深（喻势力大，根基牢固）。

2、种植，培育：树艺（“艺”，种植）。树荆棘得刺，树桃李得荫。

3、立，建立：树立。树敌。

4、量词，相当于“株”、“棵”：一树梅花。

5、姓。

- 三岁的小孩却能从不断观察中学习识别，几乎不会出错
- 机器学习：可以更“简单”的实现手工编程无法解决的复杂系统问题

为什么我们需要机器学习

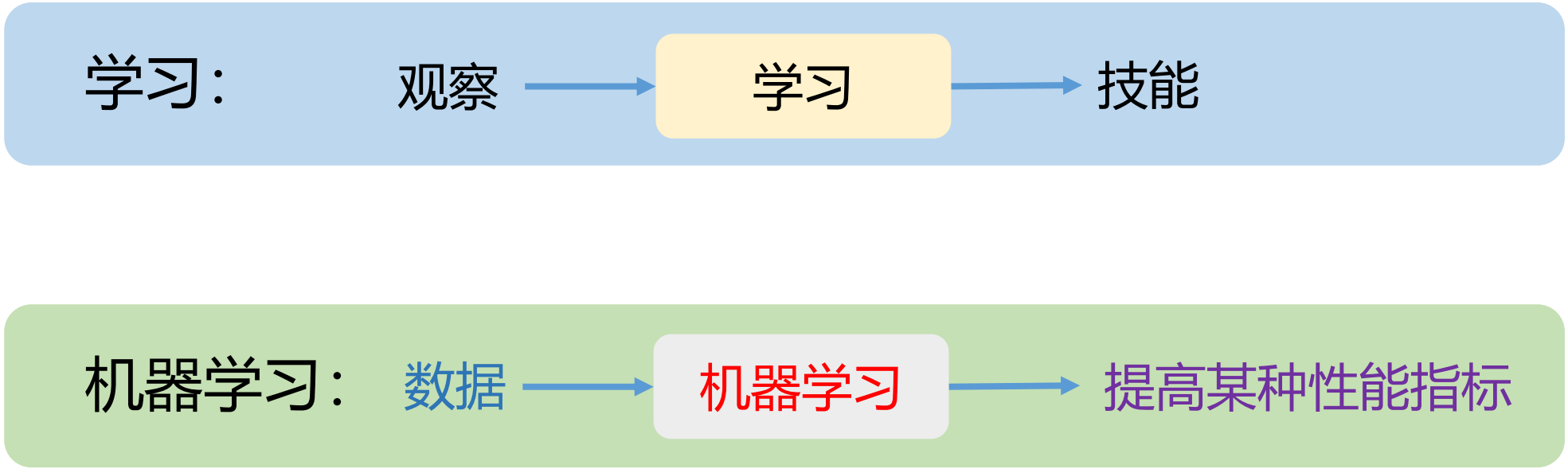
- 系统太复杂，无法显示的编程解决：
  - 自动驾驶
- 无法明确定义出一个解决方案：
  - 图像识别
- 需要非常快速的判断和决策：
  - 高频交易
- 需要处理非常大量的数据：
  - 欧洲核子对撞机撞出来的数据

授之以渔

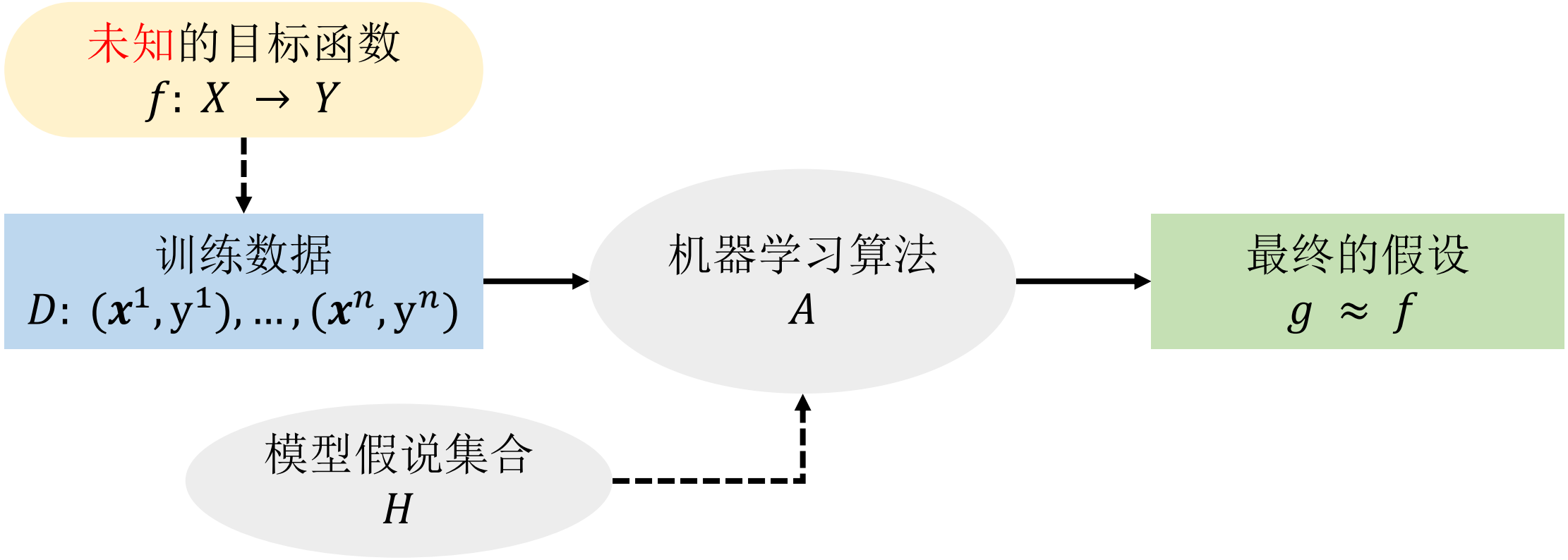
机器学习的广泛应用

- 衣：淘宝同款、根据搜索历史推荐，时尚搭配推荐
- 食：餐馆推荐
- 住：建筑楼房预算估计、房价预测
- 行：自动识别交通标志、违章识别、自动驾驶
- 机器学习无处不在

机器学习



机器学习



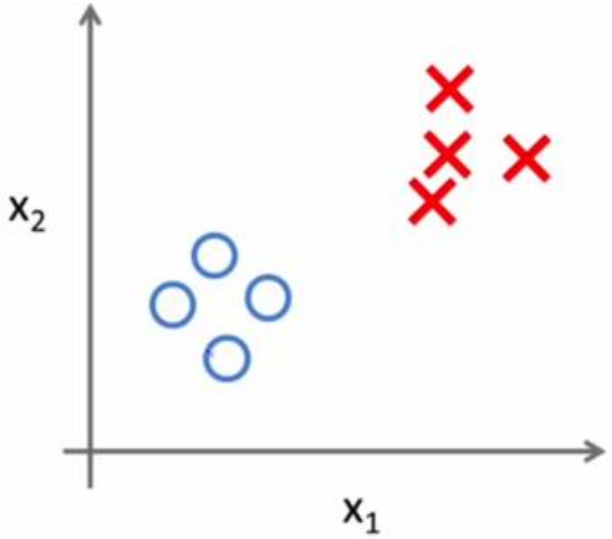
机器学习：通过数据D，计算得出一个可以近似目标函数f的假设g

机器学习的类型

- 根据输出空间分类：
  - 分类
  - 回归

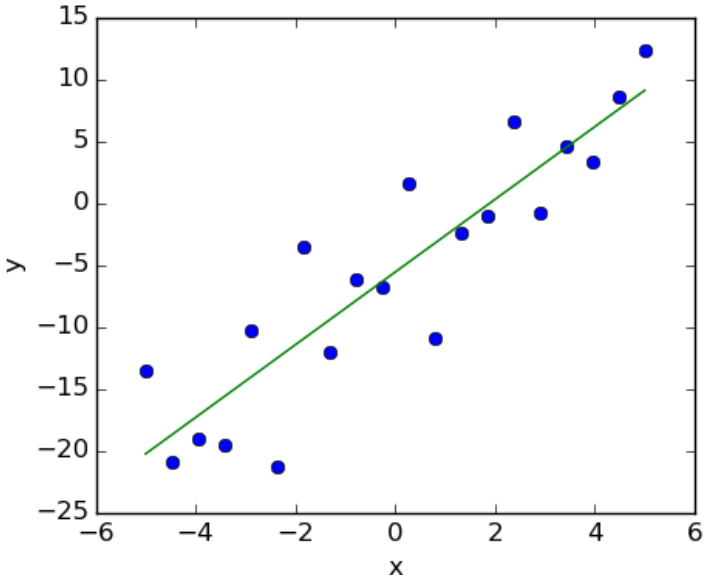
分类

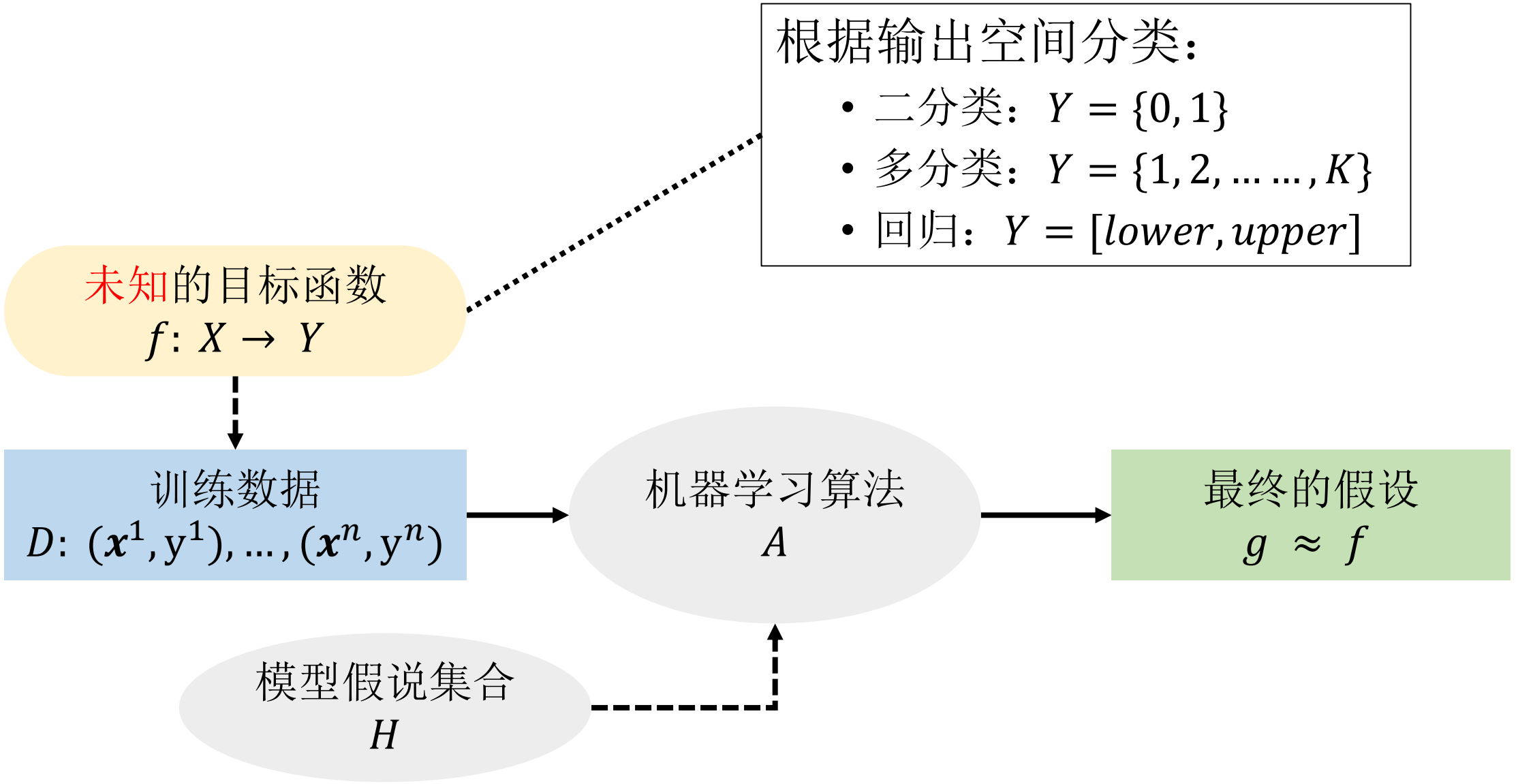
- **分类**模型预测**离散值**，例如：
  - 电子邮件是垃圾邮件还是非垃圾邮件？
  - 信用卡申请是批准还是不批准？
  - 根据病人的检查情况，判断是生病了还是没有生病？
  - 也可以是多类分类，这是一张狗、猫还是仓鼠图片？



回归

- **回归**模型预测**连续值**，例如：
  - 用户点击此广告的概率是多少？
  - 明天的气温是多少度？
- 分类还是回归？
  - 根据病人体检数据，判断是否得病
  - 根据病人体检数据，判断得了那种病
  - 根据病人体检数据，估计还需要多少天可以康复

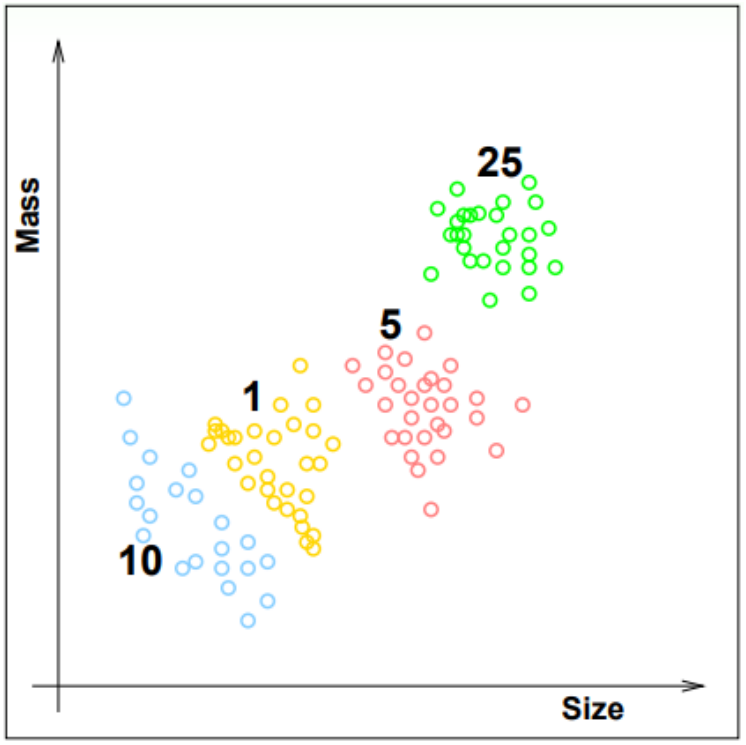



根据输出空间分类：
• 二分类：$Y = \{0, 1\}$
• 多分类：$Y = \{1, 2, \ldots\ldots, K\}$
• 回归：$Y = [lower, upper]$
未知的目标函数
$f: X \rightarrow Y$
训练数据
$D: (\boldsymbol{x}^1, \mathrm{y}^1), \ldots, (\boldsymbol{x}^n, \mathrm{y}^n)$
机器学习算法
A
最终的假设
$g \approx f$
模型假说集合
H

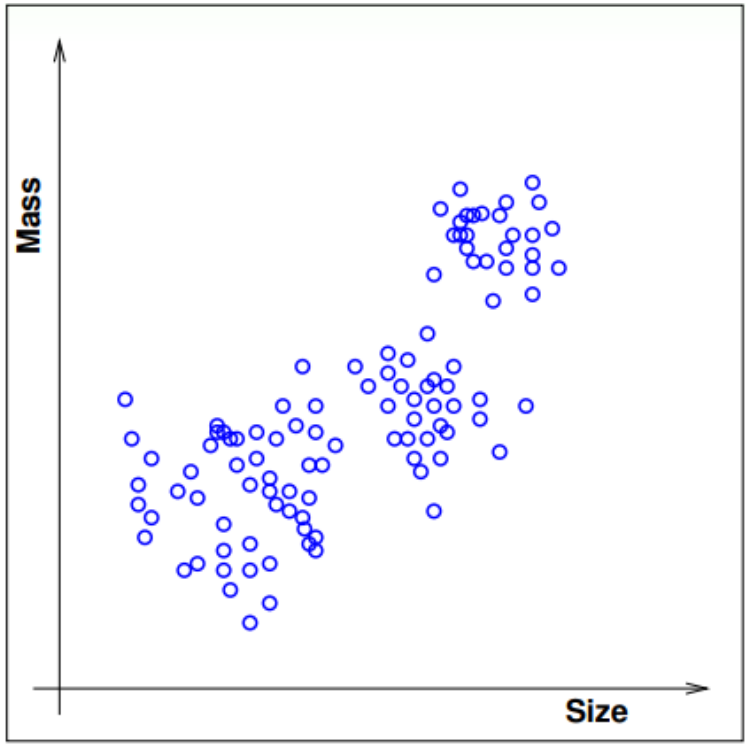
机器学习的类型

- 根据输出空间分类：
  - 分类、回归
- 根据数据标注信息分类：
  - 监督
  - 非监督
  - 半监督

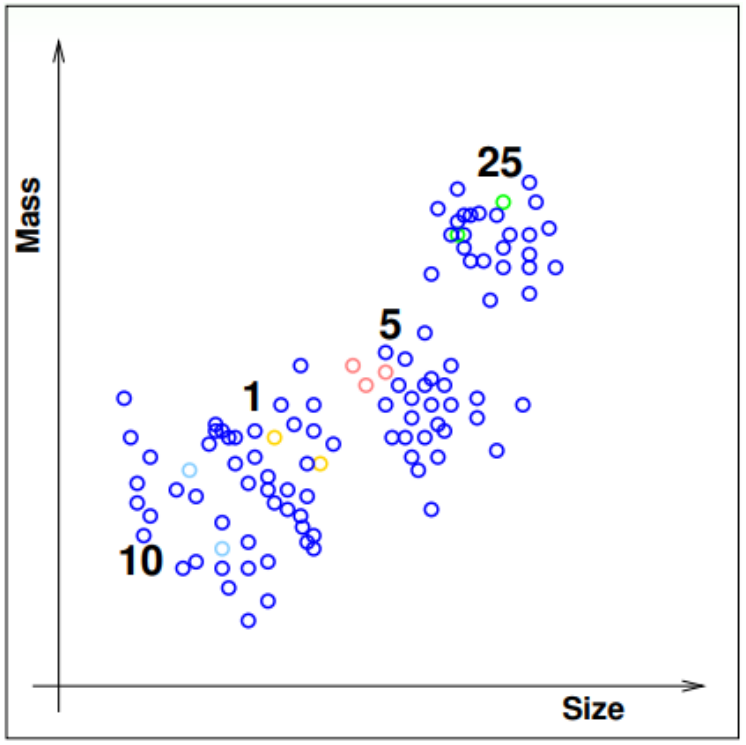
监督、非监督、半监督



监督学习



非监督学习

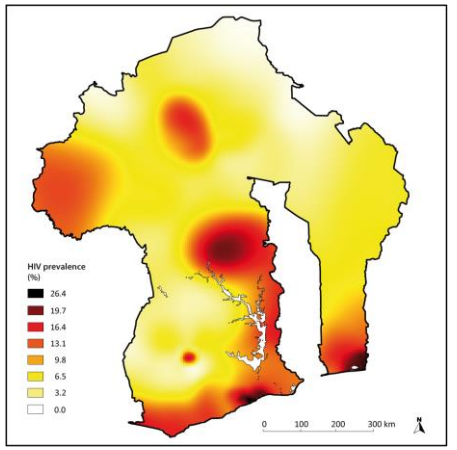


半监督学习

非监督学习

- 聚类：非监督的多类别分类
  - 根据文章内容，分类成不同的主题

- 概率密度估计：非监督的回归
  - 带有地理信息的病例，估计分布情况
- 异常检测：一种极端的非监督二分类
  - 分析日志，异常状态报警

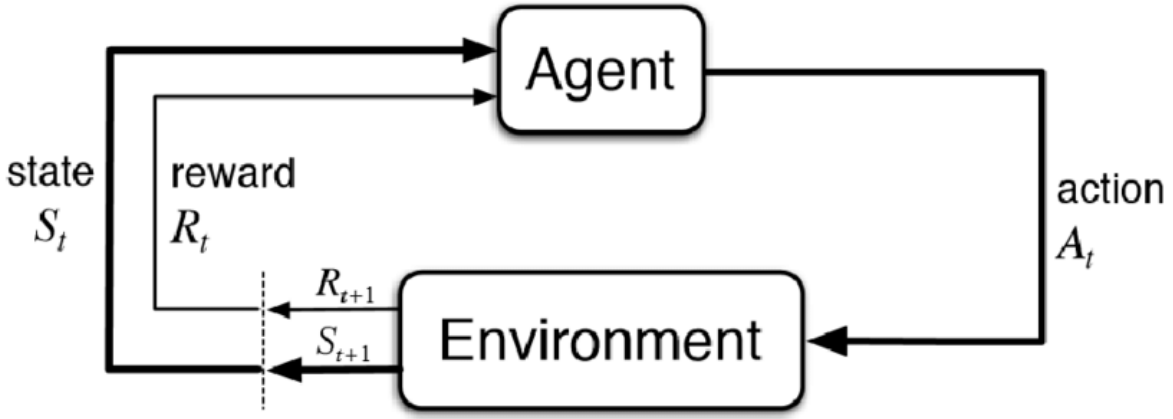


机器学习的类型

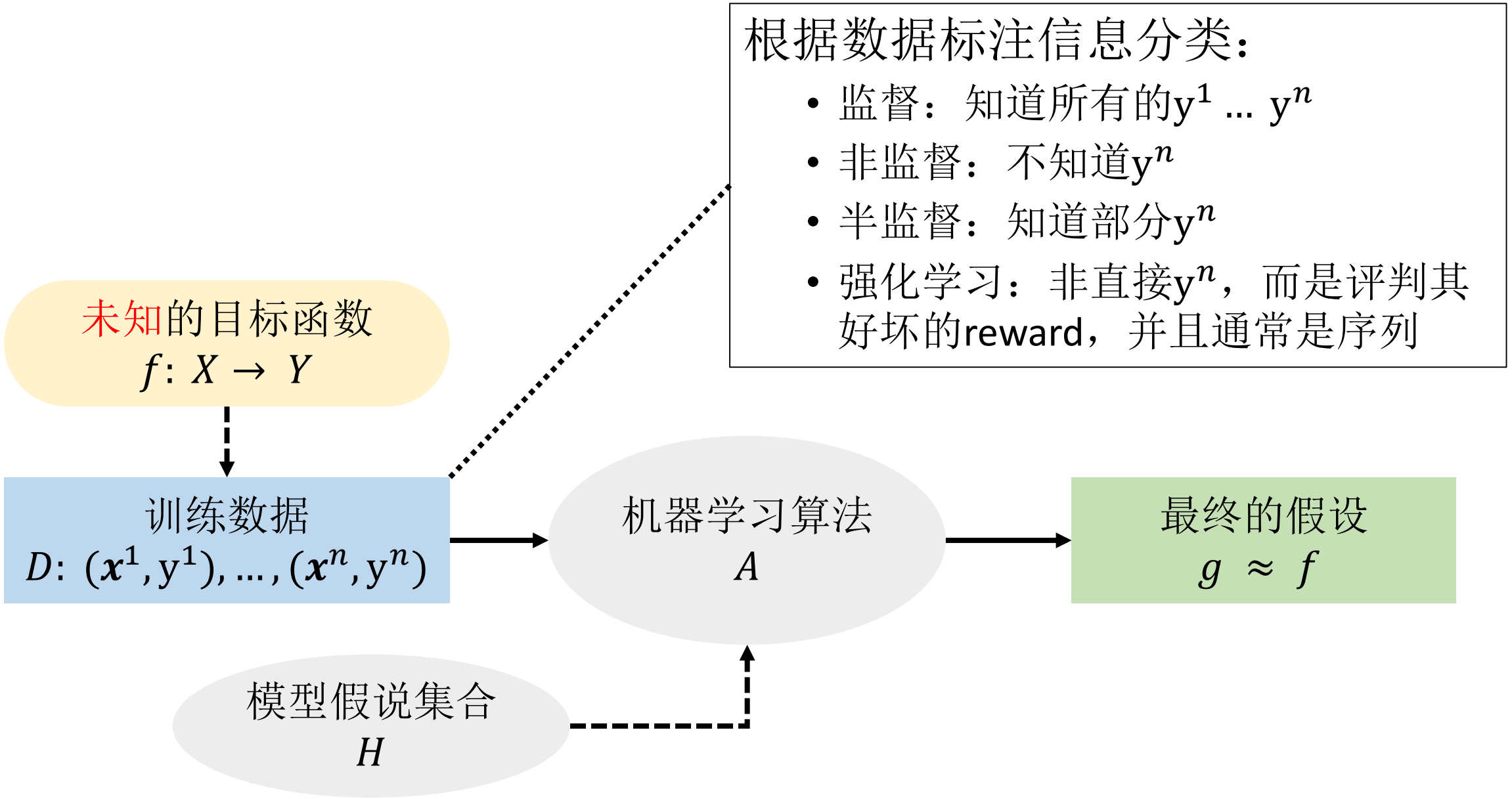
- 根据输出空间分类：
  - 分类、回归
- 根据数据标注信息分类：
  - **监督**
  - 非监督
  - 半监督
  - 弱监督
  - 自监督
  - 强化学习

强化学习

- 训练一只狗听懂“坐下”的指令：
  - 狗没有坐下，我们会呵斥它
  - 狗正确的坐下了，我们会奖励它饼干



- 其它强化学习应用：
  - 机器人控制、棋牌、游戏控制


根据数据标注信息分类：
• 监督：知道所有的$y^1 \dots y^n$
• 非监督：不知道y^n
• 半监督：知道部分y^n
• 强化学习：非直接y^n，而是评判其好坏的reward，并且通常是序列
未知的目标函数
$f: X \rightarrow Y$
训练数据
$D: (x^1, y^1), \dots, (x^n, y^n)$
机器学习算法
A
最终的假设
$g \approx f$
模型假说集合
H

与其它概念的关系

机器学习：通过数据，计算得出一个可以近似目标函数f的假设g

数据挖掘：通过大量数据，发现某种有趣的性质

- 如果“有趣的性质”=“近似f的g”，ML = DM
- 如果“有趣的性质”与“近似f的g”相关，DM可以帮助ML
- 传统的数据挖掘，还会研究“如何在超大规模数据中有效计算”
- 不过，现在通常可能很难严格区分机器学习和数据挖掘

与其它概念的关系

机器学习：通过数据，计算得出一个可以近似目标函数f的假设g

人工智能：通过计算实现某种智能行为

- 如果通过“近似f的g”可以体现出智能，机器学习是一种实现人工智能的途径
- 例如，下棋：
  - 传统AI：游戏树，搜索
  - 机器学习：从棋谱中学习

与其它概念的关系

机器学习：通过数据，计算得出一个可以近似目标函数f的假设g

统计学：通过数据，推理某个未知过程

- 在机器学习中，通常f是未知的，g是一个推理结果
  - 所以，统计学方法可以用来实现机器学习
- 传统的统计学更关注在某些数学假设条件下的证明，而非计算过程

机器如何学习？

分类

JD.京东 JD.COM 多·快·好·省

全部商品分类 首页

- 图书、音像、数字商品
- 家用电器
- 手机、数码、京东通信
- 电脑、办公
- 家居、家具、家装、厨具
- 男装、女装、内衣、珠宝
- 个护化妆
- 鞋靴、箱包、钟表、奢侈品
- 运动户外
- 整车、汽车用品
- 母婴、玩具乐器
- 食品饮料、酒类、生鲜
- 营养保健
- 彩票、旅行、充值、票务

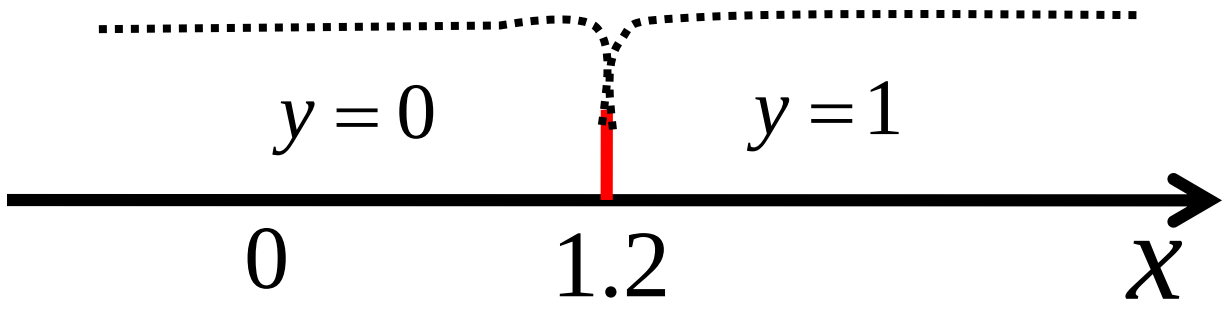
线性分类

线性分类

- 儿童免票乘车分类器

$$x: \text{身高}, y: \{0: \text{免票} \quad 1: \text{购票}\}$$

$$y = \begin{cases} 1 & x \geq 1.2m \\ 0 & x < 1.2m \end{cases}$$

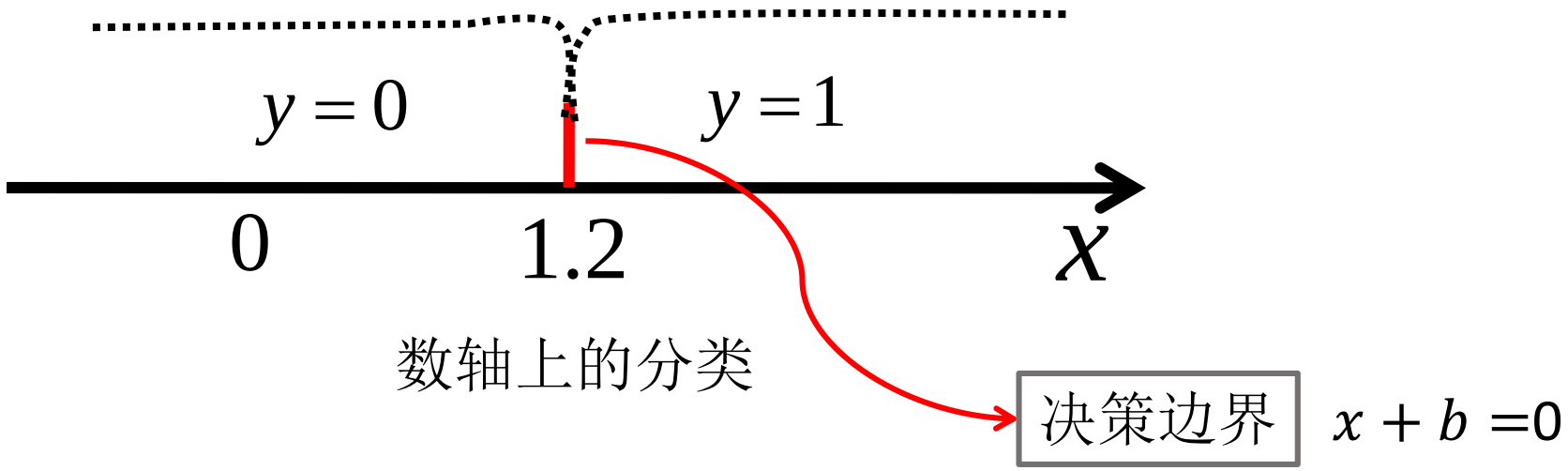


数轴上的分类

线性分类

- 儿童免票乘车分类器

$$y = g(x;b) = \begin{cases} 1 & h(x) = x + b \geq 0 \\ 0 & h(x) = x + b < 0 \end{cases} \qquad b = -1.2$$



数轴上的分类

线性分类

- 数轴上的分类
  - 单个特征x (一维特征、实数)
  - 一个参数b (偏置项)

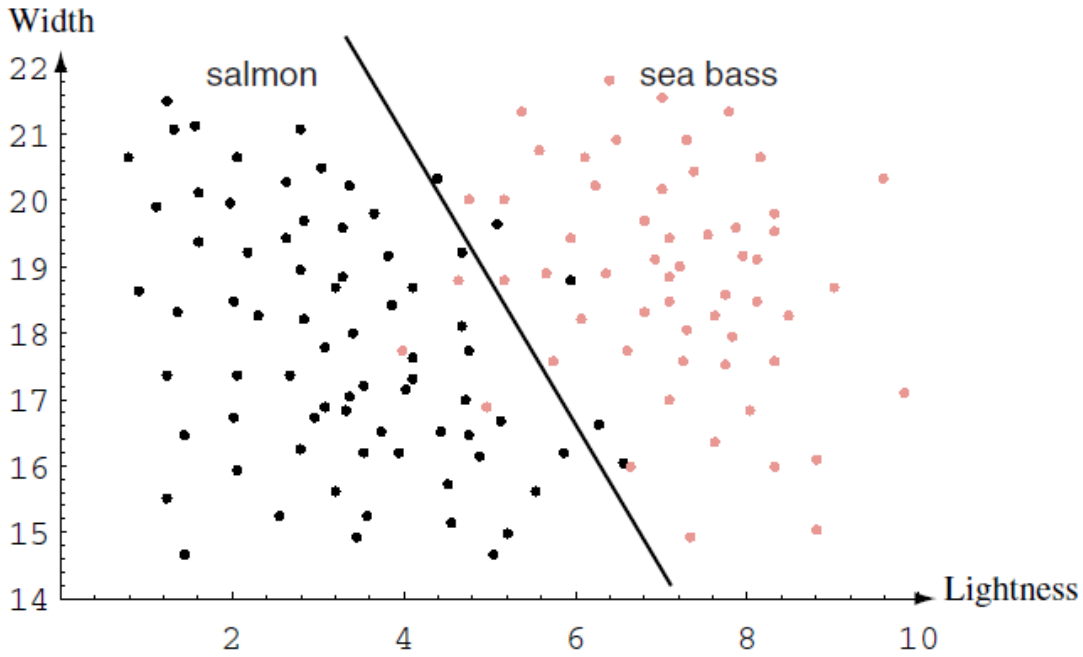
$$g(x;b)=\begin{cases}1 & h(x)=x+b\geq 0\\ 0 & h(x)=x+b<0\end{cases}$$

  - 决策边界

$$x+b=0$$

线性分类

- 三文鱼和鲈鱼的分类器



线性分类

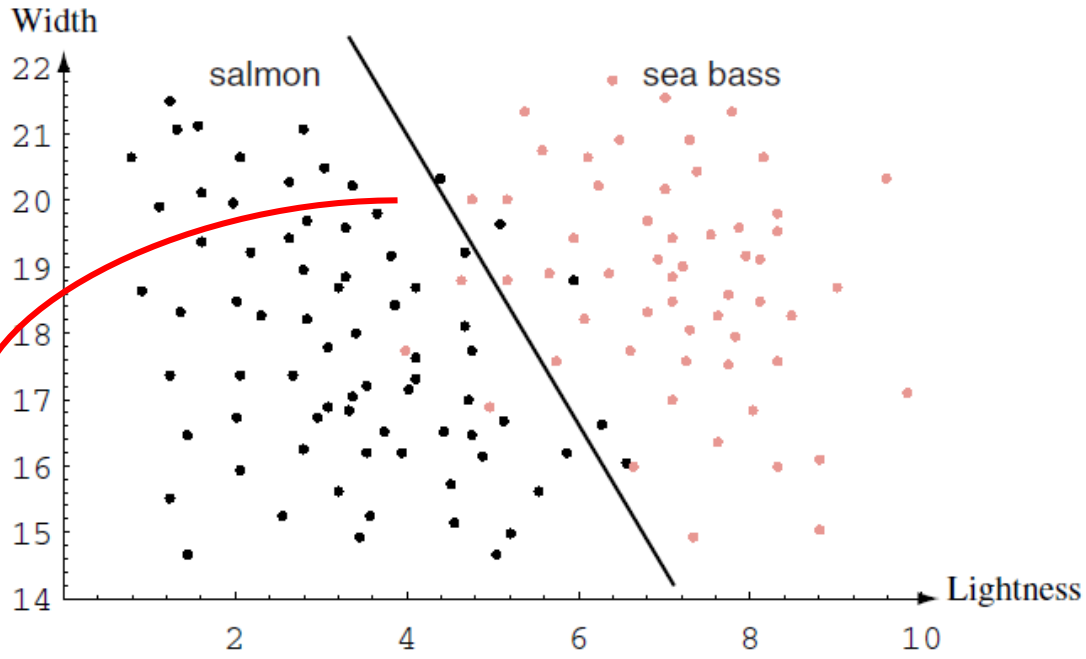
- 三文鱼和鲈鱼的分类器

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

x_1: lightness

x_2: width

y: {0: 三文鱼　1: 鲈鱼}



决策边界

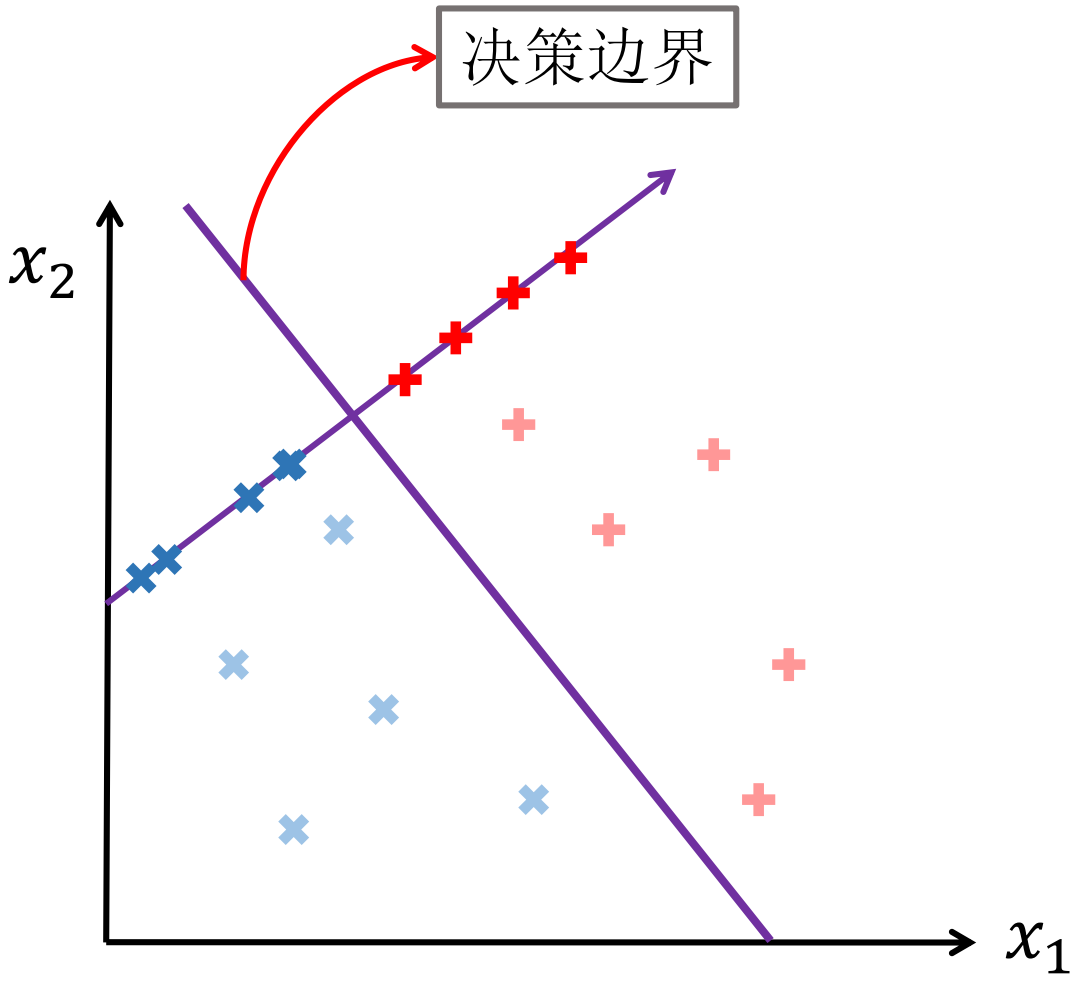
$$\boldsymbol{w}^T\boldsymbol{x} + \mathrm{b} = 0$$

$$y = g(\boldsymbol{x}; \boldsymbol{w}, b) = \begin{cases} 1 & \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \\ 0 & \boldsymbol{w}^T\boldsymbol{x} + b < 0 \end{cases}$$

线性分类

- 实际上还是数轴上的分类
  - 将向量$\boldsymbol{x}$投影到$\boldsymbol{w}$向量的方向上
- $\boldsymbol{w}$是分界线的法向量
- 也可以看作是对原始特征向量的一种加权分数
  - 根据此加权分数分类

$$\begin{aligned} & \boldsymbol{w}^T\boldsymbol{x} + \mathrm{b} \\ = & \ \mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \mathrm{b} \\ = & \ x' + b \end{aligned}$$



线性分类

- 平面上的分类
  - 特征向量$\boldsymbol{x}$ (二维向量)
  - 参数：w_1, w_2, b

$$g(\boldsymbol{x}; \boldsymbol{w}, b) = \begin{cases} 1 & \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \\ 0 & \boldsymbol{w}^T\boldsymbol{x} + b < 0 \end{cases}$$

  - 决策边界

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0$$

$\boldsymbol{w}$: 分界线的法向量

线性分类

- 高维空间上的分类：**转换为数轴上的分类**

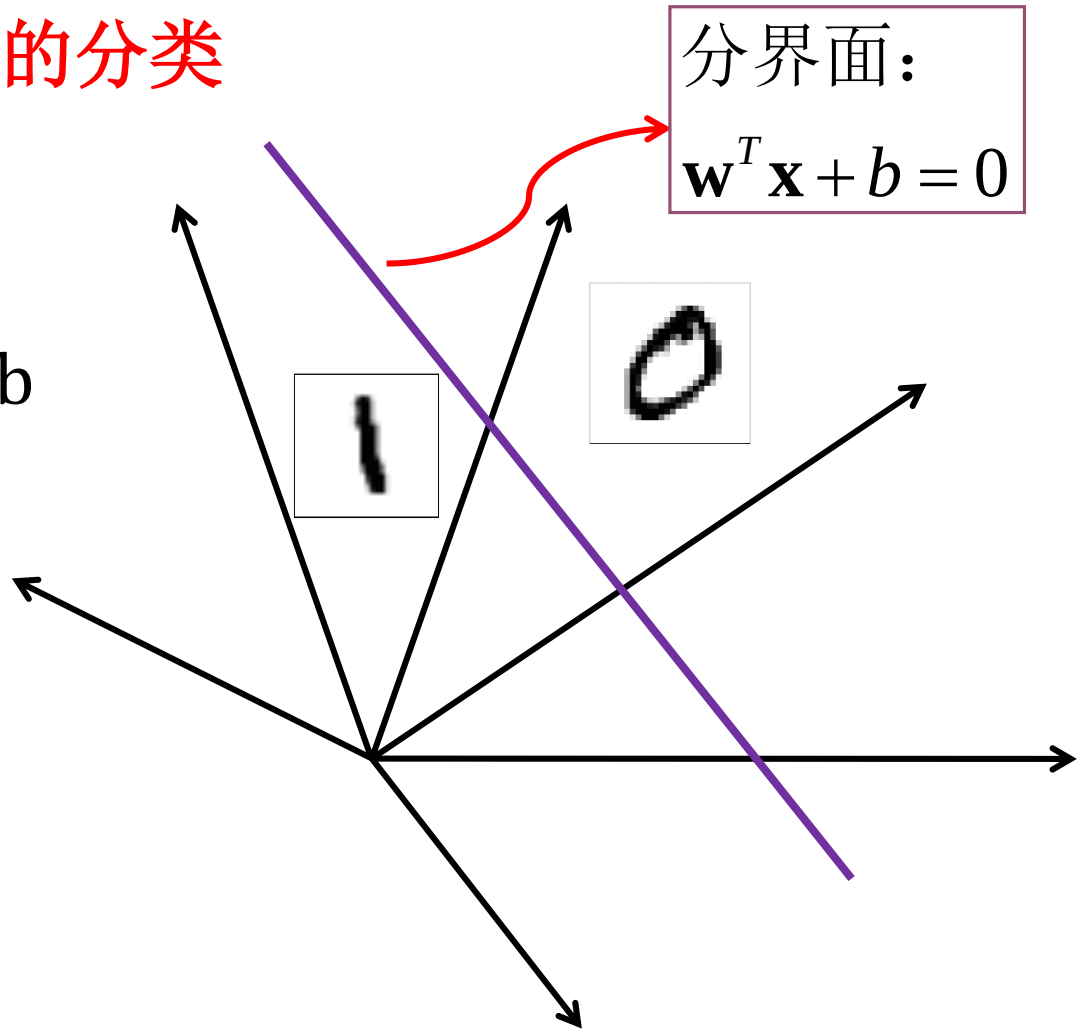
$$\boldsymbol{x} = (x_1, x_2 \ldots x_d)^T$$

$$x' = \mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \cdots + \mathrm{w}_d x_d + \mathrm{b}$$

$$x' = \boldsymbol{w}^T \boldsymbol{x}$$

$$\boldsymbol{w} = (w_1, w_2 \ldots w_d)^T$$

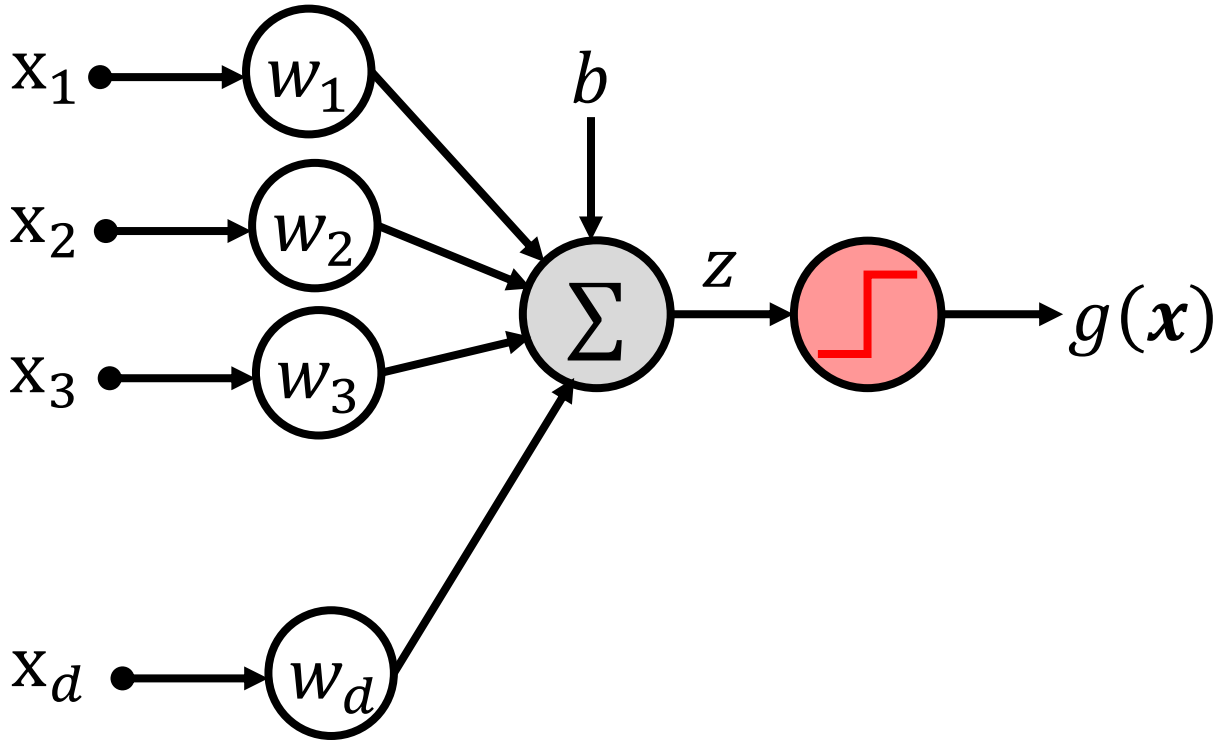
$$y = g(\boldsymbol{x}; \boldsymbol{w}, b) = sign(x' + b)$$



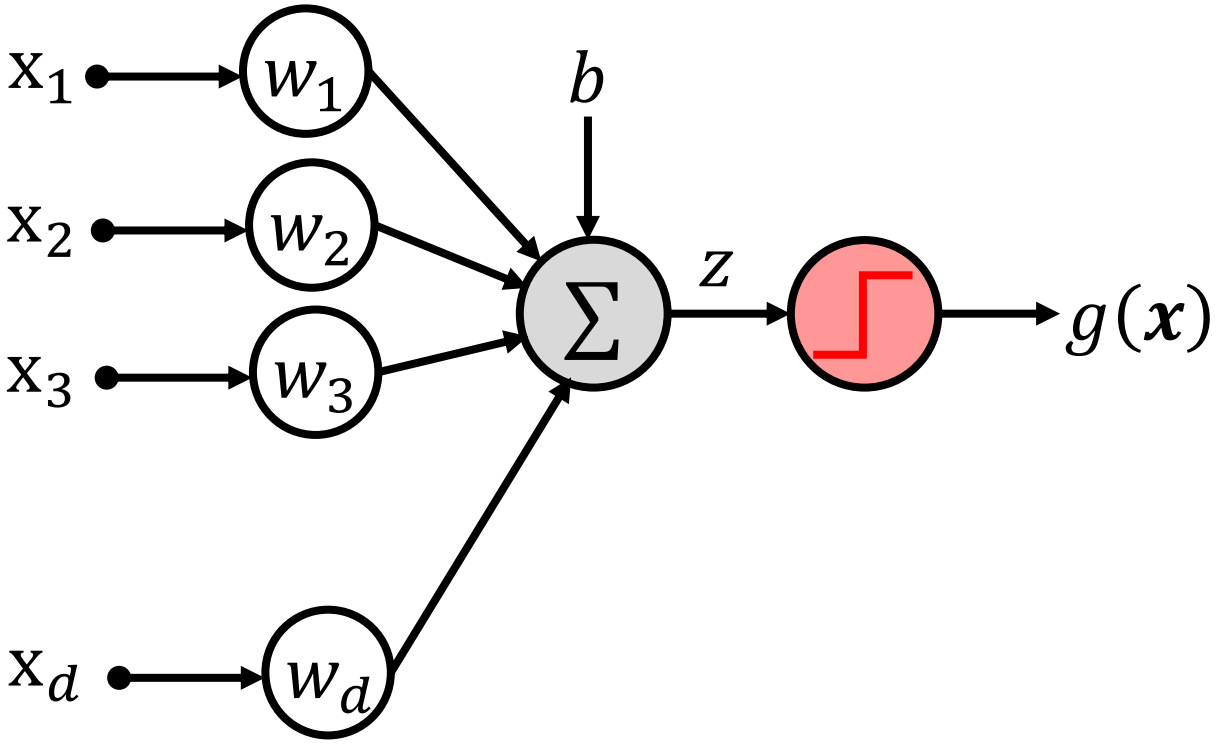
感知器 (Perceptron)

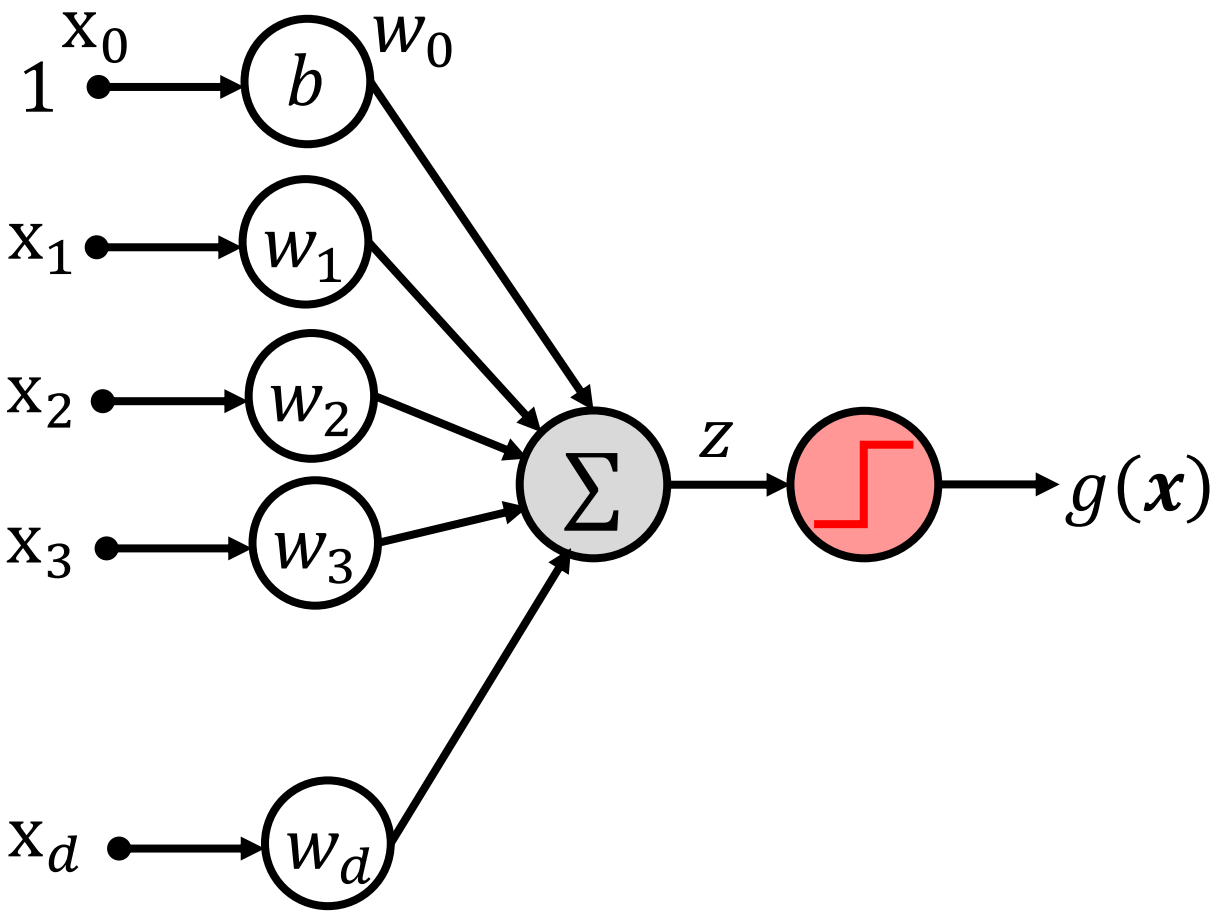
$$g(\boldsymbol{x};\boldsymbol{w},b)=\begin{cases}1 & \boldsymbol{w}^T\boldsymbol{x}+b\geq 0\\ 0 & \boldsymbol{w}^T\boldsymbol{x}+b<0\end{cases}$$

$$=sign(z)$$



参数中的b





$$g(\boldsymbol{x};\boldsymbol{w},b)=sign(\boldsymbol{w}^T\boldsymbol{x}+b)$$

$$g(\boldsymbol{x};\boldsymbol{w})=sign(\boldsymbol{w}^T\boldsymbol{x})$$

线性分类

- 某银行要根据用户的年龄、性别、年收入等情况来判断是否给该用户发信用卡
- 我们把用户的个人信息作为特征向量$\boldsymbol{x}$
- 使用线性分类器（感知器）做为模型假说H
- 现在有训练样本D，即之前用户的信息和是否对其发了信用卡
- 现在需要一个算法A学习出g
  - 即需要计算$\boldsymbol{w}$

年龄	23岁
年薪	100,000
工作年限	1
当前欠款	20,000

是否批准信用卡申请？

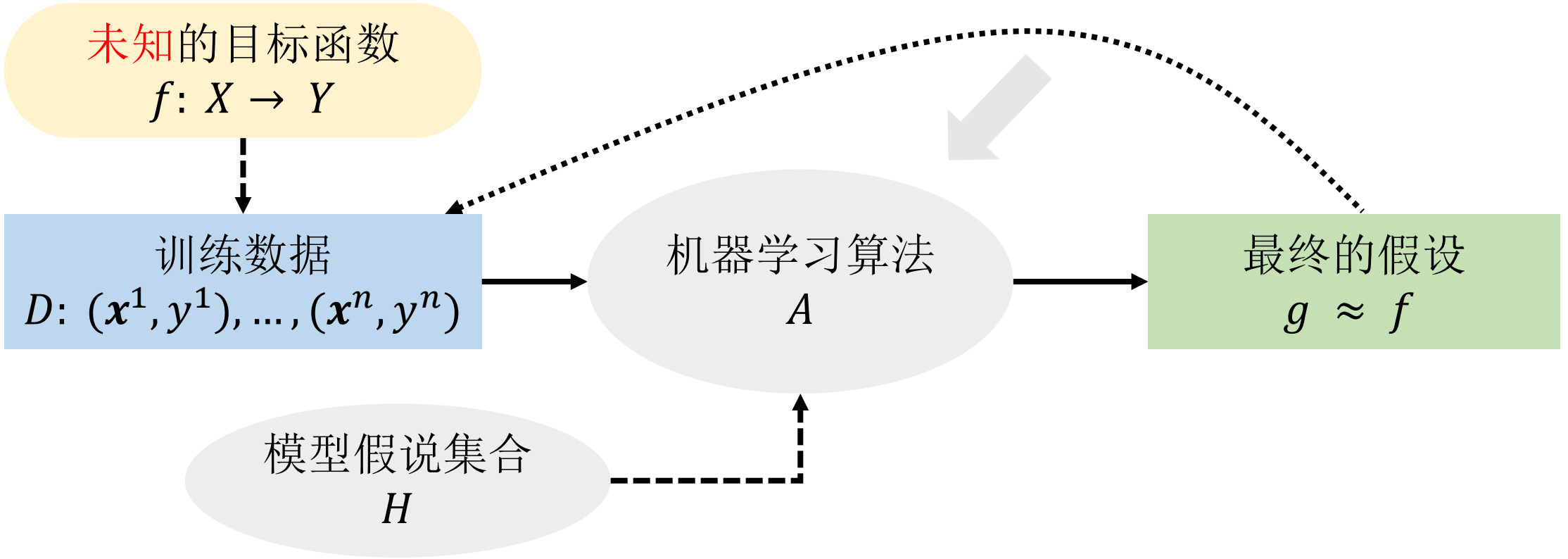
什么样的$\boldsymbol{w}$是好的？

- 经验风险最小化准则 (**ERM**, Empirical Risk Minimization)
- 定义损失函数：$l(g(\boldsymbol{x}), y)$
- 经验风险：$g(\boldsymbol{x})$在样本上进行错误的预测带来的损失

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \sum_{\mathrm{i}=1}^{n} l\left(g\left(\boldsymbol{x}^i; \boldsymbol{w}\right), y^i\right)$$

通常会取平均

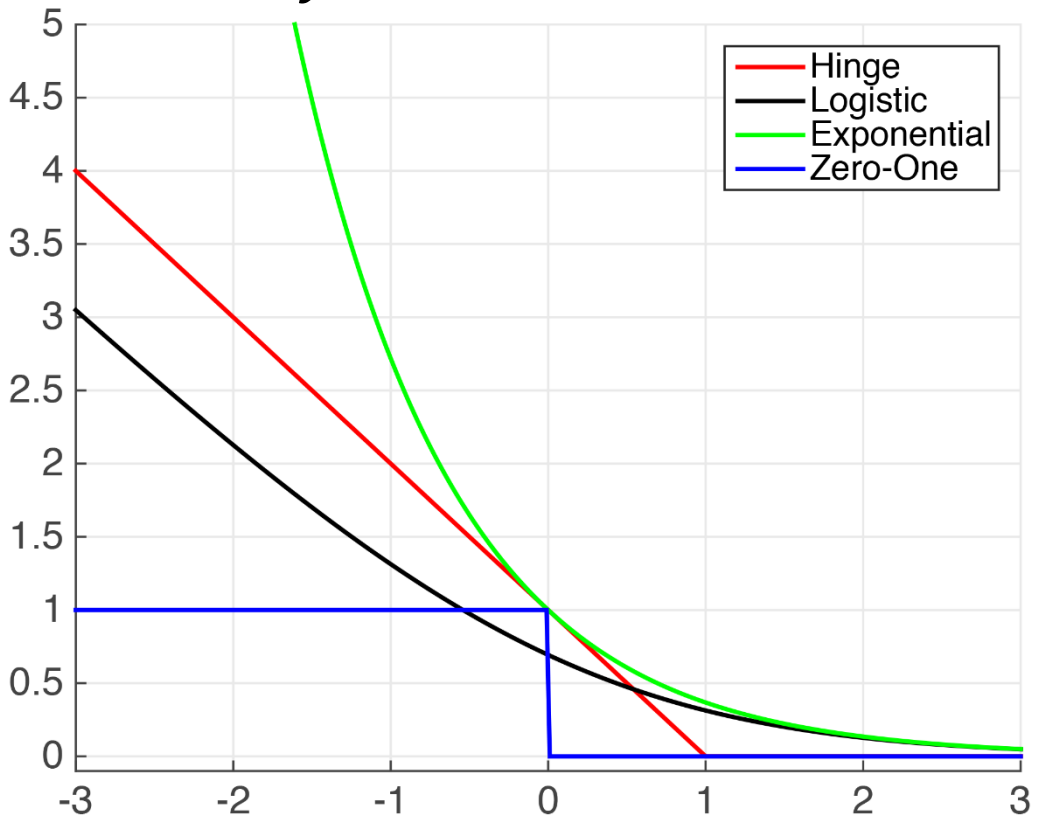
$$\boldsymbol{w}, b = \arg\min_{\boldsymbol{w}} \frac{1}{n} \sum_{\mathrm{i}=1}^{n} l\left(g\left(\boldsymbol{x}^i; \boldsymbol{w}\right), y^i\right)$$



$$\arg\min_{\boldsymbol{w}} L = \arg\min_{\boldsymbol{w}} \frac{1}{n} \sum_{\mathrm{i}=1}^{n} l\left(g\left(\boldsymbol{x}^i; \boldsymbol{w}\right), y^i\right)$$

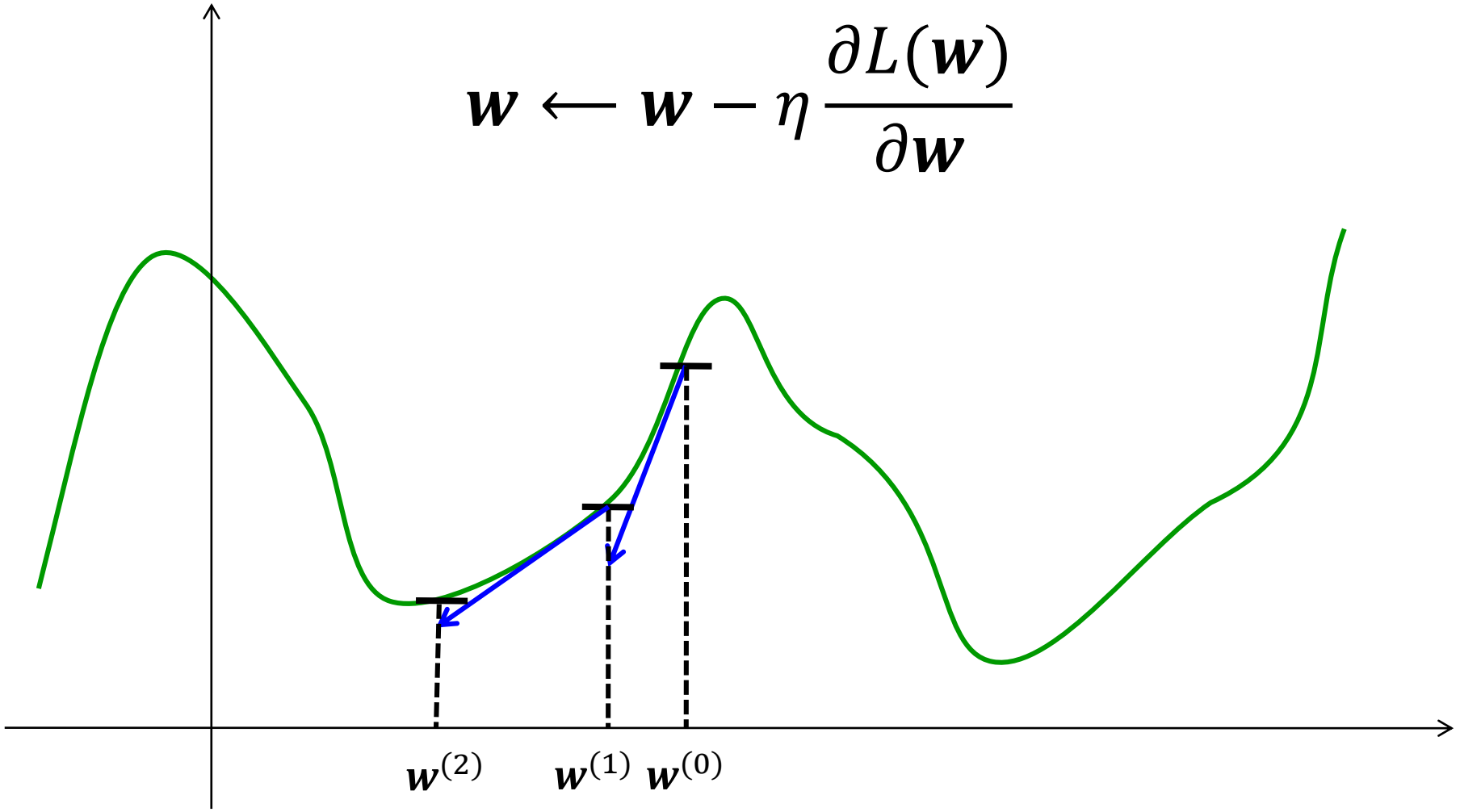
损失函数

- 是用来估量你模型的预测值$g(\boldsymbol{x})$与真实值y的不一致程度，它是一个**非负实值函数**
- 常用的损失函数包括：
  - 0-1损失函数
  - 平方损失函数
  - log损失函数
  - 铰链损失函数
- 不同的任务需要不同的损失函数
- 损失函数能影响学习的好坏



梯度下降法

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}}$$



三种梯度下降法工作模式

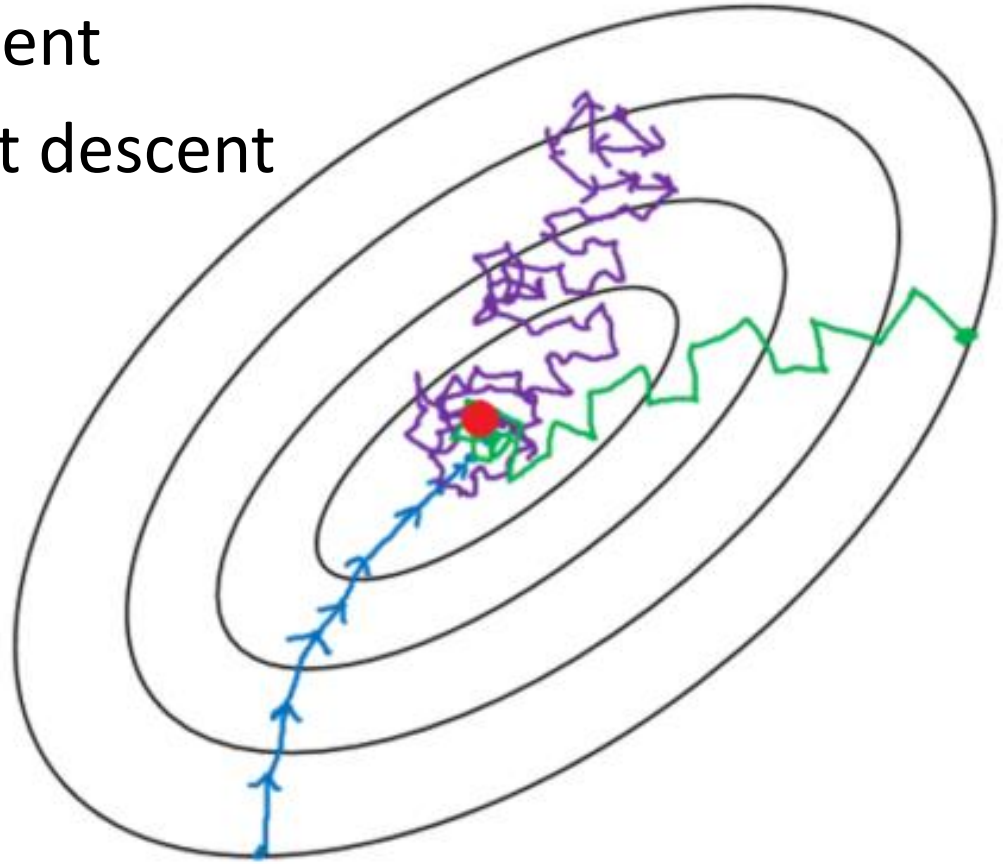
```
for i in range(num_epochs):
    grad = compute_gradient(data, params)
    params = params – learning_rate * grad

for i in range(num_epochs):
    np.random.shuffle(data)
    for example in data:
        grad = compute_gradient(example, params)
        params = params – learning_rate * grad

for i in range(num_epochs):
    np.random.shuffle(data)
    for batch in radom_minibatches(data, batch_size=32):
        grad = compute_gradient(batch, params)
        params = params – learning_rate * grad
```

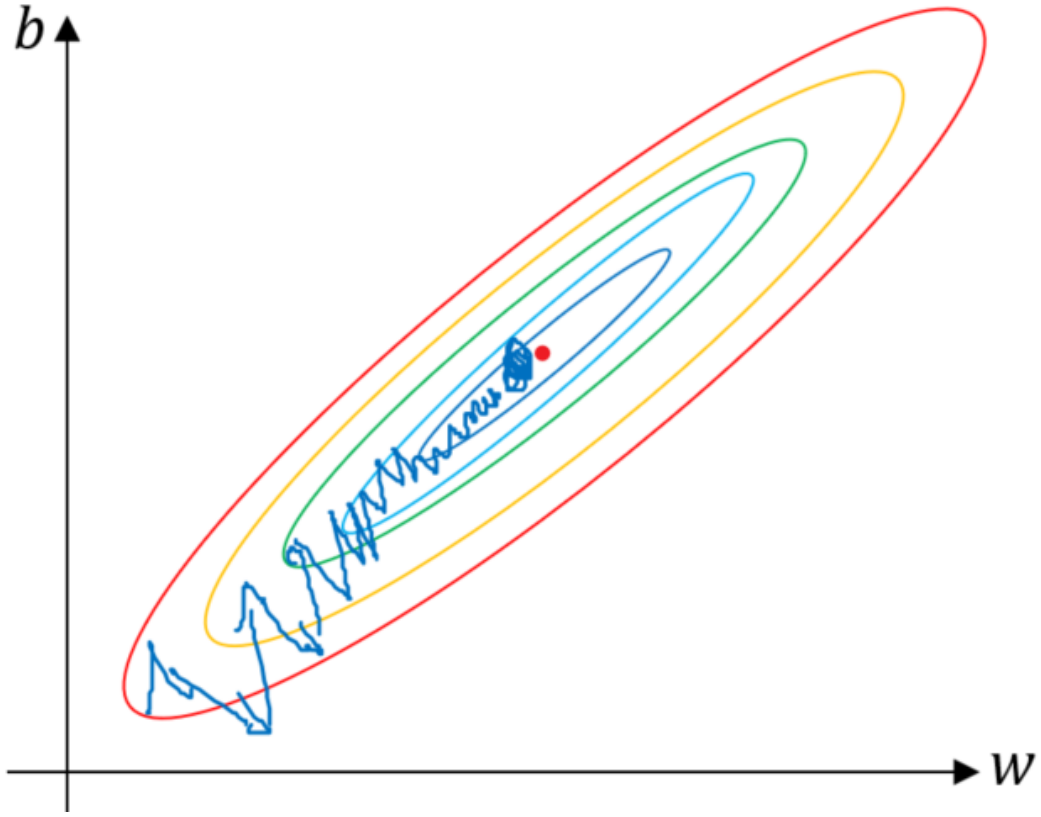
三种梯度下降法工作模式

- 批量梯度下降 batch gradient descent
- 随机梯度下降 stochastic gradient descent
- 小批随机梯度下降 mini-batch gradient descent

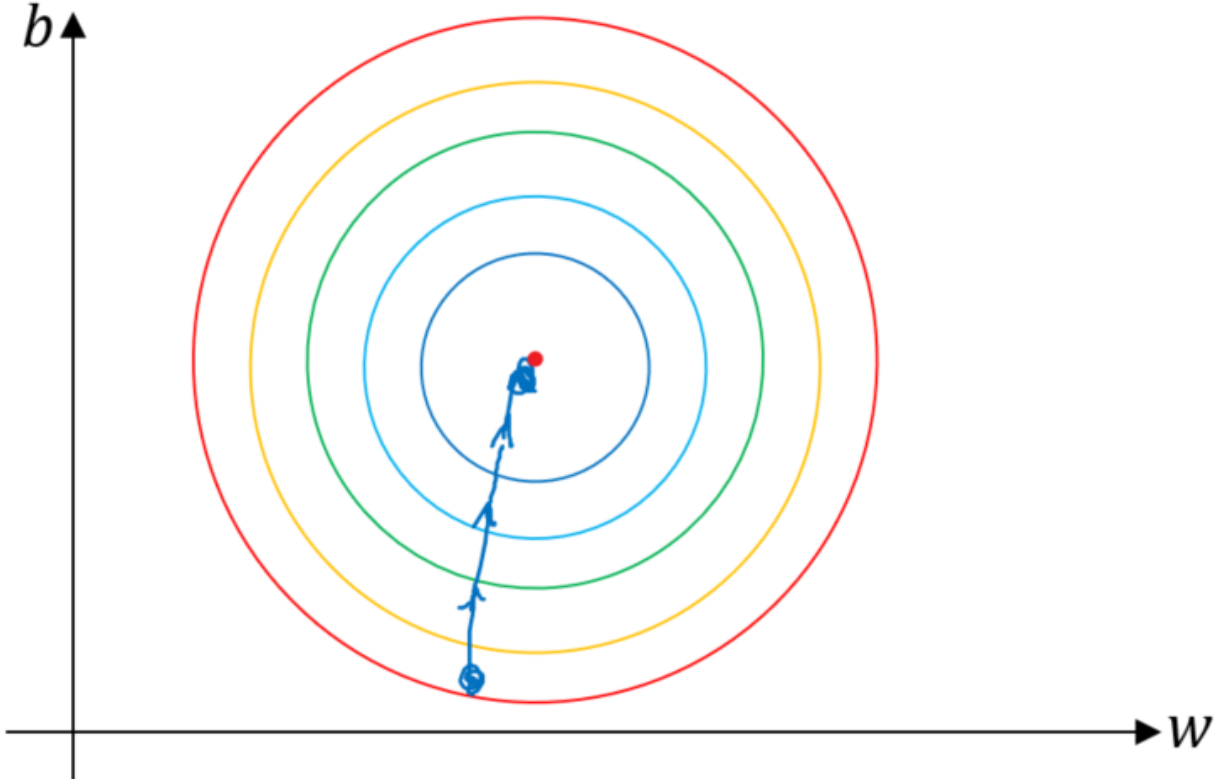


特征缩放

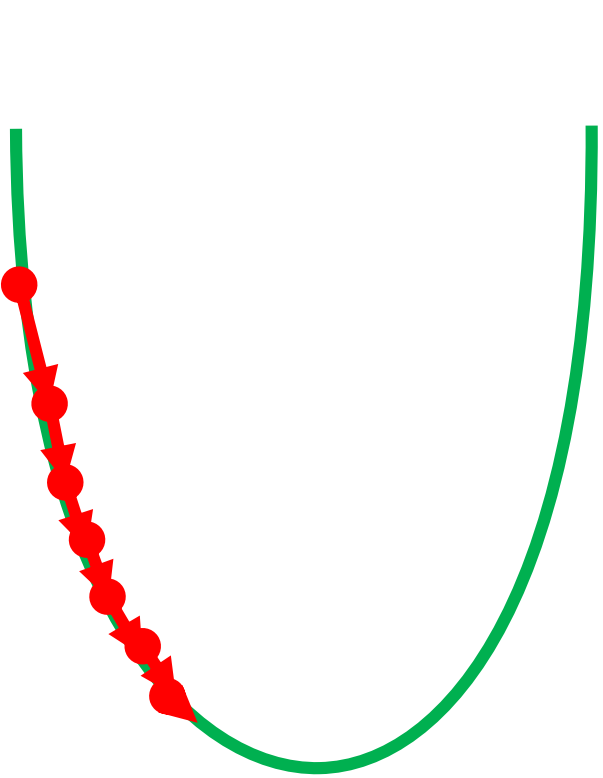
Unnormalized



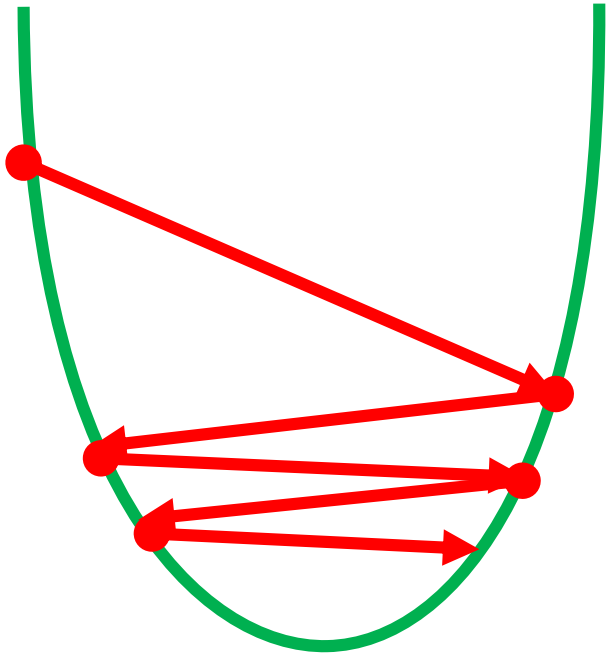
Normalized



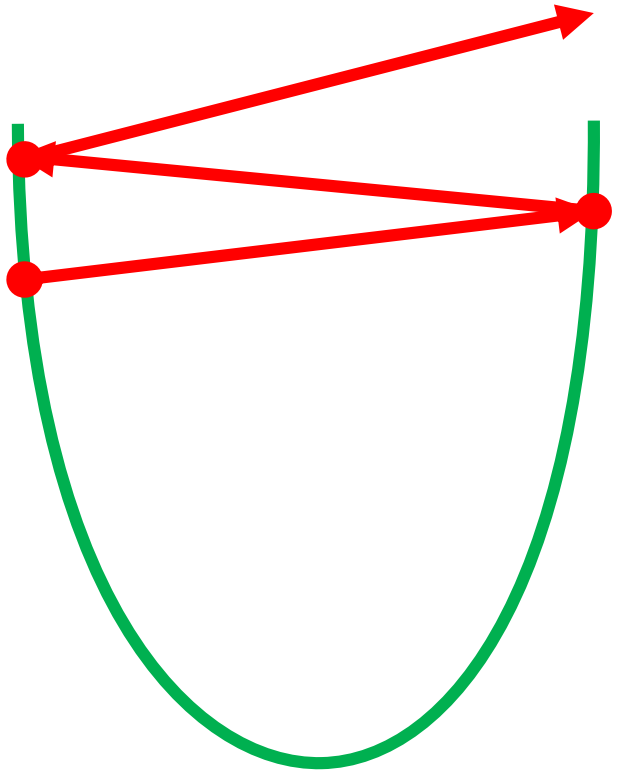
学习率



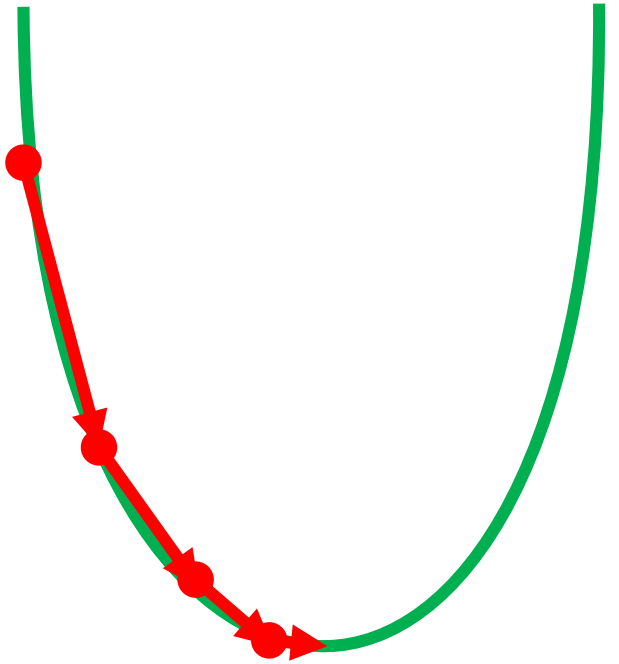
收敛太慢



波动、不稳定

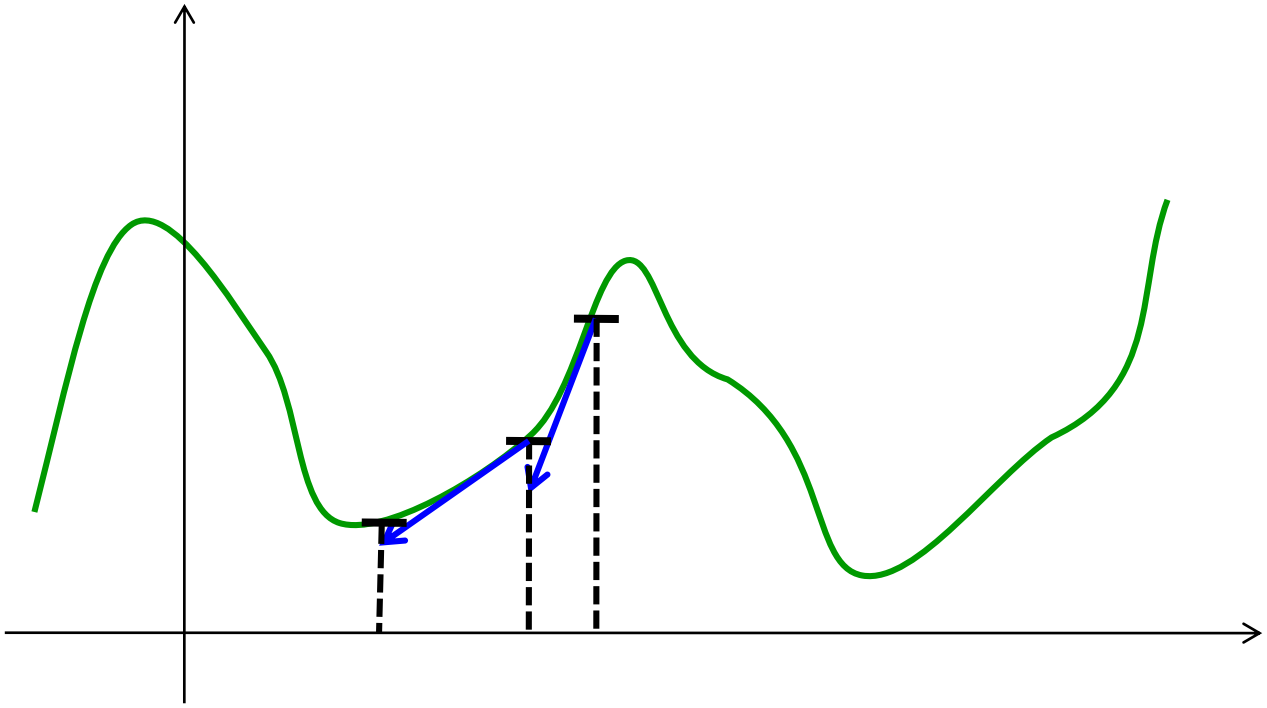


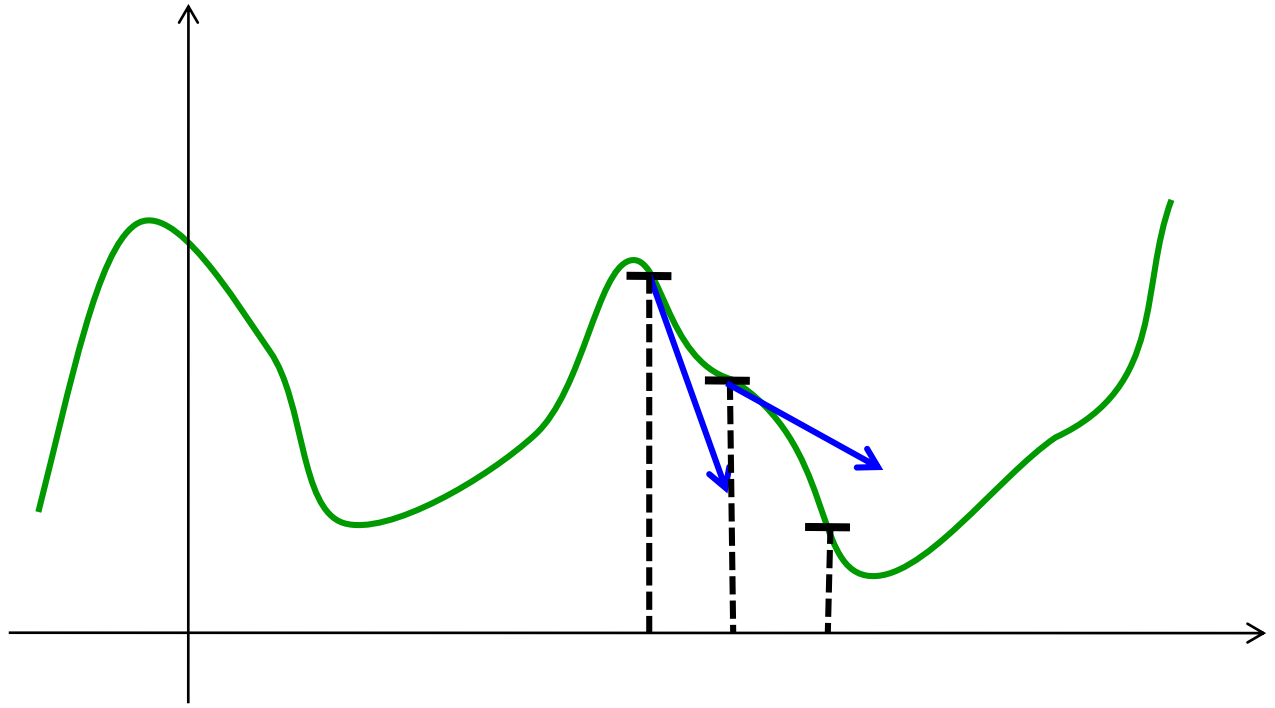
不能收敛



最佳收敛

初始值





最优的信用卡审核线性分类器？

$$g(\boldsymbol{x};\boldsymbol{w}) = sign(\boldsymbol{w}^T\boldsymbol{x})$$

$$l(g(\boldsymbol{x}), y) = \|y \neq g(\boldsymbol{x})\|_0$$

$$\arg\min_{\boldsymbol{w}} \sum_{\mathrm{i}=1}^{n} \|y^i \neq g(\boldsymbol{x}^i;\boldsymbol{w})\|_0$$

- 意义：找到错误分类个数最少的模型系数
- 不易求解：在带有噪声的数据中（非线性可分）时，NP-hard

线性回归

- 给用户发放信用卡的额度？
- 回归问题：预测函数取值在整个实数空间
- 假设H：$h(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}$
- 与感知器很像，但是没有$sign$函数

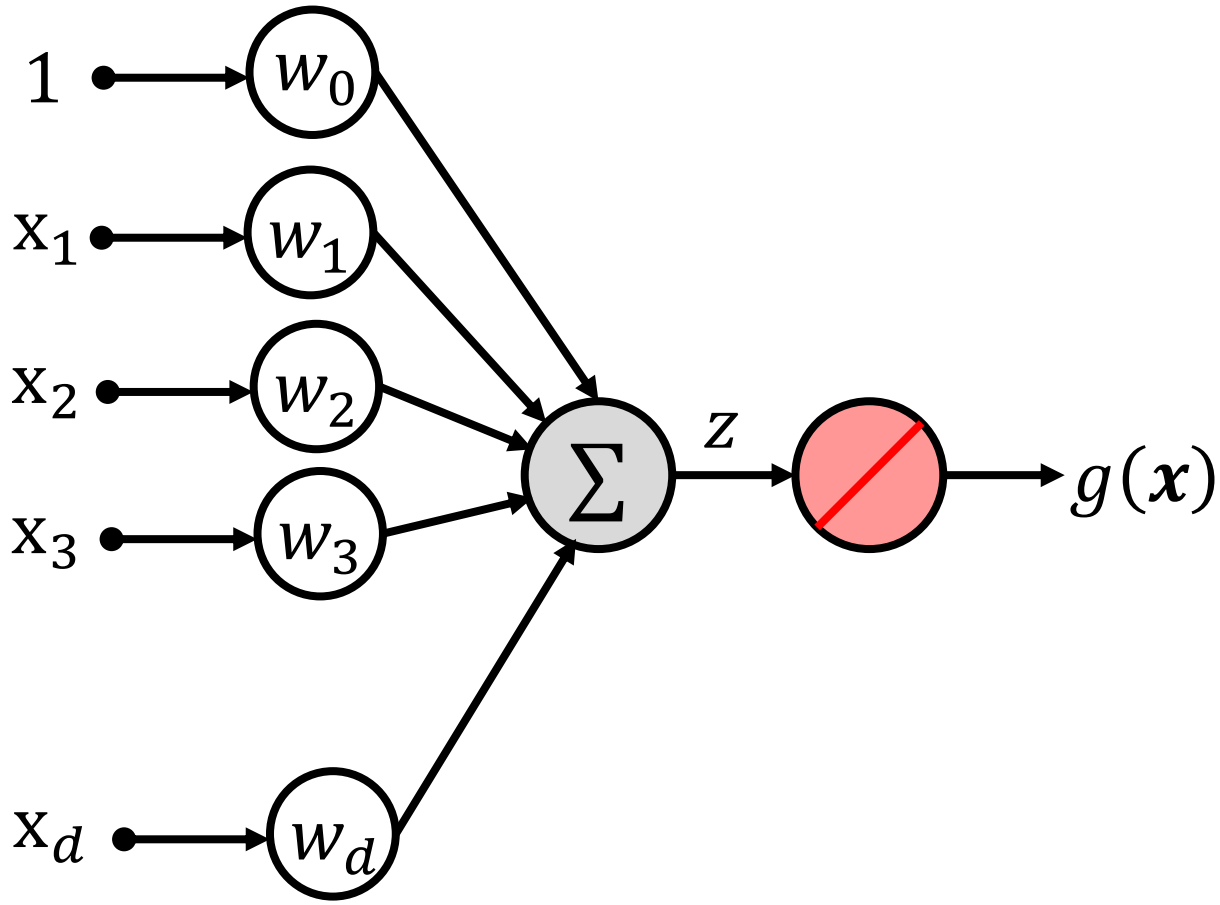
年龄	23岁
年薪	100,000
工作年限	120/95
当前欠款	240

~~是否批准信用卡申请？~~

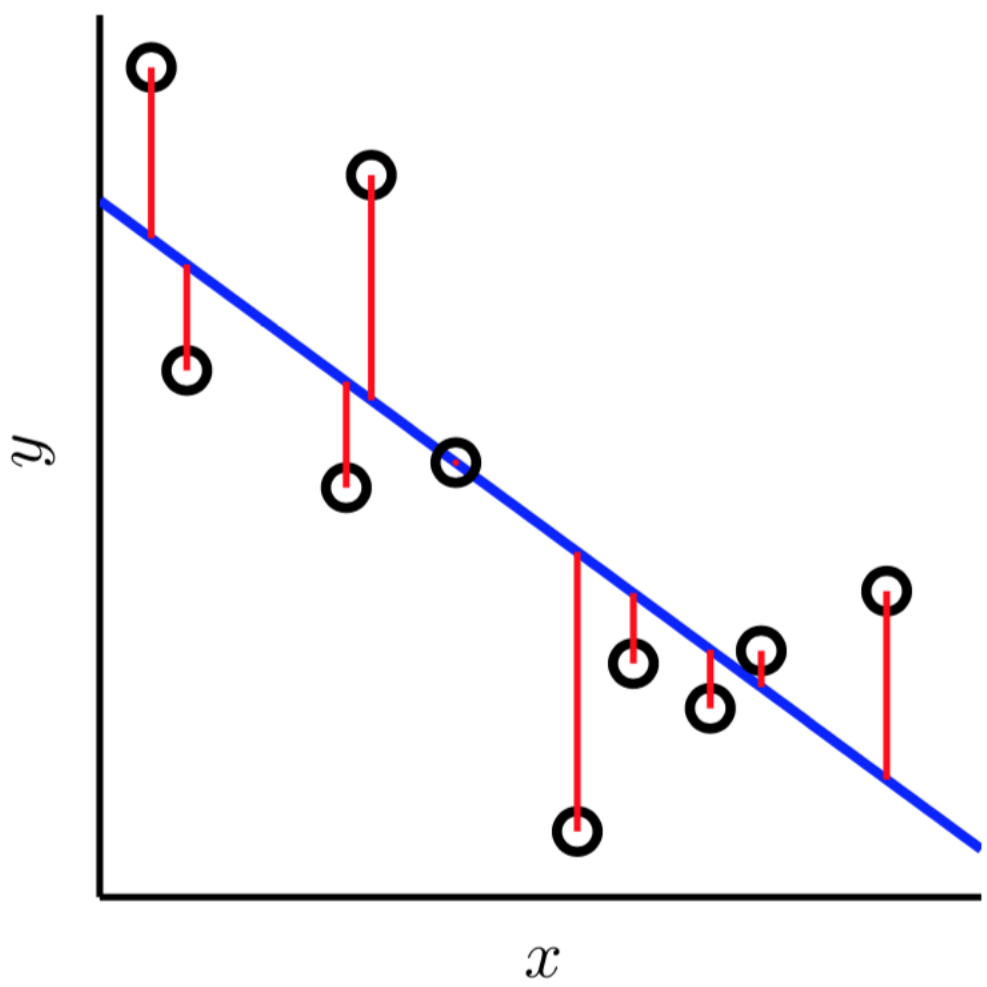
批准多少信用卡额度？

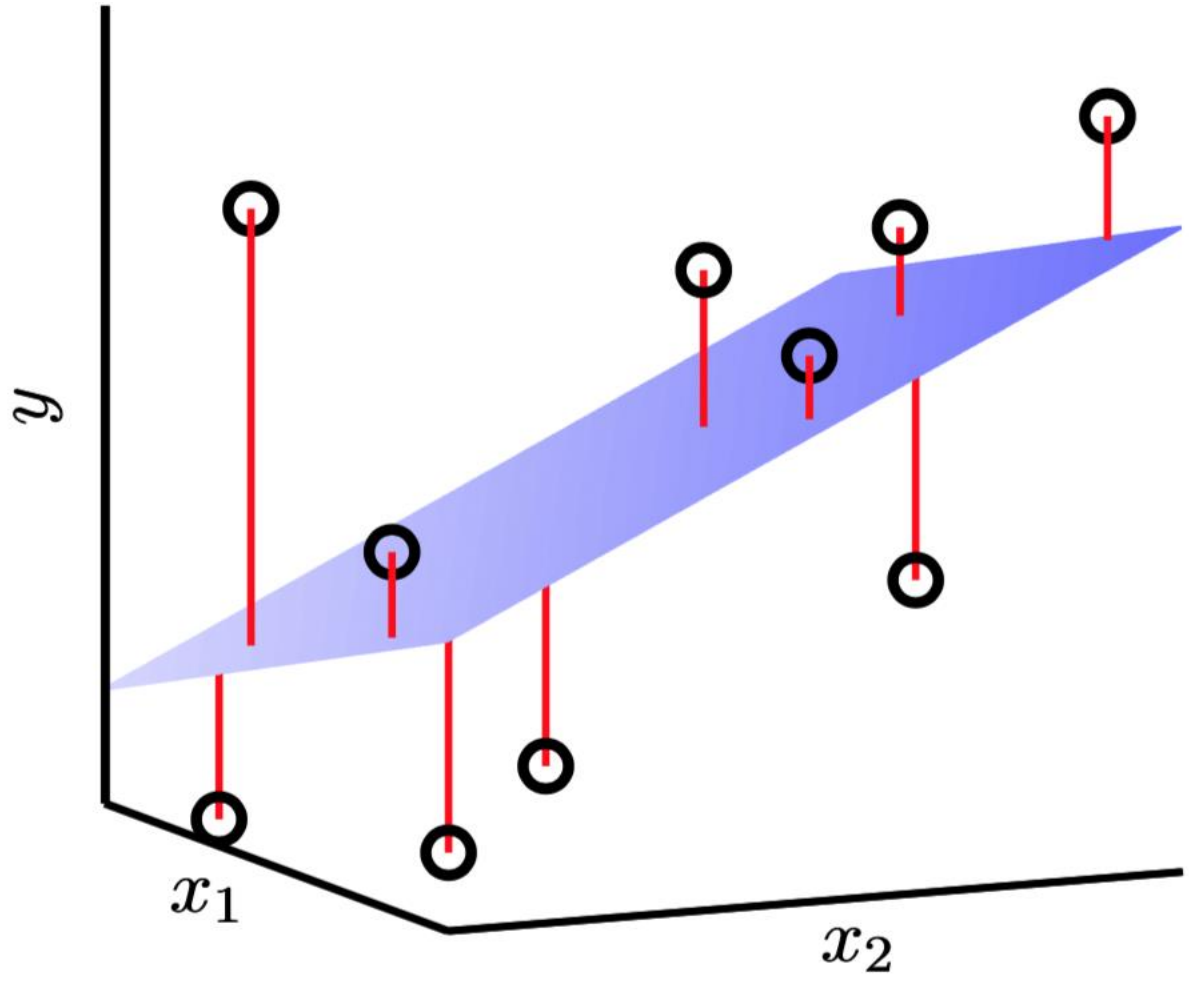
线性回归

$$g(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x}$$



线性回归





线性回归的损失函数

- 在一维或者多维空间里，线性回归的目标是找到一条直线(对应一维)、一个平面(对应二维)或者更高维的超平面，使样本集中的点更接近它，也就是残留误差最小化
- 使用平方损失函数，也被称为最小二乘法

$$l\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right),y^i\right)=\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)-y^i\right)^2$$

$$L=\frac{1}{n}\sum_{i=0}^{n}\left(\boldsymbol{w}^T\boldsymbol{x}^i-y^i\right)^2$$

梯度计算

$$L = \frac{1}{n}\sum_{i=0}^{n}\left(\boldsymbol{w}^T\boldsymbol{x}^i - y^i\right)^2$$

$$L = \frac{1}{n}\left\|\begin{bmatrix}\boldsymbol{x}^{1T}\boldsymbol{w} - y^1\\ \boldsymbol{x}^{2T}\boldsymbol{w} - y^2\\ \vdots\\ \boldsymbol{x}^{nT}\boldsymbol{w} - y^n\end{bmatrix}\right\|^2 = \frac{1}{n}\left\|\begin{bmatrix}\boldsymbol{x}^{1T}\\ \boldsymbol{x}^{2T}\\ \vdots\\ \boldsymbol{x}^{nT}\end{bmatrix}\begin{bmatrix}w_1\\ w_2\\ \vdots\\ w_d\end{bmatrix} - \begin{bmatrix}y^1\\ y^2\\ \vdots\\ y^n\end{bmatrix}\right\|^2$$

$$L = \frac{1}{n}\|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}\|^2 = \frac{1}{n}\left(\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} - 2\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}\right)$$

$$\frac{\partial L}{\partial \boldsymbol{w}} = \frac{2}{n}\left(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} - \boldsymbol{X}^T\boldsymbol{y}\right) = \frac{2}{n}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})$$

梯度下降法求解线性回归

- 批量梯度下降：

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{2}{n} \boldsymbol{X}^T (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})$$

- 随机梯度下降：

$$\text{for } i = 1,2, \dots, n\text{:}$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{2}{n} \boldsymbol{x}^{i^T} \left(\boldsymbol{w}^T \boldsymbol{x}^i - y^i\right)$$

梯度下降法求解线性回归

- 批量梯度下降：

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{2}{n} \boldsymbol{X}^T (g(\boldsymbol{X}) - \boldsymbol{y})$$

- 随机梯度下降：

$$\text{for } i = 1,2, \dots, n$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{2}{n} \boldsymbol{x}^{i\,T} (g(\boldsymbol{x}^i) - y^i)$$

线性最小二乘法的闭合(closed-form)解

- 对于此类线性回归问题，优化二次凸函数，一阶导数为零，即：

$$\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y} = 0$$

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

其中，$\boldsymbol{X}^{+}$是$\boldsymbol{X}$的伪逆矩阵(pseudo-inverse):

$$\boldsymbol{X}^{+} = \left(\boldsymbol{X}^{T}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{T}\boldsymbol{y}$$

- 非线性最小二乘法无closed-form解，必须使用迭代式优化方法

逻辑回归

- 软性二分类（soft binary classification）

$$f(\boldsymbol{x}) = P(1|\boldsymbol{x}) \in [0,1]$$

- 有什么好的hypothesis（假设）能满足要求？

$$\boldsymbol{w}^T\boldsymbol{x} \in \mathcal{R}$$

硬性二分类：$g(\boldsymbol{x};\boldsymbol{w}) = sign(\boldsymbol{w}^T\boldsymbol{x}) \in \{0,1\}$

软性二分类：$g(\boldsymbol{x};\boldsymbol{w}) = sigmoid(\boldsymbol{w}^T\boldsymbol{x})$

年龄	29岁
性别	男
血压	120/95
胆固醇量	240
体重	80kg

~~是否患有心脏病？~~

患有心脏病的可能性？

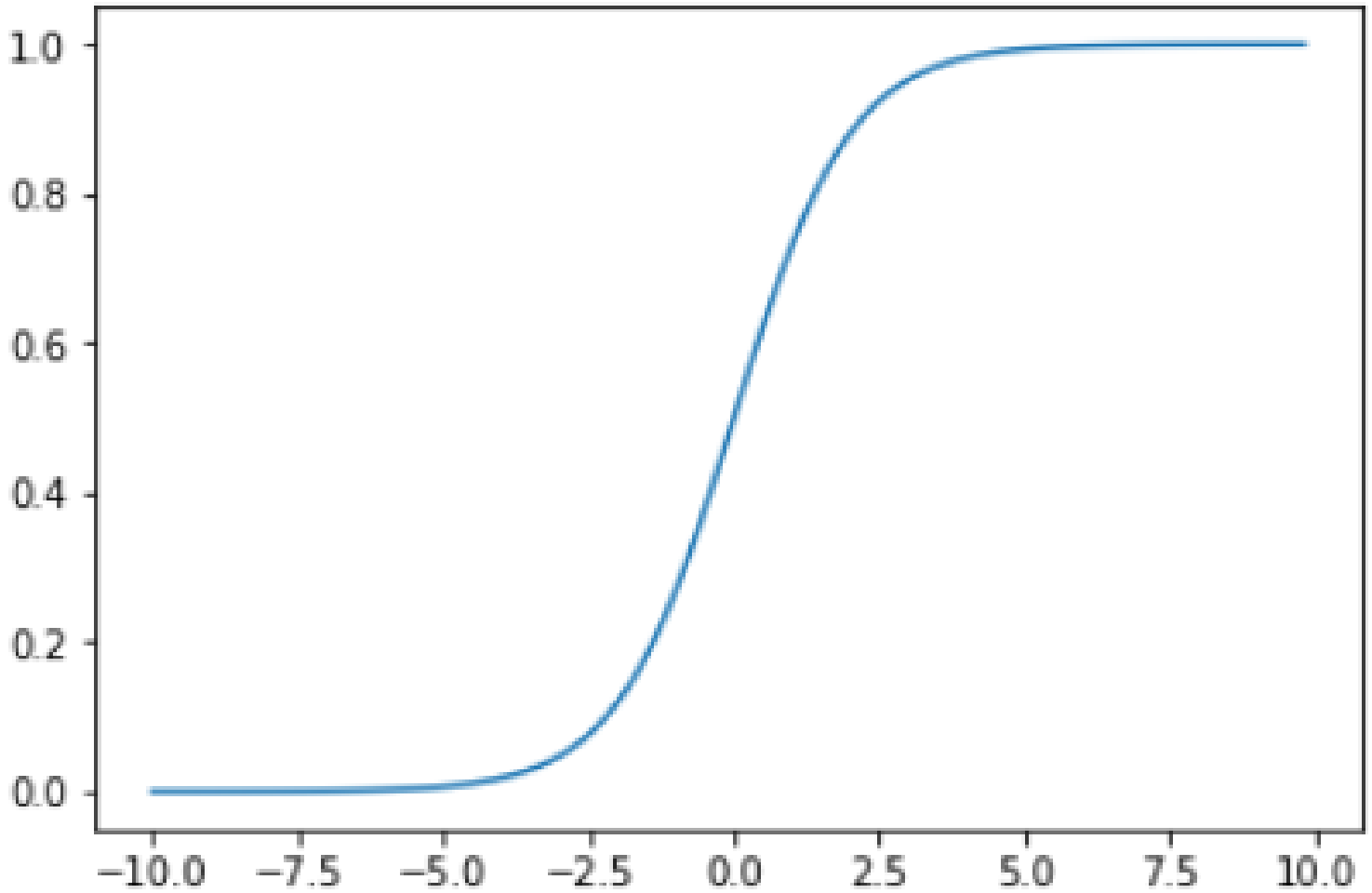
Sigmoid 函数

$$sigmoid(z) = \frac{e^z}{1 + e^z}$$

$$= \frac{1}{1 + e^{-z}}$$

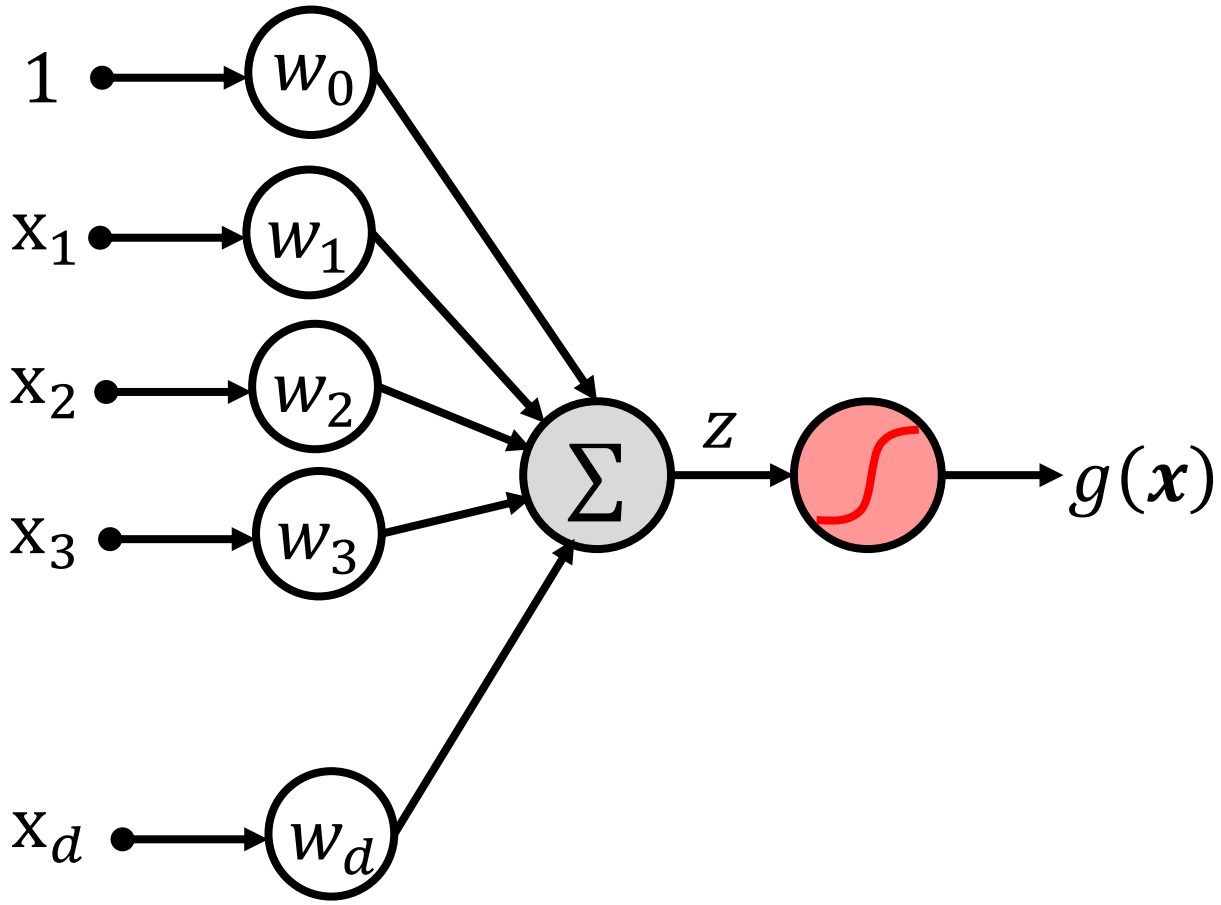
$$sigmoid'(z) = \frac{e^{-z}}{(1 + e^{-z})^2}$$

$$= sigmoid(z)(1 - sigmoid(z))$$



逻辑回归

$$g(\boldsymbol{x};\boldsymbol{w}) = sigmoid(\boldsymbol{w}^T\boldsymbol{x})$$



逻辑回归的数据

- 理想的数据：

$$(\boldsymbol{x}^1, y^1 = 0.9 = P(1|\boldsymbol{x}^1))$$

$$(\boldsymbol{x}^2, y^2 = 0.1 = P(1|\boldsymbol{x}^2))$$

$$(\boldsymbol{x}^3, y^3 = 0.6 = P(1|\boldsymbol{x}^3))$$

$$\dots$$

$$(\boldsymbol{x}^n, y^n = 0.2 = P(1|\boldsymbol{x}^4))$$

- 实际的数据：

$$(\boldsymbol{x}^1, y^1 = 1 \backsim P(y|\boldsymbol{x}^1))$$

$$(\boldsymbol{x}^2, y^2 = 0 \backsim P(y|\boldsymbol{x}^2))$$

$$(\boldsymbol{x}^3, y^3 = 1 \backsim P(y|\boldsymbol{x}^3))$$

$$\dots$$

$$(\boldsymbol{x}^n, y^n = 0 \backsim P(y|\boldsymbol{x}^4))$$

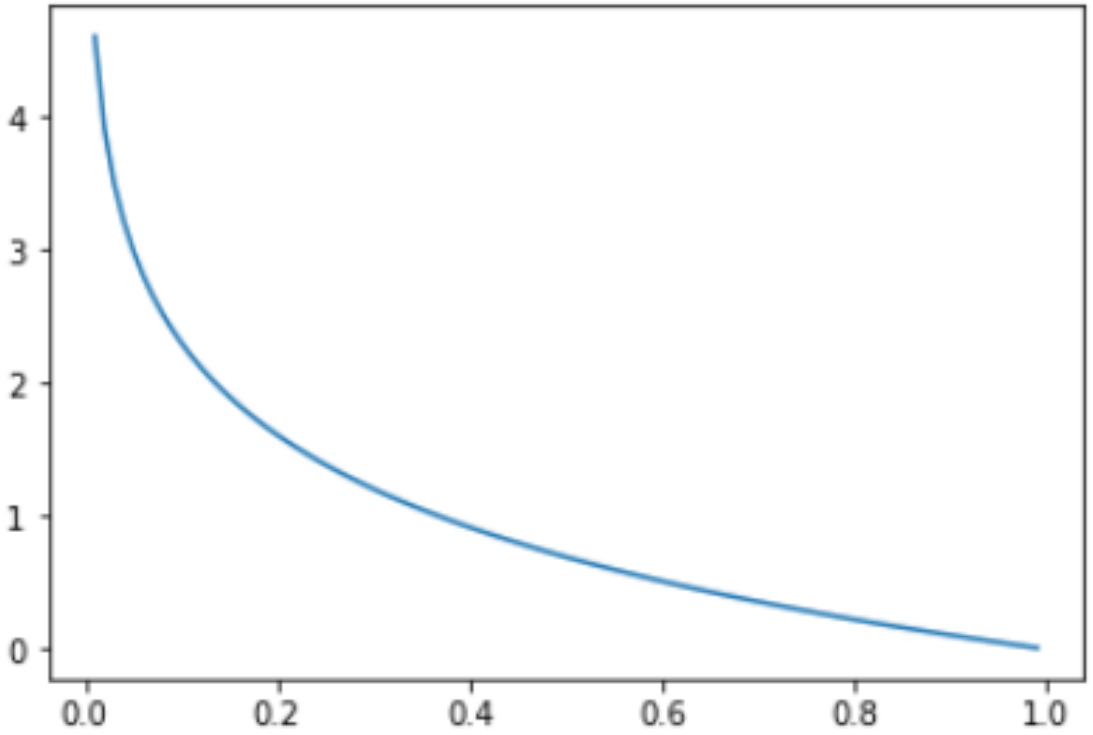
逻辑回归的损失函数

- Short answer：log 损失函数

$$l\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right),y^i\right)=\begin{cases}-log\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right) & ,y^i=1\\ -log\left(1-g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right) & ,y^i=0\end{cases}$$

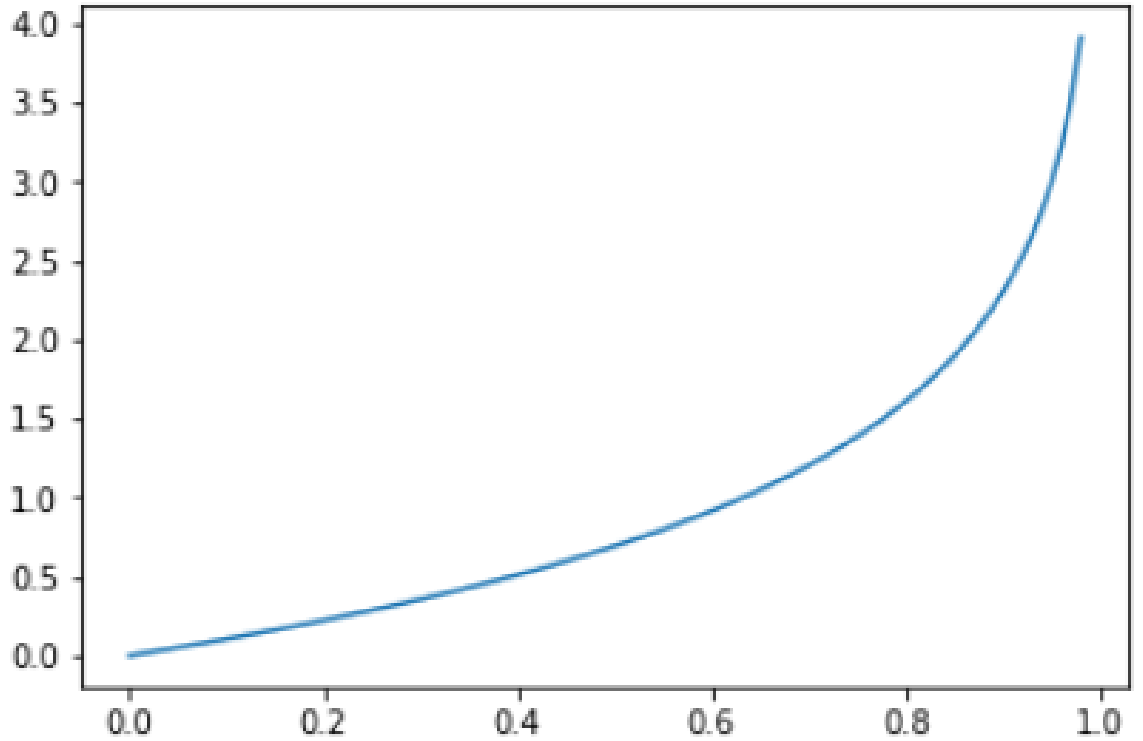
逻辑回归的损失函数

$-log(x)$



$y = 1$

$-log(1-x)$



$y = 0$

逻辑回归的损失函数

- 利用$y^i \in \{0,1\}$, 我们可以将两个合并：

$$l\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right),y^i\right) = -y^i log\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right) - \left(1-y^i\right)\left(1-g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right)$$

- 将所有数据的损失函数合起来：

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^i log\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right) + \left(1-\mathrm{y}^i\right) log\left(1-g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)\right)\right)$$

最大似然估计

- 考虑一组样本数据集 D: $(\boldsymbol{x}^1, \mathrm{y}^1), \ldots, (\boldsymbol{x}^n, \mathrm{y}^n)$，独立地由未知真实分布生成，则生成该数据的联合概率密度为：

$$P(D) = P(\boldsymbol{x}^1, \mathrm{y}^1)P\left(\boldsymbol{x}^2, \mathrm{y}^2\right) \ldots P(\boldsymbol{x}^n, \mathrm{y}^n)$$

$$P(D) = P(\boldsymbol{x}^1)P(\mathrm{y}^1|\boldsymbol{x}^1)P\left(\boldsymbol{x}^2\right)P\left(\mathrm{y}^2|\boldsymbol{x}^2\right) \ldots P(\boldsymbol{x}^n)P(\mathrm{y}^n|\boldsymbol{x}^n)$$

- 最大似然估计：使得数据联合概率密度函数最大的模型参数

$$\boldsymbol{w} = \arg\max_{\boldsymbol{w}} P(D)$$

- 因为$P\left(\boldsymbol{x}^i\right)$对于所有的$\boldsymbol{w}$是一样的，我们可以忽略它们

最大似然估计

最大似然估计：

$$\boldsymbol{w} = \arg\max_{\boldsymbol{w}} \prod_{i=1}^{n} P(\mathrm{y}^i|\boldsymbol{x}^i)$$

多个概率的乘积会因很多原因不便于计算，取log运算转化为求和运算；同时，添加负号，改为最小化问题：

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \frac{1}{n} \sum_{i=1}^{n} -log\big(P(\mathrm{y}^i|\boldsymbol{x}^i)\big)$$

最大似然估计

逻辑回归表示$y = 1$的概率意义：

$$g(\boldsymbol{x}; \boldsymbol{w}) = P(y = 1|\boldsymbol{x})$$

对应不同类别的概率：

$$P(y|\boldsymbol{x}) = \begin{cases} g(\boldsymbol{x}; \boldsymbol{w}) & \text{for } y = 1 \\ 1 - g(\boldsymbol{x}; \boldsymbol{w}) & \text{for } y = 0 \end{cases}$$

利用$y \in \{0,1\}$, 我们可以将两个合并：

$$P(y|\boldsymbol{x}) = g(\boldsymbol{x}; \boldsymbol{w})^y \big(1 - g(\boldsymbol{x}; \boldsymbol{w})\big)^{1-y}$$

最大似然估计

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} -log\left(P(\mathrm{y}^i|\boldsymbol{x}^i)\right)$$

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} -log\left(g(\boldsymbol{x}^i;\boldsymbol{w})^{\mathrm{y}^i}\left(1-g(\boldsymbol{x}^i;\boldsymbol{w})\right)^{1-\mathrm{y}^i}\right)$$

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^i log\left(g(\boldsymbol{x}^i;\boldsymbol{w})\right)+\left(1-\mathrm{y}^i\right)log\left(1-g(\boldsymbol{x}^i;\boldsymbol{w})\right)\right)$$

逻辑回归的损失函数

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^{i} log\left(g\left(\boldsymbol{x}^{i};\boldsymbol{w}\right)\right) + \left(1-\mathrm{y}^{i}\right) log\left(1-g\left(\boldsymbol{x}^{i};\boldsymbol{w}\right)\right)\right)$$

- 与之前的损失函数完全一样
- 又名**交叉熵** (cross entropy) 损失函数，也可用作计算多类分类的softmax函数的损失函数
  - 对应类别的$-log$ 损失

梯度计算

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^{i} log\left(sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\right) + \left(1-\mathrm{y}^{i}\right) log\left(1-sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\right)\right)$$

$$\frac{\partial L}{\partial w_j} = -\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\mathrm{y}^{i}}{sigmoid(\boldsymbol{w}^{T}\boldsymbol{x}^{i})} - \frac{1-\mathrm{y}^{i}}{1-sigmoid(\boldsymbol{w}^{T}\boldsymbol{x}^{i})}\right)\frac{\partial sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)}{\partial w_i}$$

$$\frac{\partial L}{\partial w_j} = -\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\mathrm{y}^{i}}{sigmoid(\boldsymbol{w}^{T}\boldsymbol{x}^{i})} - \frac{1-\mathrm{y}^{i}}{1-sigmoid(\boldsymbol{w}^{T}\boldsymbol{x}^{i})}\right) sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\left(1-sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\right)\boldsymbol{x}_j^{i}$$

$$\frac{\partial L}{\partial w_j} = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^{i}\left(1-sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\right) - \left(1-\mathrm{y}^{i}\right)sigmoid\left(\boldsymbol{w}^{T}\boldsymbol{x}^{i}\right)\right)\boldsymbol{x}_j^{i}$$

梯度计算

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^i\left(1 - sigmoid\left(\boldsymbol{w}^T\boldsymbol{x}^i\right)\right) - \left(1-\mathrm{y}^i\right)sigmoid\left(\boldsymbol{w}^T\boldsymbol{x}^i\right)\right)\boldsymbol{x}_j^i$$

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = -\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{y}^i - sigmoid\left(\boldsymbol{w}^T\boldsymbol{x}^i\right)\right)\boldsymbol{x}_j^i$$

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_j^i\left(sigmoid\left(\boldsymbol{w}^T\boldsymbol{x}^i\right) - \mathrm{y}^i\right)$$

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_j^i\left(g\left(\boldsymbol{x}^i\right) - \mathrm{y}^i\right)$$

向量化梯度计算

$$\frac{\partial L}{\partial \boldsymbol{w}_j} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_j^i \left(g\left(\boldsymbol{x}^i\right) - \mathrm{y}^i\right)$$

$$\begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{w}_0} \\ \frac{\partial L}{\partial \boldsymbol{w}_1} \\ \vdots \\ \frac{\partial L}{\partial \boldsymbol{w}_d} \end{bmatrix} = \frac{1}{n} \begin{bmatrix} \boldsymbol{x}_0^1 & \boldsymbol{x}_0^2 & \cdots & \boldsymbol{x}_0^n \\ \boldsymbol{x}_1^1 & \boldsymbol{x}_1^2 & \cdots & \boldsymbol{x}_1^n \\ & \vdots & & \\ \boldsymbol{x}_d^1 & \boldsymbol{x}_d^2 & \cdots & \boldsymbol{x}_d^n \end{bmatrix} \begin{bmatrix} g(\boldsymbol{x}^1) - \mathrm{y}^1 \\ g\left(\boldsymbol{x}^2\right) - \mathrm{y}^2 \\ \vdots \\ g(\boldsymbol{x}^n) - \mathrm{y}^n \end{bmatrix}$$

$$\frac{\partial L}{\partial \boldsymbol{w}} = \frac{1}{n}\boldsymbol{X}^T\left(g(\boldsymbol{X}) - \boldsymbol{y}\right)$$

梯度下降法求解逻辑回归

- 批量梯度下降：

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{1}{n} \boldsymbol{X}^T (g(\boldsymbol{X}) - \boldsymbol{y})$$

- 随机梯度下降：

$$\text{for } i = 1,2, \dots, n$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{1}{n} \boldsymbol{x}^{i^T} (g(\boldsymbol{x}^i) - y^i)$$

为什么不用平方损失函数？

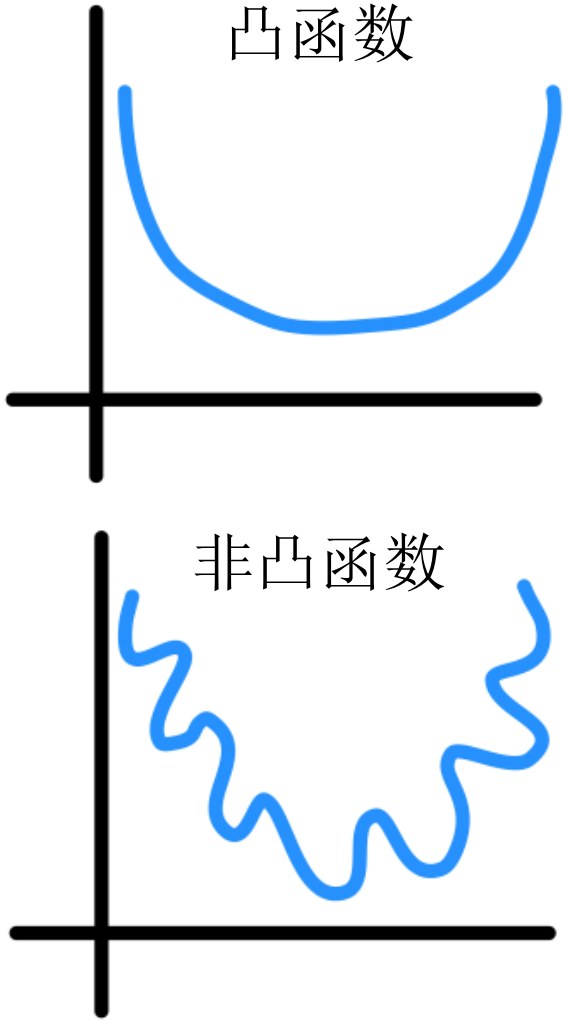
- 如果我们像线性回归中一样，定义：

$$l\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right),y^i\right)=\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right)-y^i\right)^2$$

- 则我们需要优化：

$$\boldsymbol{w}=\arg\min_{\boldsymbol{w}}\frac{1}{n}\sum_{\mathrm{i}=1}^{n}\left(sigmoid(\boldsymbol{w}^T\boldsymbol{x}^i)-y^i\right)^2$$

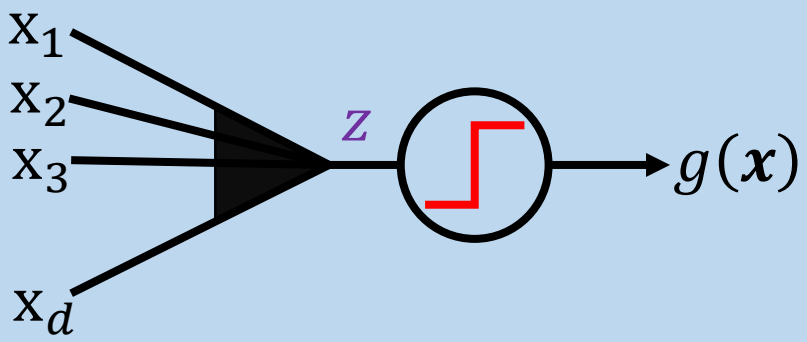
- 这是一个非凸函数（non-convex），这就意味着代价函数有着许多的局部最小值，这不利于我们的求解



三种线性模型

线性分类

$g(\boldsymbol{x}) = \text{sign}(z)$

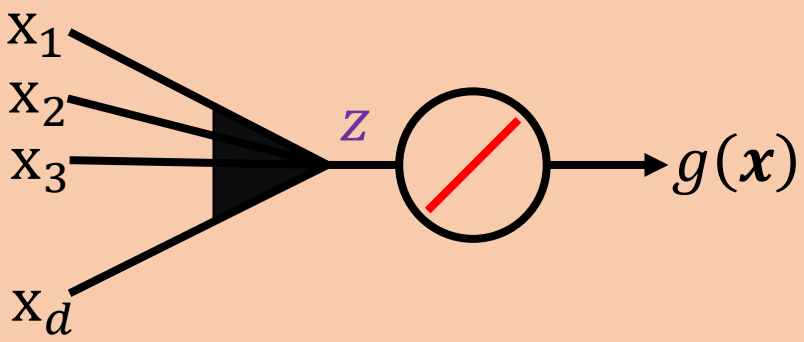


0-1损失函数

不易求解

线性回归

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

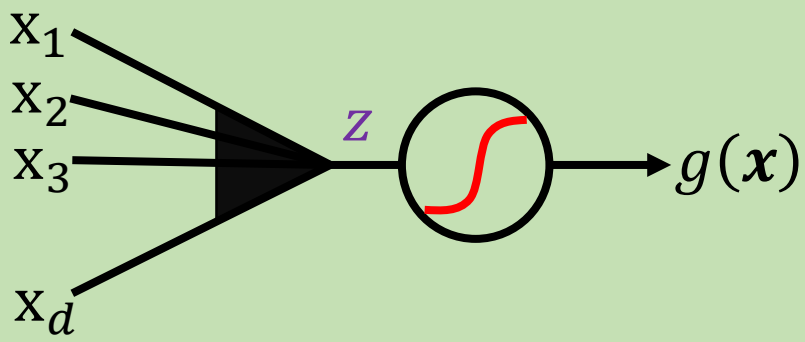


平方损失函数

有闭合解

逻辑回归

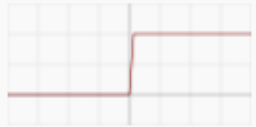
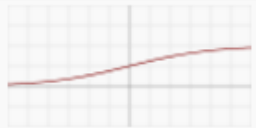
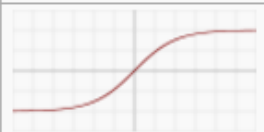
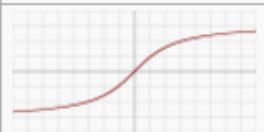
$g(\boldsymbol{x}) = sigmoid(z)$



log损失函数

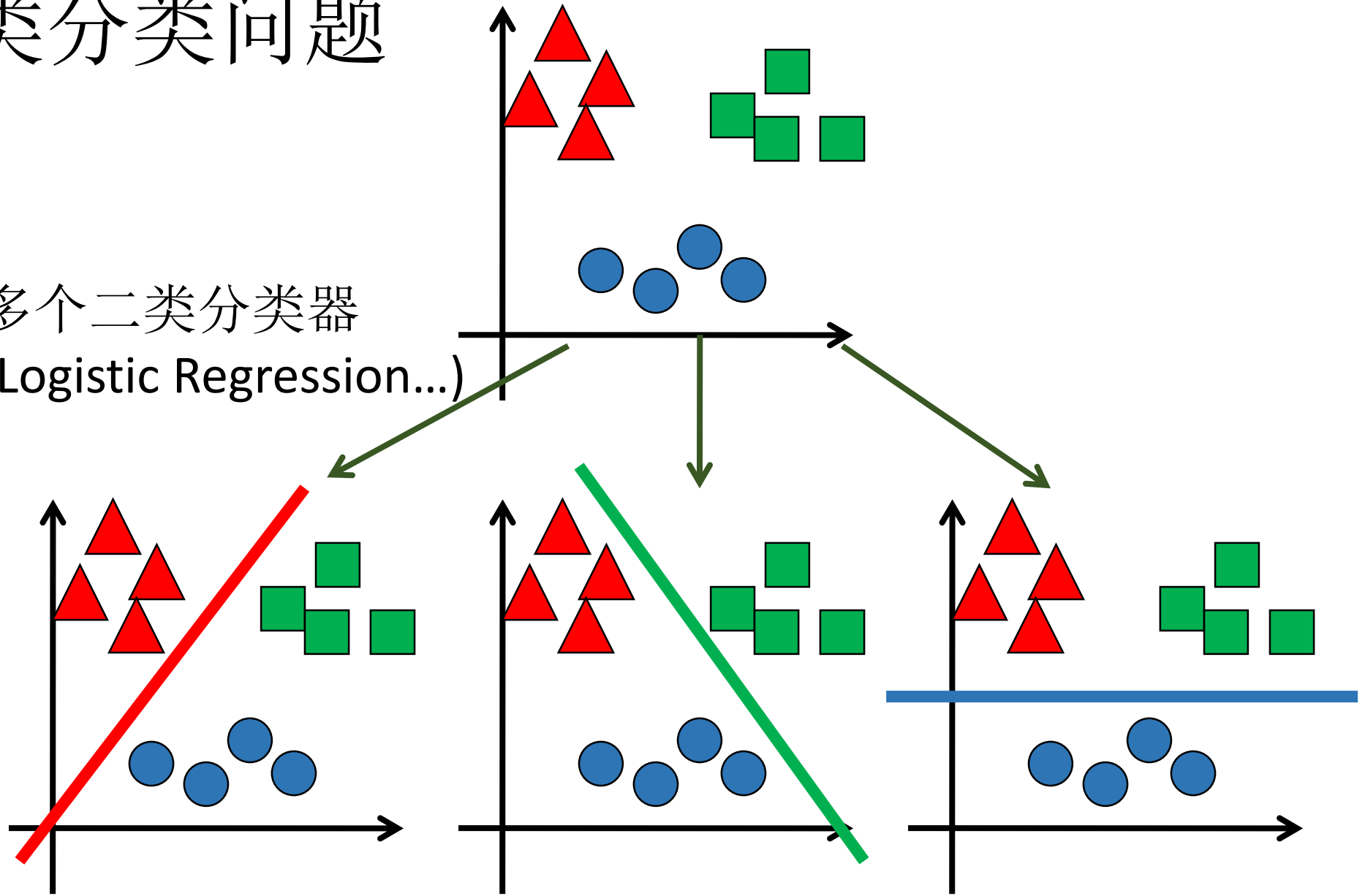
梯度下降法求解

z是关于特征$\boldsymbol{x}$的线性函数

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

多类分类问题

多个二类分类器
(Logistic Regression…)



Softmax Regression

- 假设K类，定义K个不同的线性函数：

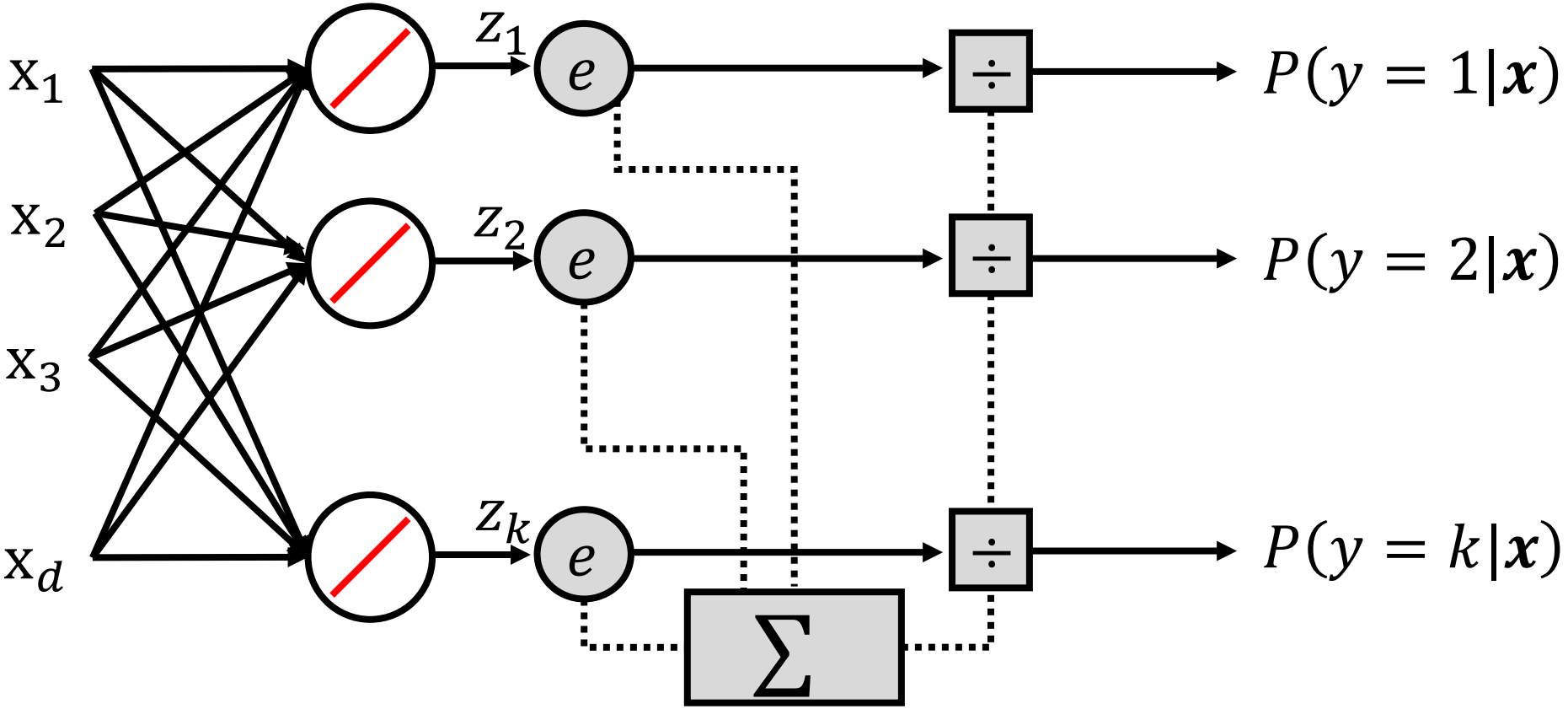
$$z_k = \boldsymbol{w}_k^T \boldsymbol{x} + b_k$$

- K类的softmax函数为：

$$P(y = i|\boldsymbol{x}) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

- 它有多个值，所有值加起来刚好等于1，每个输出都映射到了0到1区间，可以看成是概率问题
- 实际上正确的名字应该是soft argmax

Softmax


x_1
x_2
x_3
x_d
z_1
z_2
z_k
e
e
e
÷
÷
÷
$\sum$
$P(y = 1|\boldsymbol{x})$
$P(y = 2|\boldsymbol{x})$
$P(y = k|\boldsymbol{x})$

K=2时

$$P(y=1|\boldsymbol{x}) = \frac{e^{z_1}}{e^{z_0}+e^{z_1}}$$

$$P(y=1|\boldsymbol{x}) = \frac{1}{1+e^{z_0-z_1}}$$

$$P(y=1|\boldsymbol{x}) = \frac{1}{1+e^{-(z_1-z_0)}}$$

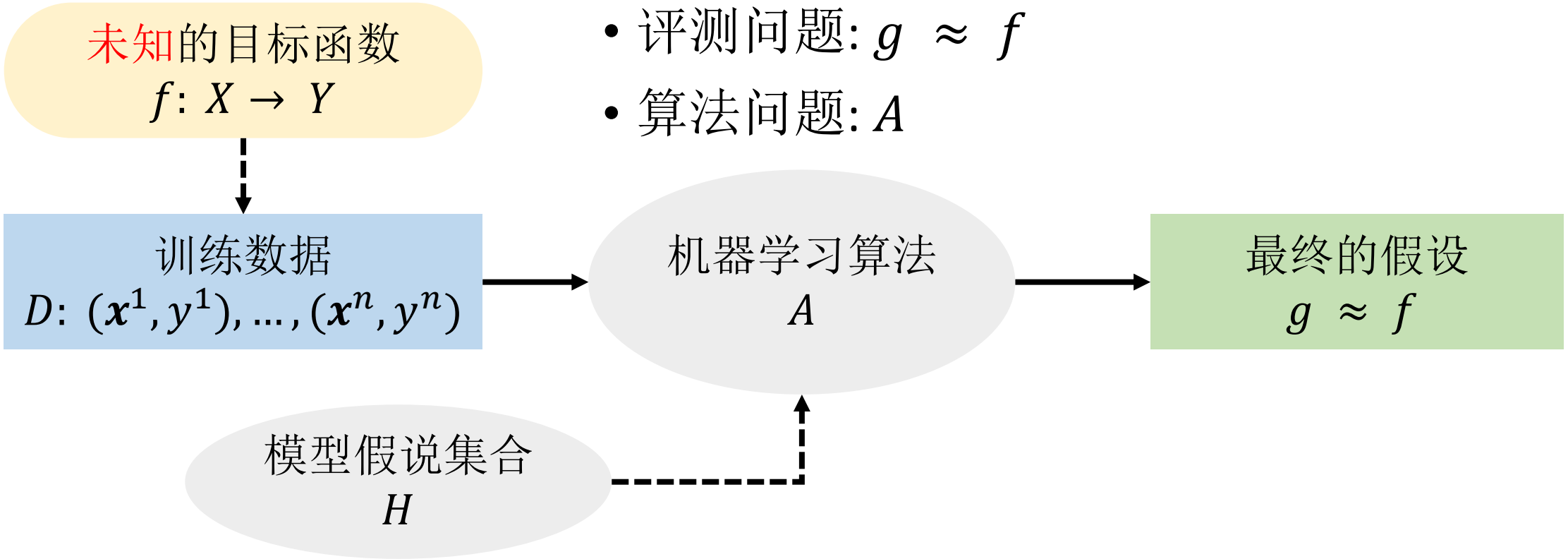
等价于逻辑回归中，只有一个输出单元：

$$P(y=1|\boldsymbol{x}) = \frac{1}{1+e^{-z_1}}$$

如何学习得更好？

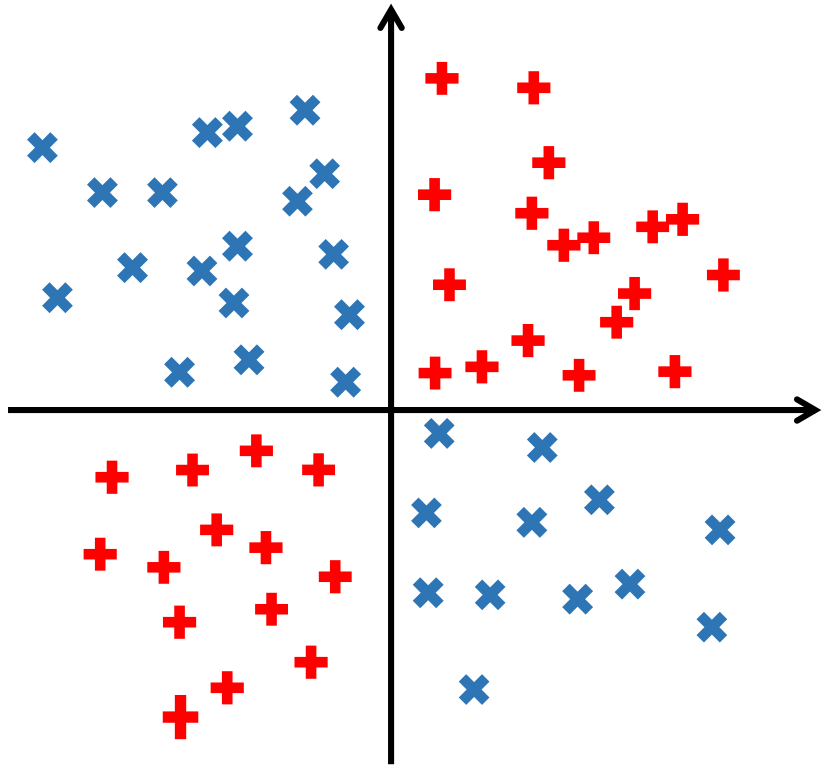
如何学习的更好

- 模型问题: H
- 数据问题: D
- 评测问题: $g \approx f$
- 算法问题: A



特征交叉

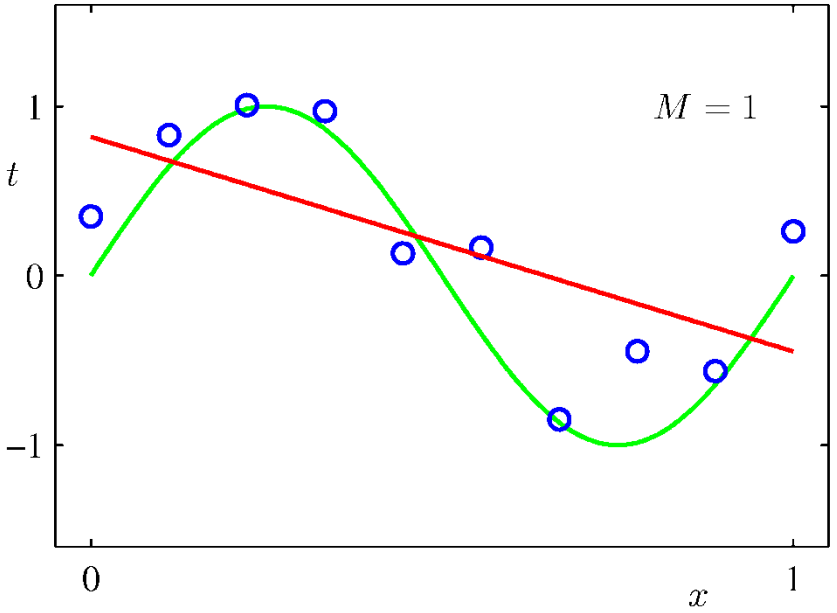
- 线性不可分
  - 任何一条线都不能很好地将两类分开
- 可以创建一个新的特征
  - 我们通过将x_1与x_2组合来创建新的x_3的特征组合
  - $x_3 = x_1 \cdot x_2$
- 在使用扩展的线性模型时辅以特征组合一直都是训练大规模数据集的有效方法



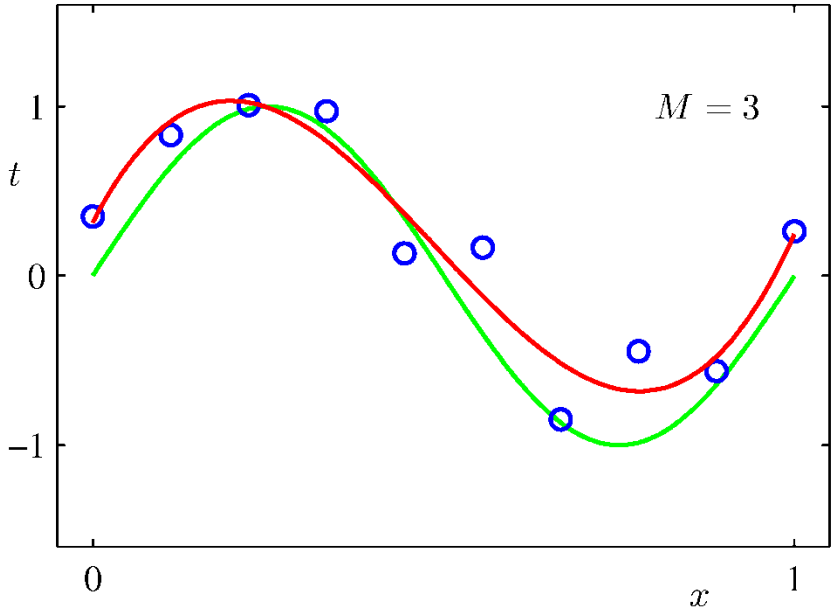
第一次AI低谷（1974-1980）

- 1969年，美国著名人工智能学者马文·明斯基和西蒙·派珀特共同出版了《感知器:计算几何简介》一书，书中论证了感知器模型的两个关键问题:
  - 单层的神经网络无法解决不可线性分割的问题，典型例子就是异或门。
  - 当时计算机的能力不足，无法满足计算量的需求。
- 由于这些问题在当时无法得到解决，感知器的发展几乎停滞，以神经网络为基础的人工智能研究开 始进入低潮，相关项目长期无法得到政府经费支持，这段时间被称为的“第一次低谷”。

多项式线性回归模型



$$g(\boldsymbol{x}) = w_0 + w_1 x$$



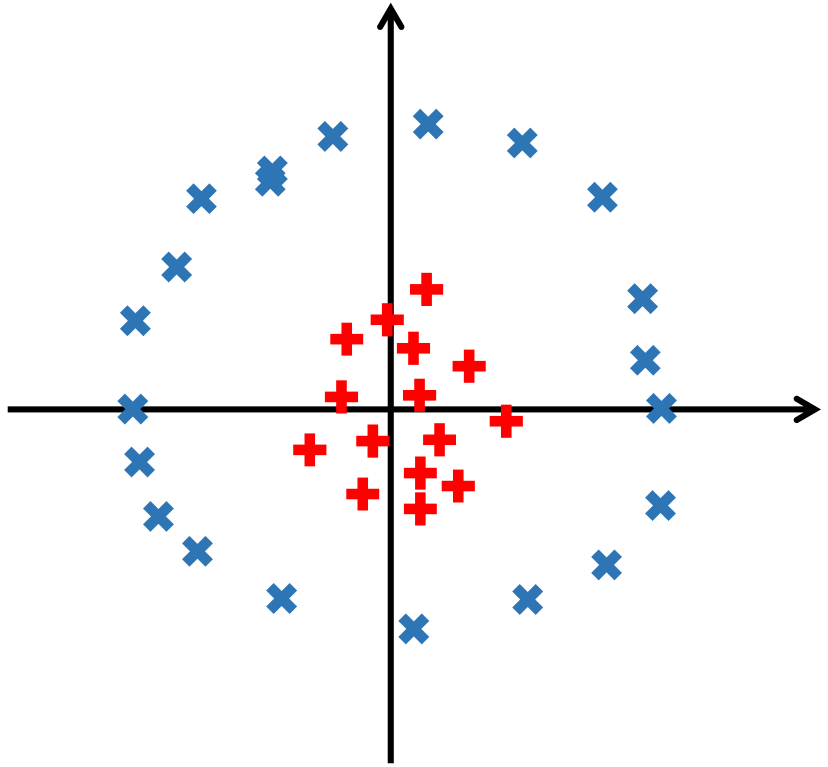
$$g(\boldsymbol{x}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

$$= w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$$

扩展的线性模型

问题？

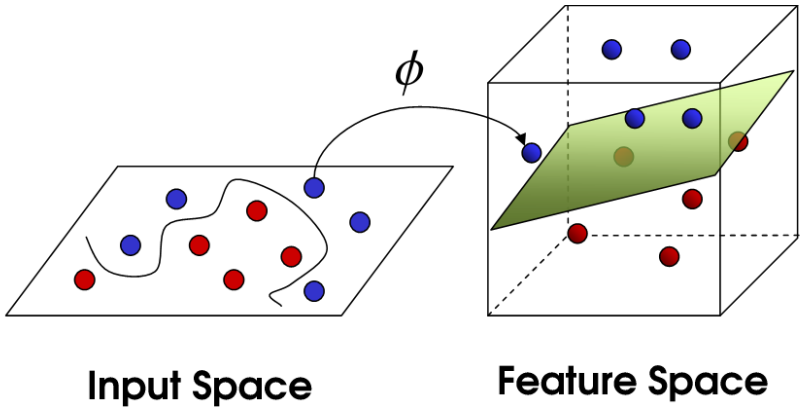
- 如何用逻辑回归分类如下数据？



$$h(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2$$

支持向量机SVM与核函数

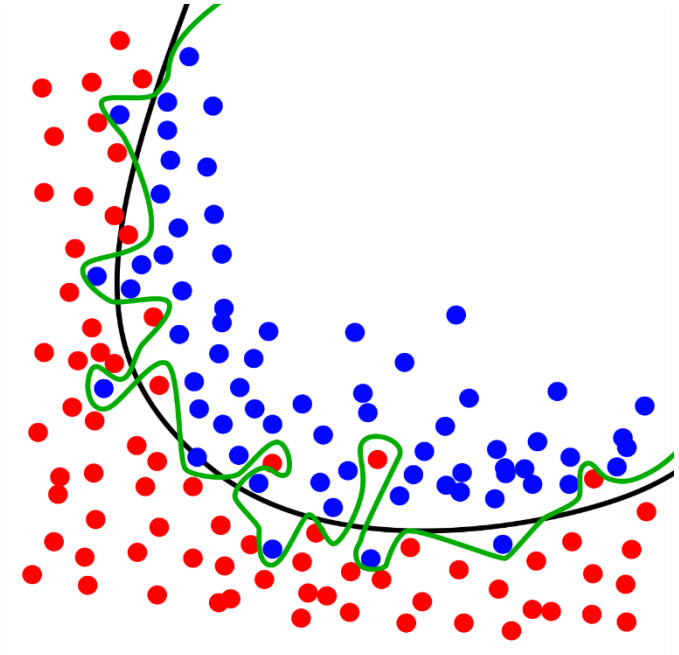
- 将原始向量从低维映射到高维是一个解决非线性可分性的重要手段
- 使用核函数(kernel)
  - 引入核函数不需再显式地定义出映射函数，就能计算两个高维空间中的向量的内积，而且时间复杂度降低
  - 核函数并不只是应用于SVM的，只要算法中出现了内积的计算，都可以考虑使用核函数，从而提高处理高维数据的性能

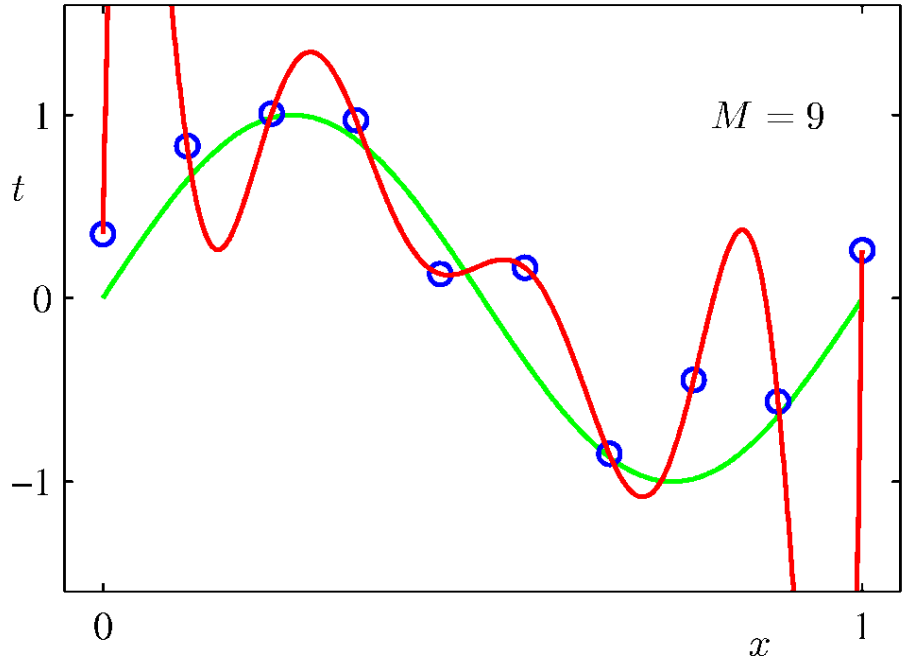


- 就是这个算法把神经网络按在地上摩擦了大概15年的时间，直到深度学习的兴起

过拟合(Overfitting)

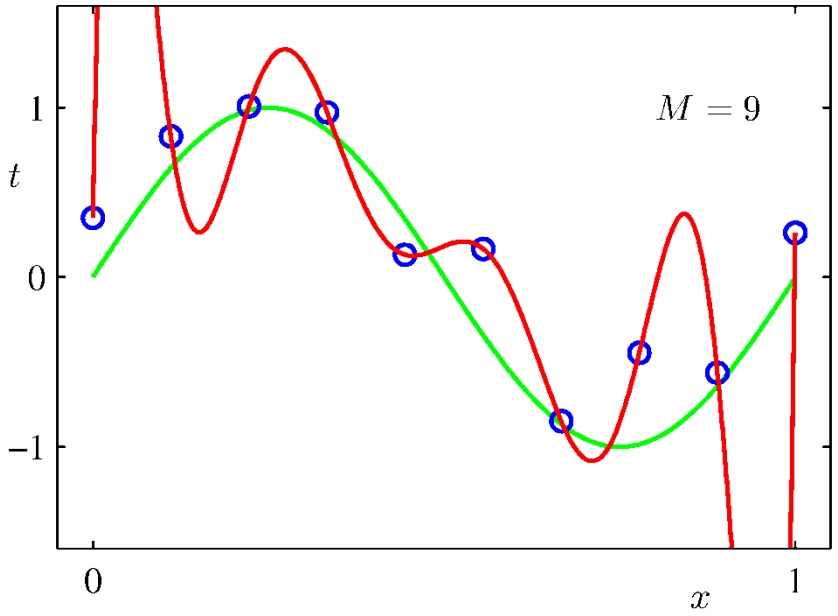
- 完全拟合训练数据不是我们真正关心的
  - 模型拟合了样本中的噪声
  - 而这样的模型在预测新数据方面表现会非常糟糕

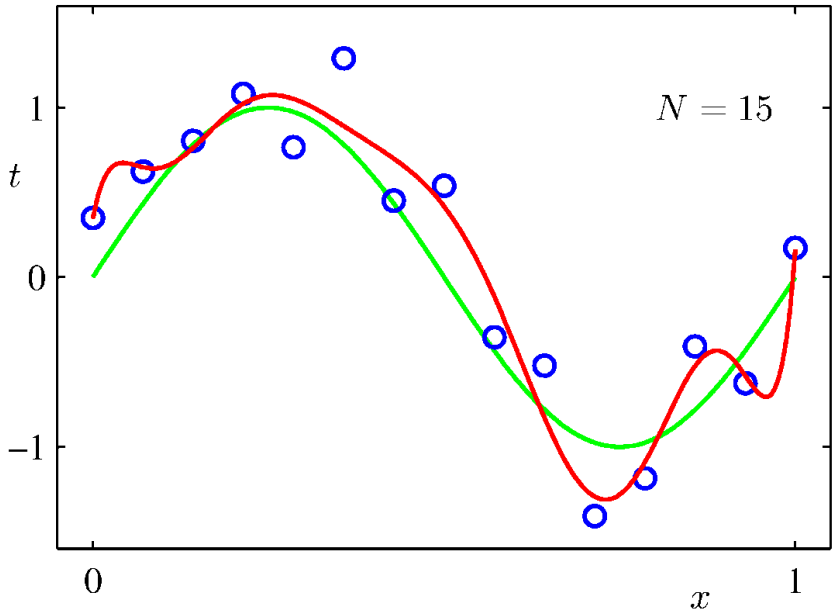


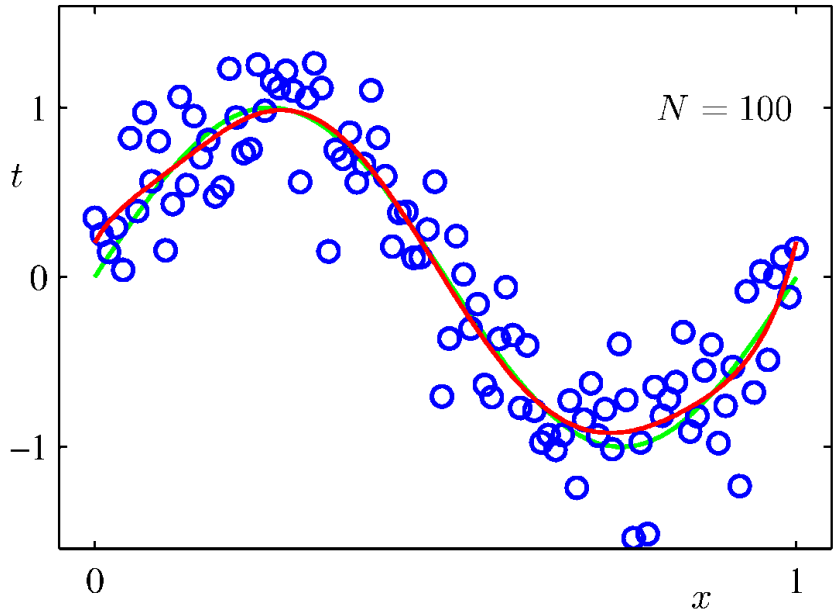


数据问题

- 使用更多的训练数据

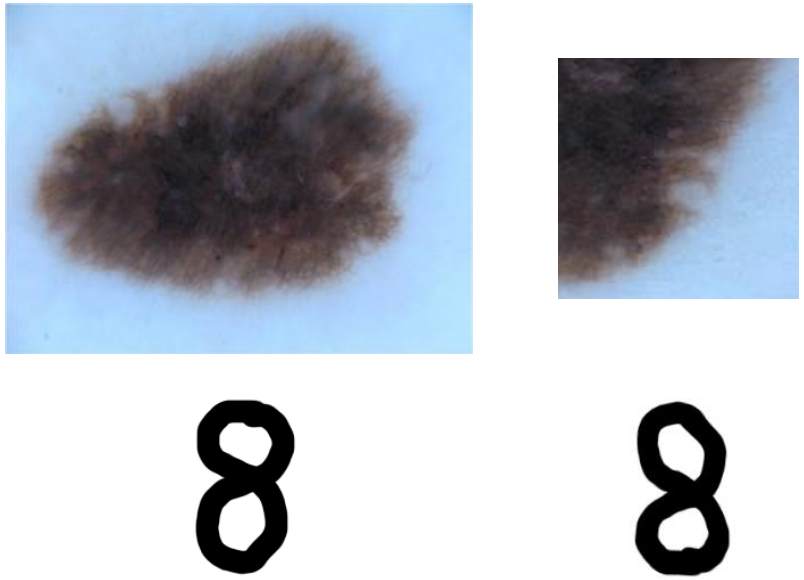


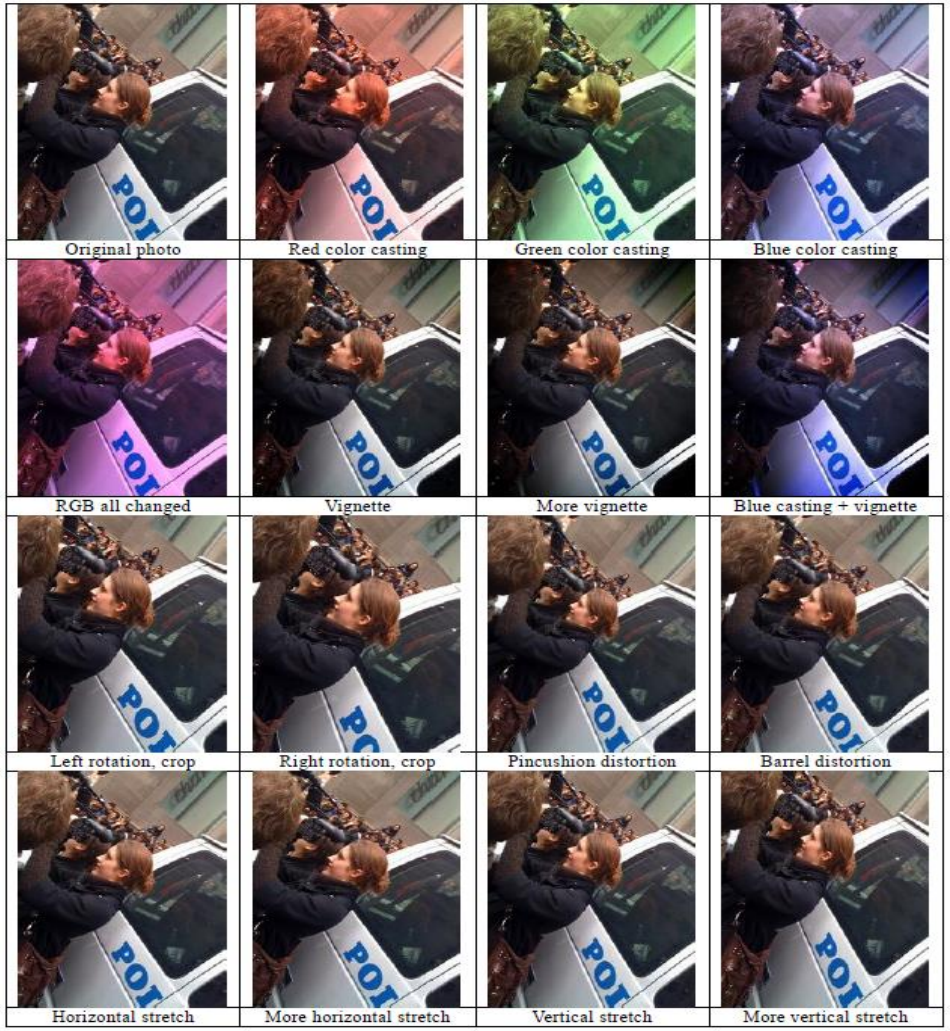




数据问题

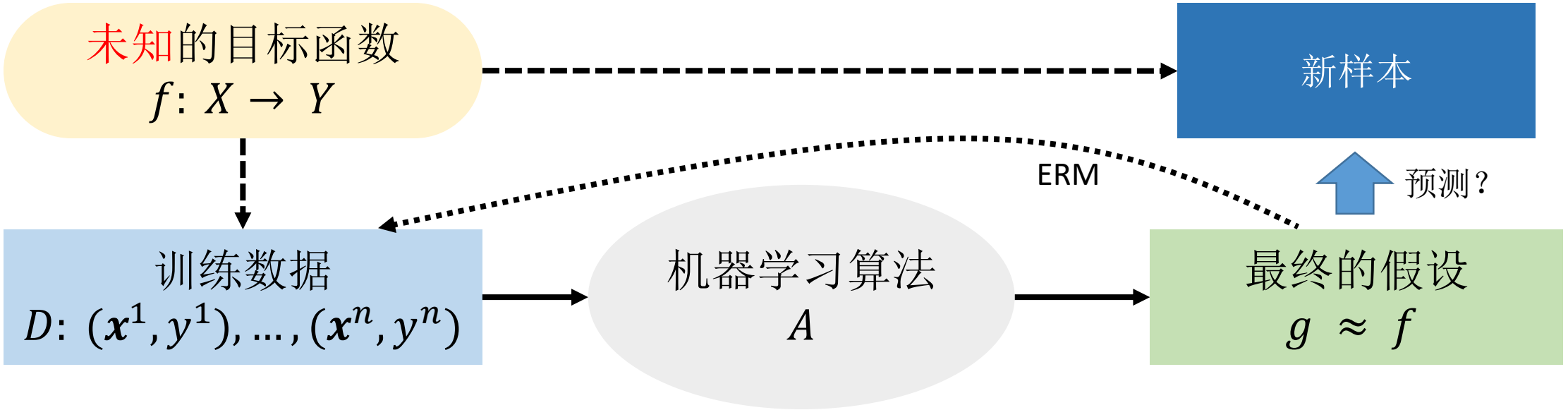
- 使用更多的训练数据
  - 获取新的数据不易
  - 标注代价非常高
- 数据增广



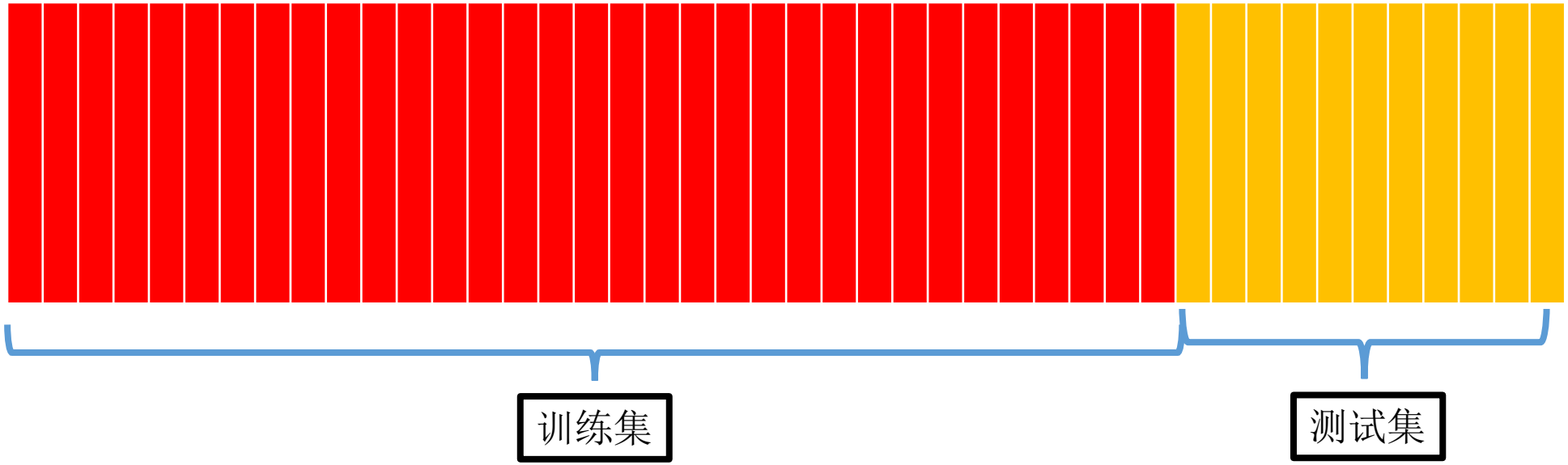


泛化

- 目标：针对从真实目标函数（分布）中产生的新数据，做出良好的预测
- 如果模型在拟合当前数据方面表现良好，那么我们如何相信该模型会对其它新样本做出良好预测呢？



拆分数据集



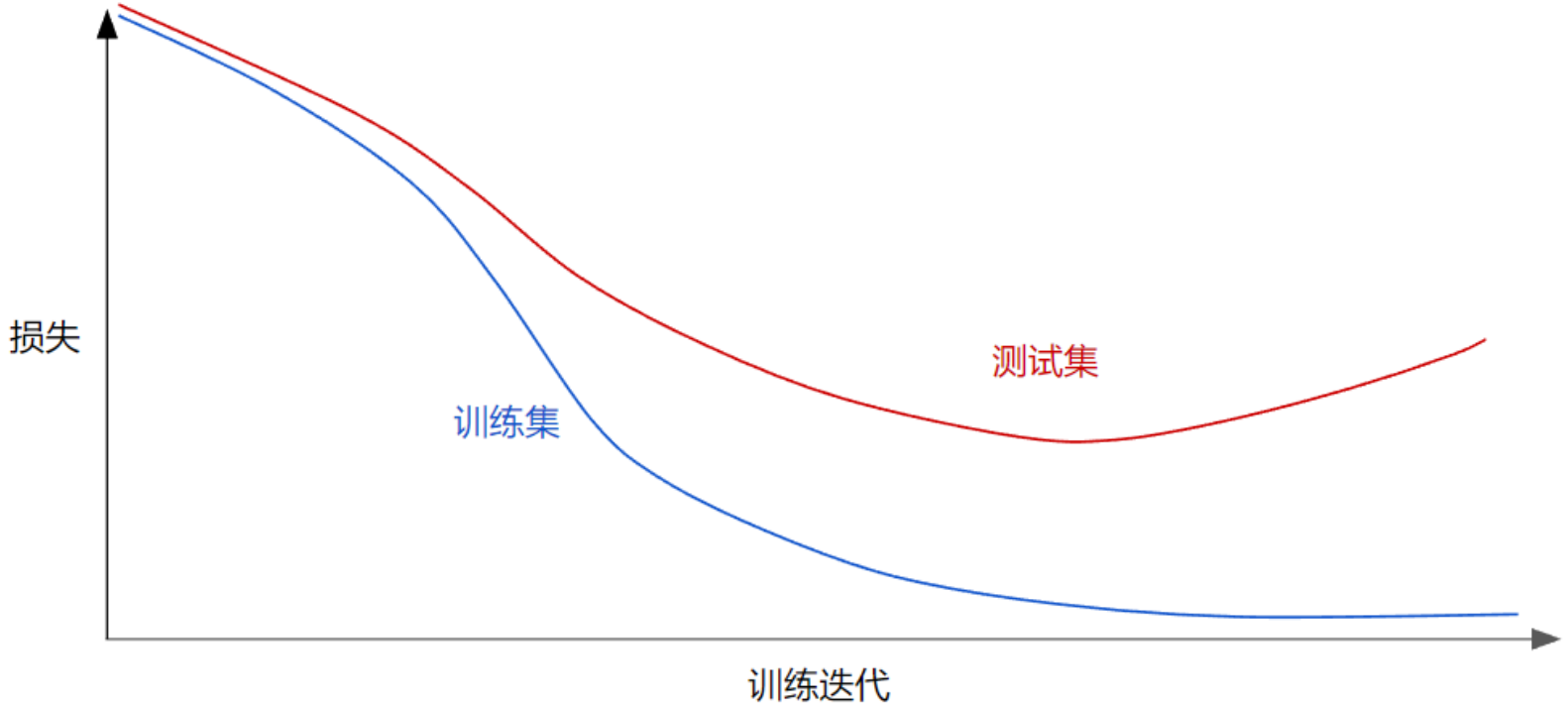
确保您的测试集满足以下两个条件：

- 规模足够大，可产生具有统计意义的结果。
- 能代表整个数据集。换言之，挑选的测试集的特征应该与训练集的特征相同。

典型陷阱：测试损失低得令人惊讶，可能意味着对测试数据进行了训练

评估测试集

- 利用测试数据集来计算代价函数
- 逻辑回归：将结果转换成分类准确率



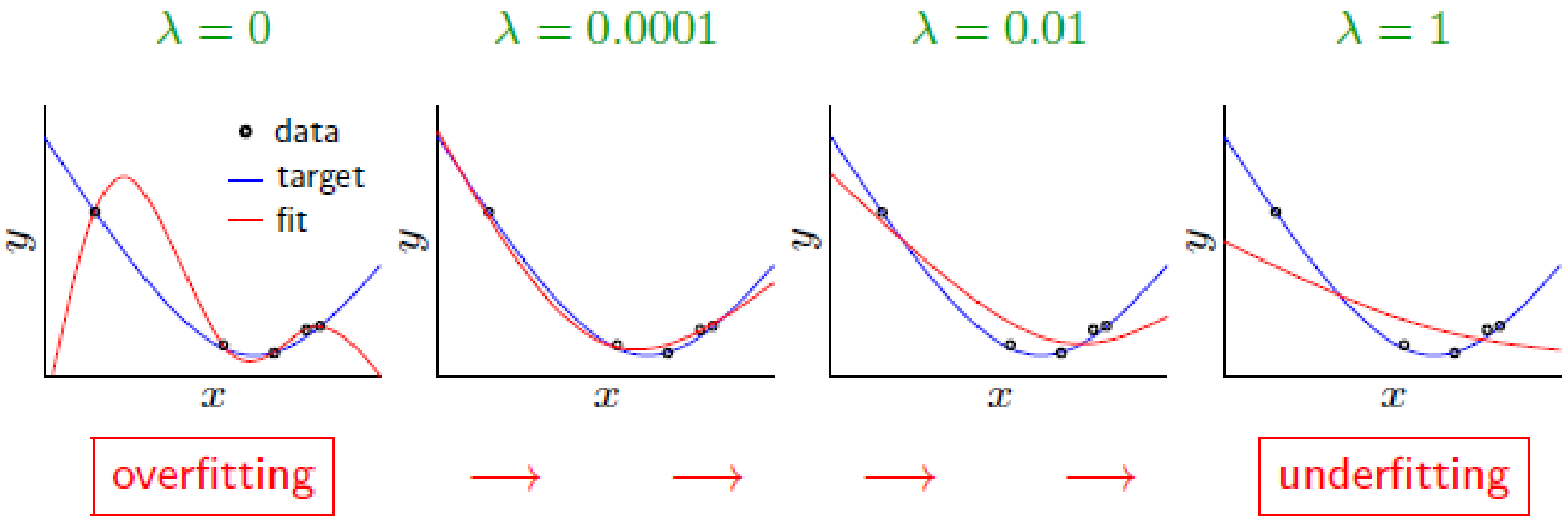
正则化

- 早停法
- 奥卡姆剃刀定律：“如无必要，勿增实体”，即“简单有效原理”
- 惩罚模型复杂度：以最小化损失和复杂度为目标
- 这称为**结构风险最小化 (SRM,** Structural Risk Minimization**)**

$$\arg\min_{\boldsymbol{w},b} \sum_{\mathrm{i}=1}^{n} l\left(g\left(\boldsymbol{x}^i;\boldsymbol{w}\right),y^i\right)+\lambda L(\boldsymbol{w})$$

- 简单化：L2 – 岭回归 Ridge Regression $L(\boldsymbol{w}) = \|\boldsymbol{w}\|_2^2$
- 稀疏化：L1 $L(\boldsymbol{w}) = \|\boldsymbol{w}\|_1$

岭回归 Ridge Regression



模型选择

- 假设我们要在10个不同次数的多项式模型之间进行选择：

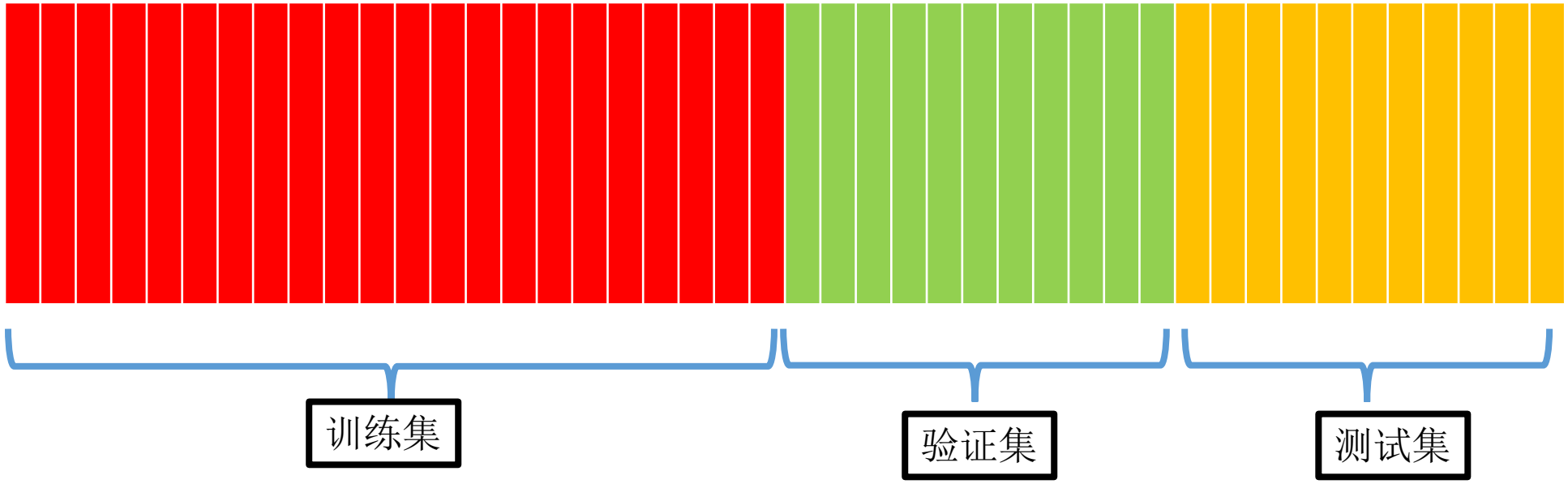
$$g(\boldsymbol{x}) = w_0 + w_1 x$$
$$g(\boldsymbol{x}) = w_0 + w_1 x + w_2 x^2$$
$$g(\boldsymbol{x}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$
$$\vdots$$
$$g(\boldsymbol{x}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_{10} x^{10}$$

- 模型选择标准：
  - Bayesian information criterion
  - False discovery rate
  - Stepwise regression
  - Cross-validation

超参数优化

- 超参数 hyperparameters
  - 正则化中的λ
  - 梯度下降中的η
- 超参数优化方法
  - Grid search
  - Bayesian Optimization
  - Random Search
  - Gradient Based Optimization
  - Cross-validation

训练、验证、测试



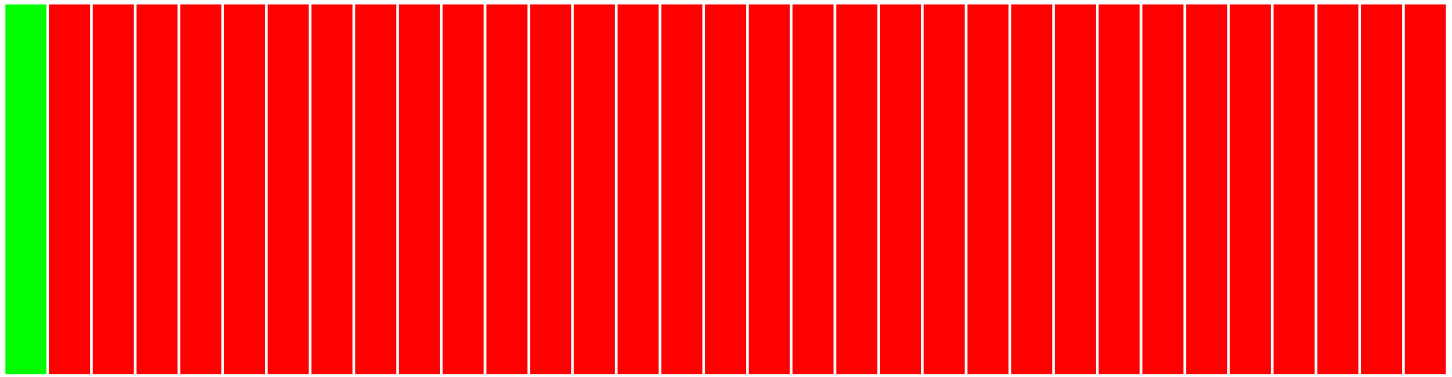
训练集：学习模型参数

验证集：选择模型/优化超参数

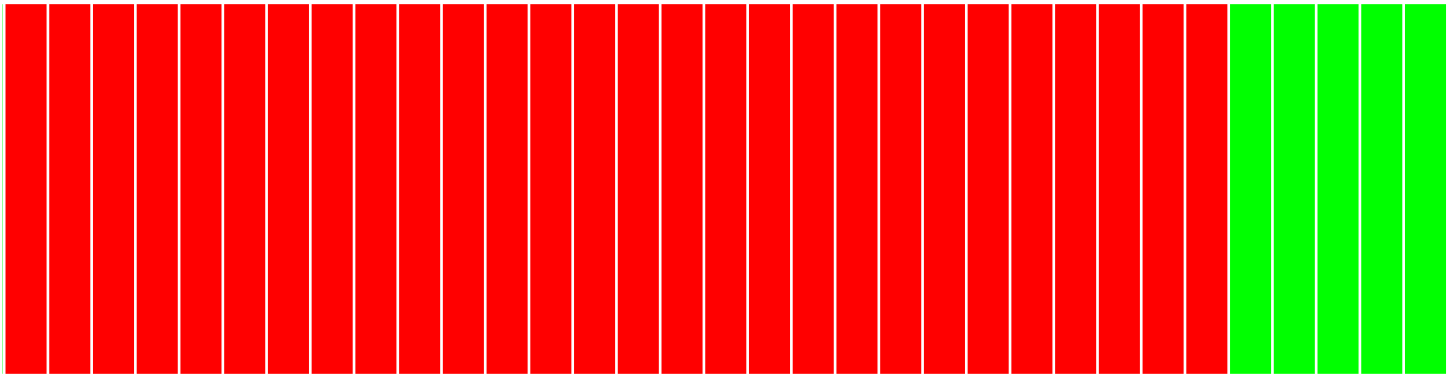
测试集：测试模型的泛化能力

交叉验证Cross-Validation

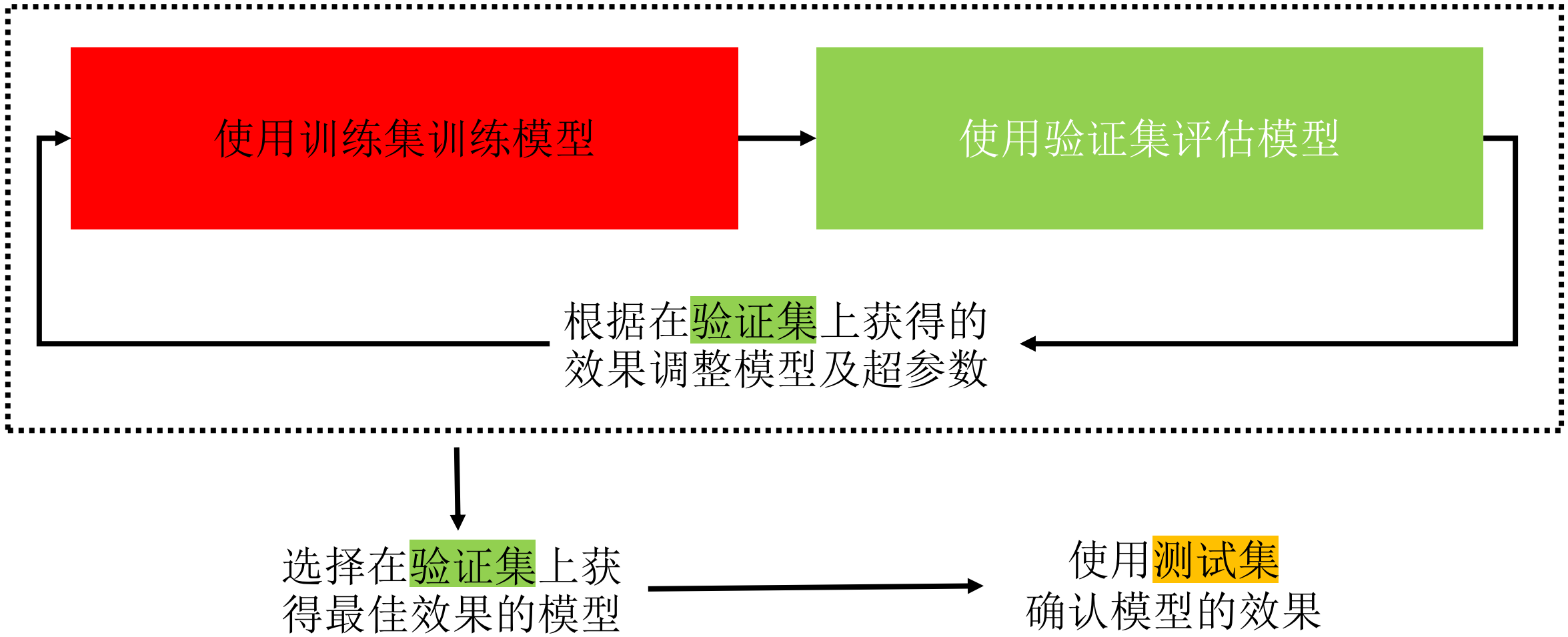
- 留一交叉验证(Leave-one-out CV, LOO-CV)



- K-fold交叉验证



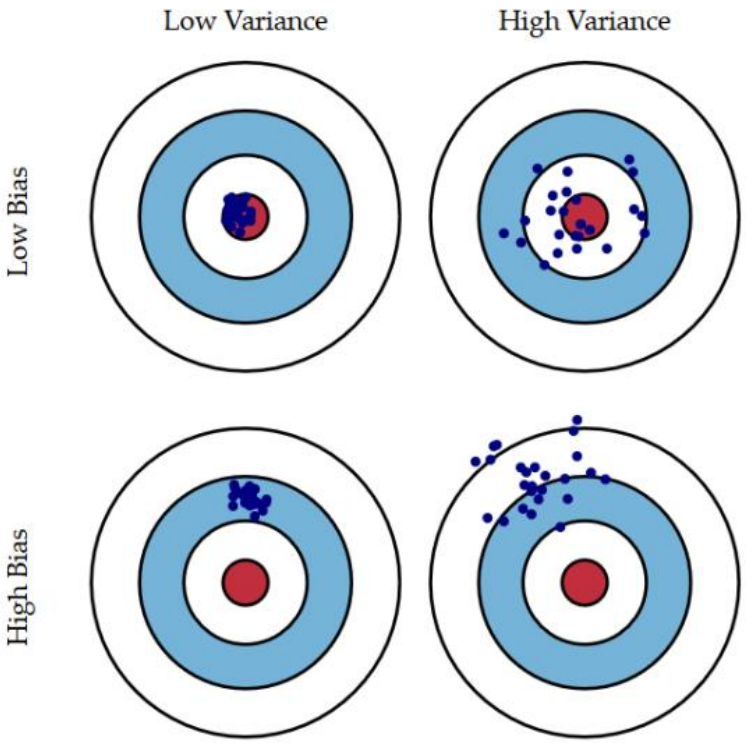
完整的机器学习工作流程



诊断偏差和方差

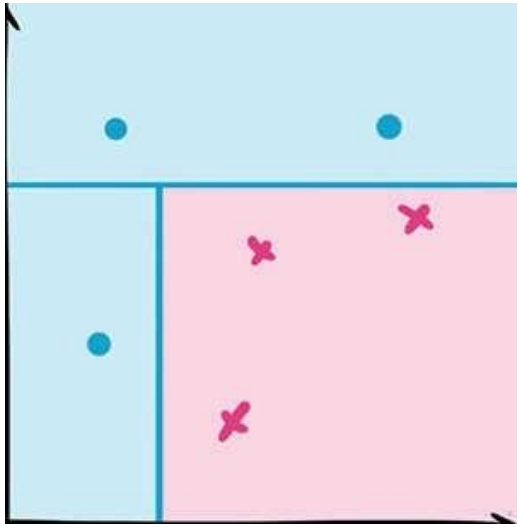
- **偏差（bias）**反映了模型无法描述数据规律
- **方差（variance）**反映了模型对训练集过度敏感，而丢失了数据规律

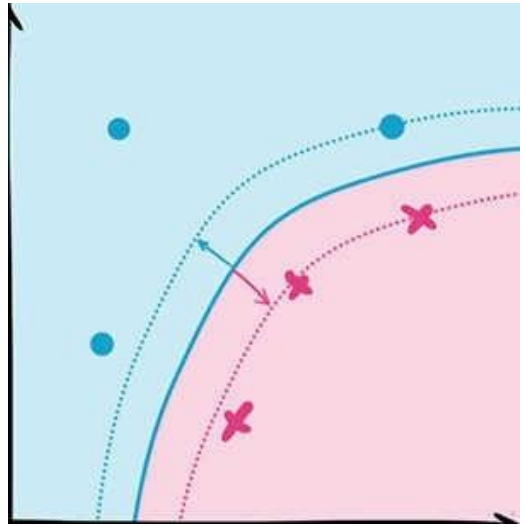
手段	使用场景
采集更多的样本	**高方差**
降低特征维度	**高方差**
采集更多的特征	**高偏差**
进行高次多项式回归	**高偏差**
降低参数 λ	**高方差**
增大参数 λ	**高偏差**

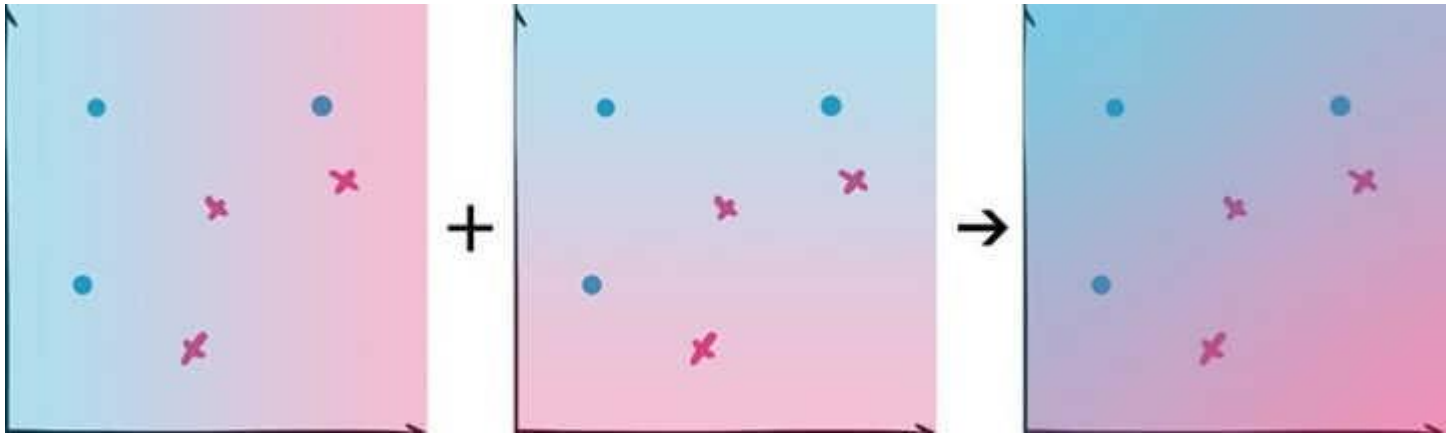


其它机器学习方法

- 决策树
- 支持向量机
- 贝叶斯
- 神经网络





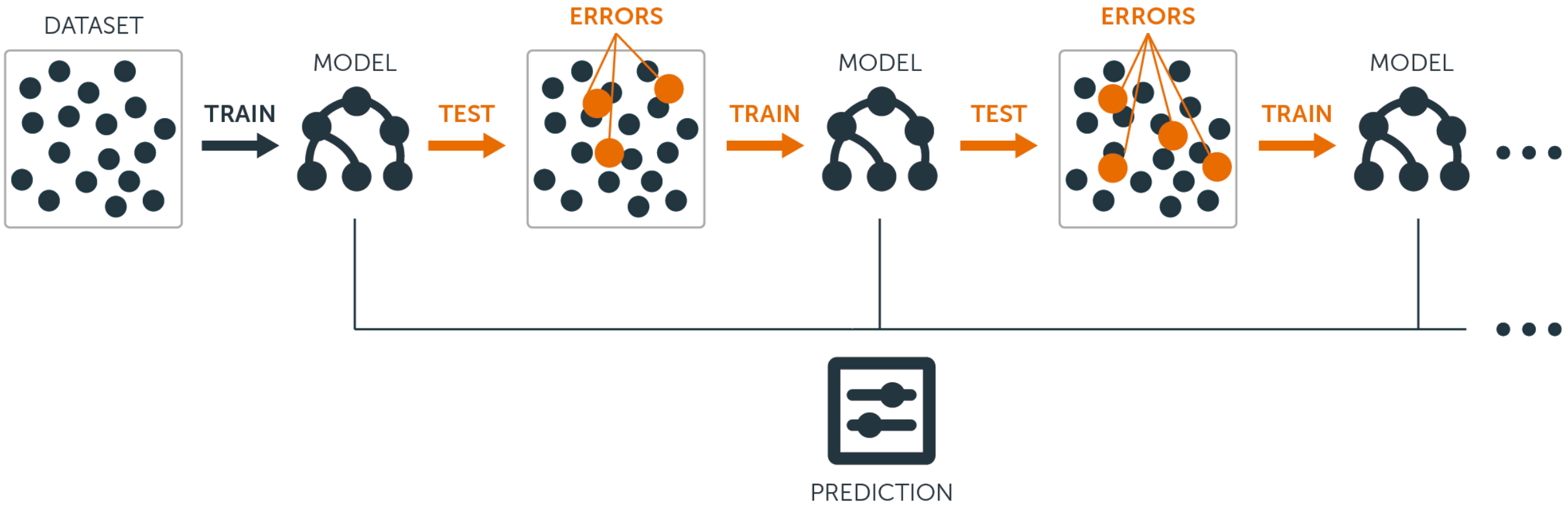


集成学习算法

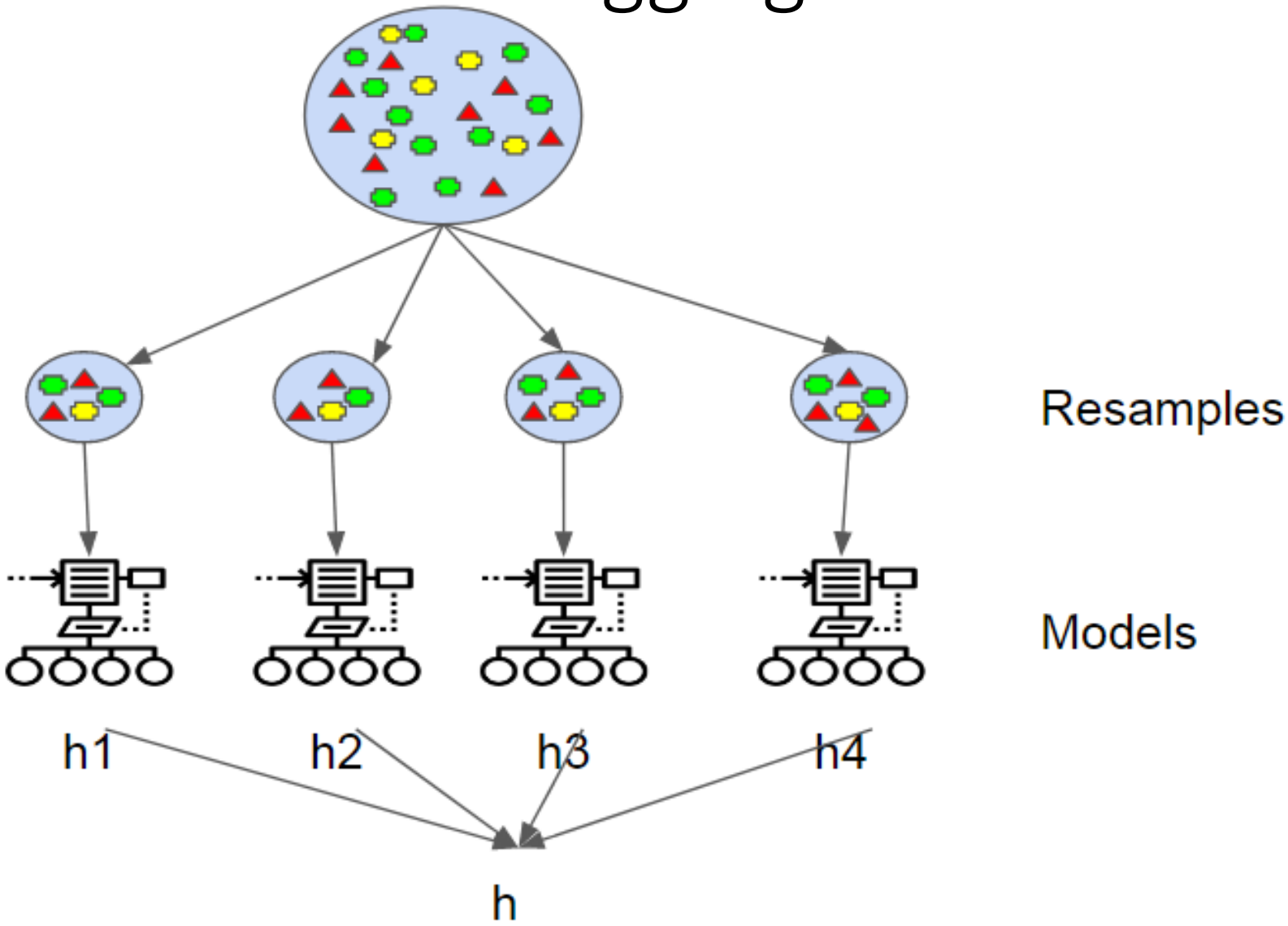
- 将多个分类器集成起来而形成的新的分类算法
  - 三个臭皮匠，顶个诸葛亮

- boosting：基于错误提升分类器性能，通过集中关注被已有分类器分类错误的样本，构建新分类器并集成
- bagging：基于数据随机重抽样的分类器构建方法
- stacking：在多个分类器的结果上，再套一个新的分类器

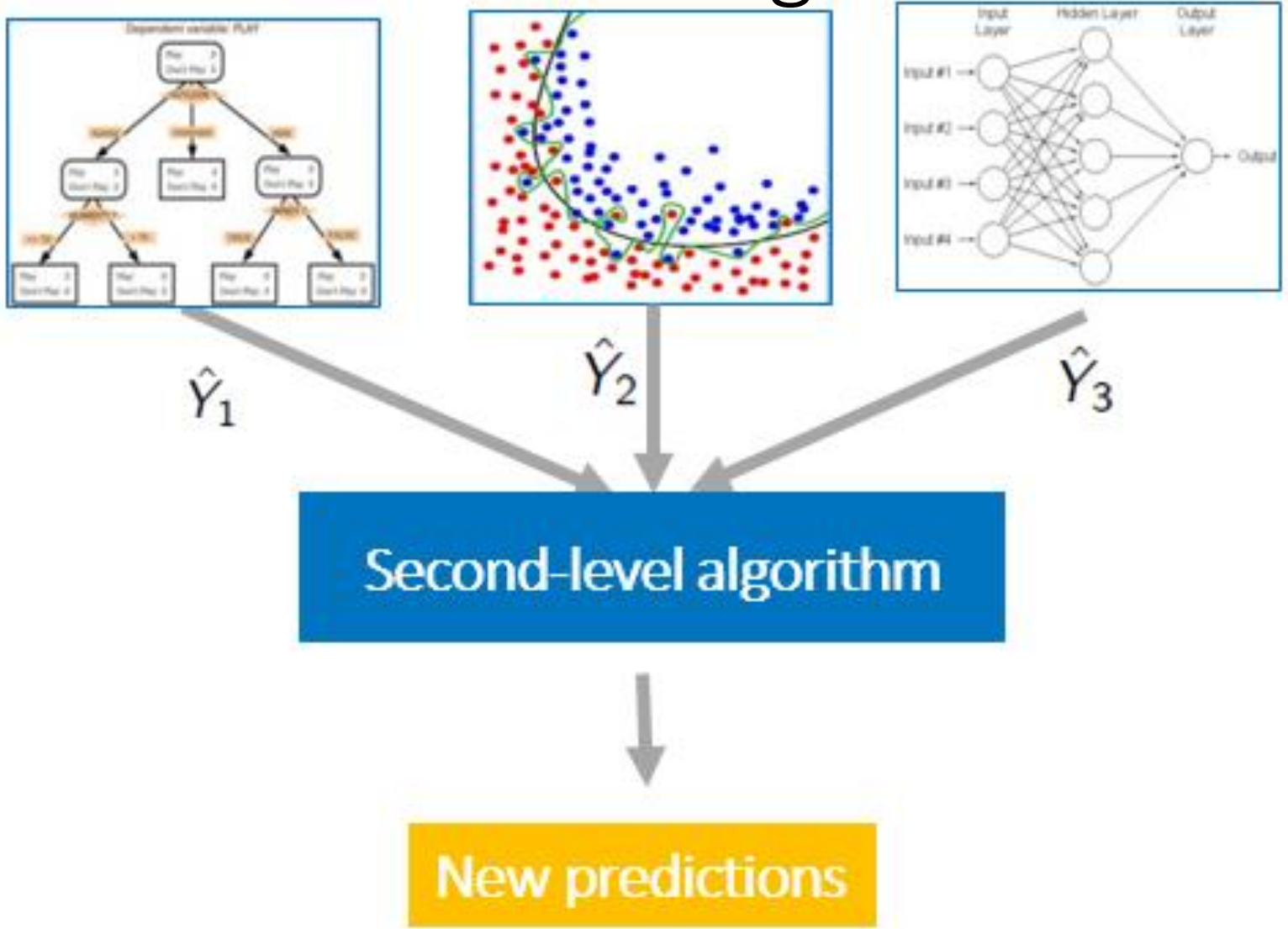
Boosting


DATASET
TRAIN
MODEL
TEST
ERRORS
TRAIN
MODEL
TEST
ERRORS
TRAIN
MODEL
PREDICTION

Bagging



Stacking


Input Layer
Hidden Layer
Output Layer
Input #1
Input #2
Input #3
Input #4
Output
$\hat{Y}_1$
$\hat{Y}_2$
$\hat{Y}_3$
Second-level algorithm
New predictions

总结

- 机器学习不是万能的
- 适用条件
- **机器学习的三个关键：**

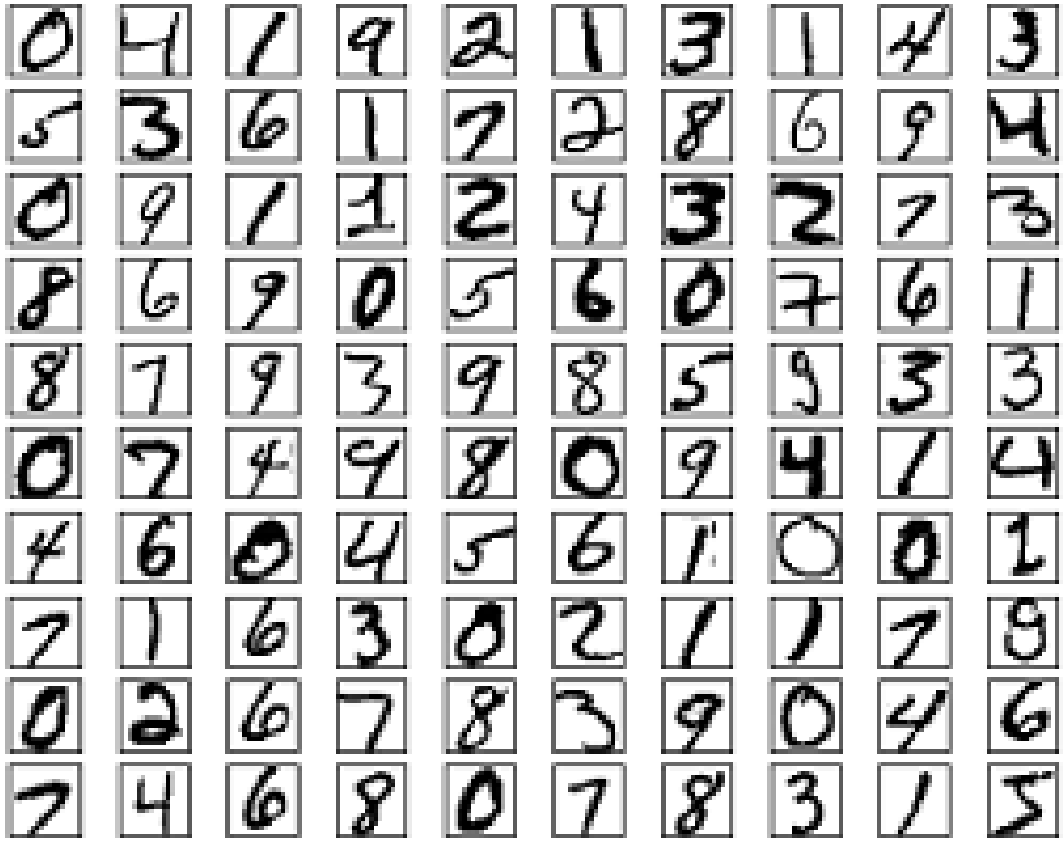
1. 存在某些潜在存在的模式供机器进行学习
2. 但我们无法把这些规则容易的写出
3. 需要有数据来学习这些模式

机器学习实战

MNIST数据集中0和1的手写字符分类

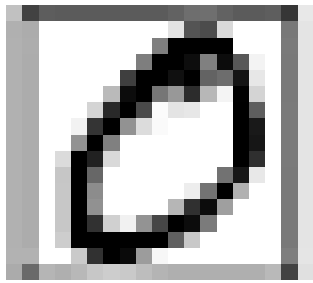
MNIST数据集

- MNIST数据集是一个手写体数据集
- 机器学习在视觉领域的hello world
- 每个手写字符图像是28x28的图像
- 总共0-9，10个类别



28x28

$\boldsymbol{x}$:



$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{784} \end{bmatrix}$$

$y \in \{0,1,2,3,4,5,6,7,8,9\}$

未知的目标函数
$f: X \rightarrow Y$

训练数据
$D: (\boldsymbol{x}^1, \mathrm{y}^1), \ldots, (\boldsymbol{x}^n, \mathrm{y}^n)$

机器学习算法
A

最终的假设
$g \approx f$

模型假说集合
H

数据下载

- 原始的数据下载地址：http://yann.lecun.com/exdb/mnist/
- **预处理好的**数据：mnist.pkl.gz
  - 准备成了pkl文件，可以非常简单的载入
  - 做了归一化处理
  - 相应的划分的训练集、验证集、测试集

逻辑回归实战

对MNIST数据集中的0、1两类手写字符图片，
使用逻辑回归的方式训练分类器

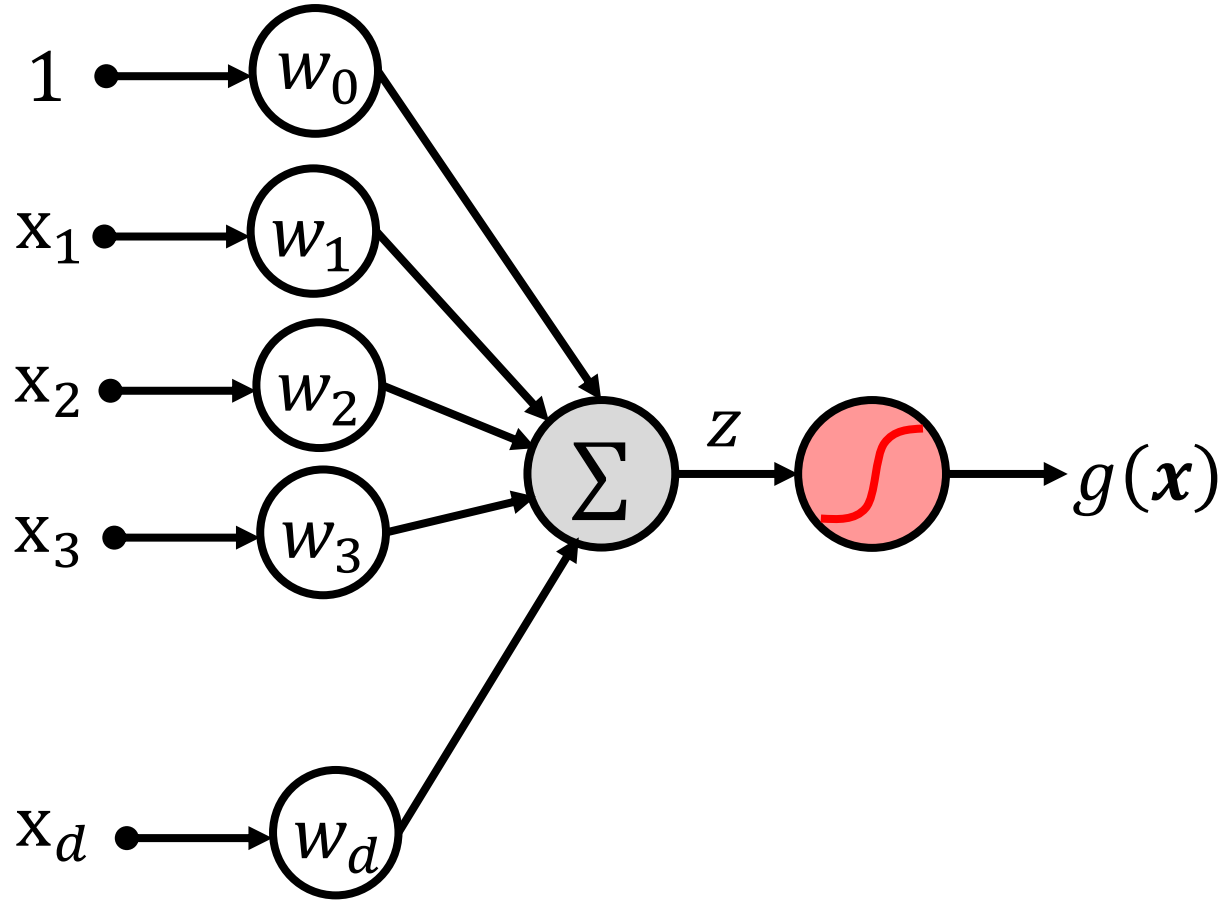
- 基本框架的mnist_logistic_regression.ipynb
- 实现梯度计算
- 实现随机梯度下降与批梯度下降
- 实现带正则化的岭回归
- 调节各种超参数，比较各自的效果

关于b的偏导？

$$\frac{\partial L}{\partial \boldsymbol{w}_0} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_0^i \left(g\left(\boldsymbol{x}^i\right) - \mathrm{y}^i\right)$$

其中$\boldsymbol{x}_0^i = 1$

$$\frac{\partial L}{\partial \boldsymbol{w}_0} = \frac{1}{n}\sum_{i=1}^{n} \left(g\left(\boldsymbol{x}^i\right) - \mathrm{y}^i\right)$$



SGD

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{1}{n} \boldsymbol{X}^T (g(\boldsymbol{X}) - \boldsymbol{y})$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \frac{1}{n} \boldsymbol{x^i}^T (g(\boldsymbol{x^i}) - y^i)$$

```
xi = X[i]
xi = X[[i]]
```