

# 第6章 机器学习

南京信息工程大学  
计算机学院

应龙  
2024年秋季

# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation maximization\*

- ✓ 周志华, “机器学习”, 2016, 清华大学出版社.
- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社.
- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社.
- ✓ 周志华等, “机器学习理论导引”, 2020, 机械工业出版社.

# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation maximization\*

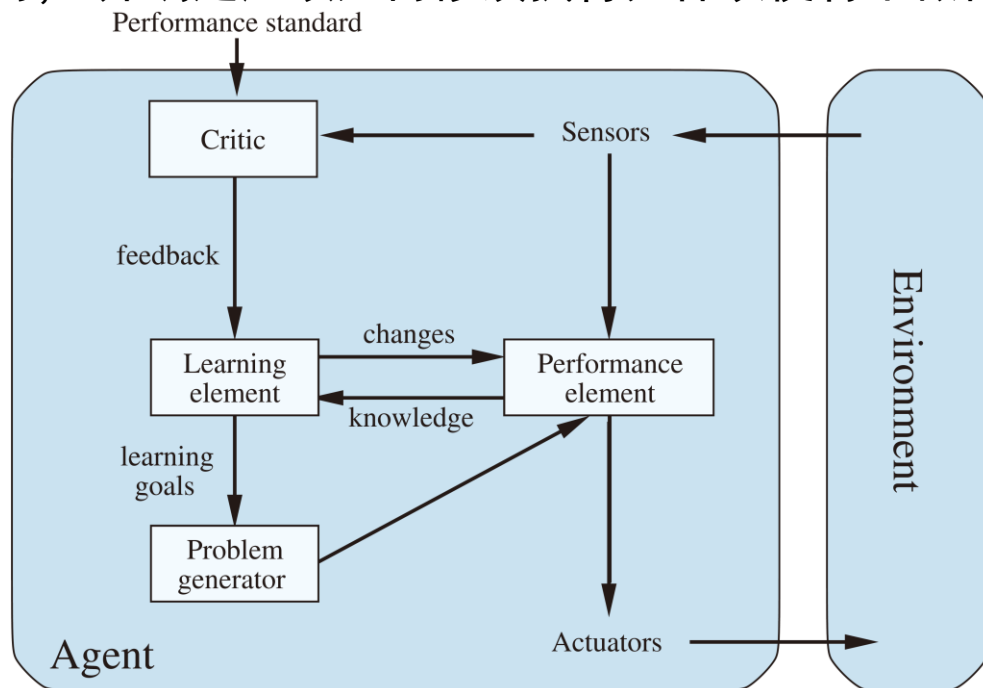
- ✓ 周志华, “机器学习”, 2016, 清华大学出版社. Ch 1
- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社. Ch 1
- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社.

# 智能体的结构

## ● Learning agents

已经给出了使用各种方法选择下一步行动的 Agent 程序。还没有说明这些 Agent 程序是如何形成的。

学习元件负责改进提高，而性能元件负责选择外部行动。**性能元件**是我们前面考虑的整个 Agent，它接受感知信息并决策。**学习元件**利用来自**评判元件**的反馈评价 Agent 做得如何，并确定应该如何修改执行元件以便将来做得更好。



学习元件的设计很大程度上依赖于性能元件的设计。对于给定的 Agent 设计，可以构造学习机制来改进 Agent 的各个部分。

评判元件根据固定的性能标准告诉学习元件 Agent 的运转情况。

学习 Agent 的最后一个组件是问题产生器。它负责建议探索性行动，得到新的和有信息的经验。

学习元件可以更改 Agent 结构图中的任何“知识”组件。最简单的情况包括直接从感知序列学习。观察环境的后继状态可以让 Agent 学到“世界是如何发展的”，而观察行动的结果可以让 Agent 学到“我的行动做了什么”。

# 机器学习的基本概念

## ◆ 从数据中学习知识



图像数据

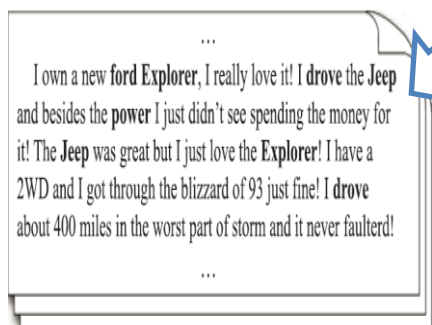
数字化图像

$f\left\{\begin{array}{cccc} 81 & 116 & \dots & 133 \\ 104 & 130 & \dots & 159 \\ \vdots & \vdots & \ddots & \vdots \\ 155 & 189 & \dots & 218 \\ 197 & 221 & \dots & 216 \end{array}\right\}$

- Person
- Dog
- ...

类别分类

- 样本 (sample) 示例 (instance)
- 特征 (feature)
- 样本空间 (sample space)
- 输入空间 (input space)
- 标记空间 (label space)  
输出空间 (output space)



文本数据

单词码

根据词典表示为 one-hot 向量

$f\{\text{car, money, drive, ...}\}$

- 喜悦
- 愤怒
- ...

情感分类

- 从原始数据中提取特征
- 学习映射函数  $f(\cdot)$
- 通过映射函数  $f(\cdot)$  将原始数据映射到语义任务空间, 即寻找数据和任务目标之间的关系



# 机器学习的基本概念

单词码

根据词典表示为  
one-hot 向量

I love this movie! It's sweet,  
but with satirical humor. The  
dialogue is great and the  
adventure scenes are fun...  
It manages to be whimsical  
and romantic while laughing  
at the conventions of the  
fairy tale genre. I would  
recommend it to just about  
anyone. I've seen it several  
times, and I'm always happy  
to see it again whenever I  
have a friend who hasn't  
seen it yet!

The Bag of Words (BOW) Representation



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

相似度计算

分类

聚类

原始数据

样本空间  
(sample space)

特征提取

特征空间  
(feature space)

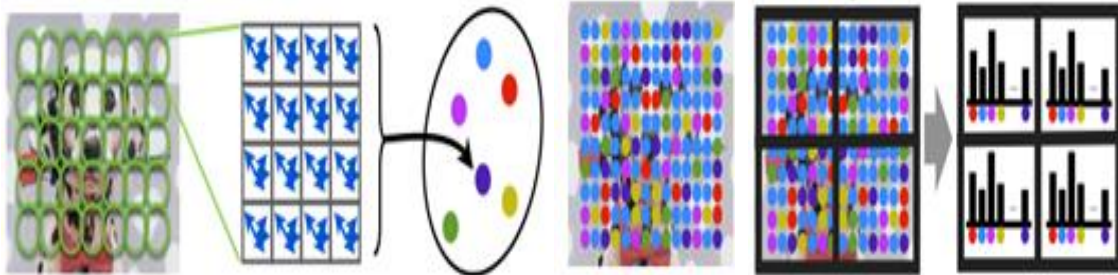
机器学习模型

数字化图像



图像

The Bag of Visual Words (BOW) Representation



稠密  
信息点

信息点  
表达

视觉  
词典

视觉  
单词

金字塔  
模型

直方图

相似度计算

分类

聚类

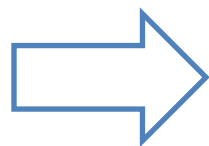
# 机器学习的基本概念

## ● 基于属性的分类模型

人员	数学好	身体好	会编程	嗓门大
程序员A	Yes	No	Yes	Yes
作家A	No	No	Yes	No
程序员B	Yes	Yes	No	No
...	...	...	...	...
医生A	Yes	Yes	Yes	Yes
程序员C	Yes	Yes	Yes	Yes
程序员D	Yes	Yes	Yes	No

标签数据

从数据  
中学习



$f$

映射函数

模式 pattern

(数学好 = Yes, 会编程 = Yes, 身体好 = ?, 嗓门大 = ?)

→ 程序员

类别

# 机器学习的基本概念

- 概念 (concept)      一种映射

概念是从样本空间  $\mathcal{X}$  到标记空间  $\mathcal{Y} \in \{-1, +1\}$  的映射, 它决定示例  $x$  的真实标记  $y$ .

- 目标概念: 如果对任何样例  $(x, y)$  均有  $c(x) = y$  成立, 则称  $c$  为目标概念.
- 概念类 (concept class): 所有我们希望学得的目标概念所构成的集合称为“概念类”, 用符号  $\mathcal{C}$  表示.

例如, 目标概念“三角形”将把所有的三角形映射为1, 把其它图形映射为-1. 如果在学习任务中考虑“三角形”“四边形”“五边形”这三种概念, 则它们组成的集合就是该任务的概念类.

- 假设空间 (hypothesis space)

给定学习算法  $\mathcal{L}$ , 它所考虑的所有可能概念的集合, 用符号  $\mathcal{H}$  表示.

- 由于学习算法事先并不知道概念类的真实存在, 因此  $\mathcal{H}$  和  $\mathcal{C}$  通常是不同的, 学习算法会把自认为可能的目标概念集中起来构成  $\mathcal{H}$ .
- 对于  $h \in \mathcal{H}$ , 由于并不能确定它是否真的是目标概念, 因此称为“假设” (hypothesis). 显然,  $h$  也是从样本空间  $\mathcal{X}$  到标记空间  $\mathcal{Y}$  的映射.
- 学习过程可以视为  $\mathcal{L}$  在  $\mathcal{H}$  中进行的搜索过程.



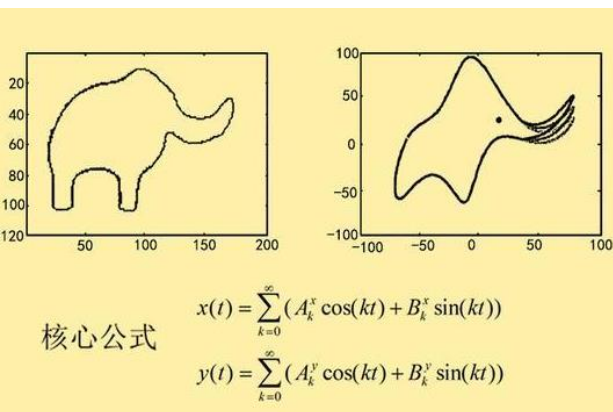
# 机器学习的基本概念

机器学习是一种“数据驱动学习 (data-driven learning)”的范式，从数据出发来学习数据中所蕴含的模式，对数据进行抽象。统计学家 Ronald Aylmer Fisher 将这一过程概括为“化繁为简 (the object of statistical methods is the reduction of data)”。

REQUIEM FOR LARGE-SCALE MODELS

Douglass B. Lee, Jr.

collapsed rather than evolved



核心公式

$$x(t) = \sum_{k=0}^{\infty} (A_k^x \cos(kt) + B_k^x \sin(kt))$$

$$y(t) = \sum_{k=0}^{\infty} (A_k^y \cos(kt) + B_k^y \sin(kt))$$

The task in this paper is to evaluate, in some detail, the fundamental flaws in attempts to construct and use large models and to examine the planning context in which the models, like dinosaurs, collapsed rather than evolved. The conclusions can be summarized in three points:

1. In general, none of the goals held out for large-scale models have been achieved, and there is little reason to expect anything different in the future.
2. For each objective offered as a reason for building a model, there is either a better way of achieving the objective (more information at less cost) or a better objective (a more socially useful question to ask).
3. Methods for long-range planning—whether they are called comprehensive planning, large-scale systems simulation, or something else—used to change drastically if planners expect to have any influence on the long run.

Almost a decade ago, John Reps presented a paper to planners in which he attacked traditional modes of land-use control and offered alternatives; his paper was titled “Requiem for Zoning.” This attack, directed at physical planners from one of their own, came at a time when many thought that mathematical models and computer data banks would overrun the field. This effort deserves a symmetrical gesture. (1964)

This paper is about large-scale urban models. The characteristics exhibited by these models are

(1) they are large in the sense that the only practical way to operate them is on a computer; (2) commonly they are spatially disaggregated, and allocate activities to geographic zones; and (3) they pertain to a single specific metropolitan area, as opposed to being generalized abstract or hypothetical models. The epitome of the genre is the comprehensive land-use model of the type constructed in the middle of the last decade.

These models were begun in the early 1960's and largely abandoned by the end of the 1960's. Considerable effort was expended on them, and a good deal was learned. Contrary to what has often been claimed, what was learned had almost nothing to do with urban spatial structure: the knowledge that was increased was our understanding of model building and its relationship to policy analysis. For that alone it was a valuable experience, but not if the lessons are ignored. For many in planning and many in a number of related fields that have recently become interested in planning, the lessons are being ignored.

Some planners never accepted models as legitimate activity of the field, and they will claim this paper vindicates their position.<sup>1</sup> This is incorrect: there was a need at that time for better analytic and quantitative procedures, and there was also a need for the development of better theory. Now, the need for both theory and method is even greater. It is not our intent to discourage those who would apply quantitative methods to urban problems, but, rather, to redirect their talents into more valuable pursuits than repeating the mistakes of the last decade.

A prototypical land-use model is broken down into subareas called zones or districts generally

<sup>1</sup>An example of this viewpoint is Raymond (in Fisher, 1970). A JOURNAL reviewer offered the Raymond article as a possible supporting reference; in fact, my position is diametrically opposed to Raymond's. Especially conscientious readers might care to compare the two articles.

16 Stimuleter/V1/3

摩尔定律: 晶体管数目2年翻倍  
黄氏定律: GPU性能逐年翻倍  
大模型参数: 每年接近10倍增长

费米与冯诺依曼

四个参数画大象之化繁为简  
预测稳定域和控制不稳定域

大模型的安魂曲

Requiem for large-scale models  
(Douglass B Lee, 1973)

数据涌现、算法模型、  
算力性能的增长

# 机器学习的基本概念

## ◆ 机器学习的分类

### 监督学习 (supervised learning)

数据有标签，一般为回归或分类等任务

### 无监督学习 (unsupervised learning)

数据无标签，一般为聚类、降维或概率密度估计等任务

### 强化学习 (reinforcement learning)

序列数据决策学习，一般为与从环境交互中学习

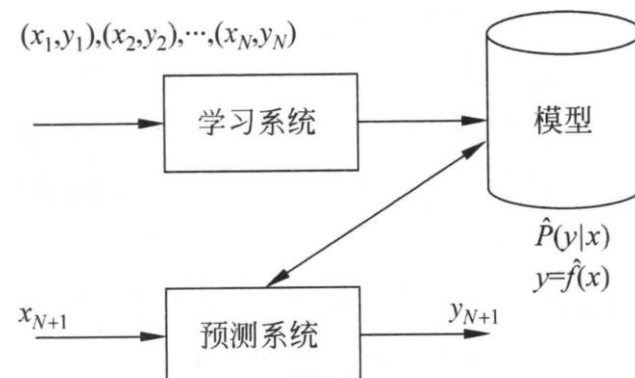


图 1.1 监督学习

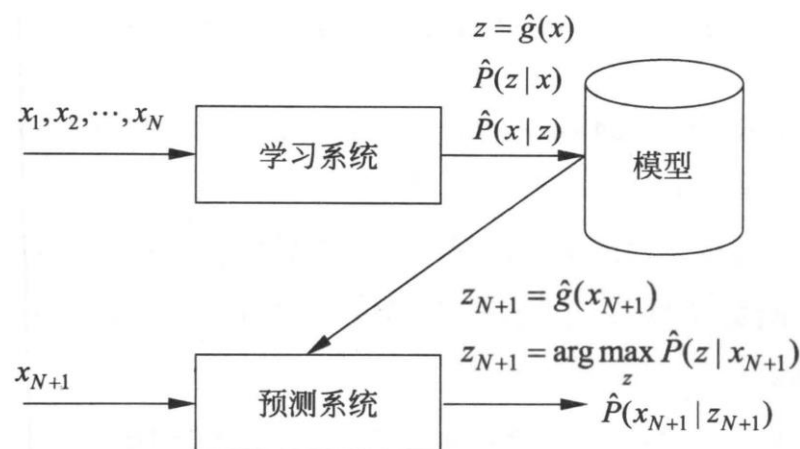


图 1.2 无监督学习

# 机器学习的基本概念

## ◆ k-means 聚类

- 输入：  $n$  个数据（无任何标注信息）
- 输出：  $k$  个聚类结果
- 目的：将  $n$  个数据聚类到  $k$  个集合（也称为类簇）

若干定义：

- $n$  个  $m$  维数据  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ ,  
 $\mathbf{x}_i \in R^m (1 \leq i \leq n)$
- 两个  $m$  维数据之间的欧氏距离为

$$d(\mathbf{x}_i, \mathbf{x}_j) =$$

$$\sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{im} - x_{jm})^2}$$

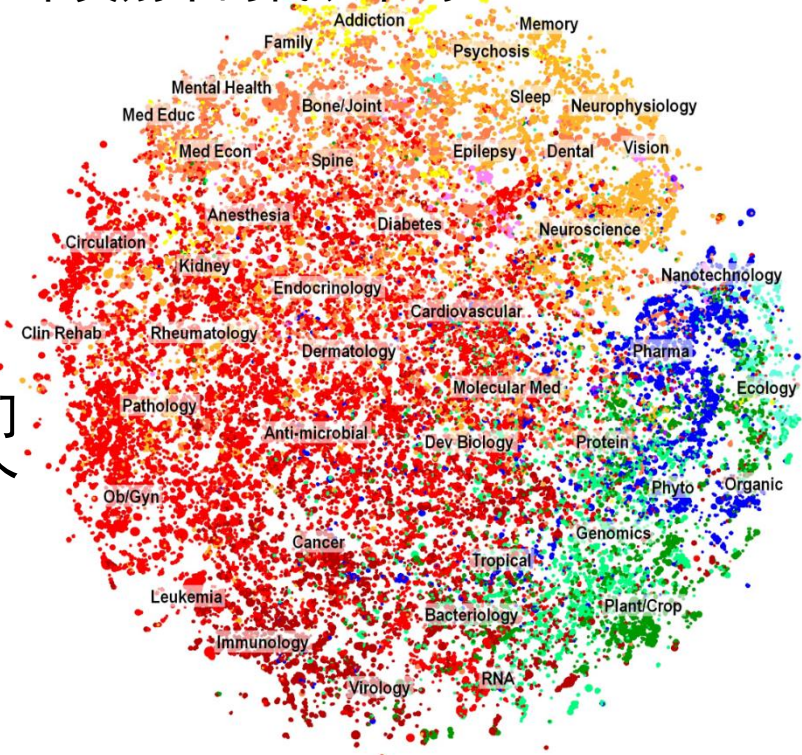
$d(\mathbf{x}_i, \mathbf{x}_j)$  值越小，表示  $\mathbf{x}_i$  和  $\mathbf{x}_j$  越相似；反之越不相似

- 聚类集合数目  $k$

问题：如何将  $n$  个数据依据其相似度大小将它们分别聚类到  $k$  个集合，使得每个数据仅属于一个聚类集合。

应用：图像压缩、文本聚类等

**文本聚类：**将200多万篇论文聚类到29,000个类别，包括化学、工程、生物、传染疾病、生物信息、脑科学、社会科学、计算机科学等及给出了每个类别中的代表单词





# 机器学习的基本概念

## ◆ k-means 聚类

### 第一步：初始化聚类质心

初始化 $k$ 个聚类质心  $c = \{c_1, c_2, \dots, c_k\}, c_j \in R^m (1 \leq j \leq k)$   
每个聚类质心 $c_j$ 所在集合记为 $G_j$

### 第二步：将每个待聚类数据放入一个聚类集合中

Expectation

计算待聚类数据 $x_i$ 和质心 $c_j$ 之间的欧氏距离 $d(x_i, c_j)$  ( $1 \leq i \leq n, 1 \leq j \leq k$ )  
将每个 $x_i$ 放入与之距离最近聚类质心所在聚类集合中, 即  $\operatorname{argmin}_{c_j \in C} d(x_i, c_j)$

### 第三步：根据聚类结果、更新聚类质心

Maximization

根据每个聚类集合中所包含的数据, 更新该聚类集合质心值:  $c_j = \frac{1}{|G_j|} \sum_{x_i \in G_j} x_i$

### 第四步：算法循环迭代, 直到满足条件

聚类迭代满足如下任意一个条件, 则聚类停止:

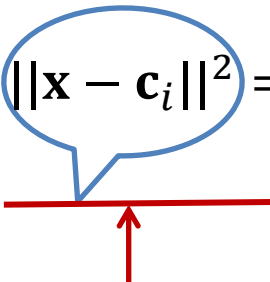
- ① 已经达到了迭代次数上限
- ② 前后两次迭代中, 聚类质心基本保持不变

# 机器学习的基本概念

## ● k-means 聚类的另一个视角

最小化每个类簇的方差

$$\arg \min_G \sum_{i=1}^k \sum_{\mathbf{x} \in G_i} \|\mathbf{x} - \mathbf{c}_i\|^2 = \arg \min_G \sum_{i=1}^k \sum_x \mathbb{I}(\mathbf{x} \in G_i) \cdot \|\mathbf{x} - \mathbf{c}_i\|^2$$



$$= \arg \min_G \sum_{i=1}^k |G_i| \text{Var} G_i$$

第  $i$  个类簇的方差:  $\text{var}(G_i) = \frac{1}{|G_i|} \sum_{\mathbf{x} \in G_i} \|\mathbf{x} - \mathbf{c}_i\|^2$

方差：用来计算样本分散程度的二阶统计量。

- 欧氏距离与方差量纲相同
- 最小化每个类簇方差将使得最终聚类结果中每个聚类集合中所包含数据呈现出来差异性最小。

## ● k-means 聚类算法的不足

- 需要事先确定聚类数目，很多时候我们并不知道数据应被聚类的数目
- 需要初始化聚类质心，初始化聚类中心对聚类结果有较大的影响
- 算法迭代执行，时间开销大
- 欧氏距离假设数据每个维度之间的重要性是一样的



# 机器学习的基本概念

## ● 超参数k的选择

常用方法: 拐点法（手肘法）和 轮廓系数法    定义新的任务

### ➤ 手肘法 (elbow method)

通过寻找损失值下降的拐点来确定k值

$$RMSSTD = \left\{ \frac{1}{\sum_i (n_i - 1)} \sum_i \sum_{x \in C_i} \|x - c_i\|^2 \right\}^{\frac{1}{2}} \quad \text{elbow}$$

### ➤ 轮廓系数法 (Average Silhouette Width)

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}, \text{ where} \quad a(i) = \frac{1}{n_{c(i)} - 1} \sum_{c(j)=c(i), j \neq i} d(\mathbf{x}_i, \mathbf{x}_j)$$
$$b(i) = \min_{k \neq c(i)} \frac{1}{n_k} \sum_{c(j)=k} d(\mathbf{x}_i, \mathbf{x}_j)$$

$$ASW(C_K) = \frac{1}{n} \sum_{i=1}^n s(i), \quad K_{ASW} = \arg \max_{k=2, \dots, k_{max}} ASW(C_K)$$

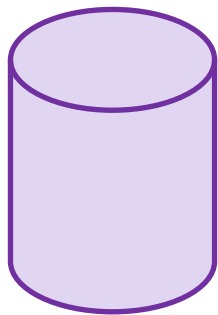
- ✓ Liu, Yanchi, Zhongmou Li, Hui Xiong, Xuedong Gao and J. Wu. "Understanding of Internal Clustering Validation Measures." *2010 IEEE International Conference on Data Mining* (2010): 911-916.

# 机器学习的基本概念

## ● 损失函数 (loss function) 监督学习

分类, 识别, 推荐, ...

训练映射函数  $f$   
使得  $f(x_i)$  预测结果尽量  
等于  $y_i$



Training dataset  
 $(x_i, y_i), i = 1, \dots, n$

- 训练集中一共有  $n$  个标注数据, 第  $i$  个标注数据记为  $(x_i, y_i)$ , 其中第  $i$  个样本数据为  $x_i$ ,  $y_i$  是  $x_i$  的标注信息。
- 从训练数据中学习得到的映射函数记为  $f$ ,  $f$  对  $x_i$  的预测结果记为  $f(x_i)$ 。损失函数就是用来计算  $x_i$  真实值  $y_i$  与预测值  $f(x_i)$  之间差值的函数。
- 很显然, 在训练过程中希望映射函数在训练数据集上得到“损失”之和最小, 即  $\min \sum_{i=1}^n \text{Loss}(f(x_i), y_i)$ 。

损失函数名称	损失函数定义
0-1损失函数	$\text{Loss}(y_i, f(x_i)) = \begin{cases} 1, & f(x_i) \neq y_i \\ 0, & f(x_i) = y_i \end{cases}$
平方损失函数	$\text{Loss}(y_i, f(x_i)) = (y_i - f(x_i))^2$
绝对损失函数	$\text{Loss}(y_i, f(x_i)) =  y_i - f(x_i) $
对数损失函数/对数似然损失函数	$\text{Loss}(y_i, P(y_i x_i)) = -\log(P(y_i x_i))$

# 机器学习的基本概念

## ◆ 机器学习的重要元素

算法 (algorithm)

模型 (model)

策略 (strategy)

监督学习

标注数据

学习模型

损失函数

- 标识了类别信息的数据  
学什么

- 如何学习得到映射模型  
如何学

- 如何对学习结果进行评估  
学到否

策略：问题的目标函数，问题形式化描述的范式

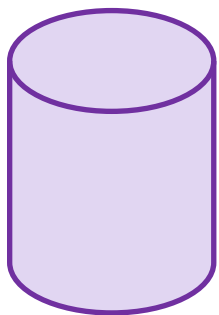
模型：假设空间（函数空间） $\mathcal{H}$  的形式化（参数化函数族）

算法：求解最优假设  $h$  的方法

# 机器学习的基本概念

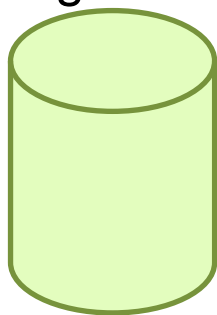
## ● 训练数据集、验证数据集和测试数据集

从训练数据集  
(training dataset)  
学习得到映射函数  
 $f$



Training dataset  
 $(\mathbf{x}_i, y_i), i = 1, \dots, n$

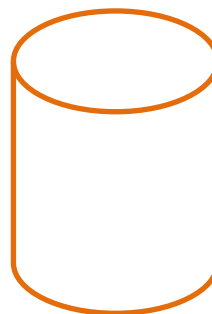
使用验证数据集  
(validation dataset)  
调整超参数和模型结  
构, 进行 early  
stopping



Validation dataset  
 $(\mathbf{x}_i^*, y_i^*), i = 1, \dots, k$

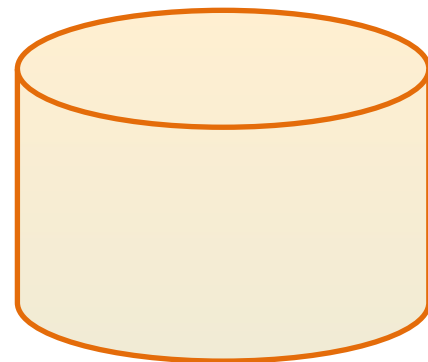
广义的训练数据集

在测试数据集  
(test dataset)  
测试映射函数  $f$



Test dataset  
 $(\mathbf{x}_i', y_i'), i = 1, \dots, m$

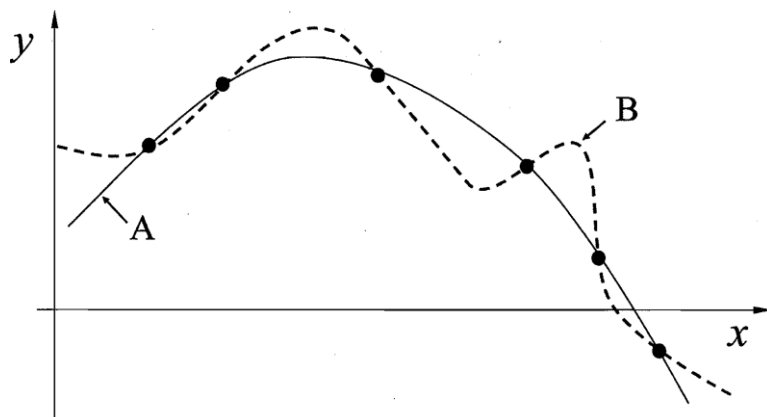
在未知数据集  
(test dataset) 上  
使用映射函数  $f$



# 机器学习的基本概念

## ● 归纳偏好 (inductive bias):

机器学习算法在学习过程中对某种类型假设的偏好。



A更好?  
B更好?

一般原则:  
奥卡姆剃刀  
(Ocam's razor)

任何一个有效的机器学习算法必有其归纳偏好，否则它将被假设空间中看似在训练集上“等效”的假设所迷惑，而无法产生确定的学习结果。

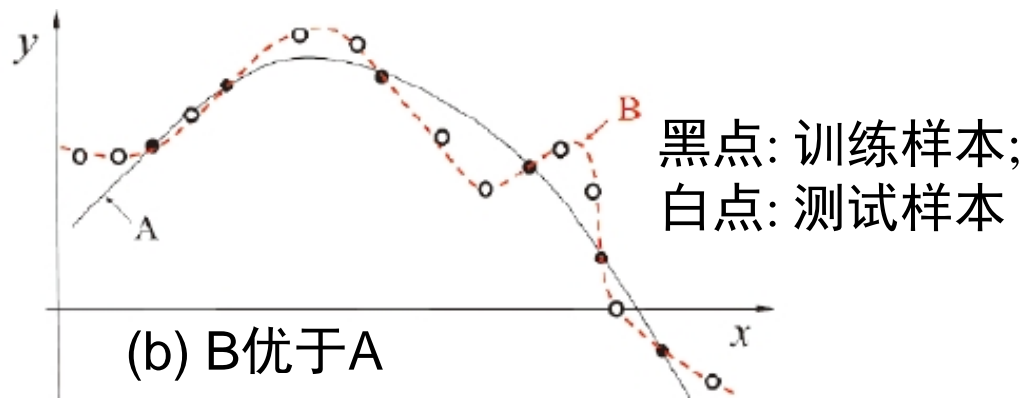
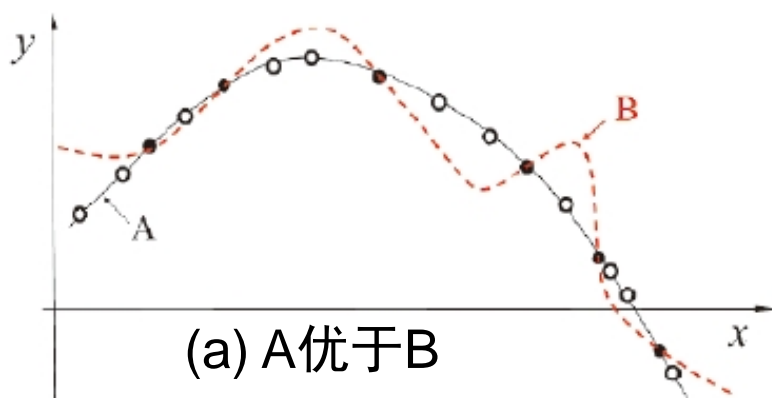
归纳偏好可看作学习算法自身在一个可能很庞大的假设空间中对假设进行选择启发式或“价值观”。

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！



# 机器学习的基本概念

- 没有免费午餐定理 (NFL): Wolpert 和 Macerday 1997年在最优化理论中提出, 指出“任何机器学习模型在**所有问题**上的性能都是相同的, 其总误差和模型本身没有关系。一种学习算法 ( $\mathcal{L}_A$ ) 在某些问题上的表现优于另一种算法 ( $\mathcal{L}_B$ ) 的同时, 一定伴随着  $\mathcal{L}_A$  在另外某一些问题上有着不如  $\mathcal{L}_B$  的表现。”



简短论述: 训练集之外所有样本上的误差

$$E_{ote}(\mathcal{L}_a|X, c) = \sum_h \sum_{\mathbf{x} \in \mathcal{X} - X} P(\mathbf{x}) \mathbb{I}(h(\mathbf{x}) \neq c(\mathbf{x})) P(h|X, \mathcal{L}_a)$$

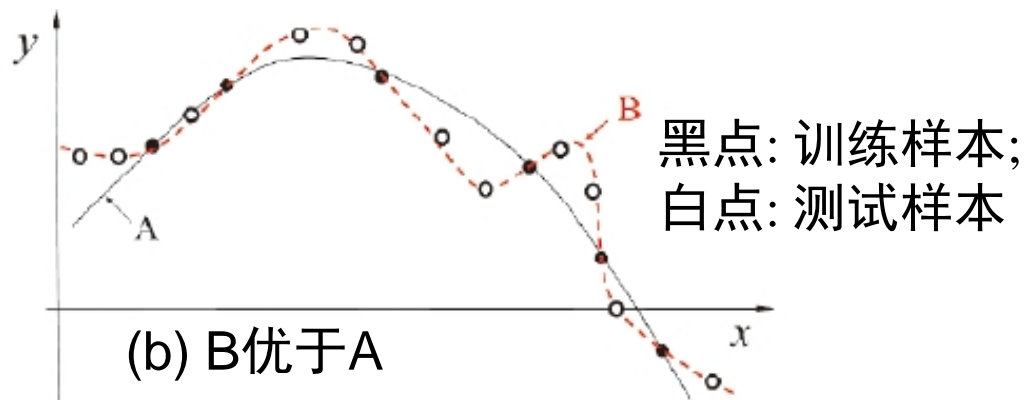
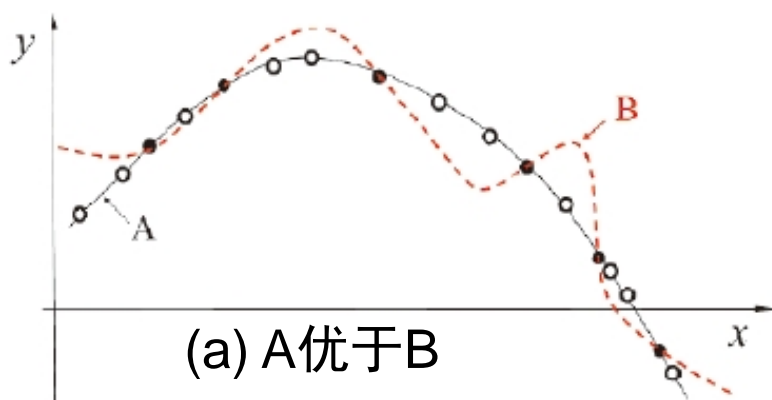
考虑二分类问题, 真实的目标函数可以是任何函数  $\mathcal{X} \mapsto \{0,1\}$ . 对所有可能的  $c$  按均匀分布对误差求和: **最一般的情况**

$$\begin{aligned} \sum_c E_{ote}(\mathcal{L}_a|X, c) &= \sum_{\mathbf{x} \in \mathcal{X} - X} P(\mathbf{x}) \sum_h P(h|X, \mathcal{L}_a) \sum_c \mathbb{I}(h(\mathbf{x}) \neq c(\mathbf{x})) \\ &= \sum_{\mathbf{x} \in \mathcal{X} - X} P(\mathbf{x}) \sum_h P(h|X, \mathcal{L}_a) \frac{1}{2} 2^{|\mathcal{X}|} = 2^{|\mathcal{X}|-1} \sum_{\mathbf{x} \in \mathcal{X} - X} P(\mathbf{x}) \cdot 1 \end{aligned}$$

- ✓ Wolpert, D.H., Macready, W.G. (1997), No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 1, 67

# 机器学习的基本概念

- 没有免费午餐定理 (NFL): Wolpert 和 Macerday 1997年在最优化理论中提出, 指出“任何机器学习模型在**所有问题**上的性能都是相同的, 其总误差和模型本身没有关系。一种学习算法 ( $\Omega_A$ ) 在某些问题上的表现优于另一种算法 ( $\Omega_B$ ) 的同时, 一定伴随着  $\Omega_A$  在另外某一些问题上有不如  $\Omega_B$  的表现。”



纯经验主义视角

**NFL定理的重要前提:** 所有“问题”出现的机会相同, 或所有问题同等重要。脱离具体问题, 空泛地谈论“什么学习算法更好”没有意义! **最一般的情况** 上面的论述中假设所有  $c$  的分布为均匀分布, 没有加入统计学样本估计的概率约束这一先验, 而实际情况并非如此。

- ✓ Wolpert, D.H., Macready, W.G. (1997), No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 1, 67

# 机器学习的基本概念

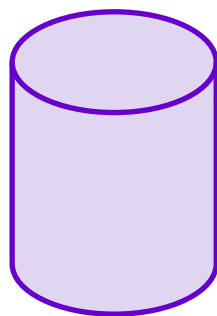
## ● 经验风险与期望风险 监督学习

从训练数据集  
(training dataset)  
学习得到映射函数  $f$

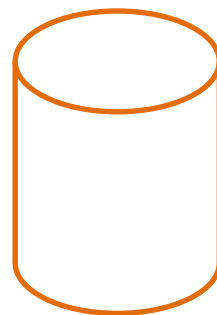
在测试数据集  
(test dataset)  
测试映射函数  $f$

经验风险 (empirical risk):

训练集中数据产生的损失。经验风险越小说明学习模型对训练数据拟合程度越好



Training dataset  
 $(x_i, y_i), i = 1, \dots, n$



Test dataset  
 $(x_i', y_i'), i = 1, \dots, m$

期望风险 (expected risk):

当测试集中存在无穷多数据时产生的损失。期望风险越小，学习所得模型越好。

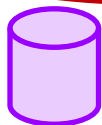
# 机器学习的基本概念

## ● 经验风险与期望风险 监督学习

映射函数训练目标：经验风险最小化  
(empirical risk minimization, ERM)

$$\min_{f \in \Phi} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f(x_i))$$

选取一个使得训练集  
所有数据损失平均值  
最小的映射函数。这  
样的考虑是否够？



训练数据集  
 $(x_i, y_i), i = 1, \dots, n$

映射函数训练目标：期望风险最小化  
(expected risk minimization)

$$\min_{f \in \Phi} \int_{x \times y} \text{Loss}(y, f(x)) P(x, y) dx dy$$



测试数据集数据无穷多  
 $(x_i', y_i'), i = 1, \dots, \infty$

- 期望风险是模型关于联合分布期望损失，经验风险是模型关于训练样本集平均损失。
- 根据大数定律，当样本容量趋于无穷时，经验风险趋于期望风险。所以在实践中很自然用经验风险来估计期望风险。
- 由于现实中训练样本数目有限，用经验风险估计期望风险并不理想，要对经验风险进行一定的约束。

# 机器学习的基本概念

## ● 过拟合 (over-fitting) 与 欠拟合 (under-fitting) 监督学习

经验风险最小化

$$\min_{f \in \Phi} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f(x_i))$$

期望风险最小化

$$\min_{f \in \Phi} \int_{x \times y} \text{Loss}(y, f(x)) P(x, y) dx dy$$

经验风险小（训练集上表现好）	期望风险小（测试集上表现好）	泛化能力强
经验风险小（训练集上表现好）	期望风险大（测试集上表现不好）	过学习（模型过于复杂）
经验风险大（训练集上表现不好）	期望风险大（测试集上表现不好）	欠学习
经验风险大（训练集上表现不好）	期望风险小（测试集上表现好）	“神仙算法”或“黄粱美梦”

表4.3 模型泛化能力与经验风险、期望风险的关系



# 机器学习的基本概念

- 结构风险最小化 (structural risk minimization)

经验风险最小化: 仅反映了局部数据

$$\min_{f \in \Phi} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f(x_i))$$

期望风险最小化: 无法得到全量数据

$$\min_{f \in \Phi} \int_{x \times y} \text{Loss}(y, f(x)) P(x, y) dx dy$$

结构风险最小化 (structural risk minimization): 为了防止过拟合, 在经验风险上加上表示模型复杂度的正则化项 (regularizer) 或惩罚项 (penalty term) :

$$\min_{f \in \Phi} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f(x_i)) + \lambda r(f)$$

经验风险                      模型复杂度

在最小化经验风险与降低模型复杂度之间寻找平衡

# 机器学习的基本概念

- 信息瓶颈理论 (information bottleneck theory)

The core of information bottleneck (IB) is to find a compressed representation of input data that captures the information most relevant to predicting a target variable. In a neural network, for example, this representation should ideally retain only what's necessary for the output prediction, discarding irrelevant data.

$$\max[I(Z; Y) - \beta I(X; Z)]$$

where  $X$  represents the original input to be compressed,  $Y$  represents the objective task,  $Z$  is the compressed representations for  $X$ , and  $I(;)$  stands for mutual information.

2023年2月OpenAI研究员 Jack Rae在斯坦福的报告“Compression For AGI”[3]，2023年8月Ilya在伯克利的报告“An observation on Generalization”[4]。报告的主要意思可以概括为“预测即压缩，压缩即泛化，泛化即智能”[5]。其中Jack Rae的报告表达了“预测即压缩”，Ilya的报告则表达了“压缩即泛化，泛化即智能”。

“Everything should be made as simple as possible, but not simpler.”

——Albert Einstein

[3] Jack Rae's talk: Compression For AGI, 2023.02.28

[4] Ilya's talk: An observation on Generalization, 2023.08.15

# 机器学习的基本概念

## ◆ 机器学习的分类

### 监督学习 (supervised learning)

数据有标签，一般为回归或分类等任务

### 无监督学习 (unsupervised learning)

数据无标签，一般为聚类、降维或概率密度估计等任务

### 强化学习 (reinforcement learning)

序列数据决策学习，一般为与从环境交互中学习

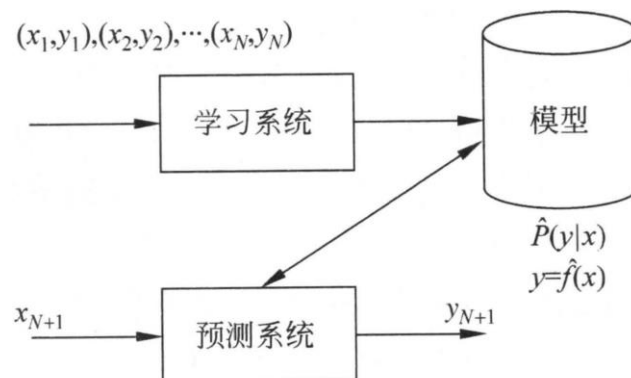


图 1.1 监督学习

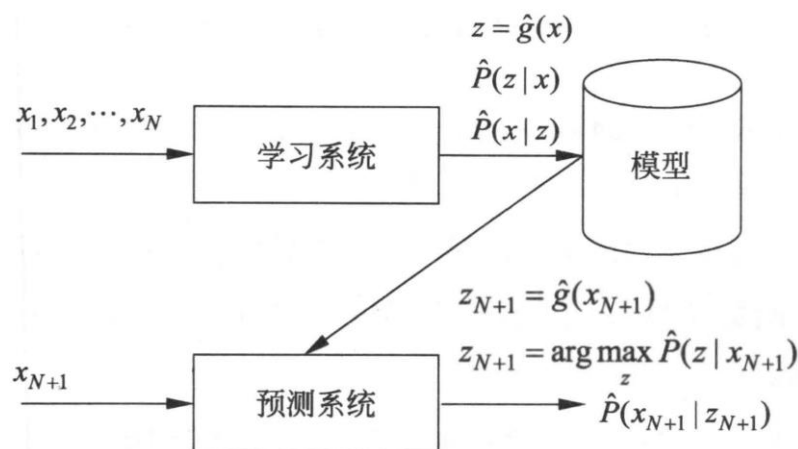


图 1.2 无监督学习

# 机器学习的基本概念

## ◆ 机器学习的分类

### 监督学习 (supervised learning)

数据有标签，一般为回归或分类等任务

### 无监督学习 (unsupervised learning)

数据无标签，一般为聚类、降维或概率密度估计等任务

### 强化学习 (reinforcement learning)

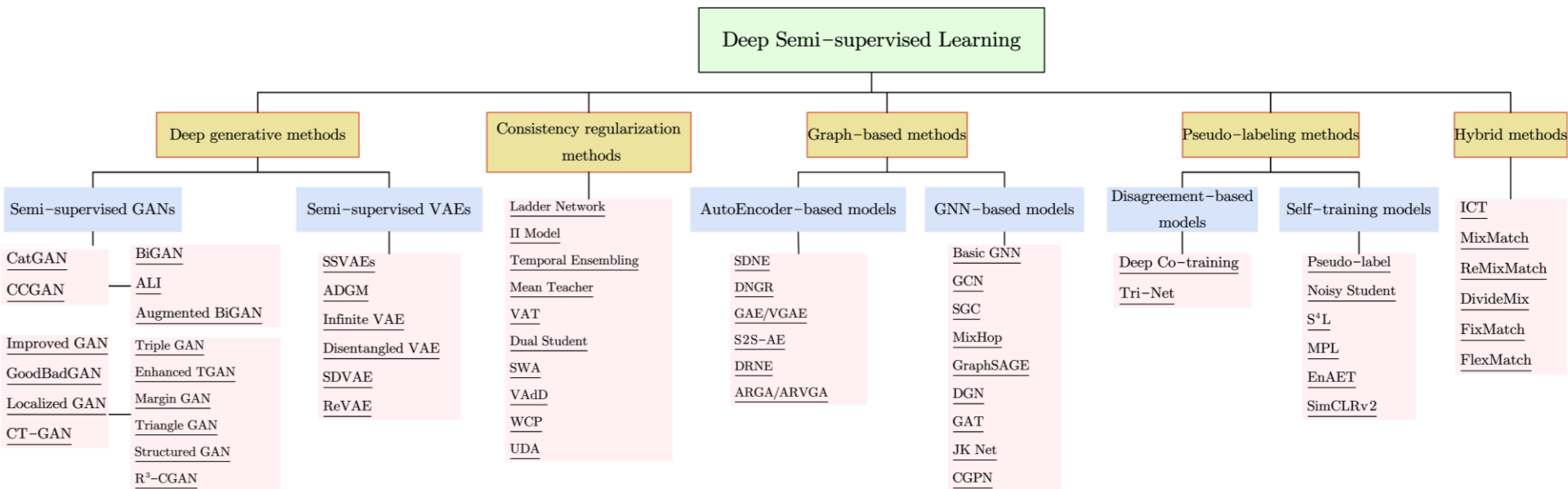
序列数据决策学习，一般为与从环境交互中学习

半监督学习  
(semi-supervised learning)

自监督学习  
(self-supervised learning)

# 机器学习的基本概念

## ◆ 机器学习的分类 半监督学习 (semi-supervised learning)



Formally, SSL aims to solve the following optimization problem,

$$\min_{\theta} \underbrace{\sum_{(x,y) \in X_L} \mathcal{L}_s(x, y, \theta)}_{\text{supervised loss}} + \alpha \underbrace{\sum_{x \in X_U} \mathcal{L}_u(x, \theta)}_{\text{unsupervised loss}} + \beta \underbrace{\sum_{x \in X} \mathcal{R}(x, \theta)}_{\text{regularization}},$$

Note that unsupervised loss terms are often not strictly distinguished from regularization terms, as regularization terms are normally not guided by label information.



# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation Maximization\*

- ✓ 周志华, “机器学习”, 2016, 清华大学出版社. Ch 12
- ✓ 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 2, 3, 4
- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社. Ch 1.6
- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社. Ch 4.5.1, 4.6.1

# 计算学习理论

- 计算学习理论 (Computational Learning Theory)
  - 可计算：什么任务是可以计算的？图灵可停机
  - 可学习：什么任务是可以被学习的，从而被学习模型来完成？
  - Leslie Valiant (2010年图灵奖获得者) 和其学生 Michael Kearns 两位学者提出了这个问题并进行了有益探索，逐渐完善了计算学习理论。
- 关注的问题
  - 怎样刻画“学习”这个过程？
  - 什么样的问题是“可学习的”？
  - 什么样的问题是“易学习的”？
  - 对于给定的学习算法，能否在理论上预测其性能？
  - 理论结果如何指导现实问题的算法设计？

# 计算学习理论

## ◆ 基础知识

- 样例集：独立同分布样本, 仅考虑二分类问题

$$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}, \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y} = \{-1, +1\}$$

- $h$  为从  $\mathcal{X}$  到  $\mathcal{Y}$  的一个映射

- 泛化误差 (Generalization error): 分类器的期望误差

$$E(h; \mathcal{D}) = P_{\mathbf{x} \sim \mathcal{D}}(h(\mathbf{x}) \neq y) = \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}}[\mathbb{I}(h(\mathbf{x}) \neq y)]$$

- 经验误差 (Empirical error): 分类器在给定样例集上的平均误差

$$\hat{E}(h; D) = \frac{1}{m} \sum_{i=1}^m \mathbb{I}(h(\mathbf{x}_i) \neq y_i)$$

由于  $D$  是  $\mathcal{D}$  的独立同分布采样, 因此  $h$  的经验误差的期望等于其泛化误差。  
在上下文明确时, 将  $E(h; \mathcal{D})$  和  $\hat{E}(h; D)$  分别简记为  $E(h)$  和  $\hat{E}(h)$ .

- 误差参数  $\epsilon$

$\epsilon$  为  $E(h)$  的上限, 即  $E(h) \leq \epsilon$ .

⇒表示预先设定的学得模型所应满足的误差要求

# 计算学习理论

## ◆ 基础知识

### ➤ 一致性 (consistency)

若  $h$  在数据集  $D$  上的经验误差为0, 则称  $h$  与  $D$  一致 (consistent), 否则为不一致 (non-consistent)。

### ➤ 不合 (disagreement) 对于任意两个映射 $h_1, h_2 \in \mathcal{X} \rightarrow \mathcal{Y}$ , 通过 “不合” 度量它们的差别: $d(h_1, h_2) = P_{x \sim \mathcal{D}}(h_1(x) \neq h_2(x))$

### ➤ 常用不等式

- Jense 不等式: 对任意凸函数  $f(x)$ , 有

$$f(\mathbb{E}(x)) \leq \mathbb{E}(f(x)) . \quad (12.4)$$

- Hoeffding 不等式 [Hoeffding, 1963]: 若  $x_1, x_2, \dots, x_m$  为  $m$  个独立随机变量, 且满足  $0 \leq x_i \leq 1$ , 则对任意  $\epsilon > 0$ , 有

$$P\left(\frac{1}{m} \sum_{i=1}^m x_i - \frac{1}{m} \sum_{i=1}^m \mathbb{E}(x_i) \geq \epsilon\right) \leq \exp(-2m\epsilon^2) , \quad (12.5)$$

$$P\left(\left|\frac{1}{m} \sum_{i=1}^m x_i - \frac{1}{m} \sum_{i=1}^m \mathbb{E}(x_i)\right| \geq \epsilon\right) \leq 2 \exp(-2m\epsilon^2) . \quad (12.6)$$

# 计算学习理论

## ◆ 基础知识

- McDiarmid 不等式 [McDiarmid, 1989]: 若  $x_1, x_2, \dots, x_m$  为  $m$  个独立随机变量, 且对任意  $1 \leq i \leq m$ , 函数  $f$  满足

$$\sup_{x_1, \dots, x_m, x'_i} |f(x_1, \dots, x_m) - f(x_1, \dots, x_{i-1}, x'_i, x_{i+1}, \dots, x_m)| \leq c_i ,$$

则对任意  $\epsilon > 0$ , 有

$$P(f(x_1, \dots, x_m) - \mathbb{E}(f(x_1, \dots, x_m)) \geq \epsilon) \leq \exp\left(\frac{-2\epsilon^2}{\sum_i c_i^2}\right) , \quad (12.7)$$

$$P(|f(x_1, \dots, x_m) - \mathbb{E}(f(x_1, \dots, x_m))| \geq \epsilon) \leq 2 \exp\left(\frac{-2\epsilon^2}{\sum_i c_i^2}\right) . \quad (12.8)$$



# 计算学习理论

## ◆ 概率近似正确 (PCA) 学习

### ➤ 概念 (concept) 一种映射

概念是从样本空间  $\mathcal{X}$  到标记空间  $\mathcal{Y}$  的映射, 它决定示例  $x$  的真实标记  $y$ .

- 目标概念: 如果对任何样例  $(x, y)$  均有  $c(x) = y$  成立, 则称  $c$  为目标概念.
- 概念类 (concept class): 所有我们希望学得的目标概念所构成的集合称为“概念类”, 用符号  $\mathcal{C}$  表示.

### ➤ 假设空间 (hypothesis space)

给定学习算法  $\mathcal{L}$ , 它所考虑的所有可能概念的集合, 用符号  $\mathcal{H}$  表示.

### ➤ 可分的与不可分的

- 可分的 (separable): 若目标概念  $c \in \mathcal{H}$ , 即  $\mathcal{H}$  中存在假设能将所有的示例完全正确分开 (按照与真实标记一致的方式), 则称该问题对学习算法  $\mathcal{L}$  是“可分的” (separable), 也称“一致的” (consistent).
- 不可分的 (non-separable): 若目标概念  $c \notin \mathcal{H}$ , 则  $\mathcal{H}$  中不存在任何假设能将所有的示例完全正确分开, 则称该问题对学习算法  $\mathcal{L}$  是“不可分的” (non-separable), 也称“不一致的” (non-consistent).

# 计算学习理论

## ◆ 概率近似正确 (Probably Approximately Correct, PCA) 学习

目标：对于给定训练集  $D$ ，我们希望基于学习算法  $\mathcal{L}$  学得模型所对应的假设  $h$  尽可能接近目标概念  $c$ 。

为什么不是希望精确地学到目标概念  $c$  呢？

机器学习过程受到很多因素的制约：

- 获得的训练集  $D$  往往仅包含有限数量的样例，因此通常会存在一些在  $D$  上“等效”的假设，学习算法无法区别这些假设；
- 从分布  $\mathcal{D}$  采样得到  $D$  的过程有一定的偶然性，即便对同样大小的不同训练集，学得结果也可能有所不同。
- 可能出现目标概念  $c \notin \mathcal{H}$ ， $\mathcal{H}$  中不存在任何假设能将所有的示例完全正确分开。

# 计算学习理论

- 概率近似正确 (Probably Approximately Correct, PAC)

我们希望以比较大的把握学得比较好的模型, 即以较大概率学得误差满足预设上限的模型. 令  $\delta$  表示置信度, 上述要求形式化为:

- 定义 **PAC辨识 (Probably Approximately Correct Identify)**

对  $0 < \epsilon, \delta < 1$ , 所有  $c \in \mathcal{C}$  和分布  $\mathcal{D}$ , 若存在学习算法  $\mathcal{L}$ , 其输出假设  $h \in \mathcal{H}$  满足:

$$P(E(h) \leq \epsilon) \geq 1 - \delta$$

则称学习算法  $\mathcal{L}$  能从假设空间  $\mathcal{H}$  中PAC辨识概念类  $\mathcal{C}$ .

这样的学习算法  $\mathcal{L}$  能以较大概率(至少  $1 - \delta$ )学得目标概念  $c$  的近似(误差最多为  $\epsilon$ ).

- 定义 **PAC可学习 (PAC Learnable)**

令  $m$  表示从分布  $\mathcal{D}$  中独立同分布采样得到的样例数目,  $0 < \epsilon, \delta < 1$ , 对所有分布  $\mathcal{D}$ , 若存在学习算法  $\mathcal{L}$  和多项式函数  $\text{poly}(\cdot, \cdot, \cdot, \cdot)$ , 使得对于任何  $m \geq \text{poly}(1/\epsilon, 1/\delta, \text{size}(x), \text{size}(c))$ ,  $\mathcal{L}$  能从假设空间  $\mathcal{H}$  中PAC辨识概念类  $\mathcal{C}$ , 则称概念类  $\mathcal{C}$  对假设空间  $\mathcal{H}$  是PAC可学习的, 有时也简称概念类  $\mathcal{C}$  是PAC可学习的。

样本数目  $m$  与误差  $\epsilon$ , 置信度  $1 - \delta$ , 数据本身的复杂度  $\text{size}(x)$ , 目标概念的复杂度  $\text{size}(c)$  有关。

# 计算学习理论

- PAC可学习性描述的是概念类  $C$  的性质, 若考虑到对应学习算法  $\mathcal{L}$  的**时间复杂度**, 则有:

## 定义 PAC学习算法(PAC Learning Algorithm)

若学习算法  $\mathcal{L}$  使概念类  $C$  为 PAC可学习的, 且  $\mathcal{L}$  的运行时间也是多项式函数  $\text{poly}(1/\epsilon, 1/\delta, \text{size}(x), \text{size}(c))$ , 则称概念类  $C$  是高效PAC可学习 (efficiently PAC learnable) 的, 称  $\mathcal{L}$  为概念类  $C$  的 PAC学习算法。

- 假定学习算法  $\mathcal{L}$  处理每个样本的时间为常数, 则  $\mathcal{L}$  的时间复杂度等价于其样本复杂度. 于是, 我们对算法时间复杂度的分析可变为对样本复杂度的分析。

## 定义 样本复杂度(Sample Complexity)

满足PAC学习算法  $\mathcal{L}$  所需的  $m \geq \text{poly}(1/\epsilon, 1/\delta, \text{size}(x), \text{size}(c))$  中最小的  $m$ , 称为学习算法  $\mathcal{L}$  的样本复杂度。

- PAC学习的意义:

给出了一个抽象地刻画机器学习能力的框架, 基于这个框架可以对很多重要问题进行理论探讨。

- 研究某任务在什么样的条件下可学得较好的模型?
- 某算法在什么样的条件下可进行有效的学习?
- 需要多少训练样例才能获得较好的模型?

把对复杂算法的**时间复杂度**的分析转为对**样本复杂度**的分析。

# 计算学习理论

## ◆ 假设空间的复杂度

假设空间  $\mathcal{H}$  的复杂度是影响可学习性的重要因素之一

- 一般而言,  $\mathcal{H}$  越大, 其包含任意目标概念的可能性越大, 但从中找到某个具体概念的难度也越大.
- $|\mathcal{H}|$  有限时, 我们称  $\mathcal{H}$  为“有限假设空间”, 否则称为“无限假设空间”.

假设空间  $\mathcal{H}$  的复杂度是影响学习任务难度的重要因素之一

恰PAC可学习 (properly PAC learnable)

假设空间  $\mathcal{H}$  包含了学习算法  $\mathcal{L}$  所有可能输出的假设, 在PAC学习中假设空间与概念类完全相同, 即  $\mathcal{H} = \mathcal{C}$  .

- 直观地看, 这意味着学习算法的能力与学习任务“恰好匹配”, 即所有候选假设都来自概念类.
- 然而在现实应用中我们对概念类  $\mathcal{C}$  通常一无所知, 设计一个假设空间与概念类恰好相同的学习算法通常是不切实际的.

研究的重点: 当假设空间与概念类不同的情形, 即  $\mathcal{H} \neq \mathcal{C}$  时.



# 计算学习理论

## ● 有限假设空间 可分情况

目标概念  $c$  属于假设空间  $\mathcal{H}$ , 即  $c \in \mathcal{H}$ .

➤ 给定包含  $m$  个样例的训练集  $D$ , 如何找出满足误差参数的假设呢?

一种简单的学习策略:

- 由于  $c$  存在于假设空间  $\mathcal{H}$  中, 因此任何在训练集  $D$  上出现标记错误的假设肯定不是目标概念  $c$ .
- 保留与  $D$  一致的假设, 剔除与  $D$  不一致的假设.
- 若训练集  $D$  足够大, 则可不断借助  $D$  中的样例剔除不一致的假设, 直到  $\mathcal{H}$  中仅剩下一个假设为止, 这个假设就是目标概念  $c$ .

通常情形下, 由于训练集规模有限, 假设空间  $\mathcal{H}$  中可能存在不止一个与  $D$  一致的“等效”假设。

➤ 到底需要多少样例才能学得目标概念  $c$  的有效近似呢?

训练集  $D$  的规模使得学习算法  $\mathcal{L}$  以概率  $1 - \delta$  找到目标假设的  $\epsilon$  近似, 则:

$$P(h \in \mathcal{H}: E(h) > \epsilon \wedge \hat{E}(h) = 0) \leq \sum_{h \in \mathcal{H}} P(E(h) > \epsilon \wedge \hat{E}(h) = 0) < |\mathcal{H}|(1 - \epsilon)^m \leq |\mathcal{H}|e^{-m\epsilon}$$

$$\text{令 } |\mathcal{H}|e^{-m\epsilon} \leq \delta \Rightarrow m \geq \frac{1}{\epsilon} (\ln |\mathcal{H}| + \ln \frac{1}{\delta})$$

可分情况下的有限假设空间  $\mathcal{H}$  都是 PAC 可学习的, 输出假设  $h$  的泛化误差随样例数目的增多而收敛到 0, 收敛速率为  $O\left(\frac{1}{m}\right)$ .

# 计算学习理论

## ● 有限假设空间 不可分情况

对于较困难的学习问题, 目标概念  $c$  不属于假设空间  $\mathcal{H}$ , 即假定对于任何  $h \in \mathcal{H}$ ,  $\hat{E}(h) \neq 0$ ,  $\mathcal{H}$  中的任何一个假设都会在训练集上出现或多或少的错误.

定理12.1 若  $\mathcal{H}$  为有限假设空间  $0 < \delta < 1$ , 则对任意  $h \in \mathcal{H}$ , 有

$$P\left(|E(h) - \hat{E}(h)| \leq \sqrt{\frac{\ln|\mathcal{H}| + \ln(2/\delta)}{2m}}\right) \geq 1 - \delta$$

证明略。详见 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 3

定理12.1表明在有限假设集的情况下, 当样本大小  $m$  足够大时,  $h$  的经验误差是其泛化误差很好的近似。此时尽管  $c \notin \mathcal{H}$ , 若能找到  $\mathcal{H}$  中泛化误差最小的假设也不失为一个较好的选择。定理12.1实际上指出了一种通用的学习原则。

**经验风险最小化 (Empirical Risk Minimization, ERM) 原则:**

令  $h$  表示学习算法  $\mathcal{L}$  输出的假设, 若  $h$  满足

$$\hat{E}(h) = \min_{h' \in \mathcal{H}} \hat{E}(h')$$

则称  $\mathcal{L}$  为满足经验风险最小化原则的算法。

# 计算学习理论

## ➤ 不可知 PAC 可学习

在  $c \notin \mathcal{H}$  时, 学习算法  $\mathcal{L}$  无法得到目标概念  $c$  的  $\epsilon$  近似。但是当假设空间  $\mathcal{H}$  给定时, 其中必存在一个泛化误差最小的假设  $\arg \min_{h \in \mathcal{H}} E(h)$ , 找出此假设的  $\epsilon$  近似也不失为一个较好的目标。可以把PAC学习的定义做如下推广:

### 定义 不可知PAC可学习(agnostic PAC Learnable)

令  $m$  表示从分布  $\mathcal{D}$  中独立同分布采样得到的样例数目,  $0 < \epsilon, \delta < 1$ , 对所有分布  $\mathcal{D}$ , 若存在学习算法  $\mathcal{L}$  和多项式函数  $\text{poly}(\cdot, \cdot, \cdot, \cdot)$ , 使得对于任何  $m \geq \text{poly}(1/\epsilon, 1/\delta, \text{size}(\mathbf{x}), \text{size}(c))$ ,  $\mathcal{L}$  能从假设空间  $\mathcal{H}$  中输出满足下式的假设:

$$P(E(h) - \min_{h' \in \mathcal{H}} \hat{E}(h') \leq \epsilon) \geq 1 - \delta$$

则称假设空间  $\mathcal{H}$  是不可知PAC可学习的 (agnostic PAC Learnable)。

- 定理12.1说明有限假设集是不可知PAC可学习的。

若学习算法  $\mathcal{L}$  的运行时间也是多项式函数  $\text{poly}(1/\epsilon, 1/\delta, \text{size}(\mathbf{x}), \text{size}(c))$ , 则称假设空间  $\mathcal{H}$  是高效不可知PAC可学的, 学习算法  $\mathcal{L}$  称为假设空间  $\mathcal{H}$  的不可知PAC可学习算法, 满足上述要求最小的  $m$  称为学习算法  $\mathcal{L}$  的样本复杂度。

# 计算学习理论

## ● 无限维假设空间

现实学习任务所面临的通常是无限假设空间，如实数域中的所有区间， $\mathbb{R}^d$ 空间中的所有线性超平面。

研究此种情形下的可学习性，需使用  $|\mathcal{H}|$  之外的方法度量假设空间的复杂性：

- VC 维 (Vapnik-Chervonenkis dimension)
- Rademacher复杂度 (Rademacher Complexity)

记号引入：

令  $\mathcal{H}$  表示假设空间，其中的假设是  $\mathcal{X}$  到  $\mathcal{Y} = \{-1, +1\}$  的映射，对于数据集  $D = \{x_1, x_2, \dots, x_m\} \subset \mathcal{X}$ ， $\mathcal{H}$  中每个假设  $h$  都能对  $D$  中示例赋予标记，标记结果称为  $h$  在数据集  $D$  上的限制 (restriction):  $h|_D = \{(h(x_1), \dots, h(x_m))\}$ ，是一个  $m$  维向量。

$\mathcal{H}|_D = \{(h(x_1), \dots, h(x_m)) | h \in \mathcal{H}\}$  称为  $\mathcal{H}$  在数据集  $D$  上的限制 (restriction)，是从  $D$  到  $\{-1, +1\}^m$  的一族映射。随着  $m$  的增大， $\mathcal{H}$  中所有假设对  $D$  中的示例所能赋予标记的可能结果数也会增大。

例如，对于二分类问题：若  $D$  中只有两个示例，则赋予标记的可能结果只有4种；若  $D$  中有3个示例，则可能结果有8种。

# 计算学习理论

## ● VC 维 (Vapnik-Chervonenkis dimension)

概念引入:

增长函数 (growth function), 对分 (dichotomy), 打散 (shattering)

### ➤ 定义 增长函数(growth function)

对所有  $m \in \mathbb{N}$ , 假设空间  $\mathcal{H}$  的增长函数  $\Pi_{\mathcal{H}}(m)$  为:

$$\Pi_{\mathcal{H}}(m) = \max_{\{x_1, \dots, x_m\} \subset \mathcal{X}} |\{(h(x_1), \dots, h(x_m)) | h \in \mathcal{H}\}|$$

对于大小为  $m$  的数据集  $D$ , 有:  $\Pi_{\mathcal{H}}(m) = \max_{|D|=m} |\mathcal{H}|_D|$

- 增长函数表示假设空间  $\mathcal{H}$  对  $m$  个示例所能赋予标记的最大可能结果数.
- $\mathcal{H}$  对示例所能赋予标记的可能结果数越大,  $\mathcal{H}$  的表示能力越强, 对学习任务的适应能力也越强.
- 增长函数表述了假设空间  $\mathcal{H}$  的表示能力, 由此反映出假设空间的复杂度.

假设空间  $\mathcal{H}$  中不同的假设对于  $D$  中示例赋予标记的结果可能相同, 也可能不同. 尽管  $\mathcal{H}$  可能包含无穷多个假设, 但是其对  $D$  中示例赋予标记的可能结果  $\mathcal{H}|_D$  是有限的: 对于  $m$  个示例, 最多有  $2^m$  个可能结果 (二分类).



# 计算学习理论

## ● VC 维 (Vapnik-Chervonenkis dimension)

- 对分 (dichotomy): 对二分类问题来说,  $\mathcal{H}$  中的假设对  $D$  中示例赋予标记的每种可能结果称为对  $D$  的一种“对分”.
- 打散 (shattering): 若假设空间  $\mathcal{H}$  能实现示例集  $D$  上的所有对分, 即  $|\mathcal{H}|_D| = 2^m$ , 则称示例集  $D$  能被假设空间  $\mathcal{H}$  “打散”, 此时  $\Pi_{\mathcal{H}}(m) = 2^m$ .

假设空间  $\mathcal{H}$  的VC维是能被  $\mathcal{H}$  打散的最大示例集的大小, 即

$$\text{VC}(\mathcal{H}) = \max\{m: \Pi_{\mathcal{H}}(m) = 2^m\}$$

$\text{VC}(\mathcal{H}) = d$  意味着存在一个大小为  $d$  的示例集能被  $\mathcal{H}$  打散。注意: 这并不意味着所有大小为  $d$  的示例集都能被假设空间  $\mathcal{H}$  打散。

➤ 证明一个假设空间  $\mathcal{H}$  的 VC 维为  $d$ :

- 存在大小为  $d$  的样本集  $D$  能被  $\mathcal{H}$  打散;
- 任意大小为  $d + 1$  的样本集  $D'$  都不能被  $\mathcal{H}$  打散.

# 计算学习理论

例1 实数域中的区间  $[a, b]$

令  $\mathcal{H}$  表示实数域中所有闭区间构成的集合  $\{h_{[a,b]}: a, b \in \mathbb{R}, a \leq b\}$ ,  $\mathcal{X} = \mathbb{R}$ . 对  $x \in \mathcal{X}$ , 若  $x \in [a, b]$ , 则  $h_{[a,b]} = +1$ , 否则  $h_{[a,b]} = -1$ .

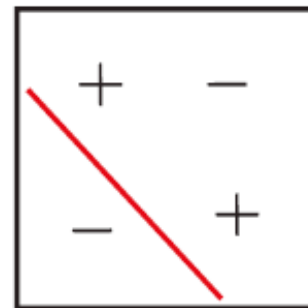
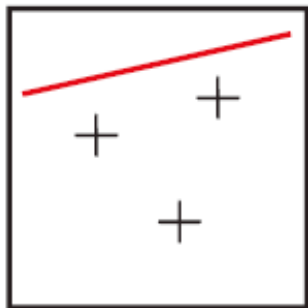
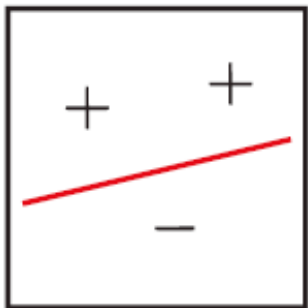
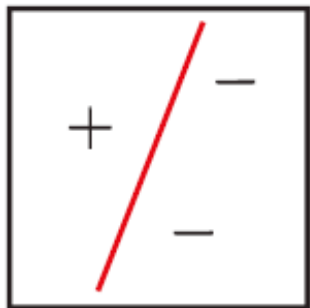
令样本  $x_1 = 0.5, x_2 = 1.5$  则假设空间  $\mathcal{H}$  中存在假设  $\{h_{[0,1]}, h_{[0,2]}, h_{[1,2]}, h_{[2,3]}\}$  将  $\{x_1, x_2\}$  打散, 所以假设空间  $\mathcal{H}$  的VC维至少为2;

对任意大小为3的示例集  $\{x_3, x_4, x_5\}$ , 不妨设  $x_3 < x_4 < x_5$ , 则  $\mathcal{H}$  中不存在任何假设  $h_{[a,b]}$  能实现对分结果  $\{(x_3, +), (x_4, -), (x_5, +)\}$ . 于是,  $\mathcal{H}$  的VC维为2.

例2 二维实平面的线性划分

令  $\mathcal{H}$  表示二维实平面上所有线性划分构成的集合,  $\mathcal{X} = \mathbb{R}^2$ . 由下图可知, 存在大小为3的示例集可被  $\mathcal{H}$  打散, 但不存在大小为4的示例集可被  $\mathcal{H}$  打散.

于是, 二维实平面上所有线性划分构成的假设空间  $\mathcal{H}$  的VC维为3.



存在这样的集合, 其  $2^3 = 8$  种对分均可被线性划分实现

对任何集合, 其  $2^4 = 16$  种对分中至少有一种不能被线性划分实现

# 计算学习理论

## ➤ VC维与增长函数之间的定量关系:

Sauer引理 (引理12.2) [Sauer, 1972] 若假设空间  $\mathcal{H}$  的VC维为  $d$  , 则对任意  $m \in \mathbb{N}$  有:

$$\Pi_{\mathcal{H}}(m) \leq \sum_{i=0}^d \binom{m}{i}$$

由Sauer引理可以计算出增长函数的上界:

推论 12.2 若假设空间  $\mathcal{H}$  的VC维为  $d$ , 则对任意整数  $m \geq d$  有:  $\Pi_{\mathcal{H}}(m) \leq \left(\frac{e \cdot m}{d}\right)^d$

## ➤ 增长函数与泛化误差

利用增长函数 (growth function) 估计经验误差与泛化误差之间的关系:

定理12.2 对假设空间  $\mathcal{H}$ ,  $m \in \mathbb{N}$ ,  $0 < \epsilon < 1$  和任意  $h \in \mathcal{H}$  有

$$P(|E(h) - \hat{E}(h)| \geq \epsilon) \leq 4\Pi_{\mathcal{H}}(2m)\exp\left(-\frac{m\epsilon^2}{8}\right)$$

证明略。详见 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 3

# 计算学习理论

## ➤ 基于VC维泛化误差界

**定理12.3** 若假设空间  $\mathcal{H}$  的VC维为  $d$ , 则对任意  $m > d$ ,  $0 < \delta < 1$  和  $h \in \mathcal{H}$  有:

$$P \left( E(h) - \hat{E}(h) \leq \sqrt{\frac{8d \ln \frac{2em}{d} + 8 \ln \frac{4}{\delta}}{m}} \right) \geq 1 - \delta$$

证明: 令  $\Pi_{\mathcal{H}}(2m) \exp \left( -\frac{m\epsilon^2}{8} \right) \leq 4 \left( \frac{2em}{d} \right)^d \exp \left( -\frac{m\epsilon^2}{8} \right) = \delta$ , 解得

$\epsilon = \sqrt{\frac{8d \ln \frac{2em}{d} + 8 \ln \frac{4}{\delta}}{m}}$ . 代入定理12.2, 于是定理12.3得证.

✓ 上式的泛化误差界只与样例数目  $m$  有关, 收敛速率为  $O\left(\frac{1}{\sqrt{m}}\right)$ .

✓ 上式的泛化误差界与数据分布  $\mathcal{D}$  及样例集  $D$  无关.

基于VC维的泛化误差界

分布无关 (distribution-free) & 数据独立 (data-independent)

类似定理12.1, 从定理12.3易看出, 当假设空间的VC维有限且样本大小  $m$  足够大时,  $h$  的经验误差  $\hat{E}(h)$  是其泛化误差  $E(h)$  的较好近似, 因此对于满足经验风险最小化原则的学习算法  $\mathcal{L}$ , 有下述定理:

**定理12.4** 任何VC维有限的假设空间  $\mathcal{H}$  都是 (不可知) PAC可学习的.

# 计算学习理论

## ● Rademacher 复杂度

基于VC维的泛化误差界是**分布无关、数据独立**的,这使得基于 VC 维的可学习性分析结果具有一定的“普适性”。但由于没有考虑数据自身,因此得到的泛化误差界通常比较松,是**假设空间所能达到的最大划分能力**。

能否将数据的分布也考虑进来?

Rademacher 复杂度 (Rademacher complexity): 另一种刻画假设空间复杂度的途径,与VC维不同的是,它**在一定程度上考虑了数据分布**。

给定训练集  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ ,  $y_i \in \{+1, -1\}$

则假设  $h$  的经验误差为

$$\hat{E}(h) = \frac{1}{m} \sum_{i=1}^m \mathbb{I}(h(\mathbf{x}_i) \neq y_i) = \frac{1}{m} \sum_{i=1}^m \frac{1 - y_i h(\mathbf{x}_i)}{2} = \frac{1}{2} - \frac{1}{2m} \sum_{i=1}^m y_i h(\mathbf{x}_i)$$

- 其中  $\frac{1}{m} \sum_{i=1}^m y_i h(\mathbf{x}_i)$  体现了预测值  $h(\mathbf{x}_i)$  与样例真实标记  $y_i$  之间的一致性。
- 若对于所有的  $i \in \{1, 2, \dots, m\}$ , 都有  $h(\mathbf{x}_i) = y_i$ , 则  $\frac{1}{m} \sum_{i=1}^m y_i h(\mathbf{x}_i)$  取最大值1。
- 经验误差最小的假设是  $\arg \max_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m y_i h(\mathbf{x}_i)$ 。



# 计算学习理论

若假设标签  $y_i$  受到随机因素的影响, 不再是  $x_i$  的真实标记. 选择假设空间  $\mathcal{H}$  中在训练集上表现最好的假设, 不如选择  $\mathcal{H}$  中事先已经考虑了随机噪声影响的假设.

- 考虑随机变量  $\sigma_i$ , 它以0.5的概率取值+1, 0.5的概率取值-1, 称为 Rademacher 随机变量。
- 基于  $\sigma_i$ , 可将  $\arg \max_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m y_i h(x_i)$  改写为  $\sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m \sigma_i h(x_i)$

对  $\sigma = (\sigma_1; \dots; \sigma_m)$  求期望可得

$$\mathbb{E}_{\sigma} \left[ \sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m \sigma_i h(x_i) \right]$$

- 上式的取值范围是  $[0,1]$  体现了假设空间  $\mathcal{H}$  的表达能力, 和增长函数有相同的作用。
- 当取值为1时, 对任意  $\sigma = (\sigma_1; \dots; \sigma_m)$ ,  $\sigma_i \in \{-1, +1\}$ , 存在  $h \in \mathcal{H}$  使得  $h(x_i) = \sigma_i$ , 即  $\Pi_{\mathcal{H}}(m) = 2^m$ ,  $\mathcal{H}$  能打散  $D$ . 取值越接近1, 假设空间的表示能力越强。

# 计算学习理论

## ➤ 定义 Rademacher 复杂度 (Rademacher complexity)

考虑实值函数空间  $\mathcal{F}: \mathcal{Z} \mapsto \mathbb{R}$ , 令  $Z = \{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ ,  $\mathbf{z}_i \in \mathcal{Z}$ , 函数空间  $\mathcal{F}$  关于  $Z$  的经验 Rademacher 复杂度为

$$\hat{R}_Z(\mathcal{F}) = \mathbb{E}_{\sigma} \left[ \sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^m \sigma_i f(\mathbf{z}_i) \right]$$

经验 Rademacher 复杂度衡量了函数空间  $\mathcal{F}$  与随机噪声在集合  $Z$  上的相关性。

函数空间  $\mathcal{F}$  关于  $Z$  上分布  $\mathcal{D}$  的 Rademacher 复杂度: 对从分布  $\mathcal{D}$  独立同分布采样得到的大小为  $m$  的集合  $Z$  求期望.

$$R_m(\mathcal{F}) = \mathbb{E}_{Z \subset \mathcal{Z}: |Z|=m} [\hat{R}_Z(\mathcal{F})]$$

需要注意到, 在 Rademacher 复杂度的定义中  $\sigma_i$  是  $\{-1, +1\}$  上服从均匀分布的随机变量, 如果将均匀分布改为其他分布, 会得到其他一些复杂度定义。

⇒ 基于 Rademacher 复杂度可得关于函数空间  $\mathcal{F}$  的泛化误差界。

# 计算学习理论

## ➤ Rademacher 复杂度与增长函数之间的关系

定理 3.4 [Massart 2000] 令  $A \subset \mathbb{R}^m$  为有限集合且  $r = \max_{x \in A} \|x\|$ , 有

$$\mathbb{E}_{\sigma} \left[ \frac{1}{m} \sup_{x \in A} \sum_{i=1}^m \sigma_i x_i \right] \leq \frac{r \sqrt{2 \ln |A|}}{m}$$

其中  $x = (x_1; \dots; x_m)$ ,  $\sigma_i$  为 Rademacher 随机变量。

证明略。详见 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 3

定理 12.7 假设空间  $\mathcal{H}$  的 Rademacher 复杂度与增长函数  $\Pi_{\mathcal{H}}(m)$  之间满足

$$\mathfrak{R}_m(\mathcal{H}) \leq \sqrt{\frac{2 \ln \Pi_{\mathcal{H}}(m)}{m}}$$

证明：由定理 3.4 可得

$$\mathfrak{R}_m(\mathcal{H}) = \mathbb{E}_D \left[ \mathbb{E}_{\sigma} \left[ \sup_{u \in \mathcal{H}|_D} \frac{1}{m} \sum_{i=1}^m \sigma_i u_i \right] \right] \leq \mathbb{E}_D \left[ \frac{\sqrt{m} \sqrt{2 \ln |\mathcal{H}|_D}}{m} \right]$$

$$\text{又因为 } \mathcal{H}|_D \leq \Pi_{\mathcal{H}}(m), \text{ 有 } \mathfrak{R}_m(\mathcal{H}) \leq \mathbb{E}_D \left[ \frac{\sqrt{m} \sqrt{2 \ln \Pi_{\mathcal{H}}(m)}}{m} \right] = \sqrt{\frac{2 \ln \Pi_{\mathcal{H}}(m)}{m}}$$

# 计算学习理论

## ➤ 基于 Rademacher 复杂度的泛化误差上界

**定理12.5** 对实值函数空间  $\mathcal{F}: \mathcal{Z} \mapsto [0,1]$ , 根据分布  $\mathcal{D}$  从  $\mathcal{Z}$  中独立同分布采样得到大小为  $m$  的示例集  $Z = \{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ ,  $\mathbf{z}_i \in \mathcal{Z}$ ,  $0 < \delta < 1$ , 对任意  $f \in \mathcal{F}$ , 以至少  $1 - \delta$  的概率有

$$\mathbb{E}(f(\mathbf{z})) \leq \frac{1}{m} \sum_{i=1}^m f(\mathbf{z}_i) + 2\mathfrak{R}_m(\mathcal{F}) + \sqrt{\frac{\ln(1/\delta)}{2m}}$$
$$\mathbb{E}(f(\mathbf{z})) \leq \frac{1}{m} \sum_{i=1}^m f(\mathbf{z}_i) + 2\hat{\mathfrak{R}}_Z(\mathcal{F}) + 3\sqrt{\frac{\ln(1/\delta)}{2m}}$$

证明略。详见 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 4  
定理12.5的适用范围是实值函数空间  $\mathcal{F}: \mathcal{Z} \mapsto [0,1]$ , 一般用于回归问题。

# 计算学习理论

## ➤ 基于 Rademacher 复杂度的泛化误差上界

**定理12.6** 对假设空间  $\mathcal{H}: \mathcal{Z} \mapsto \{-1, +1\}$ , 根据分布  $\mathcal{D}$  从  $\mathcal{Z}$  中独立同分布采样得到大小为  $m$  的示例集  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ ,  $\mathbf{x}_i \in \mathcal{X}$ ,  $0 < \delta < 1$ , 对任意  $h \in \mathcal{H}$ , 以至少  $1 - \delta$  的概率有

$$E(h) \leq \hat{E}(h) + \mathfrak{R}_m(\mathcal{H}) + \sqrt{\frac{\ln(1/\delta)}{2m}}$$
$$E(h) \leq \hat{E}(h) + \hat{\mathfrak{R}}_D(\mathcal{H}) + 3 \sqrt{\frac{\ln(1/\delta)}{2m}}$$

证明:

令  $\mathcal{Z} = \mathcal{X} \times \{-1, +1\}$ ,  $\mathcal{H}$  中的假设  $h$  可以变形为  $f_h(z) = f_h(\mathbf{x}, y) = \mathbb{I}(h(\mathbf{x}) \neq y)$ .

$$\begin{aligned} \hat{R}_Z(\mathcal{F}_{\mathcal{H}}) &= \mathbb{E}_{\sigma} \left[ \sup_{f_h \in \mathcal{F}_{\mathcal{H}}} \frac{1}{m} \sum_{i=1}^m \sigma_i f_h(\mathbf{x}_i, y_i) \right] = \mathbb{E}_{\sigma} \left[ \sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m \sigma_i \mathbb{I}(h(\mathbf{x}_i) \neq y_i) \right] \\ &= \mathbb{E}_{\sigma} \left[ \sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m \sigma_i \frac{1 - y_i h(\mathbf{x}_i)}{2} \right] = \frac{1}{2} \mathbb{E}_{\sigma} \left[ \sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m (\sigma_i h(\mathbf{x}_i)) \right] = \frac{1}{2} \hat{R}_D(\mathcal{H}) \end{aligned}$$

同时对两边取期望得  $R_m(\mathcal{F}_{\mathcal{H}}) = \frac{1}{2} R_m(\mathcal{H})$ , 由定理12.5得证。



# 计算学习理论

**定理 12.3 基于VC维的泛化误差上界**  
若假设空间  $\mathcal{H}$  的VC维为  $d$ , 则对任意  $m > d$ ,  $0 < \delta < 1$  和  $h \in \mathcal{H}$  有:

$$P \left( E(h) - \hat{E}(h) \leq \sqrt{\frac{8d \ln \frac{2em}{d} + 8 \ln \frac{4}{\delta}}{m}} \right) \geq 1 - \delta$$

- 定理12.3(基于VC维的泛化误差界)与分布无关、数据独立的;
- 定理12.6(基于Rademacher复杂度的泛化误差界)与分布  $\mathcal{D}$  或样本  $D$  有关。

基于 Rademacher 复杂度的泛化误差上界依赖于具体学习问题的数据分布, 类似于为该问题“量身定制”的, 因此它通常比基于VC维的泛化误差上界要更紧一些。从 Rademacher 复杂度和增长函数能推导出基于VC维的泛化误差上界:

由定理12.6, 定理12.7, 推论12.2 可得:  $E(h) - \hat{E}(h) \leq \sqrt{\frac{2d \ln \frac{em}{d}}{m}} + \sqrt{\frac{\ln(\frac{1}{\delta})}{2m}}$

**定理12.6 基于 Rademacher 复杂度的泛化误差上界**

对假设空间  $\mathcal{H}: \mathcal{Z} \mapsto \{-1, +1\}$ , 根据分布  $\mathcal{D}$  从  $\mathcal{Z}$  中独立同分布采样得到大小为  $m$  的示例集  $D = \{x_1, x_2, \dots, x_m\}$ ,  $x_i \in \mathcal{X}$ ,  $0 < \delta < 1$ , 对任意  $h \in \mathcal{H}$ , 以至少  $1 - \delta$  的概率有:

$$E(h) \leq \hat{E}(h) + R_m(\mathcal{H}) + \sqrt{\frac{\ln(1/\delta)}{2m}},$$
$$E(h) \leq \hat{E}(h) + \hat{R}_D(\mathcal{H}) + 3 \sqrt{\frac{\ln(1/\delta)}{2m}},$$

# 计算学习理论

无论基于VC维和Rademacher复杂度来分析泛化性能,得到的结果均与具体的学习算法无关,这使得人们能够脱离具体的学习算法来考虑学习问题本身的性质。

但另一方面,为了获得与算法有关的分析结果,则需另辟蹊径。

◆ 稳定性(stability)分析是这方面值得关注的一个方向。

考察算法在输入(训练集)发生变化时,输出是否发生较大的变化。

# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation maximization\*

- ✓ 周志华, “机器学习”, 2016, 清华大学出版社. Ch 6
- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社. Ch7
- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社.
- ✓ 周志华等, “机器学习理论导引”, 2020, 机械工业出版社. Ch 1.3, 1.4

# 支持向量机

## ◆ 结构风险最小化

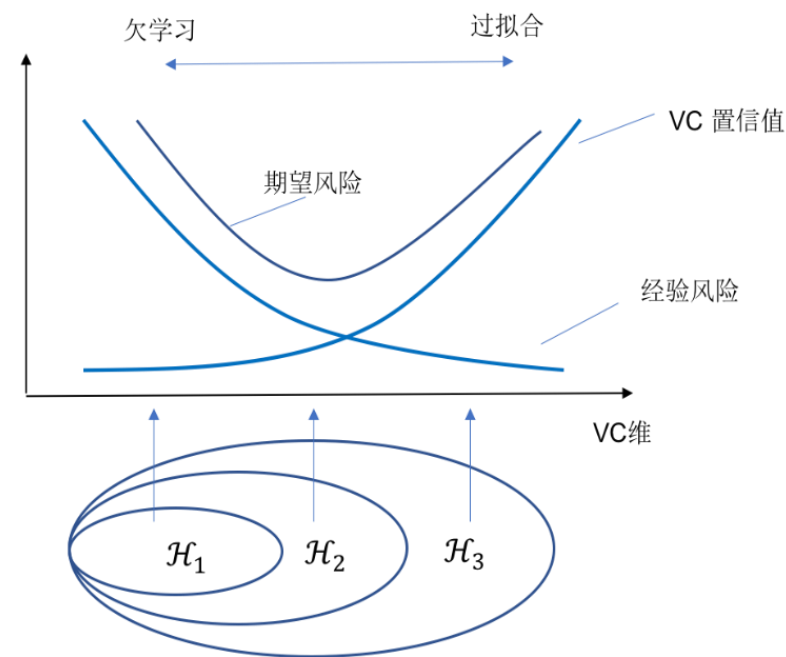
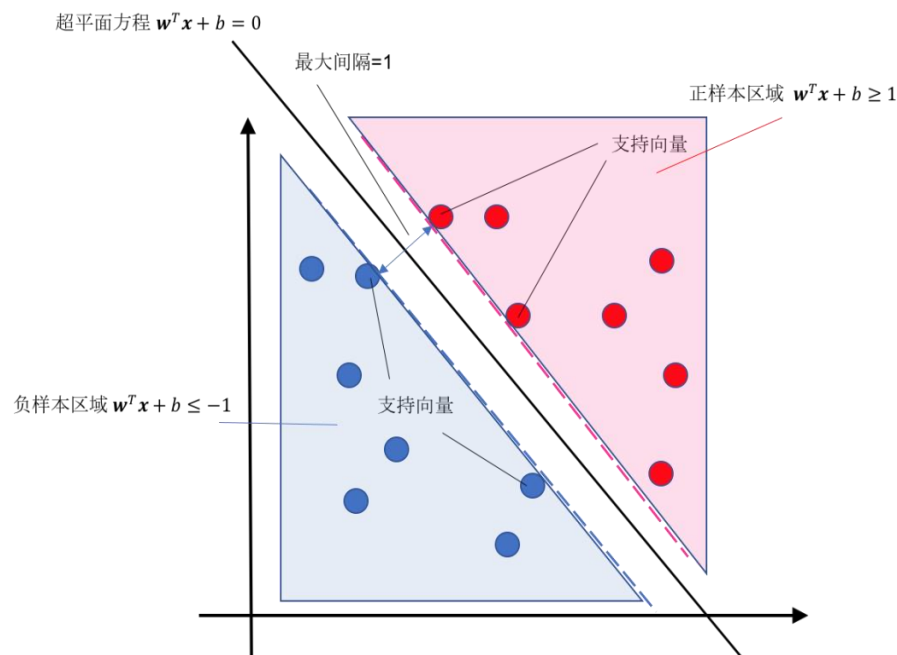
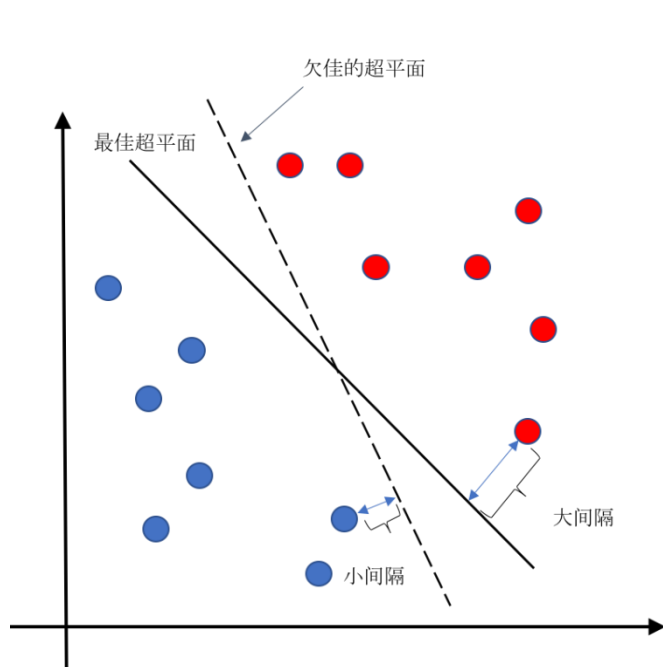


图4.11 经验风险、期望风险、VC置信度、VC维、过学习和欠学习的关系

**结构风险最小化** 通过VC理论，可认识到期望风险（即真实风险） $\mathfrak{R}$ 与经验风险 $\mathfrak{R}_{emp}$ 之间是有差别的，这个差别项被称为置信风险，它与训练样本个数和模型复杂度都有密切的关系。用复杂度高的模型去拟合小样本，往往会导致过拟合，因此需要给经验风险 $\mathfrak{R}_{emp}$ 加上一个惩罚项或者正则化项，以同时考虑经验风险与置信风险。这一思路被称为结构风险最小化。这样，在小样本情况下可取得较好性能。在保证分类精度高（经验风险小）同时，有效降低算法模型的VC维，可使算法模型在整个样本集上的期望风险得到控制。当训练样本给定时，分类间隔越大，则对应的分类超平面集合的VC维就越小。

# 支持向量机

## ◆ 线性可分支持向量机



寻找一个最优的超平面，其方程为  $w^T x + b = 0$ 。这里  $w = (w_1, w_2, \dots, w_d)$  为超平面的法向量，与超平面的方向有关； $b$  为偏置项，是一个标量，决定了超平面与原点之间的距离。



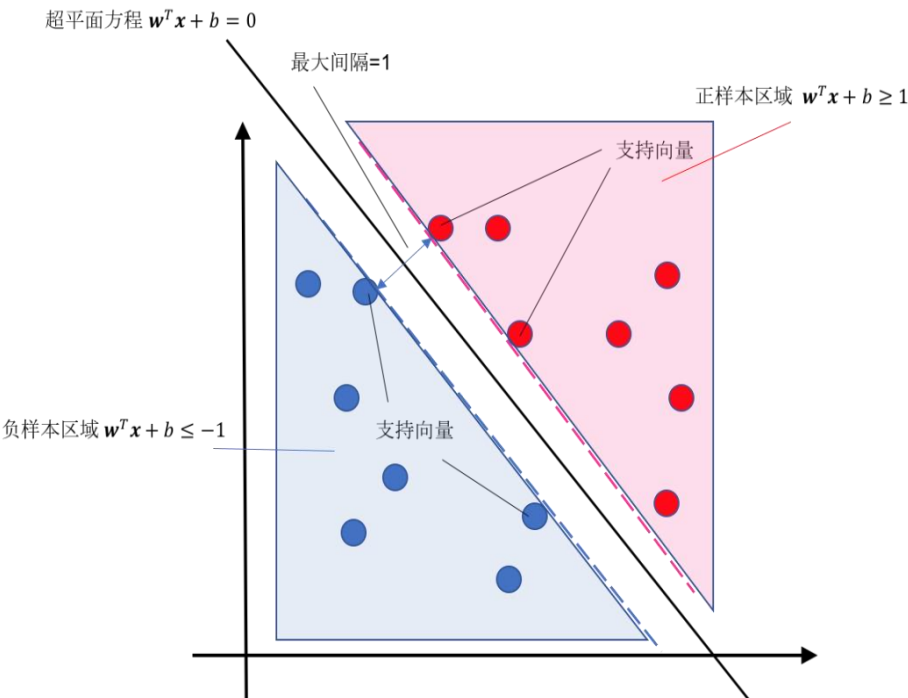
# 支持向量机

## ◆ 线性可分支持向量机

样本空间中任意样本 $x$ 到该平面距离可表示为:

$$r = d(\mathbf{w}, b, x) = \frac{|\mathbf{w}^T \mathbf{x} + b|}{\|\mathbf{w}\|_2} \quad (\|\mathbf{w}\|_2 = \sqrt{\mathbf{w}^T \mathbf{w}})$$

几何间隔



由于法向量 $\mathbf{w}$ 中的值可按比例任意缩放而不改变法向量方向，使得分类平面不唯一。为此，对 $\mathbf{w}$ 和 $b$ 添加如下约束：

$$|\mathbf{w}^T \mathbf{x}_i + b|_{\arg\min_{\mathbf{x}_i} r} = 1$$

即离超平面最近的正负样本代入超平面方程后其绝对值为1。

定义函数间隔:  $\hat{r}_i = y_i(\mathbf{w}^T \mathbf{x}_i + b)$

于是对超平面的约束变为:  $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ 。

# 支持向量机

## ◆ 线性可分支持向量机

两类样本中离分类超平面最近的数据之间的距离可如下计算：

$$\begin{aligned} d(\mathbf{w}, b) &= \min_{(x_k, y_k=1)} d(\mathbf{w}, b, \mathbf{x}_k) + \min_{(x_m, y_m=-1)} d(\mathbf{w}, b, \mathbf{x}_m) \\ &= \min_{(x_k, y_k=1)} \frac{|\mathbf{w}^T \mathbf{x}_k + b|}{\|\mathbf{w}\|_2} + \min_{(x_m, y_m=-1)} \frac{|\mathbf{w}^T \mathbf{x}_m + b|}{\|\mathbf{w}\|_2} \\ &= \frac{1}{\|\mathbf{w}\|_2} \left( \min_{(x_k, y_k=1)} |\mathbf{w}^T \mathbf{x}_k + b| + \min_{(x_m, y_m=-1)} |\mathbf{w}^T \mathbf{x}_m + b| \right) = \frac{2}{\|\mathbf{w}\|_2} \end{aligned}$$

$$\text{即: } d(\mathbf{w}, b) = \frac{2}{\|\mathbf{w}\|_2}$$

支持向量机的基本形式就是最大化分类间隔，即在满足约束的条件下找到参数  $\mathbf{w}$  和  $b$  使得  $\gamma$  最大，即等价于：

$$\begin{aligned} \min_{\mathbf{w}, b} \frac{\|\mathbf{w}\|^2}{2} &= \min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^T \mathbf{w} \\ \text{s.t. } y_i (\mathbf{w}^T \mathbf{x}_i + b) &\geq 1, i = 1, 2, \dots, n \end{aligned}$$

凸二次规划

# 支持向量机

先前介绍中假设所有训练样本数据是线性可分，即存在一个线性超平面能将不同类别样本完全隔开，这种情况称为“硬间隔”（hard margin），与硬间隔相对的是“软间隔”（soft margin）。软间隔指允许部分错分给定的训练样本。

$$\min_{w,b} \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^n \mathbb{I}(y_i \neq \text{sign}(\mathbf{w}^T \mathbf{x}_i + b))$$

s. t.  $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$  for correct  $\mathbf{x}_i$   
 $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq -\infty$  for incorrect  $\mathbf{x}_i$

$$\min_{w,b,\xi} \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^n \xi_i$$

s. t.  $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$   
 $\xi_i \geq 0, i = 1, 2, \dots, n$  松弛变量

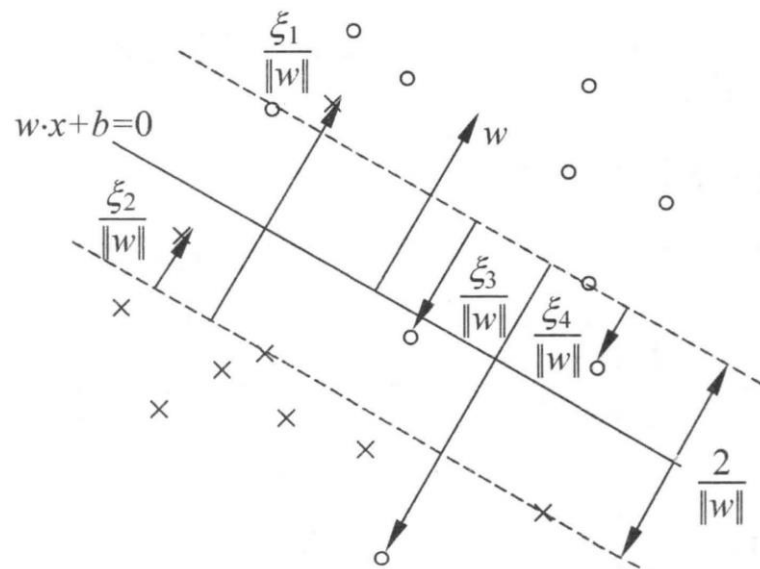
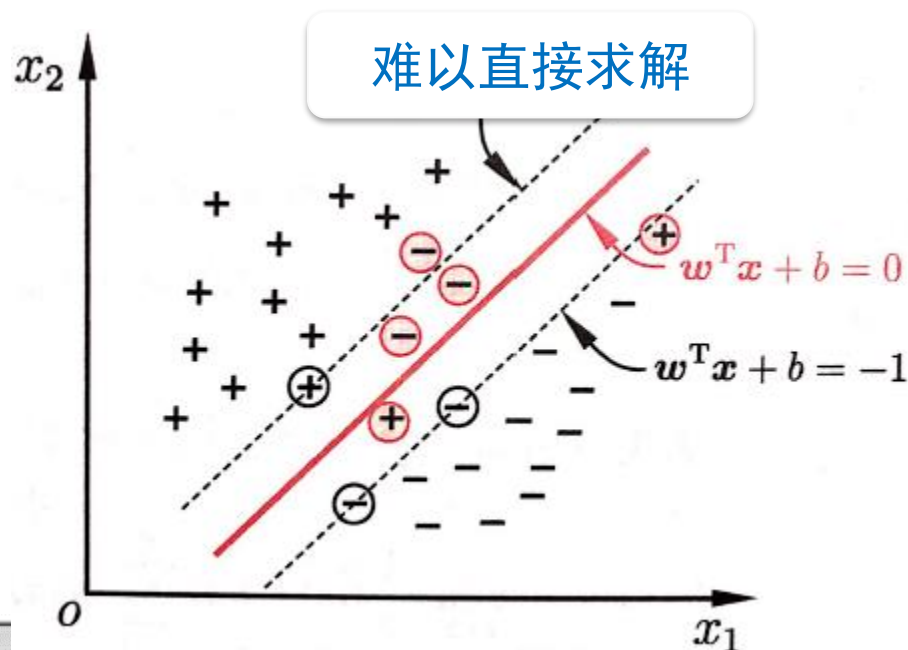


图 7.5 软间隔的支持向量

# 支持向量机

$$\begin{aligned} \min_{\mathbf{w}, b} & \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^N \xi_i \\ \text{s.t. } & y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, 2, \dots, N \end{aligned}$$

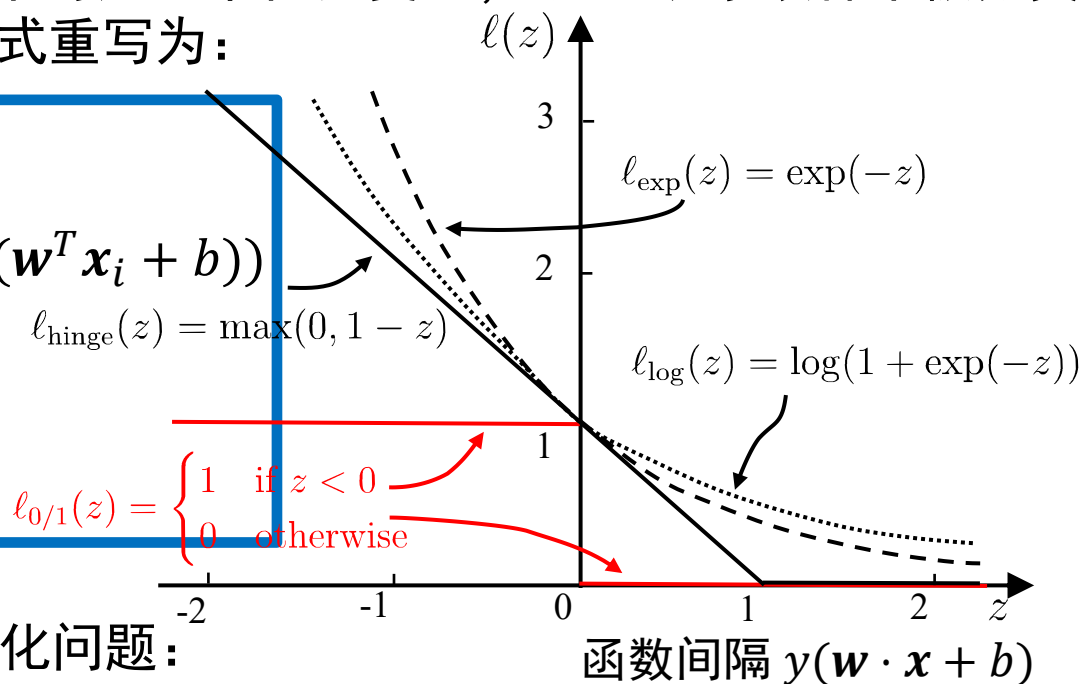
记  $\xi_i = \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$  ( $\xi_i$  被称为第  $i$  个变量的“松弛变量”，slack variables)，显然  $\xi_i \geq 0$ 。每一个样本对应一个松弛变量，用来表示该样本被分类错误所产生的损失。于是，可将上式重写为：

加入 hinge loss 项的目标函数：

$$\min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^n \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))$$

正确分类数据的 **hinge loss**：

$$\max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0$$



定理7.4 线性支持向量机原始最优化问题：

$$\text{等价于最优化问题 } \min_{\mathbf{w}, b} \sum_{i=1}^N [1 - y_i(\mathbf{w} \cdot \mathbf{x}_i)]_+ + c \|\mathbf{w}\|^2$$

# 支持向量机

软间隔支持向量机的原始问题（凸二次规划）：

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^n \xi_i \\ \text{s. t.} \quad & y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, 2, \dots, N \\ & \xi_i \geq 0, i = 1, 2, \dots, N \end{aligned}$$

定义7.5（线性支持向量机）对于给定的线性不可分的训练数据集，通过求解凸二次规划问题，即软间隔最大化问题，得到的分离超平面为

$$\mathbf{w}^* \cdot \mathbf{x} + b^* = 0$$

以及相应的分类决策函数

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^* \cdot \mathbf{x} + b^*)$$

称为线性支持向量机。



# 支持向量机

软间隔支持向量机的原始问题  
(凸二次规划):

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} & \frac{1}{2} \mathbf{w}^T \mathbf{w} + c \times \sum_{i=1}^n \xi_i \\ \text{s. t. } & y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, 2, \dots, N \\ & \xi_i \geq 0, i = 1, 2, \dots, N \end{aligned}$$

原始问题的对偶问题是:

$$\begin{aligned} \min_{\alpha} & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^N \alpha_i \\ \text{s. t. } & \sum_{i=1}^N \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1, 2, \dots, N \end{aligned}$$

原始最优化问题的拉格朗日函数是:

$$L(\mathbf{w}, b, \xi, \alpha, \mu) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i (y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^N \mu_i \xi_i$$

$\alpha_i \geq 0, \mu_i \geq 0$  为拉格朗日乘子

# 支持向量机

对偶问题是Lagrange 函数的极大极小问题：

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b, \xi, \alpha, \mu) = \mathbf{w} - \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i = 0$$

$$\nabla_b L(\mathbf{w}, b, \xi, \alpha, \mu) = -\sum_{i=1}^N \alpha_i y_i = 0$$

$$\nabla_{\xi_i} L(\mathbf{w}, b, \xi, \alpha, \mu) = C - \alpha_i - \mu_i = 0$$



$$\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i$$

$$\sum_{i=1}^N \alpha_i y_i = 0$$

$$C - \alpha_i - \mu_i = 0$$

代入  $L(\mathbf{w}, b, \xi, \alpha, \mu)$ ，得：

$$\min_{\mathbf{w}, b, \xi} L(\mathbf{w}, b, \xi, \alpha, \mu) = \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i \xi_i - \sum_{i=1}^N \mu_i \xi_i - \sum_{i=1}^N \alpha_i y_i b + \sum_{i=1}^N \alpha_i$$

$$\min_{\mathbf{w}, b, \xi} L(\mathbf{w}, b, \xi, \alpha, \mu) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^N \alpha_i$$

再对  $\min_{\mathbf{w}, b, \xi} L(\mathbf{w}, b, \xi, \alpha, \mu)$  求  $\alpha$  的极大，即得到对偶问题：

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$s.t. \quad \sum_{i=1}^N \alpha_i y_i = 0$$

$$C - \alpha_i - \mu_i = 0$$

$$\alpha_i \geq 0$$

$$\mu_i \geq 0, \quad i = 1, 2, \dots, N$$

将对偶最优化问题进行变换：利用等式约束消去  $\mu_i$ ，从而只留下变量  $\alpha_i$ ，并将约束 (7.46)~(7.48) 写成:  $0 \leq \alpha_i \leq C$

# 支持向量机

SVM的对偶问题:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^N \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^N \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, N \end{aligned}$$

定理 7.3 设  $\alpha^* = (\alpha_1^*, \alpha_2^*, \dots, \alpha_N^*)^T$  是SVM对偶问题的一个解, 若存在  $\alpha^*$  的一个分量  $\alpha_j^*$ ,  $0 < \alpha_j^* < C$ , 则SVM原始问题的解  $w^*, b^*$  可按下式求得

$$\begin{aligned} \mathbf{w}^* &= \sum_{i=1}^N \alpha_i^* y_i \mathbf{x}_i \\ b^* &= y_j - \sum_{i=1}^N \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}_j) \end{aligned}$$

证明: KKT条件

# 支持向量机

由KKT条件得：

$$y_i \in \{-1, +1\}$$

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b, \xi, \alpha, \mu) = \mathbf{w}^* - \sum_{i=1}^N \alpha_i^* y_i \mathbf{x}_i^* = 0$$

$$\nabla_b L(\mathbf{w}, b, \xi, \alpha, \mu) = -\sum_{i=1}^N \alpha_i^* y_i = 0$$

$$\nabla_{\xi} L(\mathbf{w}, b, \xi, \alpha, \mu) = C - \alpha_i^* - \mu_i^*$$

$$\alpha_i^* (y_i (\mathbf{w}^* \cdot \mathbf{x}_i^* + b^*) - 1 + \xi_i^*) = 0$$

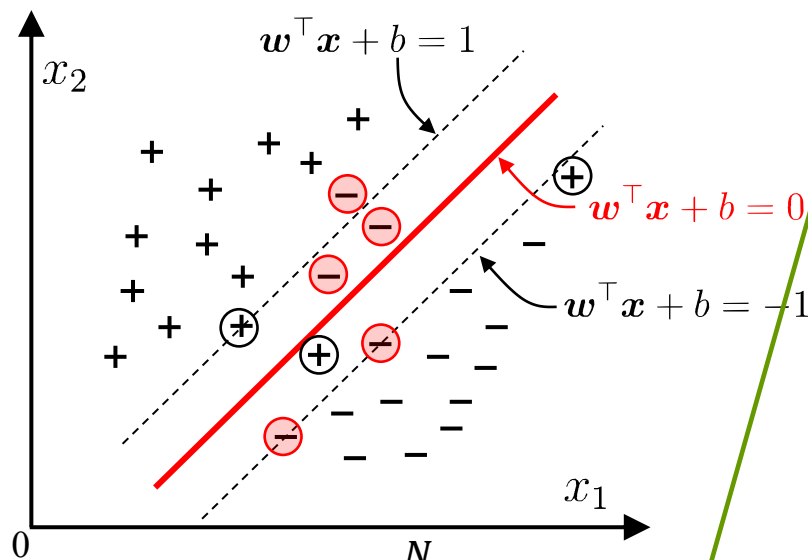
$$\mu_i^* \xi_i^* = 0$$

$$y_i (\mathbf{w}^* \cdot \mathbf{x}_i^* + b^*) - 1 + \xi_i^* \geq 0$$

$$\xi_i^* \geq 0$$

$$\alpha_i^* \geq 0$$

$$\mu_i^* \geq 0$$



$$\mathbf{w}^* = \sum_{i=1}^N y_i \alpha_i^* \mathbf{x}_i^*$$

其中至少有一个  $\alpha_j^* > 0$

反证法: 若  $\forall \alpha_j^* = 0$

$$\Rightarrow \mathbf{w}^* = 0, \forall \mu_j^* = C, \forall \xi_j^* = 0$$

$$\Rightarrow \forall (y_i b^* - 1 \geq 0)$$

$$\Rightarrow b^* \geq 1 \wedge b^* \leq -1 \text{ 矛盾}$$

当  $0 < \alpha_j^* < C$  时,  $\mu_j^* > 0, \xi_j^* = 0$ ,

同时  $y_j (\mathbf{w}^* \cdot \mathbf{x}_j^* + b^*) - 1 + \xi_j^* = 0$

样本刚好在间隔边界上. 支撑向量

当  $\alpha_j^* = 0$  时,  $\mu_j^* = C, \xi_j^* = 0$ ,

同时  $y_j (\mathbf{w}^* \cdot \mathbf{x}_j^* + b^*) - 1 + \xi_j^* \geq 0$

样本可能在间隔边界外, 也可能在边界上.

当  $\alpha_j^* = C$  时,  $\mu_j^* = 0, \xi_j^* \geq 0$ ,

同时  $y_j (\mathbf{w}^* \cdot \mathbf{x}_j^* + b^*) - 1 + \xi_j^* = 0$

样本在间隔边界内, 也可能被错分.

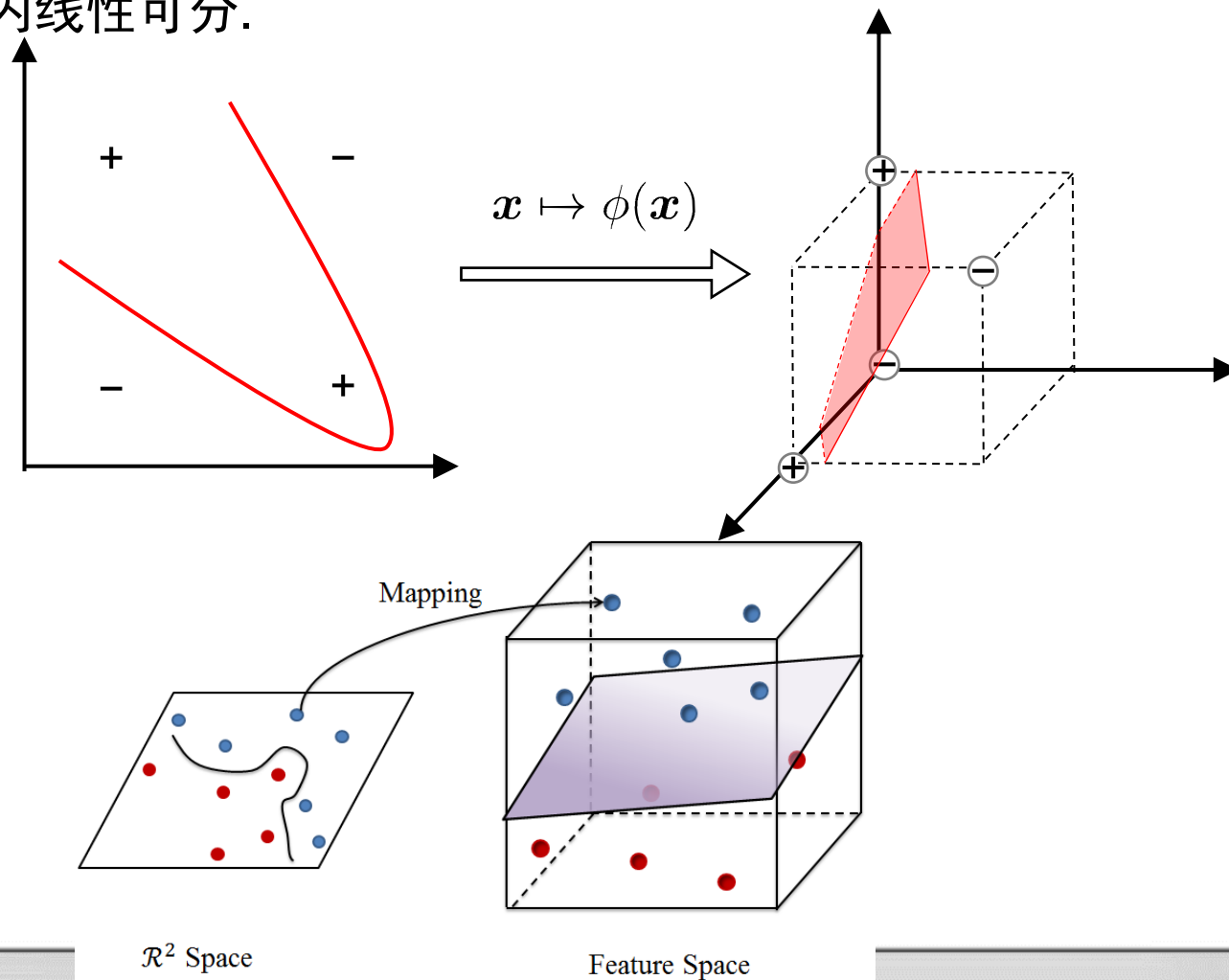
$$b^* = y_j - \mathbf{w}^* \cdot \mathbf{x}_j^* = y_j - \sum_{i=1}^N y_i \alpha_i^* (\mathbf{x}_i^* \cdot \mathbf{x}_j^*)$$

# 支持向量机

## ◆ 核支持向量机

-Q: 若不存在一个能正确划分两类样本的超平面, 怎么办? (线性不可分)

-A: 将样本从原始空间映射到一个更高维的特征空间, 使得样本在这个特征空间内线性可分.



将线性不可分样本从原始空间映射到一个更加高维的特征空间中去, 使得样本在这个特征空间中高概率线性可分。如果原始空间是有限维, 那么一定存在一个高维特征空间使样本可分 [Shawe-Taylor, J. 2004]。



# 支持向量机

## ◆ 核支持向量机

设样本特征  $\mathbf{x}$  映射后的向量为  $\phi(\mathbf{x})$ , 划分超平面为  $f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ :

原始问题  $\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2$

s.t.  $y_i(\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1, i = 1, 2, \dots, m.$

对偶问题  $\min_{\alpha} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \boxed{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)} - \sum_{i=1}^m \alpha_i$

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

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

只以内积的形式出现

# 支持向量机

## ◆ 核函数

基本想法：不显式地设计核映射  $\phi(\cdot)$ ，而是设计核函数：

$$\kappa(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

**定义 7.6 (核函数)** 设  $\mathcal{X}$  是输入空间 (欧氏空间  $\mathbf{R}^n$  的子集或离散集合)，又设  $\mathcal{H}$  为特征空间 (希尔伯特空间)，如果存在一个从  $\mathcal{X}$  到  $\mathcal{H}$  的映射

$$\phi(x) : \mathcal{X} \rightarrow \mathcal{H} \quad (7.65)$$

使得对所有  $x, z \in \mathcal{X}$ ，函数  $K(x, z)$  满足条件

$$K(x, z) = \phi(x) \cdot \phi(z) \quad (7.66)$$

则称  $K(x, z)$  为核函数， $\phi(x)$  为映射函数，式中  $\phi(x) \cdot \phi(z)$  为  $\phi(x)$  和  $\phi(z)$  的内积。

# 支持向量机

## ◆ 核函数

**定理 7.5 (正定核的充要条件)** 设  $K: \mathcal{X} \times \mathcal{X} \rightarrow \mathbf{R}$  是对称函数, 则  $K(x, z)$  为正定核函数的充要条件是对任意  $x_i \in \mathcal{X}$ ,  $i = 1, 2, \dots, m$ ,  $K(x, z)$  对应的 Gram 矩阵:

$$K = [K(x_i, x_j)]_{m \times m} \quad (7.85)$$

是半正定矩阵。

**定义 7.7 (正定核的等价定义)** 设  $\mathcal{X} \subset \mathbf{R}^n$ ,  $K(x, z)$  是定义在  $\mathcal{X} \times \mathcal{X}$  上的对称函数, 如果对任意  $x_i \in \mathcal{X}$ ,  $i = 1, 2, \dots, m$ ,  $K(x, z)$  对应的 Gram 矩阵

$$K = [K(x_i, x_j)]_{m \times m} \quad (7.87)$$

是半正定矩阵, 则称  $K(x, z)$  是正定核。

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_i, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_i, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_i, \mathbf{x}_m) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_m, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_m, \mathbf{x}_j) & \cdots & \kappa(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}$$

# 支持向量机

## ◆ 核函数

任何一个核函数都隐式地定义了一个称为“再生核希尔伯特空间”(Reproducing Kernel Hilbert Space, 简称RKHS) 的特征空间.

正定核(充分非必要): 只要一个对称函数所对应的Gram矩阵半正定, 则它就能作为核函数来使用.

常用核函数:

名称	表达式	参数
线性核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$	
多项式核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^d$	$d \geq 1$ 为多项式的次数
高斯核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ ^2}{2\delta^2}\right)$	$\delta > 0$ 为高斯核的带宽(width)
拉普拉斯核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ }{\delta}\right)$	$\delta > 0$
Sigmoid核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta \mathbf{x}_i^\top \mathbf{x}_j + \theta)$	$\tanh$ 为双曲正切函数, $\beta > 0, \theta < 0$

# 支持向量机

## ◆ 核函数

### ● 核函数的性质：

1. 对于任意正数  $\gamma_1, \gamma_2$ , 核函数的线性组合  $\gamma_1 \kappa_1 + \gamma_2 \kappa_2$  仍为核函数
2. 核函数的直积  $\kappa_1 \otimes \kappa_2(x, z) = \kappa_1(x, z) \kappa_2(x, z)$  仍为核函数
3. 设  $\kappa(x_1, x_2)$  为核函数, 则对于任意函数  $g$ ,  $g(x_1) \kappa(x_1, x_2) g(x_2)$  仍为核函数。

### ● 核函数选择成为SVM的最大变数

经验：文本数据使用线性核，情况不明使用高斯核。



# 支持向量机

## ◆ 核方法

支持向量机

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

支持向量回归

$$f(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) + b = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) \kappa(\mathbf{x}_i, \mathbf{x}) + b$$

结论: 无论是支持向量机还是支持向量回归, 学得模型总可以表示成核函数的线性组合.

更一般的结论 (表示定理 representer theorem): 令  $\mathbb{H}$  为核函数  $\kappa$  对应的再生希尔伯特空间 (RKHS), 对于任意单调增函数  $\Omega$  和任意非负损失函数  $L$ , 优化问题

$$\min_{h \in \mathbb{H}} F(h) = \Omega(\|h\|_{\mathbb{H}}) + L(h(\mathbf{x}_1), \dots, h(\mathbf{x}_m))$$

的解总可以写为  $h^*(\mathbf{x}) = \sum_{i=1}^m \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i)$ .

$h(\cdot)$  为原始特征空间到RKHS空间的映射.

# 支持向量机

## ◆ SMO算法

软间隔SVM的对偶问题：

二次规划问题

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^N \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^N \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1, 2, \dots, N \end{aligned}$$

Sequential minimal optimization (SMO) 算法 (启发式算法)

基本思路：

- 如果所有变量的解都满足此最优化问题的KKT条件，那么得到解；
- 否则，选择两个变量，固定其它变量，针对这两个变量构建一个二次规划问题，称为子问题，可通过解析方法求解，提高了计算速度。

SMO算法包括两个部分：

- 求解两个变量二次规划的解析方法
- 选择变量的启发式方法

# 支持向量机

## ◆ SMO算法

选择两个变量，其它固定，SMO最优化的子问题可以写成：

$$\min_{\alpha_1, \alpha_2} \quad W(\alpha_1, \alpha_2) = \frac{1}{2}K_{11}\alpha_1^2 + \frac{1}{2}K_{22}\alpha_2^2 + y_1y_2K_{12}\alpha_1\alpha_2 - (\alpha_1 + \alpha_2) + y_1\alpha_1 \sum_{i=3}^N y_i\alpha_i K_{i1} + y_2\alpha_2 \sum_{i=3}^N y_i\alpha_i K_{i2} \quad (7.101)$$

$$\text{s.t.} \quad \alpha_1 y_1 + \alpha_2 y_2 = - \sum_{i=3}^N y_i \alpha_i = \varsigma \quad (7.102)$$

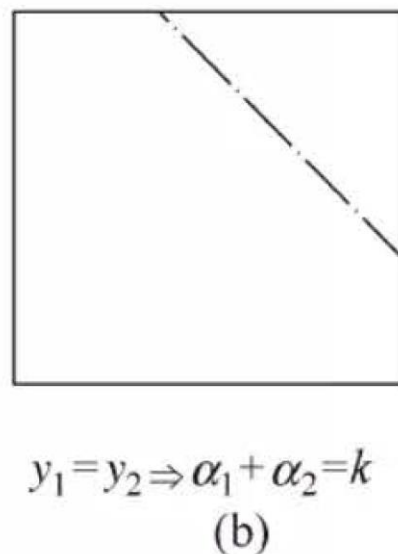
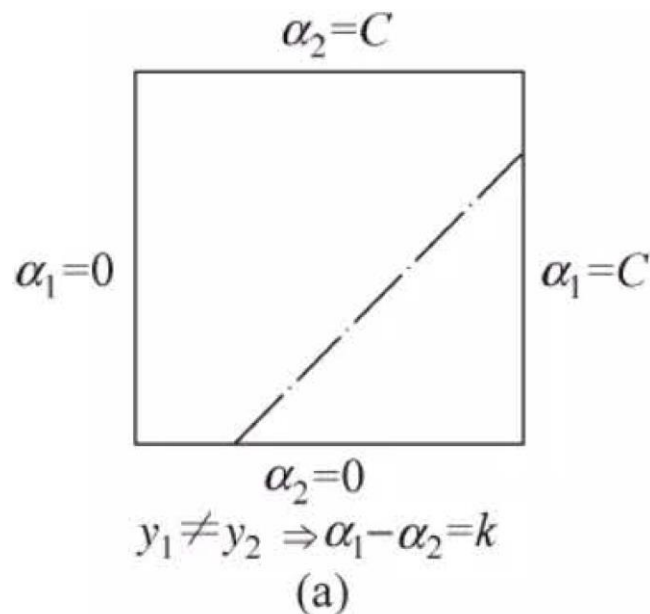
$$0 \leq \alpha_i \leq C, \quad i = 1, 2 \quad (7.103)$$

其中， $K_{ij} = \kappa(x_i, x_j)$ ,  $i, j = 1, 2, \dots, N$ ,  $\varsigma$  是常数，目标函数式 (7.101) 中省略了不含  $\alpha_1, \alpha_2$  的常数项。

两个变量的二次规划问题，可以通过解析方法求解。

# 支持向量机

## ◆ SMO算法



不等式约束 (7.103) 使得  $(\alpha_1, \alpha_2)$  在盒子  $[0, C] \times [0, C]$  内，等式约束 (7.102) 使  $(\alpha_1, \alpha_2)$  在平行于盒子  $[0, C] \times [0, C]$  的对角线的直线上，因此要求的是目标函数在一条平行于对角线的线段上的最优值。这使得两个变量的最优化问题成为实质上的单变量的最优化问题，不妨考虑为变量  $\alpha_2$  的最优化问题。

由于  $\alpha_2$  需满足不等式约束  $0 \leq \alpha_i \leq C$ ,  $i = 1, 2$  (7.103), 所以最优值  $\alpha_2^{\text{new}}$  的取值范围必须满足条件  $L \leq \alpha_2^{\text{new}} \leq H$ 。其中,  $L$  与  $H$  是  $\alpha_2^{\text{new}}$  所在的对角线段端点的界。

如果  $y_1 \neq y_2$ , 则  $L = \max(0, \alpha_2^{\text{old}} - \alpha_1^{\text{old}})$ ,  $H = \min(C, C + \alpha_2^{\text{old}} - \alpha_1^{\text{old}})$

如果  $y_1 = y_2$ , 则  $L = \max(0, \alpha_2^{\text{old}} + \alpha_1^{\text{old}} - C)$ ,  $H = \min(C, \alpha_2^{\text{old}} + \alpha_1^{\text{old}})$

# 支持向量机

## ◆ SMO算法

记  $g(\mathbf{x}) = \sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$

令  $E_i = g(\mathbf{x}_i) - y_i = (\sum_{j=1}^N \alpha_j y_j K(\mathbf{x}_j, \mathbf{x}_i) + b) - y_i, i = 1, 2$

当  $i = 1, 2$  时,  $E_i$  为函数  $g(\mathbf{x})$  对输入  $\mathbf{x}_i$  的预测值与真实输出  $y_i$  之差。

定理7.6 最优化问题 (7.101)~(7.103) 沿着约束方向未经剪辑时的解是

$$\alpha_2^{\text{new,unc}} = \alpha_2^{\text{old}} + \frac{y_2(E_1 - E_2)}{\eta} \quad (7.106)$$

$$\eta = K_{11} + K_{22} - 2K_{12} = \|\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2)\|^2 \quad (7.107)$$

$\Phi(\mathbf{x})$  是输入空间到特征空间的映射,  $E_i, i = 1, 2$ , 由式 (7.105) 给出。经剪辑后  $\alpha_2$  的解是

$$\alpha_2^{\text{new}} = \begin{cases} H, & \alpha_2^{\text{new,unc}} > H \\ \alpha_2^{\text{new,unc}}, & L \leq \alpha_2^{\text{new,unc}} \leq H \\ L, & \alpha_2^{\text{new,unc}} < L \end{cases} \quad (7.108)$$

$$\alpha_1^{\text{new}} = \alpha_1^{\text{old}} + y_1 y_2 (\alpha_2^{\text{old}} - \alpha_2^{\text{new}}) \quad (7.109)$$



# 支持向量机

## ◆ SMO算法

软间隔SVM的对偶问题：

二次规划问题

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^N \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^N \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1, 2, \dots, N \end{aligned}$$

Sequential minimal optimization (SMO) 算法

不断执行如下步骤直到收敛：

- ① 选取两个需要更新的变量  $\alpha_i$  和  $\alpha_j$ ;
- ② 固定除  $\alpha_i$  和  $\alpha_j$  之外的所有参数，简化约束得到

$$\alpha_i y_i + \alpha_j y_j = - \sum_{k \neq i, j}^N \alpha_k y_k$$

$$0 \leq \alpha_i, \alpha_j \leq C$$

- ③ 计算使得目标函数最小的  $\alpha_i$  和  $\alpha_j$  并更新

选取的  $\alpha_i$  和  $\alpha_j$  中有一个不满足KKT条件，目标函数就会在迭代后增大 [Osuna et al., 1997]. SMO 先选取违背KKT条件最大的变量；第二个变量应选取一个使得目标函数增长最快的变量，采用启发式：使得选取的两变量所对应的样本之间的间隔最大。

# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation Maximization\*

- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社. Ch 4.5
- ✓ 周志华, “机器学习”, 2016, 清华大学出版社. Ch 8.1, 8.2
- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社. Ch 8

# Boosting

From Adaptive Computation and Machine Learning

## Boosting

Foundations and Algorithms

By Robert E. Schapire and Yoav Freund

### Overview

*Boosting* is an approach to machine learning based on the idea of creating a highly accurate predictor by combining many weak and inaccurate “rules of thumb.” A remarkably rich theory has evolved around boosting, with connections to a range of topics, including statistics, game theory, convex optimization, and information geometry. Boosting algorithms have also enjoyed practical success in such fields as biology, vision, and speech processing. At various times in its history, boosting has been perceived as mysterious, controversial, even paradoxical.

This book, written by the inventors of the method, brings together, organizes, simplifies, and substantially extends two decades of research on boosting, presenting both theory and applications in a way that is accessible to readers from diverse backgrounds while also providing an authoritative reference for advanced researchers. With its introductory treatment of all material and its inclusion of exercises in every chapter, the book is appropriate for course use as well.

The book begins with a general introduction to machine learning algorithms and their analysis; then explores the core theory of boosting, especially its ability to generalize; examines some of the myriad other theoretical viewpoints that help to explain and understand boosting; provides practical extensions of boosting for more complex learning problems; and finally presents a number of advanced theoretical topics. Numerous applications and practical illustrations are offered throughout.

- 对于一个复杂的分类任务，可以将其分解为若干子任务，然后将若干子任务完成方法综合，最终完成该复杂任务。
- 将若干个弱分类器(weak classifiers)组合起来，形成一个强分类器(strong classifier)。
- 能用众力，则无敌于天下矣；**能用众智，则无畏于圣人矣**(语出《三国志·吴志·孙权传》)

Freund, Yoav; Schapire, Robert E (1997), A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences (original paper of Yoav Freund and Robert E.Schapire where AdaBoost is first introduced.)

# Boosting

## ◆ Hoeffding's inequality (霍夫丁不等式)

学习任务：统计某个电视节目在全国的收视率。

方法：不可能去统计整个国家中每个人是否观看电视节目、进而算出收视率。

只能抽样一部分人口，然后将抽样人口中观看该电视节目的比例作为该电视节目的全国收视率。

霍夫丁不等式：全国人口中看该电视节目的人口比例（记作 $x$ ）与抽样人口中观看该电视节目的人口比例（记作 $y$ ）满足如下关系：

$P(|x - y| \geq \epsilon) \leq 2e^{-2N\epsilon^2}$  ( $N$ 是采样人口总数、 $\epsilon \in (0,1)$ 是所设定的可容忍误差范围)

当 $N$ 足够大时，“全国人口中电视节目收视率”与“样本人口中电视节目收视率”差值超过误差范围 $\epsilon$ 的概率非常小。

# Boosting

- 对于统计电视节目收视率这样的任务，可以通过不同的采样方法 (即不同模型) 来计算收视率。每个模型会产生不同的误差。
- 问题：如果得到完成该任务的若干“弱模型”，是否可以将这些弱模型组合起来，形成一个“强模型”。该“强模型”产生误差很小？这就是概率近似正确 (probably approximately correct, PAC) 要回答的问题。

在概率近似正确背景下，有“强可学习模型”和“弱可学习模型”

强可学习 (strongly learnable)	学习模型能够以较高精度对绝大多数样本完成识别分类任务
弱可学习 (weakly learnable)	学习模型仅能完成若干部分样本识别与分类，其精度略高于随机猜测。(不可学习指的是学习模型所获得精度仅为50%，即相当于随机猜测)
强可学习和弱可学习是等价的，也就是说，如果已经发现了“弱学习算法”，可将其提升 (boosting) 为“强学习算法”。Ada Boosting算法就是这样的方法。具体而言，Ada Boosting将一系列弱分类器组合起来，构成一个强分类器。	

# Ada Boost 思路描述

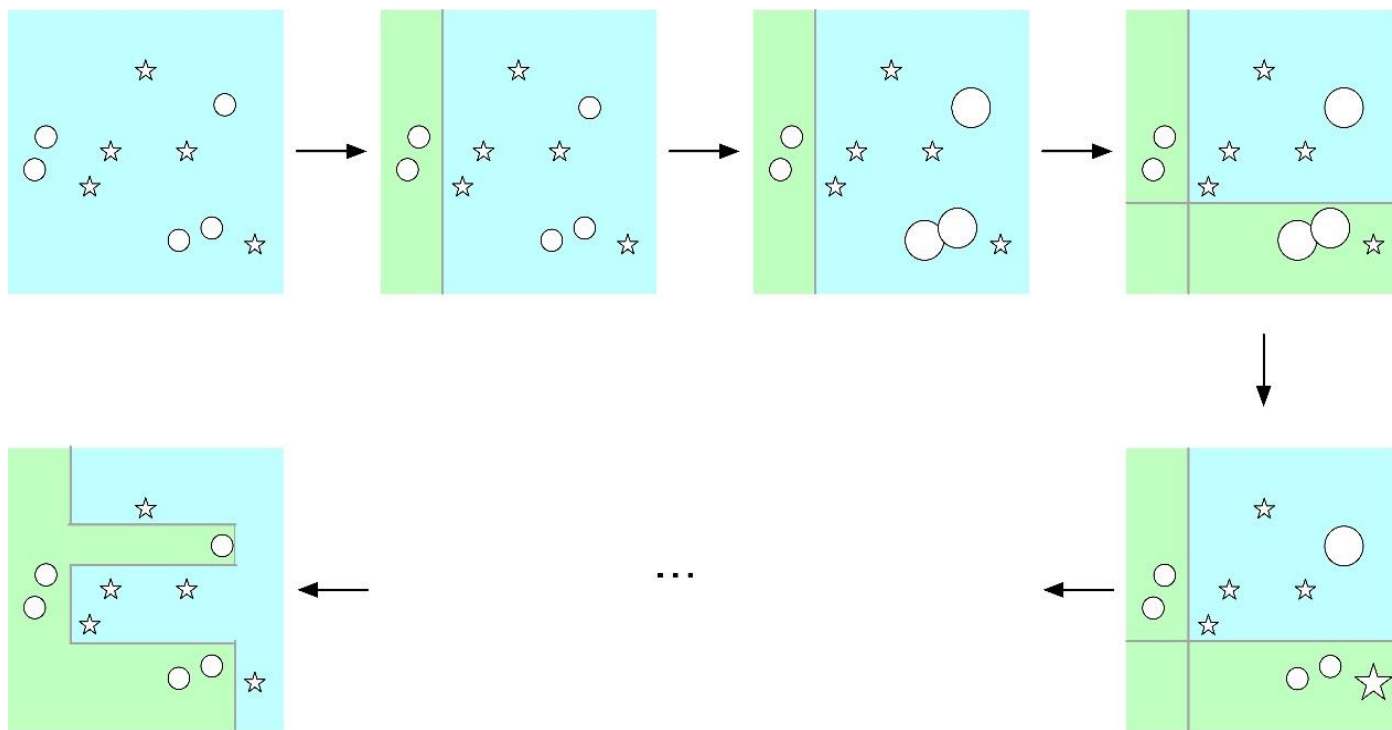


图4.9  
Ada Boosting算法  
学习过程示意图

Ada Boosting算法中两个核心问题：

- 在每个弱分类器学习过程中，如何改变训练数据的权重：提高在上一轮中分类错误样本的权重。
- 如何将一系列弱分类器组合成强分类器：通过加权多数表决方法来提高分类误差小的弱分类器的权重，让其在最终分类中起到更大作用。同时减少分类误差大的弱分类器的权重，让其在最终分类中仅起到较小作用。



# Ada Boost 算法描述

给定包含 $N$ 个标注数据的训练集合  $\Gamma = \{(x_1, y_1), \dots, (x_N, y_N)\}$ ,  
 $x_i (1 \leq i \leq N) \in X \subseteq \mathbb{R}^n$ ,  $y_i \in Y = \{-1, 1\}$

Ada Boosting 算法根据标注数据，训练得到一系列弱分类器，并将这些弱分类器线性组合得到一个强分类器。

1. 初始化每个训练样本的权重:

$$D_1 = (w_{11}, \dots, w_{1i}, \dots, w_{1N}), \text{ 其中 } w_{1i} = \frac{1}{N} (1 \leq i \leq N)$$

2. 对  $m = 1, 2, \dots, M$ :

a) 使用具有分布权重  $D_m$  的训练数据来学习得到第  $m$  个基分类器（弱分类器）  
 $G_m: G_m(x): X \rightarrow \{-1, 1\}$ .

b) 计算  $G_m(x)$  在训练数据集上的分类误差  $err_m = \sum_{i=1}^N w_{mi} \mathbb{I}(G_m(x_i) \neq y_i)$ ,  
这里:  $\mathbb{I}(\cdot) = 1$ , 如果  $G_m(x_i) \neq y_i$ ; 否则为 0.

c) 计算弱分类器  $G_m(x)$  的权重:  $\alpha_m = \frac{1}{2} \ln \frac{1 - err_m}{err_m}$ .

d) 更新训练样本数据的分布权重:  $D_{m+1} = w_{m+1,i} = \frac{w_{m,i}}{Z_m} e^{-\alpha_m y_i G_m(x_i)}$ , 其中  
 $Z_m$  是归一化因子以使得  $D_{m+1}$  为概率分布,  $Z_m = \sum_{i=1}^N w_{m,i} e^{-\alpha_m y_i G_m(x_i)}$ .

3. 以线性加权形式来组合弱分类器:  $f(x) = \sum_{i=1}^M \alpha_m G_m(x)$

得到强分类器  $G(x) = \text{sign}(f(x)) = \text{sign}(\sum_{i=1}^M \alpha_m G_m(x))$

# Ada Boost 算法描述

第 $m$ 个弱分类器 $G_m(x)$ 在训练数据集上产生的分类误差：

该误差为被错误分类的样本所具有权重的累加

$$err_m = \sum_{i=1}^N w_{m,i} \mathbb{I}(G_m(x_i) \neq y_i) \quad \text{这里：} \mathbb{I}(\cdot) = 1, \text{ 如果 } G_m(x_i) \neq y_i; \text{ 否则为 } 0$$

计算第 $m$ 个弱分类器 $G_m(x)$ 的权重 $\alpha_m$ ：
$$\alpha_m = \frac{1}{2} \ln \frac{1 - err_m}{err_m}$$

(a) 当第 $m$ 个弱分类器 $G_m(x)$  错误率为1，即  $err_m = \sum_{i=1}^N w_{m,i} \mathbb{I}(G_m(x_i) \neq y_i) = 1$ ，意味每个样本分类出错，则  $\alpha_m = \frac{1}{2} \ln \frac{1 - err_m}{err_m} \rightarrow -\infty$ ，给予第 $m$ 个弱分类器  $G_m(x)$  很低权重。

(b) 当第 $m$ 个弱分类器 $G_m(x)$ 错误率为 $\frac{1}{2}$ ，  $\alpha_m = \frac{1}{2} \ln \frac{1 - err_m}{err_m} = 0$ 。如果错误率 $err_m$  小于 $\frac{1}{2}$ ，权重 $\alpha_m$ 为正( $err_m < \frac{1}{2}$ 、 $\alpha_m > 0$ )。可知权重 $\alpha_m$ 随 $err_m$ 减少而增大，即错误率越小的弱分类器会赋予更大权重。

(c) 如果一个弱分类器的分类错误率为 $\frac{1}{2}$ ，可视为其性能仅相当于随机分类效果。

# Ada Boost 算法解释

在开始训练第 $m + 1$ 个弱分类器 $G_{m+1}(x)$ 之前对训练数据集中数据权重进行调整

$$w_{m+1,i} = \begin{cases} \frac{w_{m,i}}{Z_m} e^{-\alpha_m}, & G_m(x_i) = y_i \\ \frac{w_{m,i}}{Z_m} e^{\alpha_m}, & G_m(x_i) \neq y_i \end{cases}$$

- 可见，如果某个样本无法被第 $m$ 个弱分类器 $G_m(x)$ 分类成功，则需要增大该样本权重，否则减少该样本权重。这样，被错误分类样本会在训练第 $m + 1$ 个弱分类器 $G_{m+1}(x)$ 时会被“重点关注”。
- 在每一轮学习过程中，Ada Boosting算法均在划重点（重视当前尚未被正确分类的样本）

**弱分类器构造强分类器**  $G(x) = \text{sign}(f(x)) = \text{sign}(\sum_{i=1}^M \alpha_m G_m(x))$

- $f(x)$ 是 $M$ 个弱分类器的加权线性累加。分类能力越强的弱分类器具有更大权重。
- $\alpha_m$ 累加之和并不等于1。
- $f(x)$ 符号决定样本 $x$ 分类为1或-1。如果 $\sum_{i=1}^M \alpha_m G_m(x)$ 为正，则强分类器 $G(x)$ 将样本 $x$ 分类为1；否则为-1。

# Ada Boost 算法解释

## ● 回看霍夫丁不等式

假设有 $M$ 个弱分类器 $G_m (1 \leq m \leq M)$ ，则 $M$ 个弱分类器线性组合所产生误差满足如下条件：

$$P\left(\sum_{i=1}^M G_m(x) \neq \zeta(x)\right) \leq e^{-\frac{1}{2}M(1-2\epsilon)^2}$$

- $\zeta(x)$  是真实分类函数,  $\epsilon \in (0,1)$ 。上式表明，如果所“组合”弱分类器越多，则学习分类误差呈指数级下降，直至为零。
- 上述不等式成立有两个前提条件：1) 每个弱分类器产生的误差相互独立； 2) 每个弱分类器的误差率小于50%。因为每个弱分类器均是在同一个训练集上产生，条件1) 难以满足。也就是说，“准确性（对分类结果而言）”和“差异性（对每个弱分类器而言）”难以同时满足。
- Ada Boosting 采取了序列化学习机制。

# Ada Boost 算法解释

## ● 优化目标

Ada Boost实际在最小化如下指数损失函数 (minimization of exponential loss):

$$\sum_i e^{-y_i f(x_i)} = \sum_i e^{-y_i \sum_{m=1}^M \alpha_m G_m(x_i)}$$

Ada Boost的分类误差上界如下所示:

$$\frac{1}{N} \sum_{i=1}^N I(G(x_i) \neq y_i) \leq \frac{1}{N} \sum_i e^{-y_i f(x_i)} = \prod_m Z_m$$

在第 $m$ 次迭代中, Ada Boosting 总是趋向于将具有最小误差的学习模型选做本轮生成的弱分类器  $G_m$ , 使得累积误差快速下降。

# 主要内容

1. 机器学习基本概念

2. 计算学习理论

3. Support vector machine

4. Ada Boost\*

5. 生成式方法\*

6. 主成分分析\*

7. Expectation maximization\*

- ✓ 李航, “统计学习方法” (第2版), 2019, 清华大学出版社. Ch 1.7
- ✓ 吴飞编著, “人工智能导论: 模型与算法”, 2020, 高等教育出版社.
- ✓ 周志华, “机器学习”, 2016, 清华大学出版社.
- ✓ 周志华等, “机器学习理论导引”, 2020, 机械工业出版社.



# 生成式方法与判别式方法

监督学习方法又可以分为生成式方法 (generative approach) 和判别式方法 (discriminative approach)。所学到的模型分别称为生成模型 (generative model) 和判别模型 (discriminative model)。

- 判别式方法直接学习判别函数  $f(X)$  或者条件概率分布  $P(Y|X)$  作为预测的模型，即判别模型。
- 判别模型关心在给定输入数据下，预测该数据的输出是什么。
- 典型判别模型包括决策树、支持向量机、Ada boosting 和神经网络分类回归等。

$$f(\text{人脸}) \longrightarrow \text{人脸}$$

$$P(\text{人脸} | \text{人脸}) = 0.99$$

# 生成式方法与判别式方法

生成学习方法从数据中学习联合概率分布 $P(X, C)$ ，然后求出条件概率分布 $P(C|X)$ 作为预测模型，即 $P(c_i|x) = \frac{P(x, c_i)}{P(x)}$

$$\underbrace{P(x, c_i)}_{\text{联合概率}} = \underbrace{P(x|c_i)}_{\text{似然概率}} \times \underbrace{P(c_i)}_{\text{先验概率}} \quad \longrightarrow \quad \underbrace{P(c_i|x)}_{\text{后验概率}} = \frac{P(x, c_i)}{P(x)} = \frac{\underbrace{P(x|c_i)}_{\text{似然概率}} \times \underbrace{P(c_i)}_{\text{先验概率}}}{P(x)}$$

对每个类别，每个输入数据 $x$ 的概率 $P(x)$ 是一样的，可以去掉。  
似然概率编码了输入数据被类别数据产生的关系。

- 生成模型从数据中学习联合概率分布 $P(X, Y)$ （通过似然概率 $P(X|Y)$ 和类概率 $P(Y)$ 的乘积来求取）

$$P(Y|X) = \frac{P(X, Y)}{P(X)} \quad \text{或者} \quad P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)}$$

- 联合分布概率 $P(X, Y)$ 或似然概率 $P(X|Y)$ 求取很困难
- 常见的生成学习模型有贝叶斯方法、隐马尔可夫模型、隐狄利克雷分布 (latent dirichlet allocation, LDA)等。

似然概率：计算  
导致样本 $X$ 出现  
的模型参数值

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)}$$