

机器学习：监督学习

教学课程组

2021年

- 参考教材：吴飞，《人工智能导论：模型与算法》，高等教育出版社
- 在线课程(MOOC)：<https://www.icourse163.org/course/ZJU-1003377027>
- 在线实训平台（智海-Mo）：https://mo.zju.edu.cn/classroom/class/zju_ai_2021
- 在线共享资源（智海在线）：<http://www.wiscean.cn/online/intro/zju-01>

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

机器学习：从数据中学习知识



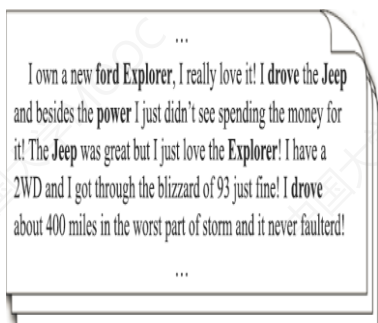
图像数据

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

- Person
- Dog
- ...

类别分类

- 从原始数据中提取特征
- 学习映射函数 f
- 通过映射函数 f 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



文本数据

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

- 喜悦
- 愤怒
- ...

情感分类

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(un-supervised learning)

数据无标签、一般为聚类或若干降维任务

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)

机器学习：分类问题

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

映射函数

模式

f

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

→ 程序员

类别

监督学习的重要元素

标注数据

- 标识了类别信息的数据
学什么

学习模型

- 如何学习得到映射模型
如何学

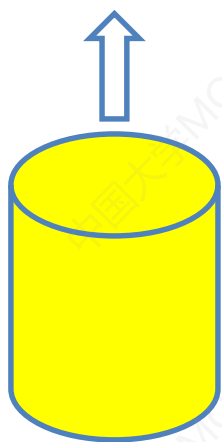
损失函数

- 如何对学习结果进行度量
学到否

监督学习：损失函数

训练映射函数 f

使得 $f(x_i)$ 预测结果尽量等于 y_i



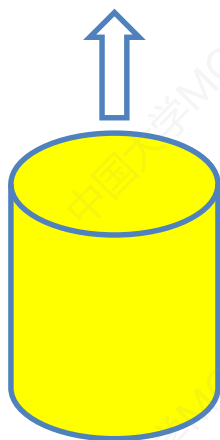
训练数据集
 $(x_i, y_i), i = 1, \dots, n$

- 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 $\min \sum_{i=1}^n \text{Loss}(f(x_i), y_i)$ 。

监督学习：损失函数

训练映射函数 f

使得 $f(x_i)$ 预测结果尽量等于 y_i



训练数据集

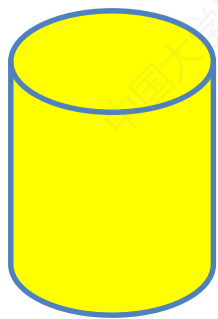
$(x_i, y_i), i = 1, \dots, n$

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

典型的损失函数

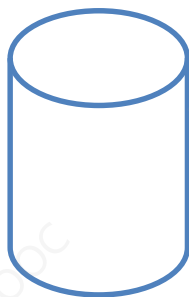
监督学习：训练数据与测试数据

从训练数据集学习
得到映射函数 f



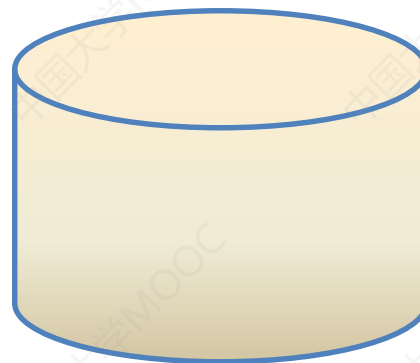
训练数据集
 $(x_i, y_i), i = 1, \dots, n$

在测试数据集
测试映射函数 f



测试数据集
 $(x_i', y_i'), i = 1, \dots, m$

未知数据集
上测试映射函数 f



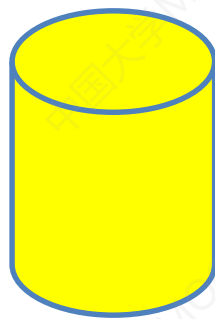
监督学习：经验风险与期望风险

从训练数据集学习得到映射函数 f

在测试数据集测试映射函数 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。



训练数据集
 $(x_i, y_i), i = 1, \dots, n$



测试数据集
 $(x'_i, y'_i), i = 1, \dots, m$

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。

监督学习：经验风险与期望风险

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。

监督学习：“过学习(over-fitting)”与“欠学习(under-fitting)”

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

监督学习: 结构风险最小

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(regulatizer)或惩罚项(penalty term):

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

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

监督学习两种方法：判别模型与生成模型

监督学习方法又可以分为生成方法 (generative approach) 和判别方法 (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。
- 典型判别模型包括回归模型、神经网络、支持向量机和Ada boosting等。

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

$$P(\text{人脸} | \text{人脸}) = 0.99$$

监督学习两种方法：判别模型与生成模型

- 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）

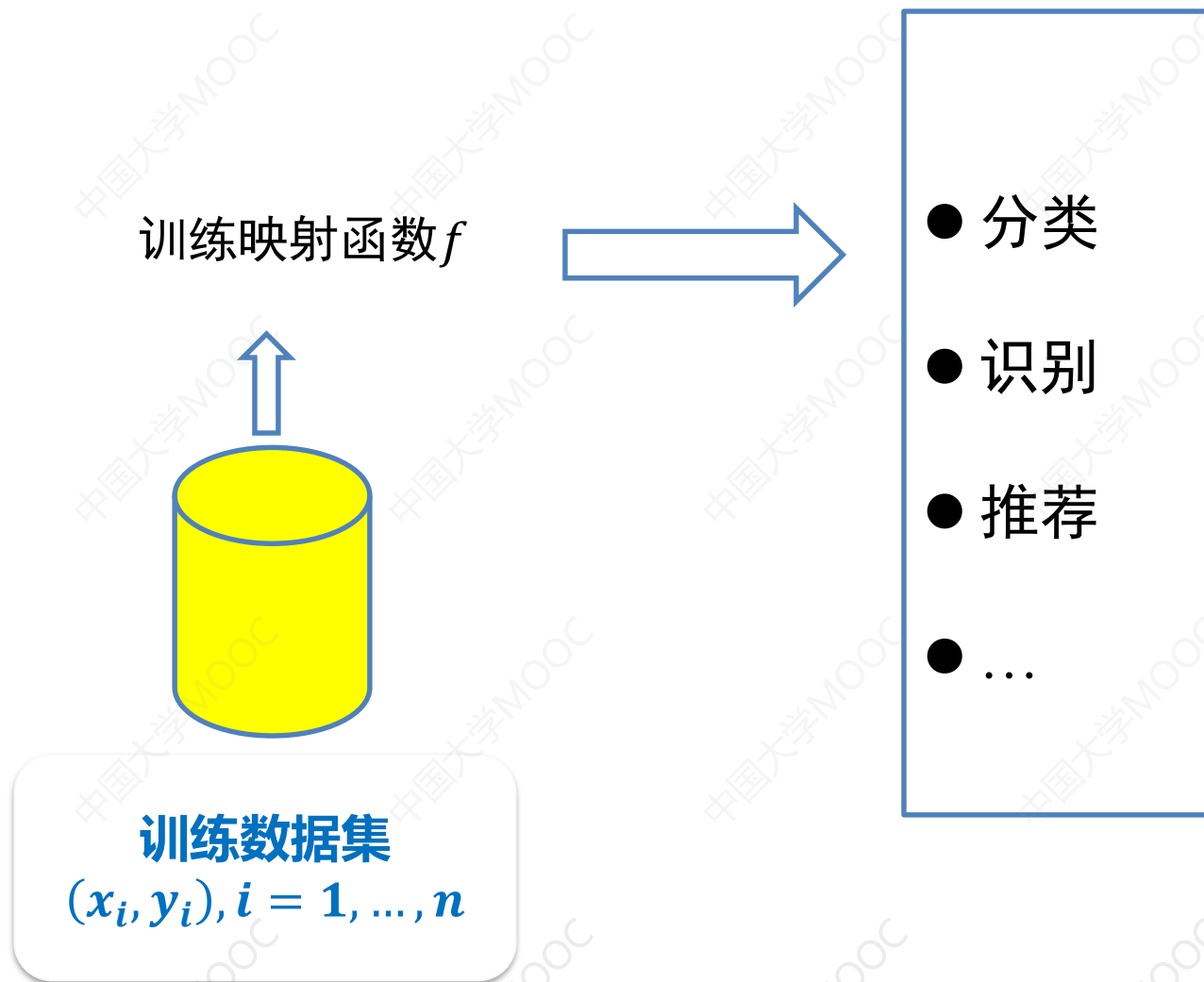
$$P(Y|X) = \frac{P(X, Y)}{P(X)} \text{ 或者 } P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)}$$

- 典型方法为贝叶斯方法、隐马尔可夫链
- 授之于鱼、不如授之于“渔”
- 联合分布概率 $P(X, Y)$ 或似然概率 $P(X|Y)$ 求取很困难

似然概率：计算
导致样本 X 出现的
模型参数值

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

监督学习



提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

线性回归 (linear regression)

- 在现实生活中，往往需要分析若干变量之间的关系，如碳排放量与气候变暖之间的关系、某一商品广告投入量与该商品销售量之间的关系等，这种分析不同变量之间存在关系的研究叫回归分析，刻画不同变量之间关系的模型被称为回归模型。如果这个模型是线性的，则称为线性回归模型。
- 一旦确定了回归模型，就可以进行预测等分析工作，如从碳排放量预测气候变化程度、从广告投入量预测商品销售量等。

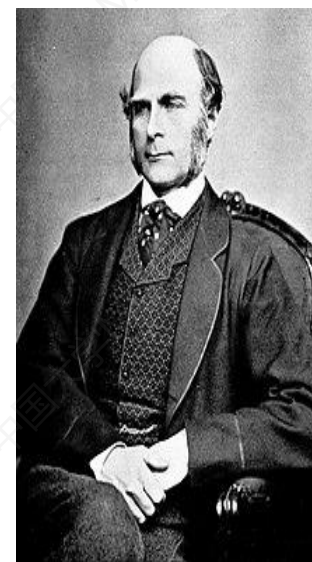
线性回归 (linear regression)

$$y = 33.73(\text{英寸}) + 0.516x$$

y : 子女平均身高

x : 父母平均身高

- 父母平均身高每增加一个单位，其成年子女平均身高只增加0.516个单位，它反映了这种“衰退 (regression)”效应（“回归”到正常人平均身高）。
- 虽然 x 和 y 之间并不总是具有“衰退”（回归）关系，但是“线性回归”这一名称就保留下来了。



英国著名生物学家兼
统计学家高尔顿
Sir Francis Galton
(1822-1911)

线性回归 (linear regression)

该回归模型中两个参数

← 需要从标注数据
中学习得到
(监督学习)

$$y = 33.73(\text{英寸}) + 0.516x$$

y : 子女平均身高

x : 父母平均身高

- 给出任意一对父母平均身高，则可根据上述方程，计算得到其子女平均身高
- 从父母平均身高来预测其子女平均身高
- 如何求取上述线性方程（预测方程）的参数？

线性回归：一元线性回归

一元线性回归模型例子

下表给出了芒提兹尼欧（Montesinho）地区发生森林火灾的部分历史数据，表中列举了每次发生森林火灾时的气温温度取值 x 和受到火灾影响的森林面积 y 。

气温温度 x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
火灾影响面积 y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

可否对气温温度与火灾所影响的森林面积之间关系进行建模呢？初步观察之后，可以使用简单的线性模型构建两者之间关系，即气温温度 x 与火灾所影响的森林面积 y 之间存在 $y = ax + b$ 形式的关系。

线性回归：一元线性回归

一元线性回归模型例子

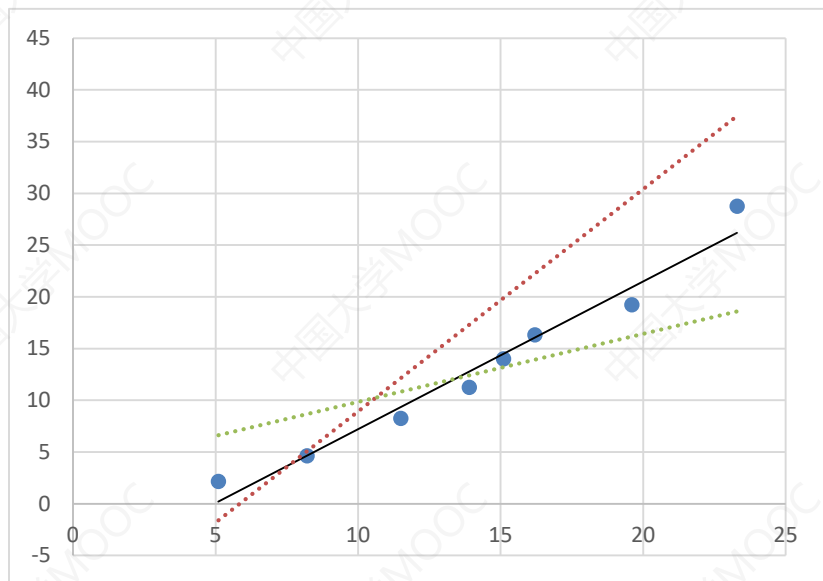


图4.2 气温温度取值和受到火灾影响森林面积之间的一元线性回归模型（实线为最佳回归模型）

$$\text{回归模型: } y = ax + b$$

求取：最佳回归模型是最小化残差平方和的均值，即要求8组 (x, y) 数据得到的残差平均值 $\frac{1}{N} \sum (y - \tilde{y})^2$ 最小。残差平均值最小只与参数 a 和 b 有关，最优解即是使得残差最小所对应的 a 和 b 的值。

线性回归：一元线性回归

回归模型参数求取： $y_i = ax_i + b$ ($1 \leq i \leq n$)

- 记在当前参数下第 i 个训练样本 x_i 的预测值为 \hat{y}_i
- x_i 的标注值（实际值） y_i 与预测值 \hat{y}_i 之差记为 $(y_i - \hat{y}_i)^2$
- 训练集中 n 个样本所产生误差总和为： $L(a, b) = \sum_{i=1}^n (y_i - a \times x_i - b)^2$

目标：寻找一组 a 和 b ，使得误差总和 $L(a, b)$ 值最小。在线性回归中，解决如此目标的方法叫最小二乘法。

一般而言，要使函数具有最小值，可对 $L(a, b)$ 参数 a 和 b 分别求导，令其导数值为零，再求取参数 a 和 b 的取值。

线性回归：一元线性回归

回归模型参数求取： $y_i = ax_i + b$ ($1 \leq i \leq n$) $\min_{a,b} L(a,b) = \sum_{i=1}^n (y_i - a \times x_i - b)^2$

$$\frac{\partial L(a,b)}{\partial a} = \sum_{i=1}^n 2(y_i - ax_i - b)(-x_i) = 0$$

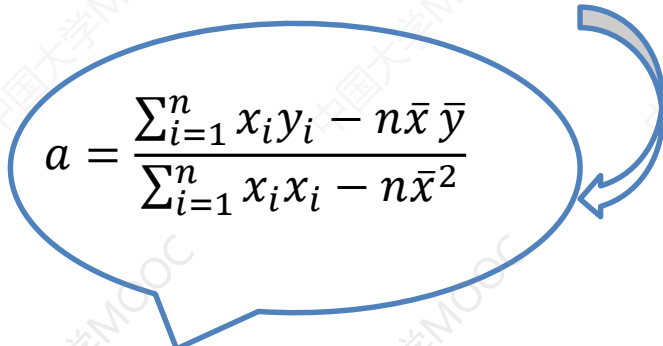
将 $b = \bar{y} - a\bar{x}$ ($\bar{y} = \frac{\sum_{i=1}^n y_i}{n}$, $\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$) 代入上式

$$\rightarrow \sum_{i=1}^n (y_i - ax_i - \bar{y} + a\bar{x})(x_i) = 0$$

$$\rightarrow \sum_{i=1}^n (y_i x_i - ax_i x_i - \bar{y} x_i + a\bar{x} x_i) = 0$$

$$\rightarrow \sum_{i=1}^n (y_i x_i - \bar{y} x_i) - a \sum_{i=1}^n (x_i x_i - \bar{x} x_i) = 0$$

$$\rightarrow \left(\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y} \right) - a \left(\sum_{i=1}^n x_i x_i - n\bar{x}^2 \right) = 0$$


$$a = \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^n x_i x_i - n\bar{x}^2}$$

线性回归：一元线性回归

回归模型参数求取： $y_i = ax_i + b$ ($1 \leq i \leq n$) $\min_{a,b} L(a,b) = \sum_{i=1}^n (y_i - a \times x_i - b)^2$

$$\frac{\partial L(a,b)}{\partial b} = \sum_{i=1}^n 2(y_i - ax_i - b)(-1) = 0$$

$$\rightarrow \sum_{i=1}^n (y_i - ax_i - b) = 0$$

$$\rightarrow \sum_{i=1}^n (y_i) - a \sum_{i=1}^n x_i - \sum_{i=1}^n b = 0$$

$$\rightarrow n\bar{y} - an\bar{x} - nb = 0$$



$$b = \bar{y} - a\bar{x}$$

$$a = \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^n x_i^2 - n\bar{x}^2}$$

可以看出：只要给出了训练样本 (x_i, y_i) ($i = 1, \dots, n$)，我们就可以从训练样本出发，建立一个线性回归方程，使得对训练样本数据而言，该线性回归方程预测的结果与样本标注结果之间的差值和最小。

线性回归：一元线性回归

回归模型参数求取： $y_i = ax_i + b$ ($1 \leq i \leq n$) $\min_{a,b} L(a, b) = \sum_{i=1}^n (y_i - a \times x_i - b)^2$

$$b = \bar{y} - a\bar{x}$$

$$a = \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^n x_i^2 - n\bar{x}^2}$$

$$a = \frac{x_1 y_1 + x_2 y_2 + \cdots + x_8 y_8 - 8\bar{x}\bar{y}}{x_1^2 + x_2^2 + \cdots + x_8^2 - 8\bar{x}^2} = 1.428$$
$$b = \bar{y} - a\bar{x} = -7.09$$

即预测芒提兹尼欧地区火灾所影响森林面积与气温温度之间的一元线性回归模型为“火灾所影响的森林面积 = $1.428 \times$ 气温温度 $- 7.09$ ”，即

$$y = 1.428x - 7.09$$

线性回归：多元线性回归

多元线性回归模型例子

接下来扩展到数据特征的维度是多维的情况，在上述数据中增加一个影响火灾影响面积的潜在因素—风力。

气温 x	5.1	8.2	11.5	13.9	15.1	16.2	19.6	23.3
风力 z	4.5	5.8	4	6.3	4	7.2	6.3	8.5
火灾影响面积 y	2.14	4.62	8.24	11.24	13.99	16.33	19.23	28.74

多维数据特征中线性回归的问题定义如下：假设总共有 m 个训练数据 $\{(x_i, y_i)\}_{i=1}^m$ ，其中 $x_i = [x_{i,1}, x_{i,2}, \dots, x_{i,D}] \in \mathbb{R}^D$ ， D 为数据特征的维度，线性回归就是要找到一组参数 $a = [a_0, a_1, \dots, a_D]$ ，使得线性函数：

$$f(x_i) = a_0 + \sum_{j=1}^D a_j x_{i,j} = a_0 + \mathbf{a}^T \mathbf{x}_i$$

线性回归：多元线性回归

最小化均方误差函数：

$$J_m = \frac{1}{m} \sum_{i=1}^m (y_i - f(x_i))^2$$

为了方便，使用矩阵来表示所有的训练数据和数据标签。

$$X = [x_1, \dots, x_m], \quad \mathbf{y} = [y_1, \dots, y_m]$$

其中每一个数据 x_i 会扩展一个维度，其值为1，对应参数 a_0 。均方误差函数可以表示为：

$$J_m(\mathbf{a}) = (\mathbf{y} - X^T \mathbf{a})^T (\mathbf{y} - X^T \mathbf{a})$$

均方误差函数 $J_n(\mathbf{a})$ 对所有参数 \mathbf{a} 求导可得：

$$\nabla J(\mathbf{a}) = -2X(\mathbf{y} - X^T \mathbf{a})$$

因为均方误差函数 $J_n(\mathbf{a})$ 是一个二次的凸函数，所以函数只存在一个极小值点，同样是最小值点，所以令 $\nabla J(\mathbf{a}) = 0$ 可得

$$XX^T \mathbf{a} = X\mathbf{y}$$

$$\mathbf{a} = (XX^T)^{-1} X\mathbf{y}$$

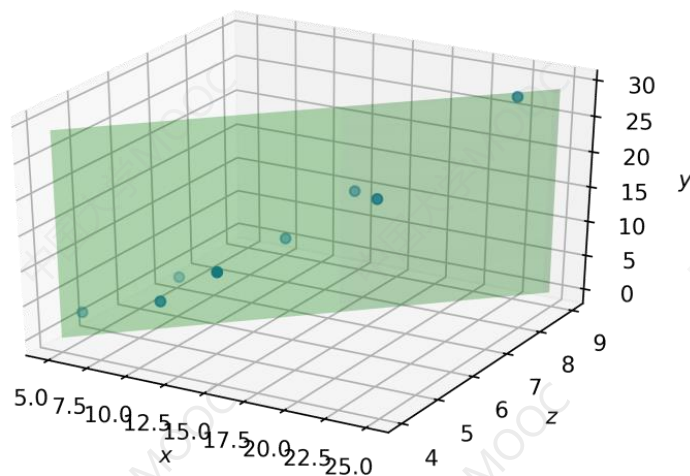
线性回归：多元线性回归

对于上面的例子，转化为矩阵的表示形式为：

$$X = \begin{bmatrix} 5.1 & 8.2 & 11.5 & 13.9 & 15.1 & 16.2 & 19.6 & 23.3 \\ 4.5 & 5.8 & 4. & 6.3 & 4. & 7.2 & 6.3 & 8.5 \\ 1. & 1. & 1. & 1. & 1. & 1. & 1. & 1. \end{bmatrix}$$
$$\mathbf{y} = [2.14 \quad 4.62 \quad 8.24 \quad 11.24 \quad 13.99 \quad 16.33 \quad 19.23 \quad 28.74]^T$$

其中矩阵 X 多出一行全1，是因为常数项 a_0 ，可以看作是数值为全1的特征的对应系数。计算可得

$$\mathbf{a} = [1.312 \quad 0.626 \quad -9.103]$$
$$\mathbf{y} = -9.103 + 1.312x + 0.626z$$



线性回归：逻辑斯蒂回归/对数几率回归

逻辑斯蒂回归/对数几率回归模型例子

线性回归一个明显的问题是对离群点（和大多数数据点距离较远的数据点，outlier）非常敏感，导致模型建模不稳定，使结果有偏，为了缓解这个问题（特别是在二分类场景中）带来的影响，可考虑逻辑斯蒂回归(logistic regression)[Cox 1958]。

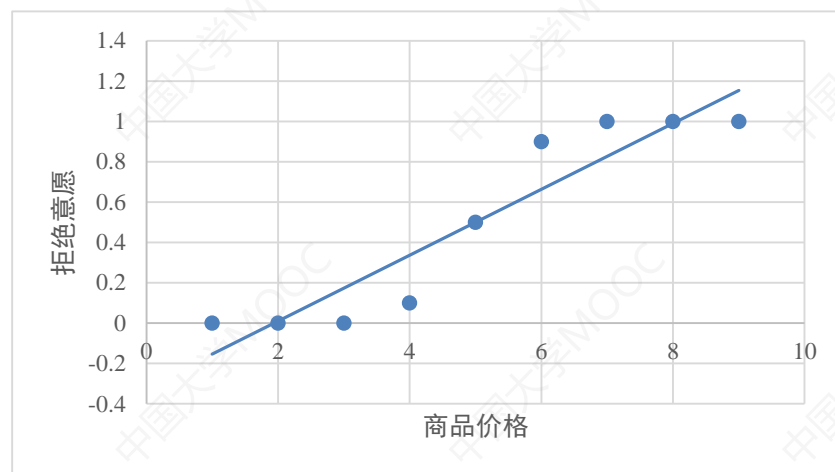


图4.4 用户对某件商品拒绝购买意愿（选择不购买商品的人数/受调查的总人数）与商品价格之间的关系

线性回归：逻辑斯蒂回归/对数几率回归

逻辑斯蒂回归/对数几率回归模型例子

线性回归一个明显的问题是对离群点（和大多数数据点距离较远的数据点，outlier）非常敏感，导致模型建模不稳定，使结果有偏，为了缓解这个问题（特别是在二分类场景中）带来的影响，可考虑逻辑斯蒂回归(logistic regression)[Cox 1958]。

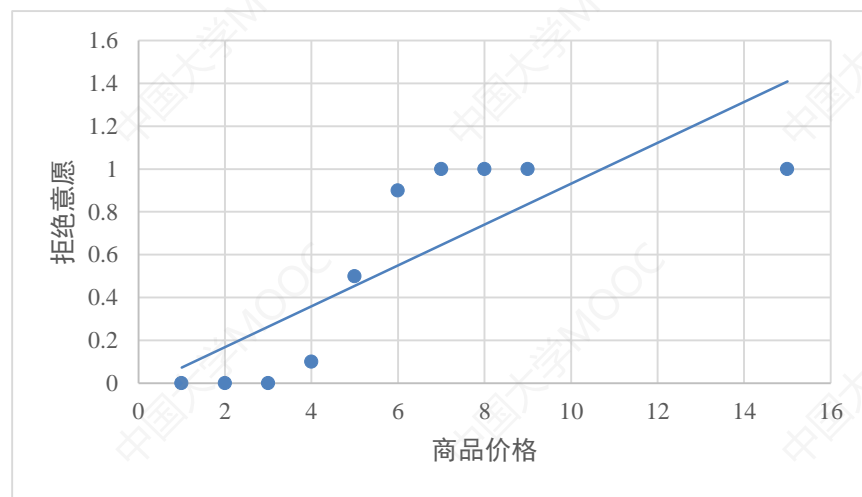


图4.5 加入一个离群点，该点表示当商品价格为15时，用户拒绝意愿为1（即用户不愿意购买该商品）

线性回归：逻辑斯蒂回归/对数几率回归

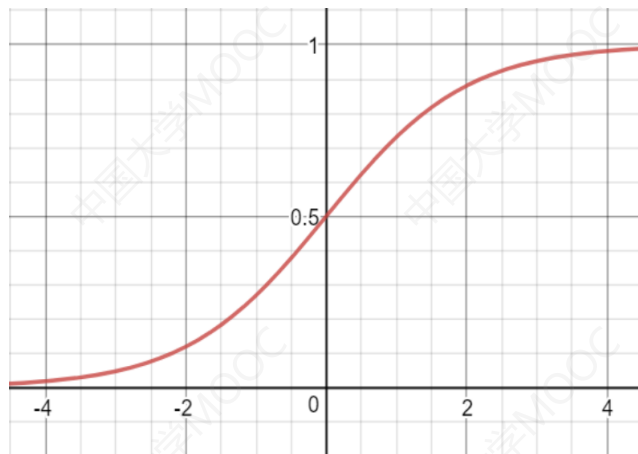


图4.6 Sigmoid函数

逻辑斯蒂回归(logistic regression)就是在回归模型中引入 sigmoid函数的一种非线性回归模型。Logistic回归模型可如下表示：

$$y = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(\mathbf{w}^T \mathbf{x} + b)}} \quad , \quad \text{其中 } y \in (0,1), z = \mathbf{w}^T \mathbf{x} + b$$

这里 $\frac{1}{1+e^{-z}}$ 是sigmoid函数、 $\mathbf{x} \in \mathbb{R}^d$ 是输入数据、 $\mathbf{w} \in \mathbb{R}^d$ 和 $b \in \mathbb{R}$ 是回归函数的参数。

线性回归：逻辑斯蒂回归/对数几率回归

逻辑斯蒂回归虽可用于对输入数据和输出结果之间复杂关系进行建模，但由于逻辑斯蒂回归函数的输出具有概率意义，使得逻辑斯蒂回归函数更多用于二分类问题（ $y = 1$ 表示输入数据 \mathbf{x} 属于正例， $y = 0$ 表示输入数据 \mathbf{x} 属于负例）。

$y = \frac{1}{1+e^{-(\mathbf{w}^T \mathbf{x} + b)}}$ 可用于计算输入数据 \mathbf{x} 属于正例概率，这里 y 理解为输入数据 \mathbf{x} 为正例的概率、 $1 - y$ 理解为输入数据 \mathbf{x} 为负例的概率，即 $p(y = 1|\mathbf{x})$ 。我们现在对比值 $\frac{p}{1-p}$ 取对数(即 $\log\left(\frac{p}{1-p}\right)$)来表示输入数据 \mathbf{x} 属于正例概率。 $\frac{p}{1-p}$ 被称为几率(odds)，反映了输入数据 \mathbf{x} 作为正例的相对可能性。 $\frac{p}{1-p}$ 的对数几率(log odds)或logit函数可表示为 $\log\left(\frac{p}{1-p}\right)$ 。

显然，可以得到 $p(y = 1|\mathbf{x}) = h_{\theta}(\mathbf{x}) = \frac{1}{1+e^{-(\mathbf{w}^T \mathbf{x} + b)}}$ 和 $p(y = 0|\mathbf{x}) = 1 - h_{\theta}(\mathbf{x}) = \frac{e^{-(\mathbf{w}^T \mathbf{x} + b)}}{1+e^{-(\mathbf{w}^T \mathbf{x} + b)}}$ 。 θ 表示模型参数（ $\theta = \{\mathbf{w}, b\}$ ）。于是有：

$$\text{logit}(p(y = 1|\mathbf{x})) = \log\left(\frac{p(y = 1|\mathbf{x})}{p(y = 0|\mathbf{x})}\right) = \log\left(\frac{p}{1-p}\right) = \mathbf{w}^T \mathbf{x} + b$$

线性回归：逻辑斯蒂回归/对数几率回归

- 如果输入数据 \mathbf{x} 属于正例的概率大于其属于负例的概率，即 $p(y = 1|\mathbf{x}) > 0.5$,

则输入数据 \mathbf{x} 可被判断属于正例。这一结果等价于 $\frac{p(y = 1|\mathbf{x})}{p(y = 0|\mathbf{x})} > 1$ ，即

$$\log\left(\frac{p(y = 1|\mathbf{x})}{p(y = 0|\mathbf{x})}\right) > \log 1 = 0, \text{ 也就是 } \mathbf{w}^T \mathbf{x} + b > 0 \text{ 成立。}$$

- 从这里可以看出，logistic回归是一个线性模型。在预测时，可以计算线性函数 $\mathbf{w}^T \mathbf{x} + b$ 取值是否大于0来判断输入数据 \mathbf{x} 的类别归属。

线性回归：逻辑斯蒂回归/对数几率回归

模型参数的似然函数被定义为 $\mathcal{L}(\theta|\mathcal{D}) = p(\mathcal{D}|\theta)$ ，其中 $\mathcal{D} = \{(x_i, y_i) | 1 \leq i \leq n\}$ 表示所有观测数据（或训练数据）， θ 表示模型参数（ $\theta = \{\mathbf{w}, b\}$ ）。在最大化对数似然函数过程中，一般假设观测所得每一个样本数据是独立同分布（independent and identically distributed, i.i.d），于是可得：

$$\mathcal{L}(\theta|\mathcal{D}) = p(\mathcal{D}|\theta) = \prod_{i=1}^n p(y_i|x, \theta) = \prod_{i=1}^n (h_{\theta}(x_i))^{y_i} (1 - h_{\theta}(x_i))^{1-y_i}$$

对上述公式取对数：

$$l(\theta) = \log(\mathcal{L}(\theta|\mathcal{D})) = \sum_{i=1}^n y_i \log(h_{\theta}(x_i)) + (1 - y_i) \log(1 - h_{\theta}(x_i))$$

最大似然估计目的是计算似然函数的最大值，而分类过程是需要损失函数最小化。因此，在上式前加一个负号得到损失函数(交叉熵)：

$$\begin{aligned} \mathcal{J}(\theta) &= -l(\theta) = -\log(\mathcal{L}(\theta|\mathcal{D})) \\ &= -\left(\sum_{i=1}^n y_i \log(h_{\theta}(x_i)) + (1 - y_i) \log(1 - h_{\theta}(x_i))\right) \end{aligned}$$

$$\mathcal{J}(\theta) \text{ 等价于: } \mathcal{J}(\theta) = \begin{cases} -\log(h_{\theta}(x_i)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x_i)) & \text{if } y = 0 \end{cases}$$

线性回归：逻辑斯蒂回归/对数几率回归

需要最小化损失函数来求解参数。数损失函数对参数 θ 的偏导如下（其中， $h'_\theta(x) = h_\theta(x)(1 - h_\theta(x))$, $\log' x = \frac{1}{x}$ ）

$$\begin{aligned}\frac{\partial J(\theta)}{\partial \theta_j} &= - \sum_{i=1}^n \left(y_i \frac{1}{h_\theta(x_i)} \frac{\partial h_\theta(x_i)}{\partial \theta_j} + (1 - y_i) \frac{1}{1 - h_\theta(x_i)} \frac{\partial (1 - h_\theta(x_i))}{\partial \theta_j} \right) \\&= - \sum_{i=1}^n \frac{1}{h_\theta(x_i)} \left(\frac{y_i}{h_\theta(x_i)} - \frac{1 - y_i}{1 - h_\theta(x_i)} \right) \\&= - \sum_{i=1}^n x_i h_\theta(x_i) (1 - h_\theta(x_i)) \left(\frac{y_i}{h_\theta(x_i)} - \frac{1 - y_i}{1 - h_\theta(x_i)} \right) \\&= - \sum_{i=1}^n x_i (y_i (1 - h_\theta(x_i)) - (1 - y_i) h_\theta(x_i)) \\&= \sum_{i=1}^n (y_i - h_\theta(x_i)) x_i\end{aligned}$$

将求导结果代入梯度下降迭代公式得：

$$\theta_j = \theta_j - \eta \sum_{i=1}^n (y_i - h_\theta(x_i)) x_i$$

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

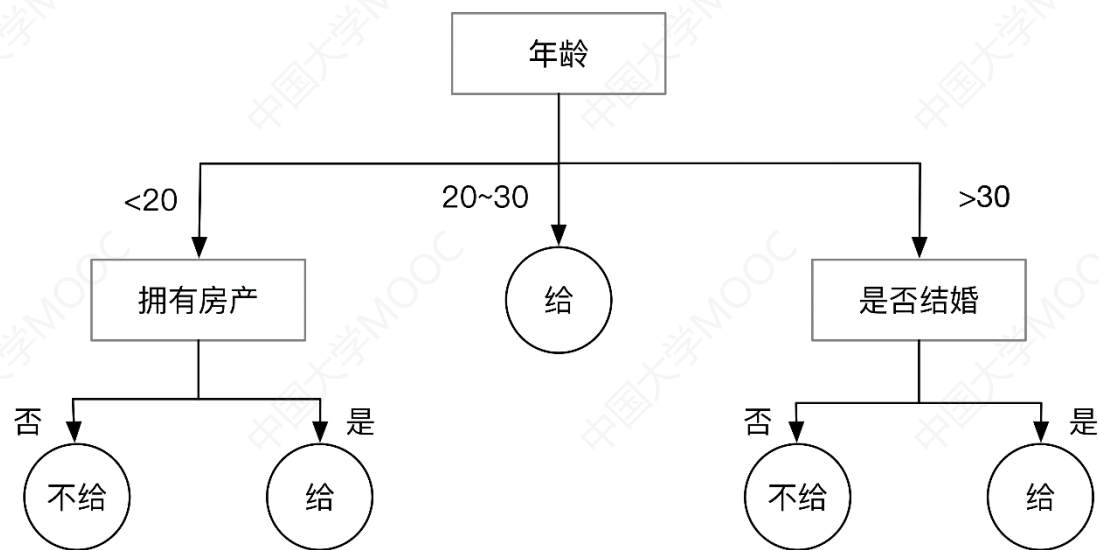
五、Ada Boosting

六、支持向量机

七、生成学习模型

决策树

决策树是一种通过树形结构来进行分类的方法。在决策树中，树形结构中每个非叶子节点表示对分类目标在某个属性上的一个判断，每个分支代表基于该属性做出的一个判断，最后树形结构中每个叶子节点代表一种分类结果，所以决策树可以看作是一系列以叶子节点为输出的决策规则（Decision Rules）[Quinlan 1987]。



决策树：决策树分类案例

决策树分类案例

序号	年龄	银行流水	是否结婚	拥有房产	是否给予贷款
1	>30	高	否	是	否
2	>30	高	否	否	否
3	20~30	高	否	是	是
4	<20	中	否	是	是
5	<20	低	否	是	是
6	<20	低	是	否	否
7	20~30	低	是	否	是
8	>30	中	否	是	否
9	>30	低	是	是	是
10	<20	中	否	是	是
11	>30	中	是	否	是
12	20~30	中	否	否	是
13	20~30	高	是	是	是
14	<20	中	否	否	否

决策树：信息熵

信息熵 (entropy)

假设有 K 个信息，其组成了集合样本 D ，记第 k 个信息发生的概率为 $p_k (1 \leq k \leq K)$ ”。如下定义这 K 个信息的信息熵：

$$E(D) = - \sum_{k=1}^K p_k \log_2 p_k$$

$E(D)$ 值越小，表示 D 包含的信息越确定，也称 D 的纯度越高。需要指出，所有 p_k 累加起来的和为1。

要点：构建决策树时划分属性的顺序选择是重要的。性能好的决策树随着划分不断进行，决策树分支结点样本集的“纯度”会越来越高，即其所包含样本尽可能属于相同类别。

表 4.7 年龄属性划分后子样本集情况统计

年龄属性 取值 a_i	">30"	"20~30"	"<20"
对应样 本数 $ D_i $	5	4	5
正负样本 数量	(2+, 3-)	(4+, 0-)	(3+, 2-)

决策树：信息熵

表 4.7 年龄属性划分后子样本集情况统计

年龄属性 取值 a_i	">30"	"20~30"	"<20"
对应样 本数 $ D_i $	5	4	5
正负样本 数量	(2+, 3-)	(4+, 0-)	(3+, 2-)

$$\text{"年龄"} > 30: Ent(D_0) = -\left(\frac{2}{5} \times \log_2 \frac{2}{5} + \frac{3}{5} \times \log_2 \frac{3}{5}\right) = 0.971$$

$$\text{"年龄"} 20 \sim 30: Ent(D_1) = -\left(\frac{4}{4} \times \log_2 \frac{4}{4} + 0\right) = 0$$

$$\text{"年龄"} < 20: Ent(D_2) = -\left(\frac{3}{5} \times \log_2 \frac{3}{5} + \frac{2}{5} \times \log_2 \frac{2}{5}\right) = 0.971$$

决策树：信息熵

表 4.7 年龄属性划分后子样本集情况统计

年龄属性 取值 a_i	">30"	"20~30"	"<20"
对应样 本数 $ D_i $	5	4	5
正负样本 数量	(2+, 3-)	(4+, 0-)	(3+, 2-)

$$\text{"年龄"} > 30: Ent(D_0) = -\left(\frac{2}{5} \times \log_2 \frac{2}{5} + \frac{3}{5} \times \log_2 \frac{3}{5}\right) = 0.971$$

$$\text{"年龄"} 20 \sim 30: Ent(D_1) = -\left(\frac{4}{4} \times \log_2 \frac{4}{4} + 0\right) = 0$$

$$\text{"年龄"} < 20: Ent(D_2) = -\left(\frac{3}{5} \times \log_2 \frac{3}{5} + \frac{2}{5} \times \log_2 \frac{2}{5}\right) = 0.971$$

决策树：信息增益

得到上述三个的信息熵后，可进一步计算使用年龄属性对原样本集进行划分后的信息增益，计算公式如下：

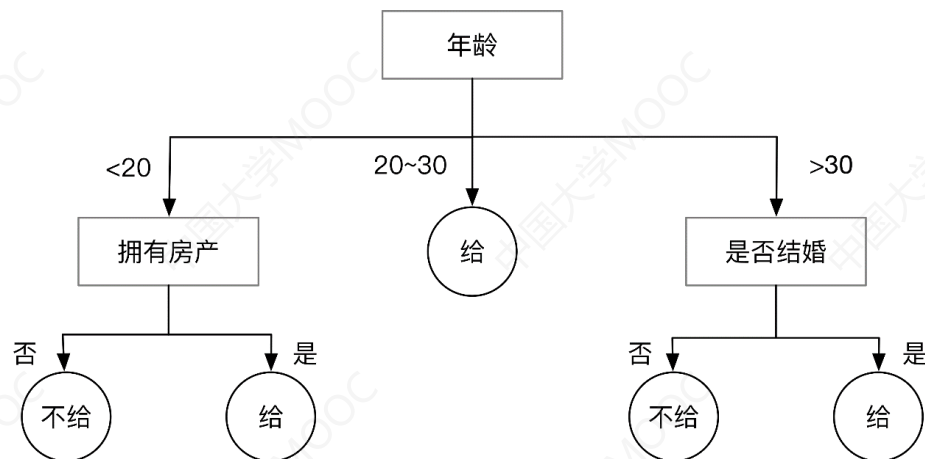
$$Gain(D, A) = Ent(D) - \sum_{i=1}^n \frac{|D_i|}{|D|} Ent(D_i)$$

将 $A = \text{年龄}$ 代入。于是选择年龄这一属性划分后的信息增益为：

$$Gain(D, \text{年龄}) = 0.940 - \left(\frac{5}{14} \times 0.971 + \frac{4}{14} \times 0 + \frac{5}{14} \times 0.971 \right) = 0.246$$

同理，可以计算银行流水、是否结婚、是否拥有房产三个人物属性的信息增益。通过比较四种属性信息增益的高低来选择最佳属性对原样本集进行划分，得到最大的“纯度”。如果划分后的不同子样本集都只存在同类样本，那么停止划分。

决策树：构建决策树



$info$ 和 $Gain - ratio$ 计算公式如下：

$$info = - \sum_{i=1}^n \frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|}$$

$$Gain - ratio = Gain(D, A) / info$$

另一种计算更简的度量指标是如下的Gini系数：

$$Gini(D) = 1 - \sum_{k=1}^K p_k^2$$

相对于信息熵的计算 $E(D) = - \sum_{k=1}^K p_k \log_2 p_k$ ，不用计算对数 \log ，计算更为简易。

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

线性区别分析

线性判别分析(linear discriminant analysis, LDA)是一种基于监督学习的降维方法,也称为Fisher线性判别分析(fisher's discriminant analysis, FDA) [Fisher 1936]。对于一组具有标签信息的高维数据样本, LDA利用其类别信息,将其线性投影到一个低维空间上,在低维空间中同一类别样本尽可能靠近,不同类别样本尽可能彼此远离。

“类内方差小、
类间间隔大”

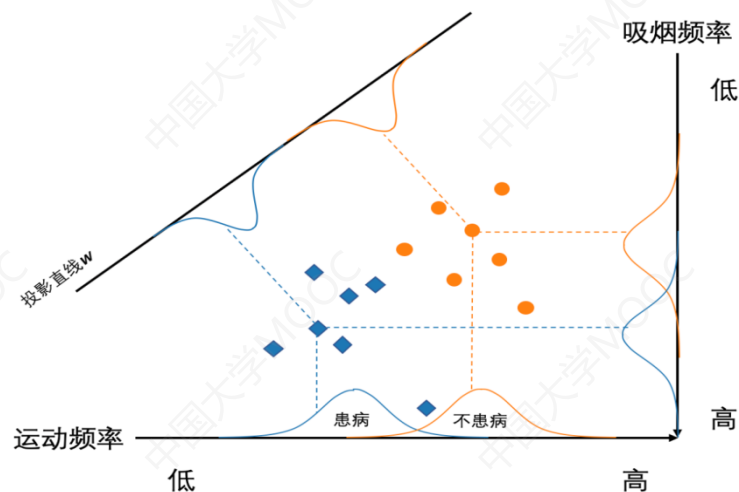


图4.8 两个类别数据所对应的不同投影方式
君子和而不同、小人同而不和

线性区别分析：符号定义

假设样本集为 $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), \dots, (\mathbf{x}_N, y_N)\}$, 样本 $\mathbf{x}_i \in \mathbb{R}^d$ 的类别标签为 y_i 。其中, y_i 的取值范围是 $\{C_1, C_2, \dots, C_K\}$, 即一共有 K 类样本。

定义 \mathbf{X} 为所有样本构成集合、 N_i 为第 i 个类别所包含样本个数、 X_i 为第 i 类样本的集合、 \mathbf{m} 为所有样本的均值向量、 \mathbf{m}_i 为第 i 类样本的均值向量。 Σ_i 为第 i 类样本的协方差矩阵, 其定义为:

$$\Sigma_i = \sum_{\mathbf{x} \in X_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$

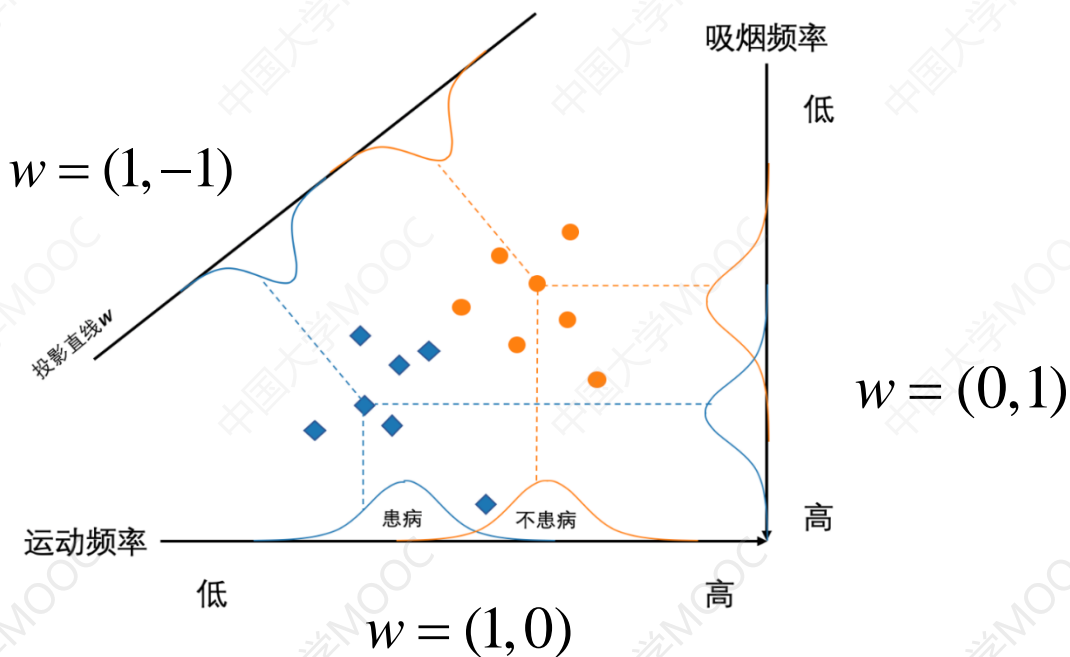
线性区别分析：二分类问题

先来看 $K = 2$ 的情况，即二分类问题。在二分类问题中，训练样本归属于 \mathcal{C}_1 或 \mathcal{C}_2 两个类别，并通过如下的线性函数投影到一维空间上：

$$y(x) = w^T x \quad (w \in \mathbb{R}^n)$$

节点 $(1,1), (2,2), (3,3), (4,4) \dots$

都会投影到同一个点。



线性区别分析：二分类问题

先来看 $K = 2$ 的情况，即二分类问题。在二分类问题中，训练样本归属于 \mathcal{C}_1 或 \mathcal{C}_2 两个类别，并通过如下的线性函数投影到一维空间上：

$$y(x) = \mathbf{w}^T \mathbf{x} \ (\mathbf{w} \in \mathbb{R}^n)$$

投影之后类别 \mathcal{C}_1 的协方差矩阵 s_1 为：

$$s_1 = \sum_{\mathbf{x} \in \mathcal{C}_1} (\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mathbf{m}_1)^2 = \mathbf{w}^T \sum_{\mathbf{x} \in \mathcal{C}_1} [(\mathbf{x} - \mathbf{m}_1)(\mathbf{x} - \mathbf{m}_1)^T] \mathbf{w}$$

同理可得到投影之后类别 \mathcal{C}_2 的协方差矩阵 s_2 。

线性区别分析：二分类问题

投影后两个协方差矩阵 s_1 和 s_2 分别为 $\mathbf{w}^T \Sigma_1 \mathbf{w}$ 和 $\mathbf{w}^T \Sigma_2 \mathbf{w}$ 。 s_1 和 s_2 可用来衡量同一类别数据样本之间“分散程度”。为了使得归属于同一类别的样本数据在投影后的空间中尽可能靠近，需要最小化 $s_1 + s_2$ 取值。

在投影之后的空间中，归属于两个类别的数据样本中心可分别如下计算：

$$m_1 = \mathbf{w}^T \mathbf{m}_1, \quad m_2 = \mathbf{w}^T \mathbf{m}_2$$

这样，就可以通过 $\|m_2 - m_1\|_2^2$ 来衡量不同类别之间的距离。为了使得归属于不同类别的样本数据在投影后空间中尽可能彼此远离，需要最大化过 $\|m_2 - m_1\|_2^2$ 取值。

同时考虑上面两点，就得到了需要最大化的目标 $J(\mathbf{w})$ ，定义如下：

$$J(\mathbf{w}) = \frac{\|m_2 - m_1\|_2^2}{s_1 + s_2}$$

线性区别分析：二分类问题

$$J(\mathbf{w}) = \frac{\|\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)\|_2^2}{\mathbf{w}^T \Sigma_1 \mathbf{w} + \mathbf{w}^T \Sigma_2 \mathbf{w}} = \frac{\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T \mathbf{w}}{\mathbf{w}^T (\Sigma_1 + \Sigma_2) \mathbf{w}} = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

其中， \mathbf{S}_b 称为类间散度矩阵(between-class scatter matrix)，即衡量两个类别均值点之间的“分离”程度，可定义如下：

$$\mathbf{S}_b = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$$

\mathbf{S}_w 则称为类内散度矩阵(within-class scatter matrix)，即衡量每个类别中数据点的“分离”程度，可定义如下：

$$\mathbf{S}_w = \Sigma_1 + \Sigma_2$$

由于 $J(\mathbf{w})$ 的分子和分母都是关于 \mathbf{w} 的二项式，因此最后的解只与 \mathbf{w} 的方向有关，与 \mathbf{w} 的长度无关，因此可令分母 $\mathbf{w}^T \mathbf{S}_w \mathbf{w} = 1$ ，然后用拉格朗日乘子法来求解这个问题。

线性区别分析：二分类问题

对应拉格朗日函数为：

$$L(\mathbf{w}) = \mathbf{w}^T \mathbf{S}_b \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{S}_w \mathbf{w} - 1)$$

对 \mathbf{w} 求偏导并使其求导结果为零，可得 $\mathbf{S}_w^{-1} \mathbf{S}_b \mathbf{w} = \lambda \mathbf{w}$ 。由此可见， λ 和 \mathbf{w} 分别是 $\mathbf{S}_w^{-1} \mathbf{S}_b$ 的特征根和特征向量， $\mathbf{S}_w^{-1} \mathbf{S}_b \mathbf{w} = \lambda \mathbf{w}$ 也被称为Fisher线性判别（Fisher linear discrimination）。

$$\mathbf{S}_w^{-1} \mathbf{S}_b \mathbf{w} = \mathbf{S}_w^{-1} (\mathbf{m}_2 - \mathbf{m}_1) \times \lambda_w = \lambda \mathbf{w}$$

由于对 \mathbf{w} 的放大和缩小操作不影响结果，因此可约去上式中的未知数 λ 和 λ_w ，得到： $\mathbf{w} = \mathbf{S}_w^{-1} (\mathbf{m}_2 - \mathbf{m}_1)$

线性区别分析：多分类问题

假设 n 个原始高维数据所构成的类别种类为 K 、每个原始数据被投影映射到低维空间中的维度为 r 。

令投影矩阵 $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_r)$ ，可知 \mathbf{W} 是一个 $n \times r$ 矩阵。于是， $\mathbf{W}^T \mathbf{m}_i$ 为第 i 类样本数据中心在低维空间的投影结果， $\mathbf{W}^T \Sigma_i \mathbf{W}$ 为第 i 类样本数据协方差在低维空间的投影结果。

类内散度矩阵 \mathbf{S}_w 重新定义如下：

$$\mathbf{S}_w = \sum_{i=1}^K \Sigma_i, \text{ 其中 } \Sigma_i = \sum_{x \in \text{class } i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$

在上式中， \mathbf{m}_i 是第 i 个类别中所包含样本数据的均值。

类间散度矩阵 \mathbf{S}_b 重新定义如下：

$$\mathbf{S}_b = \sum_{i=1}^K \frac{N_i}{N} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$$

线性区别分析：多分类问题

将多类LDA映射投影方向的优化目标 $J(W)$ 改为：

$$J(W) = \frac{\prod_{diag} W^T S_b W}{\prod_{diag} W^T S_w W}$$

其中， $\prod_{diag} A$ 为矩阵 A 主对角元素的乘积。

继续对 $J(W)$ 进行变形：

$$J(W) = \frac{\prod_{diag} W^T S_b W}{\prod_{diag} W^T S_w W} = \frac{\prod_{i=1}^r w_i^T S_b w_i}{\prod_{i=1}^r w_i^T S_w w_i} = \prod_{i=1}^r \frac{w_i^T S_b w_i}{w_i^T S_w w_i}$$

显然需要使乘积式子中每个 $\frac{w_i^T S_b w_i}{w_i^T S_w w_i}$ 取值最大，这就是二分类问题的求解目标，

即每一个 w_i 都是 $S_w^{-1} S_b W = \lambda W$ 的一个解。

线性区别分析：线性判别分析的降维步骤

对线性判别分析的降维步骤描述如下：

1. 计算数据样本集中每个类别样本的均值
2. 计算类内散度矩阵 S_w 和类间散度矩阵 S_b
3. 根据 $S_w^{-1}S_bW = \lambda W$ 来求解 $S_w^{-1}S_b$ 所对应前 r 个最大特征值所对应特征向量 (w_1, w_2, \dots, w_r) ，构成矩阵 W
4. 通过矩阵 W 将每个样本映射到低维空间，实现特征降维。

线性区别分析：与主成分分析法的异同

	线性判别分析	主成分分析
是否需要样本标签	监督学习	无监督学习
降维方法	优化寻找特征向量 \mathbf{w}	优化寻找特征向量 \mathbf{w}
目标	类内方差小、类间距大	寻找投影后数据之间方差最大的投影方向
维度	LDA降维后所得到维度是与数据样本的类别个数 K 有关	PCA对高维数据降维后的维数是与原始数据特征维度相关

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

Boosting (adaptive boosting, 自适应提升)

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.)

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。

计算学习理论：霍夫丁不等式(Hoeffding's inequality)

- 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：

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

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 ϵ 的概率非常小。

计算学习理论：概率近似正确 (probably approximately correct, PAC)

- 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”产生误差很小呢？这就是概率近似正确（PAC）要回答的问题。

计算学习理论：概率近似正确 (probably approximately correct, PAC)

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。
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 Boosting: 思路描述

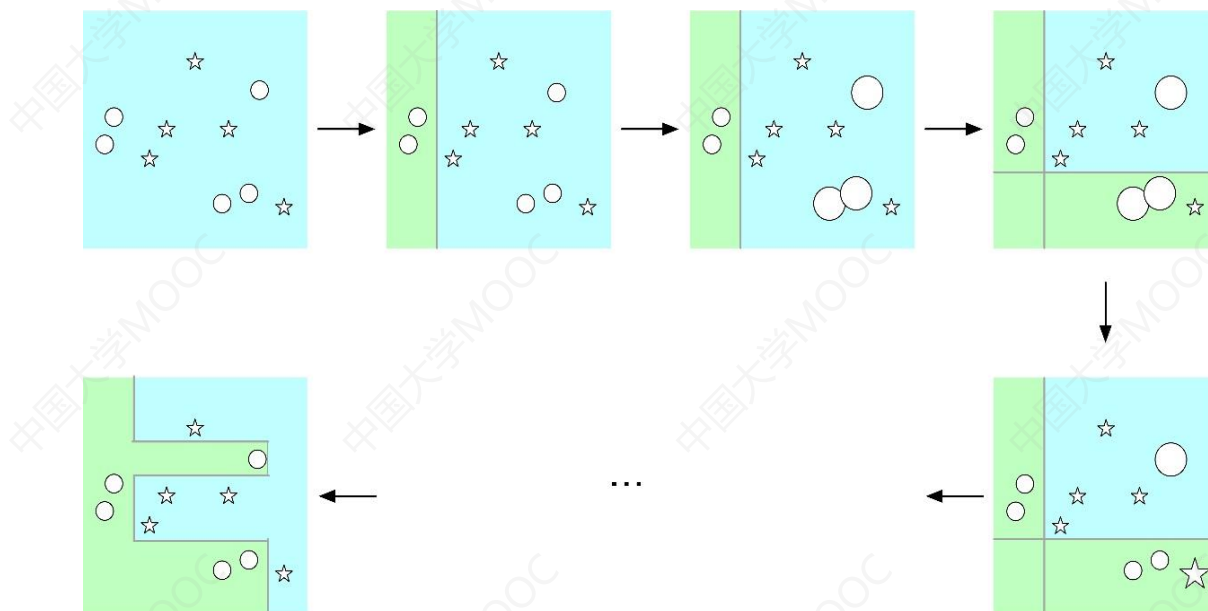


图4.9 Ada Boosting算法学习过程示意图

Ada Boosting: 思路描述

- 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 Boosting: 算法描述---数据样本权重初始化

- 给定包含 N 个标注数据的训练集合 Γ , $\Gamma = \{(x_1, y_1), \dots, (x_N, y_N)\}$ 。

$$x_i (1 \leq i \leq N) \in X \subseteq R^n, y_i \in Y = \{-1, 1\}$$

- Ada Boosting算法将从这些标注数据出发, 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

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

Ada Boosting: 算法描述---第 m 个弱分类器训练

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

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

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, $Z_m = \sum_{i=1}^N w_{m,i} e^{-\alpha_m y_i G_m(x_i)}$

Ada Boosting: 算法描述---弱分类器组合成强分类器

3. 以线性加权形式来组合弱分类器 $f(x)$

$$f(x) = \sum_{i=1}^M \alpha_m G_m(x)$$

得到强分类器 $G(x)$

$$G(x) = \text{sign}(f(x)) = \text{sign}\left(\sum_{i=1}^M \alpha_m G_m(x)\right)$$

Ada Boosting: 算法解释

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

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

Ada Boosting: 算法解释

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

(a) 当第 m 个弱分类器 $G_m(x)$ 错误率为1, 即 $\text{err}_m = \sum_{i=1}^N w_{mi} I(G_m(x_i) \neq y_i) = 1$, 意味每

个样本分类出错, 则 $\alpha_m = \frac{1}{2} \ln \frac{1 - \text{err}_m}{\text{err}_m} \rightarrow -\infty$, 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 $\frac{1}{2}$, $\alpha_m = \frac{1}{2} \ln \frac{1 - \text{err}_m}{\text{err}_m} = 0$ 。如果错误率 err_m 小于 $\frac{1}{2}$,

权重 α_m 为正($\text{err}_m < \frac{1}{2}$ 、 $\alpha_m > 0$)。可知权重 α_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 Boosting: 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）

Ada Boosting: 算法解释

弱分类器构造强分类器

$$f(x) = \sum_{i=1}^M \alpha_m G_m(x)$$

$$G(x) = \text{sign}(f(x)) = \text{sign}\left(\sum_{i=1}^M \alpha_m G_m(x)\right)$$

- $f(x)$ 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。
- α_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 Boosting: 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 Boosting: 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。

Ada Boosting: 例子

通过一个简单两类分类例子来介绍Ada Boosting算法过程。表4.5.1给出了10个数据点 $x_i (i \in \{1, 2, \dots, 10\})$ 取值及其所对应的类别标签 $y_i \in \{1, -1\} (i \in \{1, 2, \dots, 10\})$ 。

	1	2	3	4	5	6	7	8	9	10
x	-9	-7	-5	-3	-1	1	3	5	7	9
y	-1	-1	1	1	-1	-1	-1	-1	1	1

表4.8 两类分类问题数据

根据表4.8所给出的数据，要构造若干个弱分类器，然后将这些弱分类器组合为一个强分类器，完成表4.8所示数据的分类任务。

Ada Boosting: 例子

这里定义每个弱分类器 G 为一种分段函数，由一个阈值 ε 构成，形式如下：

$$G(x_i) = \begin{cases} -1 & x_i < \varepsilon \\ 1 & x_i > \varepsilon \end{cases} \quad \text{或} \quad G(x_i) = \begin{cases} 1 & x_i < \varepsilon \\ -1 & x_i > \varepsilon \end{cases}$$

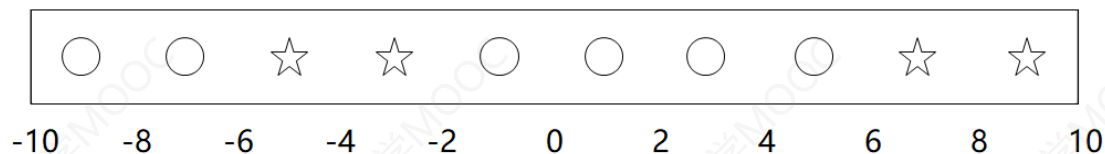
当然，在实际中，可根据需要使用其他的弱分类器

Ada Boosting: 例子

(1) 数据样本权重初始化

$$D_1 = (w_{1,1}, \dots, w_{1,i}, \dots, w_{1,10}), \text{ 其中 } w_{1,i} = \frac{1}{10} (1 \leq i \leq 10)$$

下面用图来辅助说明算法流程。图中圆圈所代表数据点标签为-1、五角星所代表数据点标签为1，每个形状的颜色深浅代表这些数据被当前所学习得到（组合）分类器给出标签预测值大小，颜色越深表示越接近标签值-1、颜色越浅表示越接近标签值1。



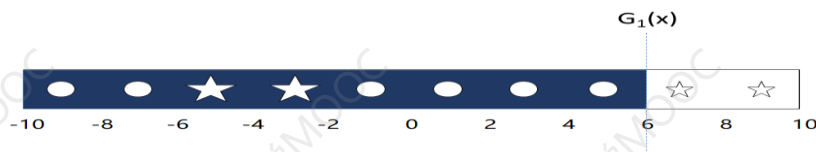
Ada Boosting: 例子

(2) 分别训练M个基分类器 (弱分类器)

对 $m = 1$

- 使用具有分布权重 D_1 的训练数据来学习得到第 $m = 1$ 个基分类器 G_1 。不难看出，当阈值 $\epsilon = 6$ 时，基分类器 G_1 具有最小错误率。 G_1 分类器如下表示：
- $G_1(x_i) = \begin{cases} -1 & x_i < 6 \\ 1 & x_i > 6 \end{cases}$
- 计算 $G_1(x)$ 在训练数据集上的分类误差，样例3、4被错误分类，因此 G_1 的分类误差为 $\text{err}_1 = \sum_{i=1}^N w_{1,i} I(G_1(x_i) \neq y_i) = 0.1 + 0.1 = 0.2$
- 根据分类误差计算弱分类器 $G_1(x)$ 的权重： $\alpha_1 = \frac{1}{2} \ln \frac{1 - \text{err}_1}{\text{err}_1} = 0.6931$
- 更新下一轮第 $m = 2$ 个分类器训练时第 i 个训练样本的权重： $D_2 = \{w_{2,i}\}_0^{10}, w_{2,i} = \frac{w_{1,i}}{Z_1} e^{-\alpha_1 y_i G_1(x_i)}$ ，可得到数据样本新的权重：
 $D_2 = (0.0625, 0.0625, 0.25, 0.25, 0.0625, 0.0625, 0.0625, 0.0625, 0.0625, 0.0625)$
- 通过加权线性组合得到当前的分类器 $f_1(x) = \sum_{i=1}^M \alpha_m G_m(x) = 0.6931 G_1(x)$

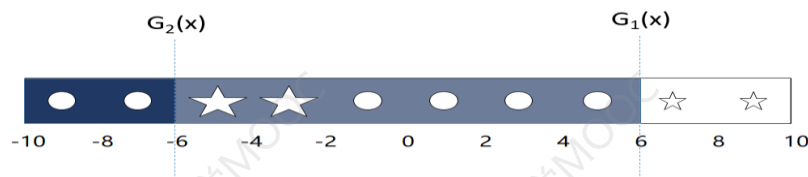
在下图中，被分类错误数据样本的形状尺寸比其它数据样本形状稍大，以表示被分类错误的样本数据权重增大。



Ada Boosting: 例子

对 $m = 2$

- 对于具有分布权重为 D_2 的训练数据，当阈值 $\varepsilon = -6$ 时，基分类器 G_2 具有最小的错误率。 G_2 分类器如下表示：
- $G_2(x_i) = \begin{cases} -1 & x_i < -6 \\ 1 & x_i > -6 \end{cases}$
- 分类误差 $err_2 = \sum_{i=1}^N w_{2,i} I(G_2(x_i) \neq y_i) = 0.25$
- 弱分类器 $G_2(x)$ 的权重 $\alpha_2 = \frac{1}{2} \ln \frac{1-err_2}{err_2} = 0.5439$
- 当进行下一轮分类器训练时，样本权重更新如下： $D_3 =$
(0.04166667, 0.04166667, 0.16666667, 0.16666667, 0.125, 0.125, 0.125, 0.125, 0.04166667, 0.04166667)
, 0.125, 0.125, 0.04166667, 0.04166667)
- 通过加权线性组合得到当前的分类器 $f_2(x) = 0.6931G_1(x) + 0.5439G_2(x)$



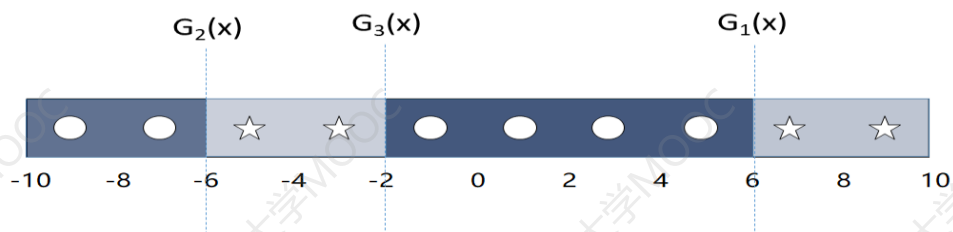
Ada Boosting: 例子

对 $m = 3$

- 对于具有分布权重为 D_3 的训练样本数据，当阈值 $\varepsilon = -2$ 时，基分类器 G_3 具有最小的错误率。 G_3 分类器表示如下：

$$G_3(x_i) = \begin{cases} -1 & x_i > -2 \\ 1 & x_i < -2 \end{cases}$$

- 分类误差 $err_3 = \sum_{i=1}^N w_{3,i} I(G_3(x_i) \neq y_i) = 0.1667$
- 弱分类器 $G_3(x)$ 的权重 $\alpha_3 = \frac{1}{2} \ln \frac{1-err_3}{err_3} = 0.8047$
- 下一轮弱分类器训练时，训练数据样本的权重更新如下： $D_4 = (0.125, 0.125, 0.1, 0.1, 0.075, 0.075, 0.075, 0.075, 0.125, 0.125)$
- 通过加权线性组合得到当前的分类器 $f_3(x) = 0.6931G_1(x) + 0.5439G_2(x) + 0.8047G_3(x)$
- 在 $f_3(x)$ 的基础上，构造强分类器 $G(x) = \text{sign}(f_3(x)) = \text{sign}(0.6931G_1(x) + 0.5439G_2(x) + 0.8047G_3(x))$ 。
- 这里 $\text{sign}(\cdot)$ 是符号函数，其输入值大于0时，符号函数输出为1，反之为-1。由于 $G(x)$ 在训练样本上分类错误率为0，算法终止，得到最终的强分类器。



回归与分类的区别

□ 两者均是学习输入变量和输出变量之间潜在关系模型，基于学习所得模型将输入变量映射到输出变量。

$$f: x \rightarrow y, \quad x \in A, \quad y \in B$$



- 监督学习分为回归和分类两个类别。
- 在回归分析中，学习得到一个函数将输入变量映射到连续输出空间，如价格和温度等，即值域是连续空间。
- 在分类模型中，学习得到一个函数将输入变量映射到离散输出空间，如人脸和汽车等，即值域是离散空间。

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

支持向量机：VC维与结构风险最小化

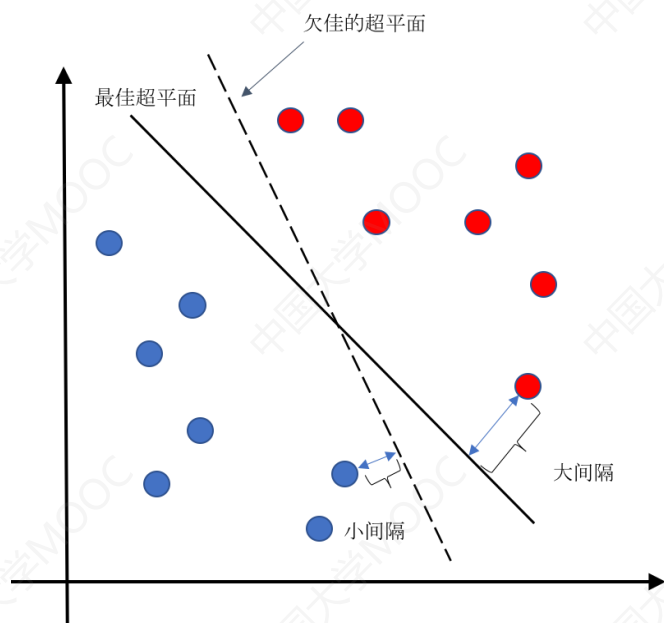


图4.10 两类分类问题的最佳分类平面

传统识别理论认为，算法模型性能可用从训练样本集所得经验风险（empirical risk）来衡量。经验风险指算法模型在训练样本集中所有数据上所得误差的累加。很显然，经验风险越小，算法模型对训练数据拟合程度越好。在实际中，一味要求经验风险小，往往会造成过学习问题（over-fitting）。

支持向量机（support vector machine, SVM）通过结构风险最小化（structural risk minimization）来解决过学习问题。如图4.10给出了一个两分类问题。图中存在多个可将样本分开的超平面(hyper-plane)。但是，在这些超平面中，支持向量机学习算法会去寻找一个最佳超平面，使得每个类别中距离超平面最近的样本点到超平面的最小距离最大。

支持向量机：VC维与结构风险最小化

在理论上，支持向量机认为：分类器对未知数据（即测试数据）进行分类时所产生的期望风险（expected risk，即真实风险）不是由经验风险单独决定的，而是由两部分组成：1）从训练集合数据所得经验风险（如果经验风险小、期望风险很大，则是过学习）；2）置信风险（confidence risk），它与分类器的VC维及训练样本数目有关。

如果 $0 \leq \eta \leq 1$ ，Vapnik推导出期望风险 \mathfrak{R} 和经验风险 \mathfrak{R}_{emp} 以 $1 - \eta$ 的概率满足如下关系：

$$\mathfrak{R} \leq \mathfrak{R}_{emp} + \sqrt{\frac{h(\log(2n/h) + 1) - \log(\eta/4)}{n}}$$

其中， $\sqrt{\frac{h(\log(2n/h)+1)-\log(\eta/4)}{n}}$ 叫做“VC置信值”， n 是训练样本个数， h 是反映学习机复杂程度的VC维。因为期望风险 \mathfrak{R} 代表了分类器对未知数据分类推广能力，所以 \mathfrak{R} 越小越好。在上面不等式中，VC置信值是 h 的增函数， \mathfrak{R}_{emp} 是 h 的减函数，于是选择一个折中的 h 值可以使期望风险 \mathfrak{R} 达到最小。支持向量机使用结构风险最小化准则来选取VC维 h ，使每一类别数据之间的分类间隔（Margin）最大，最终使实期望风险 \mathfrak{R} 最小。

支持向量机：VC维与结构风险最小化

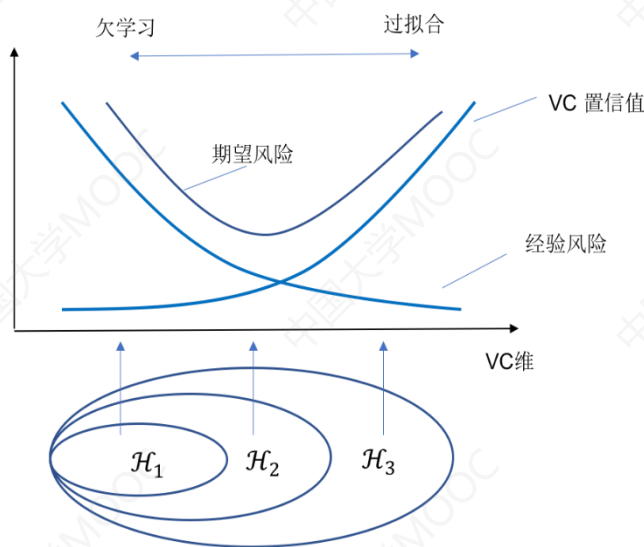


图4.11 经验风险、期望风险、VC置信度、VC维、过学习和欠学习的关系

- 假设空间的VC维 将 n 个数据分为两类，可以有 2^n 种分法，即可理解成有 2^n 个学习问题。若存在一个假设 \mathcal{H} ，能准确无误地将 2^n 种问题进行分类，那么 n 就是 \mathcal{H} 的VC维。
- 一般地，在 r 维空间中，线性决策面的VC维为 $r + 1$ 。VC维就是对假设空间 \mathcal{H} 复杂度的一种度量。当样本数 n 固定时，如果VC维越高，则算法模型的复杂性越高。VC维越大，通常推广能力越差，置信风险会变大。如果算法模型一味降低经验风险，则会提高模型复杂度，导致VC维很高，置信风险大，使得期望风险就大。

支持向量机：VC维与结构风险最小化

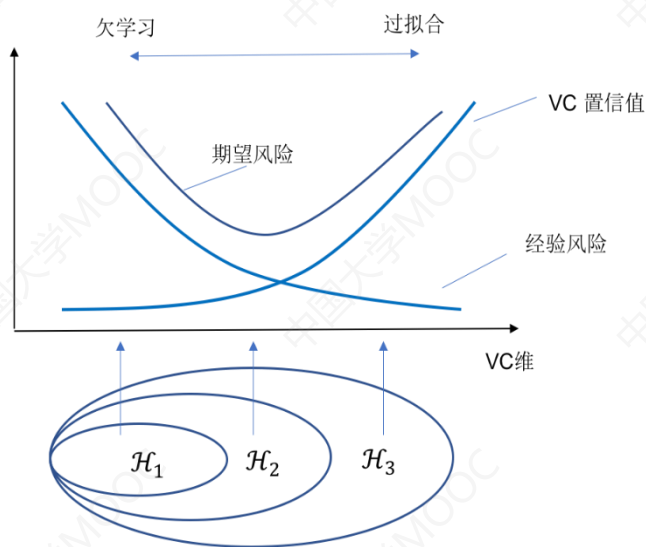


图4.11 经验风险、期望风险、VC置信度、VC维、过学习和欠学习的关系

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

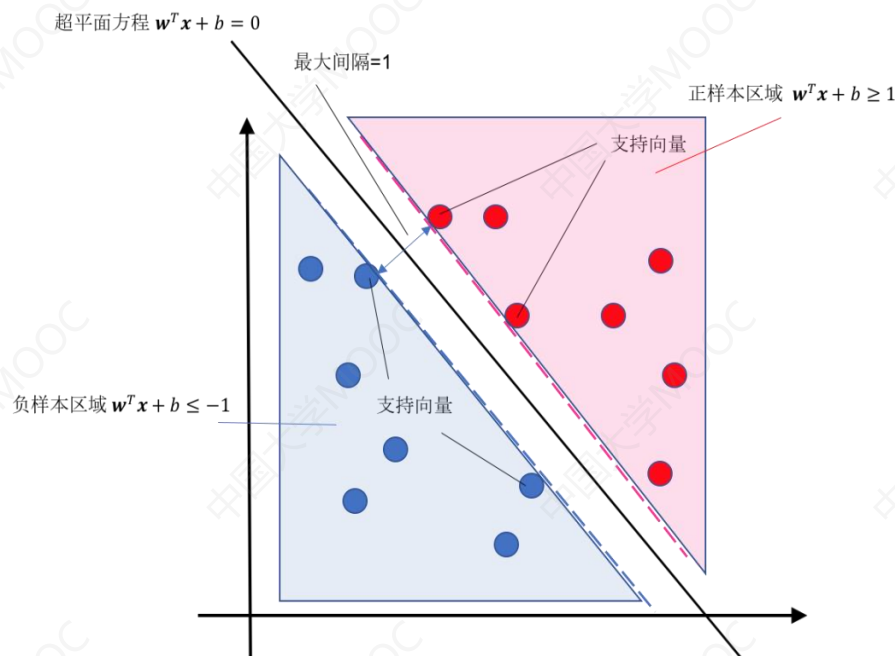


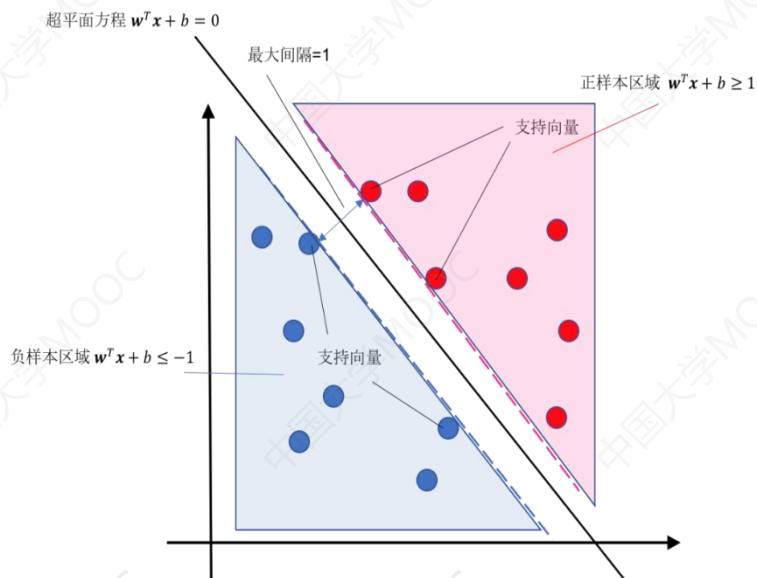
图4.12 经验风险、期望风险、VC置信度、VC维、过学习和欠学习的关系

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，其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 \mathbf{x} 到该平面距离可表示为：

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



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：

$$\min_i |\mathbf{w}^T \mathbf{x}_i + b| = 1$$

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。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：

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

即：

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

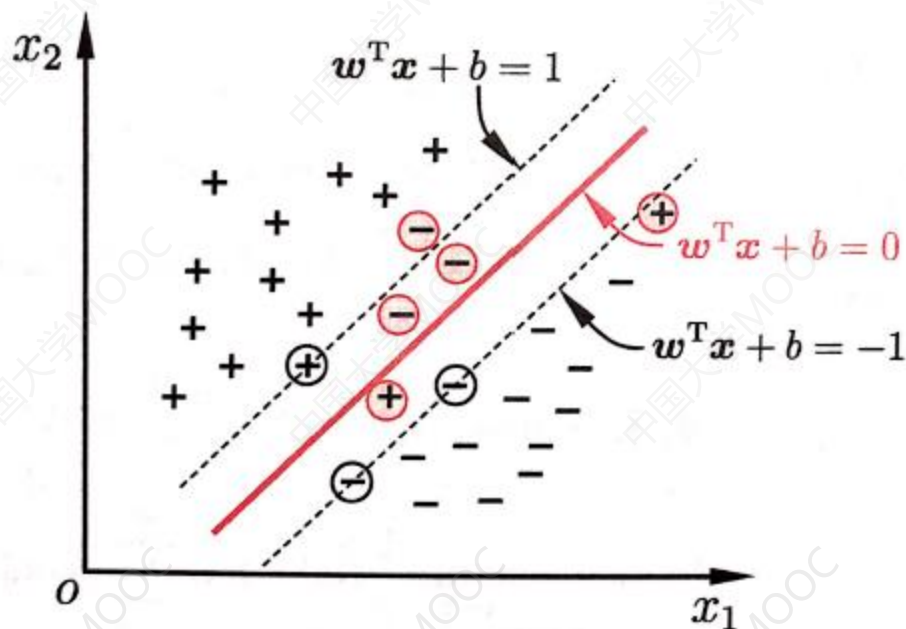
支持向量机的基本形式就是最大化分类间隔，即在满足约束的条件下找到参数 \mathbf{w} 和 b 使得 γ 最大，即等价于：

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

凸二次规划

支持向量机： 松弛变量，软间隔与hinge损失函数

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。



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

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

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq -\infty \text{ for incorrect } \mathbf{x}_i$$

难以直接求解

支持向量机： 松弛变量，软间隔与hinge损失函数

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

正确分类数据的hinge损失： $\max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0$

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

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

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

拉格朗日乘子法

支持向量机：核函数

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]。

常见核函数

线性	$\kappa(x_i, x_j) = x_i x_j$
多项式	$\kappa(x_i, x_j) = (\gamma x_i x_j + c)^n$
Radial basis function	$\kappa(x_i, x_j) = e^{-\frac{\ x_i - x_j\ ^2}{2\sigma^2}}$
Sigmoid	$\kappa(x_i, x_j) = \tanh(\gamma(x_i, x_j - \gamma))$

提纲

一、机器学习基本概念

二、回归分析

三、决策树

四、线性区别分析

五、Ada Boosting

六、支持向量机

七、生成学习模型

生成学习模型

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 $P(c_i|x) = \frac{P(x, c_i)}{P(x)}$ 。

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

表4.10 输入数据-类别标签的样本分布

类别 输入	阳性	阴性
$x = 0$	6	2
$x = 1$	0	4

在表4.10中，一共有12个样本-标签数据，其中 $(x=0, c=\text{阳性})$ 样本出现了6次、 $(x=0, c=\text{阴性})$ 样本出现了2次、 $(x=1, c=\text{阳性})$ 样本没有出现、 $(x=1, c=\text{阴性})$ 样本出现了4次。

生成学习模型

生成学习

输入样本和类别标签的联合概率分布为： $P(0, \text{阳性}) = \frac{6}{12} = \frac{1}{2}$ 、 $P(0, \text{阴性}) = \frac{2}{12} = \frac{1}{6}$ 、 $P(1, \text{阳性}) = 0$ 、 $P(1, \text{阴性}) = \frac{4}{12} = \frac{1}{3}$ 。一旦给出输入数据，假定输入数据的概率为某个常数，就可以通过计算 $\frac{P(0, \text{阳性})}{P(0)}$ 或者 $\frac{P(0, \text{阴性})}{P(0)}$ 以及 $\frac{P(1, \text{阳性})}{P(1)}$ 或者 $\frac{P(1, \text{阴性})}{P(1)}$ ，将输入数据归属到所得结果最大所对应的类别。这里要注意，样本-标签数据的联合概率分布累加之和为 1。

生成学习模型

判别式学习

输入样本和类别标签的条件概率分布为： $P(\text{阳性}|0) = \frac{6}{8} = \frac{3}{4}$ 、 $P(\text{阳性}|1) = \frac{2}{8} = \frac{1}{4}$ ； $P(\text{阴性}|0) = \frac{0}{4} = 0$ 、 $P(\text{阴性}|1) = \frac{4}{4} = 1$ 。这里要注意，输入样本为 0 或为 1 前提下，其所对应的类别概率累加之和为 1。

常见的生成学习模型有隐马尔可夫模型、隐狄利克雷分布(latent dirichlet allocation, LDA)等。

作业内容

图像恢复重建

图像是一种非常常见的信息载体，但是在图像的获取、传输、存储的过程中可能由于各种原因使得图像受到噪声的影响。如何去除噪声的影响，恢复图像原本的信息是计算机视觉中的重要研究问题。

常见的图像恢复算法有基于空间域的中值滤波、基于小波域的小波去噪、基于偏微分方程的非线性扩散滤波等，在本次实验中，我们要对图像添加噪声，并对添加噪声的图像进行基于模型的去噪。

4月12日（周一）布置

实验要求

A. 生成受损图像。

- 受损图像 (X) 是由原始图像 ($I \in \mathcal{R}^{H \times W \times C}$) 添加了不同噪声遮罩 (noise masks) ($M \in \mathcal{R}^{H \times W \times C}$) 得到的 ($X = I \odot M$)，其中 \odot 是逐元素相乘。
- 噪声遮罩仅包含 $\{0,1\}$ 值。对原图的噪声遮罩的可以每行分别用 0.8/0.4/0.6 的噪声比率产生的，即噪声遮罩每个通道每行 80%/40%/60% 的像素值为 0，其他为 1。

B. 使用你最擅长的算法模型，进行图像恢复。

C. 评估误差为所有恢复图像 (R) 与原始图像 (I) 的 2-范数之和，此误差越小越好。 $\text{error} = \sum_{i=1}^3 \text{norm}(R_i(:) - I_i(:), 2)$ ，其中 $(:)$ 是向量化操作，其他评估方式包括 Cosine 相似度以及 SSIM 相似度。