Contents

I 绪论

1. 机械学习定义

经典定义: 利用经验改善系统自身的性能.

宽泛定义: 以数据为经验的载体，利用经验数据不断提高性能的计算机系统/程序/算法

机械学习三大顶会: NeurIPS, ICML, IKL

II 基础术语与模型评估

1. 基本术语

1.1 数据

输入 → 输出.
机械学习是面向输出的技术.
术语扩展:

- 特征 (属性)
- 属性值: 离散或连续
- 样本维度: 特征个数
- 特征张成的空间 (属性空间/特征空间/输入空间)
- 标记张成的空间 (标记空间/输出空间)
- 示例/样本: 一个对象的输入 (不含标记)
- 样例: 示例 + 标记
- 训练集: 一组训练样例
- 测试集: 一组测试样例

1.2 任务

预测任务 : 根据标记取值的情况

- 分类任务: 标记为离散值
  二分类
  多分类
- 回归任务: 标记为连续值
- 聚类任务: 标记为空值, 对示例进行分组

根据标记完整性的情况

- 监督学习: 所有示例有标记
  e.g. 分类, 回归
- 无监督学习: 所有示例无标记
  e.g. 聚类
- 半监督学习: 少量示例有标记, 大量示例无标记

1.3 目标

泛化能力: 应对未来未见测试样本的能力.
I.I.D (独立同分布) 假设: 历史与未来来自相同分布

2. 概念学习

最理想的机器学习技术是学习到概念 (语言的边界就是思维的边界)

2.1 假设空间

假设空间: 枚举所有可能的假设
版本空间 (假设空间的子集): 跟训练集一致的 “假设集合”.

2.2 归纳偏好

Inductive bias: 学习过程中对某种类型假设的偏好称作归纳偏好

Theorem II.1 (No Free Lunch)**.** 一个算法 ζ_a 如果在某些问题上比另一个算法 ζ_b 好, 必然存在另一些问题, ζ_b 比 ζ_a 好.

因为未来数据是不知道的，总有一种未来数据的分布让你失败

证明. 假设样本空间 $\mathcal{X}$ 和假设空间 $\mathcal{H}$ 都是离散的. 令 $P(h|X,\mathfrak{L}_a)$ 代表算法 $\mathfrak{L}_a$ 基于训练数据 X 产生假设 h 的概率, 再令 f 代表我们希望学习的真实目标函数. $\mathfrak{L}_a$ 的 “训练集外误差”, 即 $\mathfrak{L}_a$ 在训练集之外的所有样本上的误差为

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

其中 $\mathbb{I}(\cdot)=\begin{cases}1 & \cdot \text{ is true}\\ 0 & \text{otherwise}\end{cases}$ 为指示函数.

以二分类问题为例, $\forall f:\mathcal{X}\mapsto\{0,1\}$, 函数空间 (所有可能映射的数量) 为 $\{0,1\}^{|\mathcal{X}|}$. 对所有 f 求和, 有

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

$\forall\mathfrak{L}_a,\mathfrak{L}_b$, 有

$$\sum_f E_{ote}(\mathfrak{L}_a|X,f)=\sum_f E_{ote}(\mathfrak{L}_b|X,f)$$

即不同学习算法的性能期望一致 (严谨证明更加复杂) QED

若 f 均匀分布，则有一半的 f 对 $\boldsymbol{x}$ 的预测与 $h(x)$ 不一致. 所以 $\sum_f\mathbb{I}(h(\boldsymbol{x})\neq f(\boldsymbol{x}))=\frac{1}{2}2^{|\mathcal{X}|}$
证明了总误差与学习算法无关.

3. 模型评估与选择

3.1 经验误差与过拟合

经验误差:

- 错误率 error rate: 错分样本的占比 $E = \frac{a}{m}$
- 误差 error: 样本真实输出与预测输出之间的差异
  训练误差: 训练集的误差
  测试误差: 测试集的误差

拟合:

- 过拟合: 将训练样本本身的特点, 当作所有样本的一般性质.
- 欠拟合: 对训练样本的一般性质尚未学好

3.2 评估方法

- 留出法 (hold-out): 分为两个互斥集合
- 交叉验证法 (cross-validation): 分为 k 个子集, 轮流当测试集.
- 自助法 (bootstrapping): 以一个概率采样样本, 允许重复, 适用于小数据.
- 在验证集上调参

3.3 性能度量

性能度量衡量泛化能力.
对于回归任务, 度量最常用的是均方误差:

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

对于分类任务:

- 错误率: 分错样本占总样本数的比例
- 精度: 分对样本占总样本数的比例

错误率 + 精度 =1

混淆矩阵

Table II.1: 混淆矩阵

	Relevant	Irrelevant
Retrieved	R_R	I_R
Not Retrieved	R_N	I_N

$$\text{Precision(查准率) } P = \frac{R_R}{R_R + I_R}$$
$$\text{Recall(查全率) } R = \frac{R_R}{R_R + R_N}$$

P-R 曲线 查准率-查全率曲线, 平衡点: Precision=Recall.

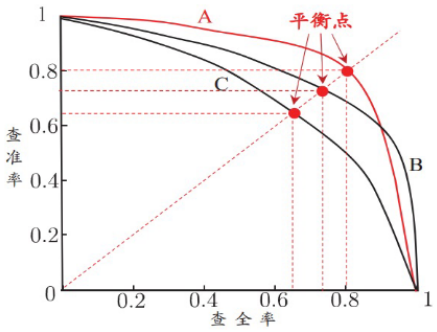

Figure II.1: P-R 曲线与平衡点

F_β **度量** 更常用

$$F_\beta = \frac{(1+\beta)^2 \times P \times R}{(\beta^2 \times P) + R}$$

- $\beta = 1$: 标准 F_1
- $\beta > 1$: 偏重 Recall
- $\beta < 1$: 偏重 Precision

其来自对 P, R 的加权调和平均:

$$\frac{1}{F_\beta} = \frac{1}{1+\beta^2}\left(\frac{1}{P} + \frac{\beta^2}{R}\right)$$

ROC 与 AUC ROC 全称受试者工作特征 (Receiver Operating Characteristic) 曲线. AUC 值: 衡量样本预测的排序质量.

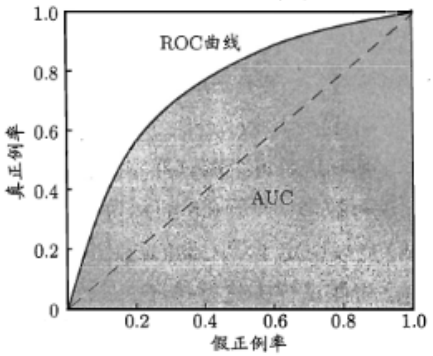

Figure II.2: ROC 曲线与 AUC

代价敏感错误率 为错误赋予 “非均等代价”(unequal cost).

3.4 比较检验

- 二项检验
- T-检验
- 交叉验证 T-检验

3.5 偏差与方差

偏差-方差分解帮助解释泛化性能.

对测试样本 $\boldsymbol{x}$, 令 y_D 为 $\boldsymbol{x}$ 在数据集中的标记, y 为 $\boldsymbol{x}$ 的真实标记, $f(\boldsymbol{x};D)$ 为训练集 D 上学得模型 f 在 $\boldsymbol{x}$ 上的预测输出. 以回归任务为例, 学习算法的期望预测为

$$\bar{f}(\boldsymbol{x}) = \mathbb{E}_D\left[f(\boldsymbol{x};D)\right]$$

使用样本数相同的不同训练集产生的方差为

$$var(\boldsymbol{x}) = \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right]$$

噪声为

$$\epsilon^2 = \mathbb{E}\left[(y_D - y)^2\right]$$

期望输出与真实标记的差别称为偏差 (bias), 即

$$bias^2(\boldsymbol{x}) = \left(\bar{f}(\boldsymbol{x}) - y\right)^2$$

为便于讨论, 假定噪声期望为零，即 $\mathbb{E}_D[y_D - y] = 0$. 通过简单的多项式展开合并，可对算法的期望泛化误差进行分解:

$$\begin{aligned}
E(f;D) =& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - y_D\right)^2\right] \\
=& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x}) + \bar{f}(\boldsymbol{x}) - y_D\right)^2\right] \\
=& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right] + \mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y_D\right)^2\right] \\
&+ \mathbb{E}_D\left[2\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)\left(\bar{f}(\boldsymbol{x}) - y_D\right)\right] \\
=& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right] \\
&+ \mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y + y - y_D\right)^2\right] \\
=& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right] \\
&+ \mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y\right)^2\right] + \mathbb{E}_D\left[\left(y - y_D\right)^2\right] \\
&+ \mathbb{E}_D\left[2\left(\bar{f}(\boldsymbol{x}) - y\right)\left(y - y_D\right)\right] \\
=& \mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)^2\right] \\
&+ \mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y\right)^2\right] + \mathbb{E}_D\left[\left(y - y_D\right)^2\right] \\
=& bias^2(\boldsymbol{x}) + var(\boldsymbol{x}) + \epsilon^2
\end{aligned}$$

这里, 因为 $\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)$ 与 $\left(\bar{f}(\boldsymbol{x}) - y_D\right)$ 可能是独立的, 所以有

$$\begin{aligned}
&\mathbb{E}_D\left[2\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)\left(\bar{f}(\boldsymbol{x}) - y_D\right)\right] \\
=& 2\mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - \bar{f}(\boldsymbol{x})\right)\right]\mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y_D\right)\right] \\
=& 2\mathbb{E}_D\left[\left(f(\boldsymbol{x};D) - f(\boldsymbol{x};D)\right)\right]\mathbb{E}_D\left[\left(\bar{f}(\boldsymbol{x}) - y_D\right)\right] = 0
\end{aligned}$$

同理, 因为 $\mathbb{E}_D[y_D - y] = 0$,

$$\mathbb{E}_D\left[2\left(\bar{f}(\boldsymbol{x}) - y\right)\left(y - y_D\right)\right] = 0$$

泛化误差可分解为偏差、方差与噪声之和. 方差越往大, 越过拟合.

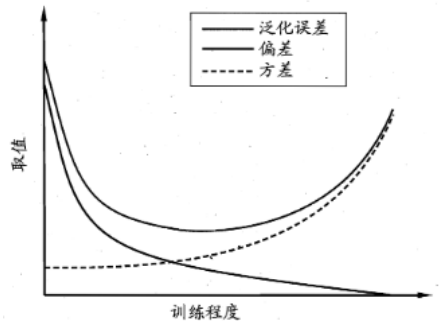

Figure II.3: 泛化误差与偏差, 方差的关系示意图

III 线性模型

1. 引言

1.1 基本形式

一般形式:

$$f(\boldsymbol{x}) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b$$

向量形式:

$$f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$$

其中, $\boldsymbol{w} = \begin{pmatrix} w_1 \\ w_2 \\ \cdots \\ w_d \end{pmatrix}, \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_d \end{pmatrix}$

1.2 Perceptron 感知机

对于线性分类器, 误分类则:

$$-y_i(wx_i + b) > 0$$

定义损失函数

$$L(w,b) = -\sum_{x_i \in M} y_i(wx_i + b)$$

梯度

$$\begin{aligned} \nabla_w L(w,b) &= -\sum_{x_i \in M} y_i x_i \\ \nabla_b L(w,b) &= -\sum_{x_i \in M} y_i \\ \therefore\ w &:= w + \eta y_i x_i \\ b &:= b + \eta y_i \end{aligned}$$

1.3 梯度下降

- 一阶方法
- 无约束优化

考虑无约束优化问题 $\min_{\boldsymbol{x}} f(\boldsymbol{x})$, 其中 $f(\boldsymbol{x})$ 连续可微, 若能构造 $\boldsymbol{x}^0, \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots$ 满足

$$f(\boldsymbol{x}^{t+1}) < f(\boldsymbol{x}^t), t = 0, 1, 2, \ldots$$

有

$$\begin{aligned} f(x + \Delta x) &= f(x) + \Delta x^\top \nabla f(x) \\ \Delta x^\top \nabla f(x) &< 0 \\ \therefore\ \Delta x &= -\gamma \nabla f(x) \end{aligned}$$

1.4 优缺点

优点:

- 简单
- 可解释
- 基础

缺点:

- 有线性不可分的数据

2. 回归任务

2.1 线性回归

给定数据集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, 其中 $\boldsymbol{x}_i = (x_{i1}; x_{i2}; \ldots; x_{id})$, $y_i \in \mathbb{R}$.

试图学得一个线性模型以尽可能准确地预测实值输出标记.

离散属性处理:

- 有序: 化为连续值
- 无序: 转化为 k 维向量

单属性: $f(x) = wx_i + b$, 有 $f(x_i) \simeq y_i$

最小二乘法 基于均方误差最小化来进行模型求解

$$\begin{aligned} (w^*, b^*) &= \underset{(w,b)}{\operatorname{argmin}} \sum_i^m (f(x_i) - y_i)^2 \\ &= \underset{(w,b)}{\operatorname{argmin}} \sum_i^m (y_i - wx_i - b)^2 \end{aligned}$$

均方误差

$$E_{(w,b)} = \sum_i^m (y_i - wx_i - b)^2$$

对 w 与 b 求导

$$\begin{aligned} \frac{\partial E_{(w,b)}}{\partial w} &= \sum_i^m 2x_i (y_i - wx_i - b) \\ &= 2\left(w\sum_i^m x_i^2 - b\sum_i^m x_i + \sum_i^m x_i y_i\right) \\ \frac{\partial E_{(w,b)}}{\partial b} &= \sum_i^m 2 (y_i - wx_i - b) \\ &= 2\left(-w\sum_i^m x_i - mb + \sum_i^m y_i\right) \end{aligned}$$

令二者为 0, 可得

$$w=\left(\frac{\sum_{i=1}^{m}y_ix_i-\sum_{i=1}^{m}y_i\sum_{i=1}^{m}x_i}{\sum_{i=1}^{m}x_i^2-\frac{1}{m}\left(\sum_{i=1}^{m}x_i\right)^2}\right)$$

$$b=\frac{1}{m}\sum_{i=1}^{m}(y_i-wx_i)$$

2.2 多元线性回归

给定数据集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\dots,(\boldsymbol{x}_m,y_m)\}$, 其中 $\boldsymbol{x}_i=(x_{i1};x_{i2};\dots;x_{id})$, $y_i\in\mathbb{R}$.

目标: $f(\boldsymbol{x}_i)=\boldsymbol{w}^\top\boldsymbol{x}_i+b$, 有 $f(\boldsymbol{x}_i)\simeq y_i$

齐次表达 把 $\boldsymbol{w}$ 和 b 吸收入向量形式 $\hat{\boldsymbol{w}}=(\boldsymbol{w};b)$, 数据集表示为

$$\boldsymbol{X}=\begin{pmatrix}x_{11}&x_{12}&\cdots&x_{1d}&1\\x_{21}&x_{22}&\cdots&x_{2d}&1\\\vdots&\vdots&\ddots&\vdots&\vdots\\x_{m1}&x_{m2}&\cdots&x_{md}&1\end{pmatrix}=\begin{pmatrix}\boldsymbol{x}_1^\top&1\\\boldsymbol{x}_2^\top&1\\\vdots&\vdots\\\boldsymbol{x}_m^\top&1\end{pmatrix}$$

$$\boldsymbol{y}=(y_1;y_2;\dots;y_m)$$

最小二乘 类似

$$\hat{\boldsymbol{w}}^*=\underset{\hat{\boldsymbol{w}}}{\operatorname{argmin}}(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{w}})^\top(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{w}})$$

令 $E_{\hat{\boldsymbol{w}}}=(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{w}})^\top(\boldsymbol{y}-\boldsymbol{X}\hat{\boldsymbol{w}})$, 对 $\hat{\boldsymbol{w}}$ 求导有

$$\frac{\partial E_{\hat{\boldsymbol{w}}}}{\partial\hat{\boldsymbol{w}}}=2\boldsymbol{X}^\top(\boldsymbol{X}\hat{\boldsymbol{w}}-\boldsymbol{y})$$

令其为 0 可得 $\hat{\boldsymbol{w}}$ 最优解.

满秩讨论 最优解涉及矩阵求逆, 比较复杂, 需要讨论.

若 $\boldsymbol{X}^\top\boldsymbol{X}$ 为满秩或正定矩阵, 则

$$\hat{\boldsymbol{w}}^*=(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$$

线性回归模型为

$$f(\hat{\boldsymbol{x}}_i)=\hat{\boldsymbol{x}}_i^\top(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$$

若 $\boldsymbol{X}^\top\boldsymbol{X}$ 不是满秩, 需要引入正则化.

2.3 对数线性回归

输出标记的对数为线性模型逼近的目标

$$\ln y=\boldsymbol{w}^\top\boldsymbol{x}+b$$

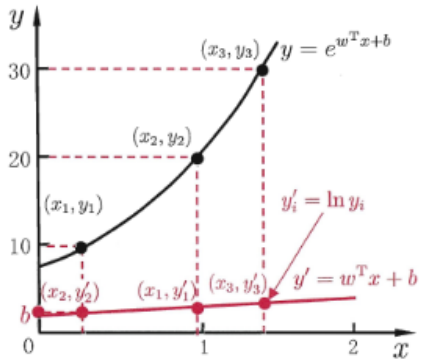

Figure III.1: 对数线性回归示意图

2.4 广义线性模型

考虑单调可微函数 $g(\cdot)$, 令

$$y=g^{-1}\left(\boldsymbol{w}^\top\boldsymbol{x}+b\right)$$

称 g 为联系函数 (link function)

3. 二分类任务

预测值与输出标记

$$z=\boldsymbol{w}^\top\boldsymbol{x}+b$$
$$y\in\{0,1\}$$

寻找函数将分类标记与线性回归模型输出联系起来.

最理想的函数 — 单位阶跃函数

$$y=\begin{cases}0&z<0\\0.5&z=0\\1&z>0\end{cases}$$

但其不连续

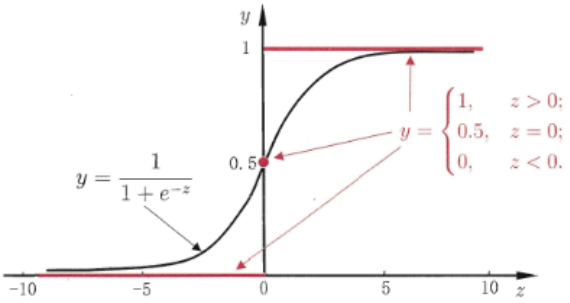

Figure III.2: 单位阶跃函数与对数几率函数

3.1 对数几率回归

对数几率函数 (logistic function) 于是使用对数几率函数替代.

$$y = \frac{1}{1+e^{-z}}$$
$$y = \frac{1}{1+e^{-(\boldsymbol{w}^\top \boldsymbol{x}+b)}}$$
$$\ln \frac{y}{1-y} = \boldsymbol{w}^\top \boldsymbol{x} + b$$

Definition III.1 (对数几率 (log odds)). *若将 y 视为样本 $\boldsymbol{x}$ 作为正例的可能性, 则 $1-y$ 是其反例可能性, 两者的比值称为 "几率"(odds), 反映了 $\boldsymbol{x}$ 作为正例的相对可能性. 对几率取对数则得到 "对数几率"(log odds, 亦称 logit)*

$$\ln \frac{y}{1-y}$$

优势:

1) 无需假设数据分布
2) 可得 "类别" 的近似概率预测
3) 可直接应用现有数值优化算法求最优解

极大似然法 来估计 $\boldsymbol{w}$ 与 b.

若将 y 视为类后验概率估计 $p(y=1|\boldsymbol{x})$, 有

$$\ln \frac{p(y=1|\boldsymbol{x})}{p(y=0|\boldsymbol{x})} = \boldsymbol{w}^\top \boldsymbol{x} + b$$
$$p(y=1|\boldsymbol{x}) = \frac{e^{(\boldsymbol{w}^\top \boldsymbol{x}+b)}}{1+e^{(\boldsymbol{w}^\top \boldsymbol{x}+b)}}$$
$$p(y=0|\boldsymbol{x}) = \frac{1}{1+e^{(\boldsymbol{w}^\top \boldsymbol{x}+b)}}$$

给定数据集 $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, 最大化对数似然函数

$$\ell(\boldsymbol{w}, b) = \sum_{i=1}^m \ln p(y_i|\boldsymbol{x}_i; \boldsymbol{w}_i, b)$$

令 $\boldsymbol{\beta} = (\boldsymbol{w}; b), \hat{\boldsymbol{x}} = (\boldsymbol{x}, 1)$, $\boldsymbol{w}^\top \boldsymbol{x} + b = \boldsymbol{\beta}^\top \hat{\boldsymbol{x}}$

$$p_1(\hat{\boldsymbol{x}}; \boldsymbol{\beta}) = p(y=1|\hat{\boldsymbol{x}}; \boldsymbol{\beta})$$
$$p_0(\hat{\boldsymbol{x}}; \boldsymbol{\beta}) = p(y=0|\hat{\boldsymbol{x}}; \boldsymbol{\beta}) = 1 - p_1(\hat{\boldsymbol{x}}; \boldsymbol{\beta})$$
$$p(y_i|\boldsymbol{x}_i; \boldsymbol{w}, b) = y_i p_1(\hat{\boldsymbol{x}}; \boldsymbol{\beta}) + (1-y_i) p_0(\hat{\boldsymbol{x}}; \boldsymbol{\beta})$$

因为 $y_i = 0, 1$, 有

$$\ell(\boldsymbol{\beta}) = \begin{cases} \boldsymbol{\beta}^\top \hat{\boldsymbol{x}}_i - \ln\left(1+e^{\boldsymbol{\beta}^\top \hat{\boldsymbol{x}}_i}\right) & y_i = 1 \\ -\ln\left(1+e^{\boldsymbol{\beta}^\top \hat{\boldsymbol{x}}_i}\right) & y_i = 0 \end{cases}$$
$$= \sum_{i=1}^m \left(y_i \boldsymbol{\beta}^\top \hat{\boldsymbol{x}}_i - \ln\left(1+e^{\boldsymbol{\beta}^\top \hat{\boldsymbol{x}}_i}\right)\right)$$

关于 $\boldsymbol{\beta}$ 的高阶可导连续凸函数, 可以使用经典的数值优化算法求解得

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} -\ell(\boldsymbol{\beta})$$

对数几率回归总结 主要思想:

- 引入 sigmod, 关联了离散标记与线性模型
- sigmod 光滑, 高阶可导, 接近离散标记
- 得到类别接近概率似然估计
- 构建极大似然目标函数
- 优化求解: 梯度下降, 牛顿法等

3.2 线性判别分析 (LDA)

(Linear Discriminant Analysis)LDA 视为一种监督降维技术.

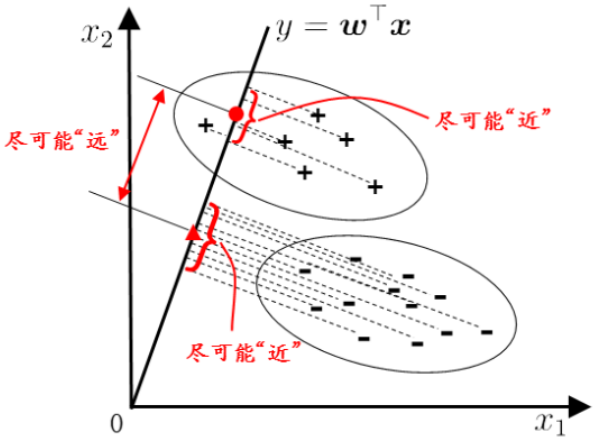

Figure III.3: LDA 的二维示意图

给定训练样例集, 设法将样例投影到一条直线上, 使得同类样例的投影点尽可能接近、异类样例的投影点尽可能远离.

相关变量:

- $\boldsymbol{w}$: 目标直线
- X_i: 第 i 类示例的集合
- $\boldsymbol{\mu}_i$: 第 i 类示例的均值向量
- Σ_i: 第 i 类示例的协方差矩阵

$$\Sigma_i = \sum_{\boldsymbol{x} \in X_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^\top$$

- $\boldsymbol{w}^\top \boldsymbol{\mu}_0$, $\boldsymbol{w}^\top \boldsymbol{\mu}_1$: 两类样本中心在直线上的投影.
- $\boldsymbol{w}^\top \Sigma_0 \boldsymbol{w}$, $\boldsymbol{w}^\top \Sigma_1 \boldsymbol{w}$: 两类样本的协方差

最大化目标:

$$J = \frac{\left\|\boldsymbol{w}^\top \boldsymbol{\mu}_0 - \boldsymbol{w}^\top \boldsymbol{\mu}_1\right\|_2^2}{\boldsymbol{w}^\top \Sigma_0 \boldsymbol{w} + \boldsymbol{w}^\top \Sigma_1 \boldsymbol{w}}$$
$$= \frac{\boldsymbol{w}^\top (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^\top \boldsymbol{w}}{\boldsymbol{w}^\top (\Sigma_0 + \Sigma_1) \boldsymbol{w}}$$

定义类内散度矩阵与类间散度矩阵:

$$\begin{aligned} S_w &= \Sigma_0 + \Sigma_1 \\ &= \sum_{\boldsymbol{x}\in X_0} (\boldsymbol{x}-\boldsymbol{\mu}_0)(\boldsymbol{x}-\boldsymbol{\mu}_0)^\top + \sum_{\boldsymbol{x}\in X_1} (\boldsymbol{x}-\boldsymbol{\mu}_1)(\boldsymbol{x}-\boldsymbol{\mu}_1)^\top \\ S_b &= (\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^\top \end{aligned}$$

于是最大化目标可重写为 S_b 与 S_w 的广义瑞利商 (generalized Rayleigh quotient)

$$J = \frac{\boldsymbol{w}^\top S_b \boldsymbol{w}}{\boldsymbol{w}^\top S_w \boldsymbol{w}}$$

令 $\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w} = 1$ 最大化上式等价形式为

$$\begin{aligned} &\min_{\boldsymbol{w}} -\boldsymbol{w}^\top S_b \boldsymbol{w} \\ &\text{s.t. } \boldsymbol{w}^\top S_w \boldsymbol{w} = 1 \end{aligned}$$

运用拉格朗日乘子法

$$S_b \boldsymbol{w} = \lambda S_w \boldsymbol{w}$$

拉格朗日乘子法 用了个对偶问题, 略过过程, 最后有

$$\boldsymbol{w} = S_w^{-1}(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)$$

使用奇异值分解求解, i.e. $S_w = U\Sigma V^\top$, 然后得到 $S_w^{-1} = V\Sigma^{-1}U^\top$.

LDA 也可以用贝叶斯决策论解释.

LDA 推广—多分类任务 假定存在 N 个类, 且第 i 类示例数位 m_i.

定义全局散度矩阵

$$\begin{aligned} S_t &= S_b + S_w \\ &= \sum_{i=1}^{m} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^\top \end{aligned}$$

其中 $\boldsymbol{\mu}$ 是所有示例的均值向量.

再重定义类内散度矩阵, 类间散度矩阵

$$\begin{aligned} S_w &= \sum_{i=1}^{N} S_{w_i} \\ &= \sum_{i=1}^{N} \left(\sum_{\boldsymbol{x}\in X_i} (\boldsymbol{x}-\boldsymbol{\mu}_i)(\boldsymbol{x}-\boldsymbol{\mu}_i)^\top \right) \\ S_b &= S_t - S_w \\ &= \sum_{i=1}^{N} m_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^\top \end{aligned}$$

使用 S_b, S_w, S_t 三者中的任何两个即可, 常见的是采用优化目标

$$\max_W \frac{tr(W^\top S_b W)}{W^\top S_w W}$$

其中 $W \in \mathbb{R}^{d\times(N-1)}$, 通过如下广义特征值问题求解

$$S_b W = \lambda S_w W$$

W 的闭式解则是 $S_w^{-1}S_b$ 的 $N-1$ 个最大广义特征值所对应的特征向量组成的矩阵.

多分类 LDA 将样本投影到 $N-1$ 维空间, $N-1$ 通常远小于数据原有的属性数, 因此 LDA 也常被视为一种经典的监督降维技术.

4. 多分类任务

常用技巧: 利用二分类学习器解决多分类问题.

4.1 一对一

4.2 一对其余

4.3 比较一对一与一对其余

4.4 多对多

Many vs Many (MvM): 若干类为正类, 若干类为反类.

纠错输出码 (Error Correcting Output Code, ECOC)

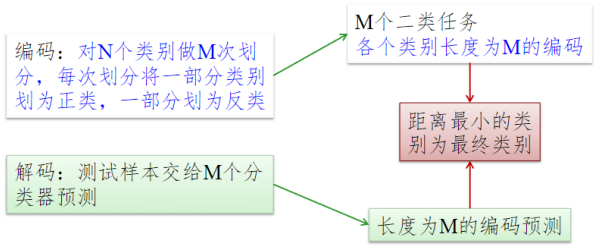

Figure III.4: ECOC

5. 类别不平衡任务

IV 决策树

1. 决策树简介 (基本流程)

处理未见的能力

输入: 训练集 D, 属性集 A

递归构建决策树, 三个条件停止递归:

1) 当前结点包含的样本全部属于同一类别, 无需划分
2) 当前属性集为空, 或所有样本在所有属性上取值相同, 无法划分
3) 当前结点包含的样本集合为空, 不能划分

2. 决策树算法的关键：划分选择

最优划分属性, 希望纯度 (purity) 愈来愈高.

2.1 信息增益

“信息熵” (information entropy) 是度量样本集合纯度最常用的一种指标. 假定当前样本集合。中第 k 类样本所占的比例为 $p_k(k=1,2,\dots,|\mathcal{Y}|)$, 则 D 的信息熵定义为

$$Ent(D)=\sum_{k=1}^{|\mathcal{Y}|}p_k\log_2 p_k$$

$Ent(D)$ 越小, D 纯度越高. $0\le Ent(D)\le\log_2|\mathcal{Y}|$. 计算时约定: $p=0$ 时, $p\log_2 p=0$.

假定离散属性 a 有 V 个可能的取值 $\{a^1,a^2,\dots,a^V\}$, 若使用 a 来对样本集 D 进行划分, 则会产生 V 个分支结点, 其中第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本, 记为 D^v. 再考虑到不同的分支结点所包含的样本数不同, 给分支结点赋予权重 $\frac{|D^v|}{|D|}$, 即样本数越多的分支结点的影响越大, 于是可计算出用属性 a 对样本集 D 进行划分所获得的 “信息增益”(information gain)

$$Gain(D,a)=Ent(D)-\sum_{i=1}^{V}\frac{|D^v|}{|D|}Ent(D^v)$$

一般而言, 信息增益越大, 则意味着使用属性 a 来进行划分所获得的 “纯度提升” 越大. 因此, 我们可用信息增益来进行决策树的划分属性选择, 即选择属性 $a_*=\underset{a\in A}{\operatorname{argmin}}\,Gain(D,a)$. 著名的 ID3 决策树学习算就是以信息增益为准则来选择划分属性.

偏向于类目多的属性.

2.2 增益率

为减少这种偏好可能带来的不利影响，著名的 C4.5 决策树算法使用 “增益率”(gain ratio) 来选择最优划分属性, 增益率定义为

$$Gain\_ratio(D,a)=\frac{Gain(D,a)}{IV(a)}$$

$$IV(a)=-\sum_{v=1}^{V}\frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

其中 $IV(a)$ 称为属性 a 的 “固有值” (intrinsic value). 属性 a 的可能取值数目越多 (即 V 越大)，则 $IV(a)$ 的值通常会越大.

偏向于类目少的属性, 相当于给增益做了规范化.

2.3 基尼系数

CART 决策树使用 “基尼指数”(Gini index) 来选择划分属性.

$$Gini(D)=\sum_{k=1}^{|\mathcal{Y}|}\sum_{k'\ne k}p_kp_{k'}=1-\sum_{k=1}^{\mathcal{Y}}p_k^2$$

反映了从数据集 D 中随机抽取两个样本，其类别标记不一致的概率. 因此, $Gini(D)$ 越小，则数据集 D 的纯度越高.

属性 a 的基尼指数定义为

$$Gini\_index(D,a)=\sum_{v=1}^{V}\frac{|D^v|}{|D|}Gini(D^v)$$

在候选属性集合 A 中，选择那个使得划分后基尼指数最小的属性作为最优划分属性，即 $a_*=\underset{a\in A}{\operatorname{argmin}}\,Gini\_index(D,a)$

3. 克服过拟合的问题：剪枝处理

没有先验, 非常易过拟合.

3.1 预剪枝

边建树, 边剪枝

划分验证集, 若划分后验证集精度下降则不进行划分.

优点:

- 降低过拟合
- 减少训练与测试时间

缺点: 基于贪心, 可能欠拟合

3.2 后剪枝

先建树, 后剪枝

优点: 后剪枝比预剪枝保留了更多的分支，欠拟合风险小，泛化性能往往优于预剪枝决策树

缺点: 训练时间开销大：后剪枝过程是在生成完全决策树之后进行的，需要自底向上对所有非叶结点逐一考察；其训练时间要远大于预剪枝决策树

4. 处理多种类型数据：连续与缺失值

4.1 连续值离散化 *(二分法)*

1) 假定 a 在 D 上出现了 n 个不同的取值, 从小到大进行排序，记为 $\{a^1, a^2, \dots, a^n\}$, 基于划分点 t 可将 D 分为子集 D_t^- 和 D_t^+. 其中 D_t^- 包含那些在属性 a 上取值不大于 t 的样本，而 D_t^+ 则包含那些在属性 a 上取值大于 t 的样本. 考察包含 $n-1$ 个元素的候选划分点集合

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \middle| 1 \le i \le n-1 \right\}$$

即把区间 $[a^i, a^{i-1})$ 的中位点作为候选划分点.

2) 考察这些划分点, 选取最优划分点进行样本集合的划分.

与离散属性不同，若当前结点划分属性为连续属性，该属性还可作为其后代结点的划分属性.

4.2 缺失值

两个问题:

1) 如何在属性值缺失的情况下进行划分属性选择?

跟传统决策树一致, 只不过仅在有属性值的子集上计算信息增益, 不考虑无属性值的样本

2) 给定划分属性, 若样本在该属性上的值缺失, 如何对样本进行划分？

就是让同一个样本以不同的概率划入到不同的子节点中去

给定训练集 D 和属性 a, 令 $\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集. 显然我们仅可根据 $\tilde{D}$ 来判断属性 a 的优劣. 假定属性 a 有 V 个可取值 $\{a^1, a^2, \dots, a^V\}$, 令 $\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集, $\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类 $(k = 1, 2, \dots, |\mathcal{Y}|)$ 的样本子集，则显然有 $\tilde{D} = \bigcup_{k=1}^{|\mathcal{Y}|} \tilde{D}_k$, $\tilde{D} = \bigcup_{v=1}^{V} \tilde{D}_k$. 定我们为每个样本 $\boldsymbol{x}$ 赋予一个权重 $w_{\boldsymbol{x}}$ 并定义

$$\begin{aligned}
\rho &= \frac{\sum_{x \in \tilde{D}} w_{\boldsymbol{x}}}{\sum_{x \in D} w_{\boldsymbol{x}}} \\
\tilde{p}_k &= \frac{\sum_{x \in \tilde{D}_k} w_{\boldsymbol{x}}}{\sum_{x \in D} w_{\boldsymbol{x}}} \ (1 \le k \le |\mathcal{Y}|) \\
\tilde{r}_v &= \frac{\sum_{x \in \tilde{D}^v} w_{\boldsymbol{x}}}{\sum_{x \in D} w_{\boldsymbol{x}}} \ (1 \le v \le V)
\end{aligned}$$

对属性 a, ρ 表示无缺失值样本所占的比例, $\tilde{p}_k$ 表示无缺失值样本中第 k 类所占的比例，$\tilde{r}_v$ 则表示无缺失值样本中在属性 a 上取值 a^v 的样本所占的比例.

基于上述定义，我们可将信息增益的计算式推广为

$$\begin{aligned}
Gain(D, a) &= \rho \times Gain(\tilde{D}, a) \\
&= \rho \times \left(Ent(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v Ent(\tilde{D}^v) \right) \\
Ent(\tilde{D}) &= -\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k
\end{aligned}$$

跟传统决策树一致，只不过仅在有属性值的子集上计算信息增益，不考虑无属性值的样本.

若样本 $\boldsymbol{x}$ 在划分属性 a 上的取值未知，则将 $\boldsymbol{x}$ 同时划入所有子结点，且样本权值在与属性值 a^v 对应的子结点中调整为 $\tilde{r}_v \cdot w_{\boldsymbol{x}}$

5. 决策树的变体：多变量决策树

非叶节点不再是仅对某个属性, 而是对属性的线性组合. 多变量本质就是模型, 相当于叶节点是小模型.

V 神经网络

1. 神经元模型

Definition V.1. 神经网络是由具有适应性的简单单元组成的广泛并行互联的网络, 它的组织能够模拟生物神经系统对真实世界物体所作出的反应.

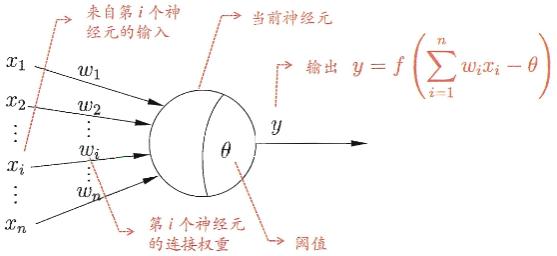

Figure V.1: M-P 神经元模型

输入-> 处理-> 激活函数-> 输出

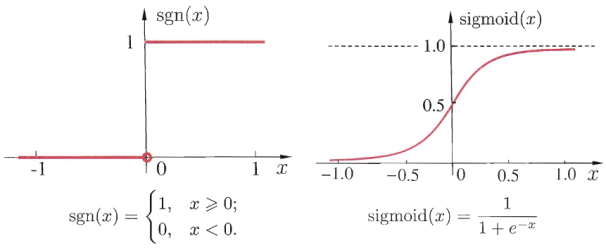

Figure V.2: 典型的神经元激活函数

2. 感知机与多层网络

2.1 感知机

感知机 (Perceptron) 由两层神经元组成, 输入层接收外界输入信号后传递给输出层，输出层是 M-P 神经元. 感知机能容易地实现逻辑与、或、非运算.

给定训练数据集, 权重 $w_i(i=1,2,\ldots,n)$ 以及阈值 θ 可通过学习得到.

两类模式是线性可分的，则感知机的学习过程一定会收敛, 否则感知机学习过程将会发生振荡. 单层感知机的学习能力非常有限, 只能解决线性可分问题.

2.2 多层感知机

输出层与输入层之间的一层神经元, 被称之为隐层或隐含层, 隐含层和输出层神经元都是具有激活函数的功能神经元.

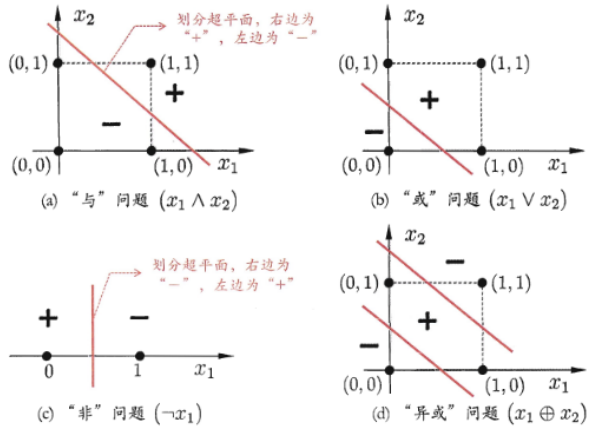

Figure V.3: 线性可分的“与“或”“非”问题与非线性可分的“异或”问题

2.3 多层前馈神经网络

定义：每层神经元与下一层神经元全互联, 神经元之间不存在同层连接也不存在跨层连接.

前馈：输入层接受外界输入, 隐含层与输出层神经元对信号进行加工, 最终结果由输出层神经元输出

学习：根据训练数据来调整神经元之间的“连接权”以及每个功能神经元的“阈值”

3. 误差逆传播算法

误差逆传播 (Error BackPropagation, 简称 BP) 算法是迄今最成功的神经网络学习算法.

给定训练集 $D=\{(\boldsymbol{x}_1,\boldsymbol{y}_1),(\boldsymbol{x}_2,\boldsymbol{y}_2),\ldots,(\boldsymbol{x}_m,\boldsymbol{y}_m)\}$, $\boldsymbol{x}_i\in\mathbb{R}^d$, $\boldsymbol{y}_i\in\mathbb{R}^l$, 即输入示例由 d 个属性描述, 输出 l 维实值向量. 以拥有 d 个输入神经元、, l 个输出神经元、q 个隐层神经元的多层前馈网络结构为例

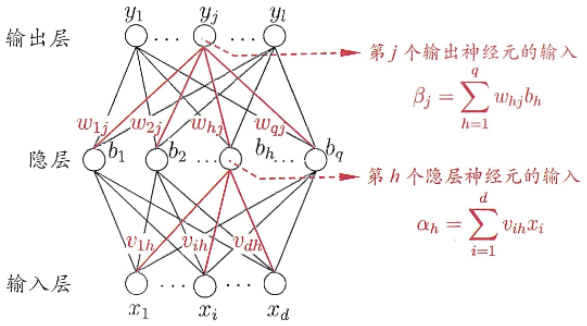

Figure V.4: BP 网络及算法中的变量符号

- θ_j: 输出层第 j 个神经元阈值
- γ_h: 隐含层第 h 个神经元阈值
- v_{ih}: 输入层与隐层神经元之间的连接权重

- w_{hj}: 隐层与输出层神经元之间的连接权重

BP 算法基于梯度下降 (gradient descent) 策略，以目标的负梯度方向对参数进行调整. 误差 E_k 给定学习率 η

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}$$

标准 BP 与累计 BP

多层前馈网络表示能力: 只需一个包含足够多神经元的隐层，多层前馈网络就能以任意精度逼近任意复杂度的连续函数.

缓解过拟合的策略:

- 早停
- 正则化 (惩罚复杂性)

4. 全局最小与局部最小

定义了全局最小与局部最小
以及跳出局部最小的方法

5. 其他常见神经网络

- RBF
- ART
- SOM
- 级联相关
- Elman
- Boltzmann

6. 深度学习

典型的深度学习模型就是很深层的神经网络

VI 支持向量机

1. 间隔与支持向量

给定训练样本集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, $y_i \in \{-1, +1\}$.

划分超平面方程:

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

其中 $\boldsymbol{w}$ 为法向量, 决定了超平面的方向, b 为位移项, 决定了超平面与原点之间的距离. 记 $(\boldsymbol{w}, b)$ 为超平面. 样本空间中任意 $\boldsymbol{w}$ 到超平面 $(\boldsymbol{w}, b)$ 的距离为

$$r = \frac{|\boldsymbol{w}^\top \boldsymbol{x} + b|}{\|\boldsymbol{w}\|}$$

假设超平面划分正确, 有

$$\begin{cases} \boldsymbol{w}^\top \boldsymbol{x}_i + b \geq +1 & y_i = +1 \\ \boldsymbol{w}^\top \boldsymbol{x}_i + b \leq -1 & y_i = -1 \end{cases}$$

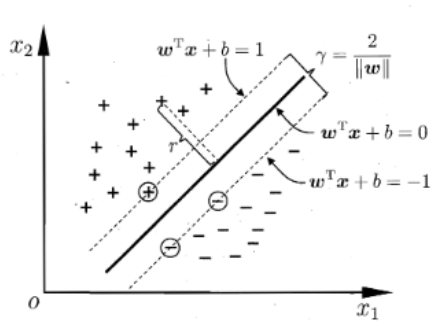

Figure VI.1: 间隔与支持向量

如图所示, 距离超平面最近的这几个训练样本点使等号成立, 它们被称为“支持向量”(support vector), 两个异类支持向量到超平面的距离之和被称为“间隔”(margin), 为

$$\gamma = \frac{2}{\|\boldsymbol{w}\|}$$

最大间隔: 寻找 $\boldsymbol{w}, b$, 使 γ 最大.

$$\max_{\boldsymbol{w}, b} \frac{2}{\|\boldsymbol{w}\|}$$
$$\text{s.t. } y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, i = 1, 2, \ldots, m$$

等价于

$$\min_{\boldsymbol{w}, b} \frac{1}{2} \|\boldsymbol{w}\|^2$$
$$\text{s.t. } y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, i = 1, 2, \ldots, m$$

这就是支持向量机 (Support Vector Machine, 简称 SVM) 的基本型.

2. 对偶问题

用拉格朗日乘子法可得到其“对偶问题”(dual problem).

1) 引入拉格朗日乘子 $a_i \geq 0$ 得到拉格朗日函数

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{m} \alpha_i \left(1 - y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right)\right)$$

其中 $\boldsymbol{\alpha} = (\alpha_1; \alpha_2; \ldots; \alpha_m)$

2) 令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 对 $\boldsymbol{w}, b$ 的偏导为 0

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i$$

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

3) 回代

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i^\top\right)\left(\sum_{j=1}^{m} \alpha_j y_j \boldsymbol{x}_j\right)$$

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

这个优化问题可以使用 SMO 方法求解.

最终模型:

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

因为有不等式约束, 所以需要满足 KKT 条件, 即

$$\begin{cases} \alpha_i \geq 0 \\ y_i f(\boldsymbol{x}_i) - 1 \geq 0 \\ \alpha(y_i f(\boldsymbol{x}_i) - 1) = 0 \end{cases}$$

支持向量机解的稀疏性: 训练完成后，大部分的训练样本都不需保留，最终模型仅与支持向量有关

2.1 SMO 方法

不断执行如下两个步骤直至收敛:

1) 取一对需更新的变量 α_i, α_j

2) 固定 α_i, α_j 以外的参数, 求解优化问题获得更新后的 α_i, α_j.

仅考虑 α_i 与 α_j 时, 对偶问题约束变为

$$\alpha_i y_i + \alpha_j y_j = -\sum_{k \neq i,j} a_k y_k, \alpha_i \geq 0, \alpha_j \geq 0$$

用一个变量表示另一个变量, 回代入对偶问题可. 得一个单变量的二次规划, 该问题具有闭式解.

偏移项 b：通过支持向量来确定

3. 核函数

若不存在一个能正确划分两类样本的超平面, 则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.

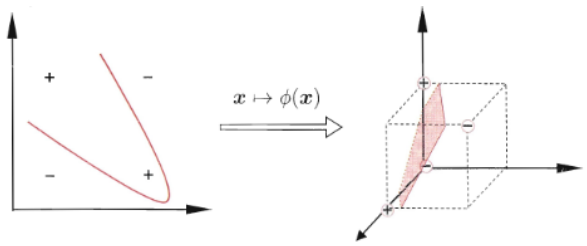

Figure VI.2: 异或问题与非线性映射

令 $\phi(\boldsymbol{x})$ 表示将 $\boldsymbol{x}$ 映射后的特征向量, 划分超平面为 $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x}) + b$.

原始问题:

$$\min_{\boldsymbol{w}, b} \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t. } y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) + b) \geq 1, i = 1, 2, \ldots, m$$

对偶问题:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\left(\sum_{i=1}^{m} \alpha_i y_i \phi(\boldsymbol{x}_i)^\top\right)\left(\sum_{j=1}^{m} \alpha_j y_j \phi(\boldsymbol{x}_j)\right) - \sum_{i=1}^{m} \alpha_i$$

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

预测:

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

不显示设计核映射, 而是设计核函数

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$$

Theorem VI.1 (Mercer)**.** 只要一个对称函数所对应的核矩阵半正定,. 它就能作为核函数使用.

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

Figure VI.3: 常用核函数

4. 软间隔与正则化

4.1 软间隔

在现实任务中往往很难确定合适的核函数使得训练样本在特征空间中线性可分, 同时一个线性可分的结果也很难断定是否是有过拟合造成的.

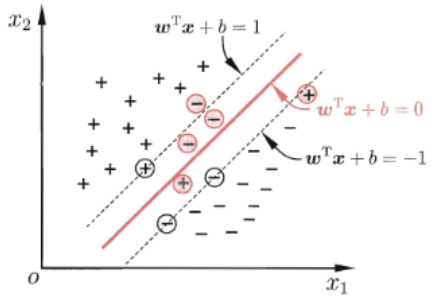

Figure VI.4: 软间隔示意图

引入”软间隔”的概念, 允许支持向量机在一些样本上不满足约束.

基本想法：最大化间隔的同时, 让不满足约束的样本应尽可能少

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m} \ell_{0/1}\left(y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right) - 1\right)$$

其中 $C>0$ 为常数, $\ell_{0/1}$ 是 0/1 损失函数

$$\ell_{0/1}(z) = \begin{cases} 1 & \text{if } z<0 \\ 0 & \text{otherwise} \end{cases}$$

但其非凸, 非连续.

于是替代损失函数, 就是软间隔支持向量机, 引入松弛变量 $\zeta_i \geq 0$, 有

原始问题:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\zeta_i$$

$$\text{s.t. } y_i(\boldsymbol{w}^\top \phi(\boldsymbol{x}_i)+b) \geq 1-\zeta_i, \zeta_i \geq 0, i=1,2,\ldots,m$$

对偶问题:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\left(\sum_{i=1}^{m}\alpha_i y_i \phi(\boldsymbol{x}_i)^\top\right)\left(\sum_{j=1}^{m}\alpha_j y_j \phi(\boldsymbol{x}_j)\right) - \sum_{i=1}^{m}\alpha_i$$

$$\text{s.t. } \sum_{i=1}^{m}\alpha_i y_i = 0, 0 \leq \alpha_i \leq C, i=1,2,\ldots,m$$

预测:

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

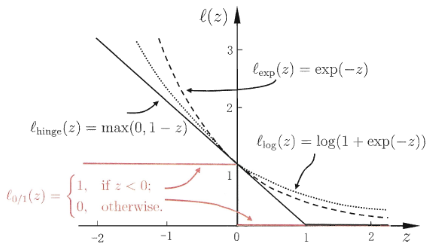

Figure VI.5: 三种常见的替代损失函数: hinge 损失、指数损失、对率损失

软间隔支持向量机的最终模型仅与支持向量有关，即通过采用 hinge 损失函数仍保持了稀疏性.

4.2 正则化

支持向量机学习模型的更一般形式

$$\min_f \Omega(f) + C\sum_{i=1}^{m}\ell(f(\boldsymbol{x}_i), y_i)$$

其中, $\Omega(f)$ 称为结构风险, 也称正则化项, 用于描述模型 f 的某些性质; $\ell(f(\boldsymbol{x}_i), y_i)$ 称为经验风险, 用于描述模型与训练数据的契合程度. C 用于对二者进行折中, 也称正则化常数.

正则化是对复杂性的惩罚.

5. 支持向量回归

允许模型输出和实际输出间存在 2ϵ 的偏差.

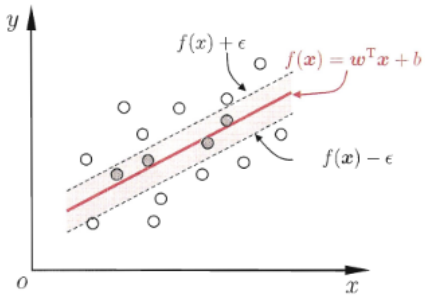

Figure VI.6: 支持向量回归示意图

落入中间 2ϵ 间隔带的样本不计算损失, 从而使得模型获得稀疏性.

SVR 问题可形式化, 引入松弛变量 $\zeta_i, \hat{\zeta}_i$

原始问题:

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}(\zeta_i - \hat{\zeta}_i)$$

$$\begin{aligned} \text{s.t. } & f(\boldsymbol{x}_i) - y_i \leq \epsilon + \zeta_i \\ & y_i - f(\boldsymbol{x}_i) \leq \epsilon + \zeta_i \\ & \zeta_i \geq 0, \hat{\zeta}_i \geq 0, i=1,2,\ldots,m \end{aligned}$$

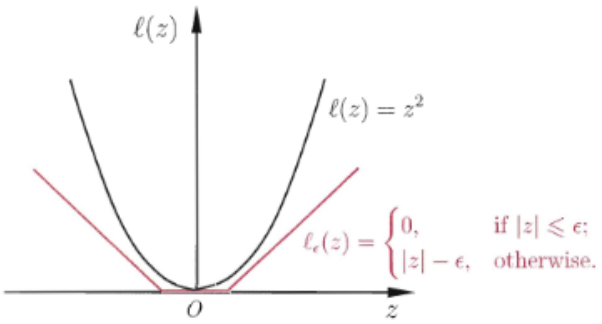

Figure VI.7: ϵ-不敏感损失函数

预测:

$$f(\boldsymbol{x})=\sum_{i=1}^{m}(\hat{\alpha}_i-\alpha_i)y_i\boldsymbol{x}_i^\top\boldsymbol{x}+b$$

6. 核方法

Theorem VI.2 (表示定理). 令 $\mathbb{H}$ 为核函数 κ 对应的再生核希尔伯特空间, $\|h\|_{\mathbb{H}}$ 表示 $\mathbb{H}$ 空间中国关于 h 的范数, 对于任意单调递增函数 $\Omega:[0,\infty]\mapsto\mathbb{R}$ 和任意非负损失函数 $\ell:\mathbb{R}^m\mapsto[0,\infty]$, 优化问题

$$\min_{h\in\mathbb{H}}F(h)=\Omega(\|h\|_{\mathbb{H}})+\ell(h(\boldsymbol{x}_1),h(\boldsymbol{x}_2),\ldots,h(\boldsymbol{x}_m))$$

的解总可以写为

$$h^*(\boldsymbol{x})=\sum_{i=1}^{m}\alpha_i\kappa(\boldsymbol{x},\boldsymbol{x}_i)$$

VII 贝叶斯分类器

1. 贝叶斯决策论 (Bayesian decision theory)

假设有 N 种可能的类别标记, i.e. $\mathcal{Y}=\{c_1,c_2,\ldots,c_N\}$, λ_{ij} 是将一个真实标记为 c_j 的样本误分类为 c_i 所产生的损失, 基于后验概率 $P(c_i|\boldsymbol{x})$ 可获得将样本 $\boldsymbol{x}$ 分类为 c_i 所产生的期望损失, 即样本 $\boldsymbol{x}$ 上的条件风险

$$R(c_i|\boldsymbol{x})=\sum_{j=1}^{N}\lambda_{ij}P(c_j|\boldsymbol{x})$$

任务是寻找一个判定准则 $h:\mathcal{X}\mapsto\mathcal{Y}$ 以最小化总体风险

$$R(h)=\mathbb{E}_{\boldsymbol{x}}\left[R(h(\boldsymbol{x})|\boldsymbol{x})\right]$$

根据贝叶斯判定准则 (Bayes decision rule): 为最小化总体风险，只需在每个样本上选择那个能使条件风险 $R(c|\boldsymbol{x})$ 最小的类别标记，即

$$h^*(\boldsymbol{x})=\operatorname*{argmin}_{c\in\mathcal{Y}}R(c|\boldsymbol{x})$$

此时 h^* 称为贝叶斯最优分类器 (Bayes optimal classifier), 与之对应的总体风险 $R(h^*)$ 称为贝叶斯风险 (Bayes risk). $1-R(h^*)$ 反映了分类器所能达到的最好性能，即通过机器学习所能产生的模型精度的理论上限.

1.1 判别式 vs. 生成式

欲使用贝叶斯判定准则来最小化决策风险，首先要获得后验概率 $P(c|\boldsymbol{x})$.

- 判别式: 直接对 $P(c|\boldsymbol{x})$ 建模
- 生成式: 先对 $P(\boldsymbol{x},c)$ 建模, 再获得 $P(c|\boldsymbol{x})=\dfrac{P(\boldsymbol{x},c)}{P(\boldsymbol{x})}$

1.2 贝叶斯定理

基于贝叶斯定理, $P(c|\boldsymbol{x})$ 可写为

$$P(c|\boldsymbol{x})=\frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})}$$

其中，$P(c)$ 是类“先验”(prior) 概率; $P(\boldsymbol{x}|c)$ 是样本 $\boldsymbol{x}$ 相对于类标记 c 的类条件概率 (class-conditional probability), 或称为“似然” (likelihood); $P(\boldsymbol{x})$ 是用于归一化的“证据” (evidence) 因子. 对给定样本 $\boldsymbol{x}$ 证据因子 $P(\boldsymbol{x})$ 与类标记无关，因此估计 $P(c|\boldsymbol{x})$ 的问题就转化为如何基于训练数据 D 来估计先验 $P(c)$ 和似然 $P(\boldsymbol{x}|c)$.

1.3 极大似然估计

先假设某种概率分布形式，再基于训练样例对参数进行估计. 估计结果的准确性严重依赖于所假设的概率分布形式是否符合潜在的真实分布

假定 $P(\boldsymbol{x}|c)$ 具有确定的形式并且被参数向量 $\boldsymbol{\theta}_c$ 唯一确定, 任务就是利用训练集 D 估计参数 $\boldsymbol{\theta}_c$, i.e. 估计 $P(\boldsymbol{x}|\boldsymbol{\theta}_c)$ 作为 $P(\boldsymbol{x}|c)$ 使用.

令 D_c 表示训练集 D 种第 c 类样本组成的集合, 假设样本独立同分布, 则参数 $\boldsymbol{\theta}_c$ 对训练集 D_c 的似然是

$$P(D_c|\boldsymbol{\theta}_c) = \prod_{\boldsymbol{x}\in D_c} P(\boldsymbol{x}|\boldsymbol{\theta}_c)$$

其对数似然

$$\begin{aligned} LL(\boldsymbol{\theta}_c) &= \log P(D_c|\boldsymbol{\theta}_c) \\ &= \sum_{\boldsymbol{x}\in D_c} \log P(\boldsymbol{x}|\boldsymbol{\theta}_c) \end{aligned}$$

$\boldsymbol{\theta}_c$ 的极大似然估计 $\hat{\boldsymbol{\theta}}_c$ 为

$$\hat{\boldsymbol{\theta}}_c = \underset{\boldsymbol{\theta}_c}{\operatorname{argmin}} LL(\boldsymbol{\theta}_c)$$

2. 朴素贝叶斯分类器

"属性条件独立性假设" (attribute conditional independence assumption): 对已知类别，假设所有属性相互独立. 换言之, 假设每个属性独立地对分类结果发生影响. 基于此, 有

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})} = \frac{P(c)}{P(\boldsymbol{x})}\prod_{i=1}^{d} P(x_i|c)$$

其中 d 为属性数目，x_i 为 $\boldsymbol{x}$ 在第 i 个属性上的取值.

由于对所有类别来说 $P(\boldsymbol{x})$ 相同，因此贝叶斯判定准则有

$$h_{nb}(\boldsymbol{x}) = \underset{c\in\mathcal{Y}}{\operatorname{argmin}} P(c)\prod_{i=1}^{d} P(x_i|c)$$

估计出类先验概率

$$P(c) = \frac{|D_c|}{|D|}$$

对离散属性而言, 令 D_{c,x_i} 表示 D_c 中在第 i 个属性上取值为 x_i 的样本组成的集合, 则条件概率 $P(x_i|c)$ 可估计为

$$P(x_i|c) = \frac{|D_{c,x_i}|}{|D_c|}$$

对连续属性可考虑概率密度函数, 假定

$$p(x_i|c) \sim \mathcal{N}(\mu_{c,i}, \sigma_{c,i}^2)$$

求出 $\mu_{c,i}$, $\sigma_{c,i}$ 即可知其分布.

2.1 拉普拉斯修正 (Laplacian correction)

若某个属性值在训练集中没有与某个类同时出现过，则直接计算会出现问题，因为概率连乘将"抹去"其他属性提供的信息.

令 N 表示训练集 D 中可能的类别数, N_i 表示第 i 个属性可能的取值数, 修正式为

$$\begin{aligned} \hat{P}(c) &= \frac{|D_c|+1}{|D|+N} \\ \hat{P}(x_i|c) &= \frac{|D_{c,x_i}|+1}{|D_c|+N_i} \end{aligned}$$

假设了属性值与类别的均匀分布，这是额外引入的 bias.

3. 半朴素贝叶斯分类器 (semi-naïve Bayes classifier)

基本思路：适当考虑一部分属性间的相互依赖信息.

最常用策略：独依赖估计 (One-Dependent Estimator, ODE), 假设每个属性在类别之外最多仅依赖一个其他属性

$$P(c|\boldsymbol{x}) \propto P(c)\prod_{i=1}^{d} P(x_i|c, pa_i)$$

其中 pa_i 为属性 x_i 所依赖的属性，称为 x_i 的父属性.

确定每个属性的父属性有几种常见方法:

- SPODE (Super-Parent ODE)

  假设所有属性都依赖于同一属性，称为 "超父" (Super-Parent)，然后通过交叉验证等模型选择方法来确定超父属性

- TAN (Tree Augmented naïve Bayes)

  以属性间的条件 "互信息"(mutual information) 为边的权重，构建完全图，再利用最大带权生成树算法，仅保留强相关属性间的依赖性

- AODE (Averaged One-Dependent Estimator)

  尝试将每个属性作为超父来构建 SPODE , 然后将那些具有足够训练数据支撑的 SPODE 集成起来作为最终结果，

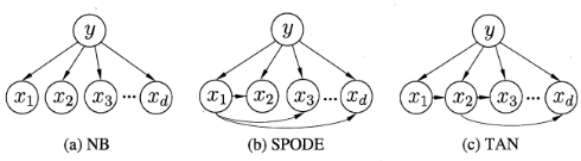

Figure VII.1: 朴素贝叶斯与两种半朴素贝叶斯分类器所考虑的属性依赖关系

4. 贝叶斯网 (Bayesian network)

贝叶斯网 (Bayesian network) 亦称“信念网”(belief network), 它借助有向无环图 (Directed Acyclic Graph , 简称 DAG) 来刻画属性之间的依赖关系，并使用条件概率表 (Conditional Probability Table, 简称 CPT) 来描述属性的联合概率分布.

一个贝叶斯网 B 由结构 G 和参数 Θ 两部分构成，即 $B = (G, \Theta)$. 参数 Θ 定量描述这种依赖关系, 假设属性 x_i 在 G 中的父结点集为 π_i, 则 Θ 包含了每个属性的条件概率表 $\theta_{x_i|\pi_i} = P_B(x_i|\pi_i)$

给定父结点集，贝叶斯网假设每个属性与其非后裔属性独立

4.1 结构

4.2 学习

4.3 推断

近似推断: 吉布斯采样

进行 T 次采样，每次采样中逐个考察每个非证据变量：假定所有其他属性取当前值，推断出采样概率，然后根据该概率采样

5. EM 算法

未观测变量的学名是“隐变量”(latent variable). EM (Expectation-Maximization) 算法是常用的估计参数隐变量的利器.

令 $\boldsymbol{X}$ 表示已观测变量集，$\boldsymbol{Z}$ 表示隐变量集，Θ 表示模型参数. 若欲对 Θ 做极大似然估计，则应最大化对数似然

$$LL(\Theta|\boldsymbol{X}, \boldsymbol{Z}) = \ln P(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$$

通过对 Z 计算期望，来最大化已观测数据的对数“边际似然”(marginal likelihood)

$$LL(\Theta|\boldsymbol{X}) = \ln P(\boldsymbol{X}|\Theta) = \ln \sum_{\boldsymbol{Z}} P(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$$

以初始值 Θ^0 为起点，对其可迭代执行以下步骤直至收敛

1) 基于 Θ^t 推断隐变量 $\boldsymbol{Z}$ 的期望，记为 $\boldsymbol{Z}^t$
2) 基于已观测变量 $\boldsymbol{X}$ 和 $\boldsymbol{Z}^t$ 对参数 Θ 做极大似然估计，记为 Θ^{t+1}

进一步, 不是取 $\boldsymbol{Z}$ 的期望, 而是基于 Θ^t 计算隐变量 $\boldsymbol{Z}$ 的概率分布 $P(\boldsymbol{Z}|\boldsymbol{X}, \Theta^t)$, 则 EM 的两步为:

1) E 步: 以当前参数 Θ^t 推断隐变量分布 $P(\boldsymbol{Z}|\boldsymbol{X}, \Theta^t)$, 并计算对数似然 $LL(\Theta|\boldsymbol{X}, \boldsymbol{Z})$ 关于 $\boldsymbol{Z}$ 的期望

$$Q(\Theta|\Theta^t) = \mathbb{E}_{\boldsymbol{Z}|\boldsymbol{X},\Theta^t} LL(\Theta|\boldsymbol{X}, \boldsymbol{Z})$$

2) M 步: : 寻找参数最大化期望似然，即

$$\Theta^{t+1} = \underset{\Theta}{\operatorname{argmax}}\, Q(\Theta|\Theta^t)$$

VIII 集成学习

1. 个体与集成

集成学习 (ensemble learning) 通过构建并结合多个学习器来提升性能

在一定条件下，随着集成分类器数目的增加，集成的错误率将指数级下降，最终趋向于 0

2. Boosting

先从初始训练集训练出一个基学习器，再根据基学习器的表现对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续受到更多关注, 然后基于调整后的样本分布来训练下一个基学习器；如此重复进行，直至基学习器数目达到事先指定的值 T , 最终将这 T 个基学习器进行加权结合.

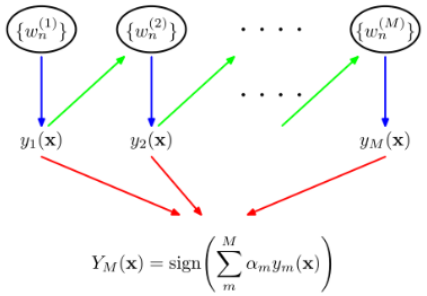

Figure VIII.1: Boosting

2.1 Adaboost

AdaBoost 算法有多种推导方式. 跳过推导了.

数据分布的学习: 重赋权法, 重采样法.

重启动，避免训练过程过早停止

3. Bagging 与随机森林

3.1 Bagging

Bagging 是并行式集成学习方法最著名的代表.

使用可重复采样采样出 T 个含 m 个训练样本的采样集，然后基于每个采样集训练出一个基学习器，再将这些基学习器进行结合.

结果通过投票法或比较置信度确认.

训练一个 bagging 集成与直接使用基学习器的复杂度同阶.

3.2 随机森林

随机森林 (Random Forest, 简称 RF) 是 Bagging 的一个扩展变体. RF 在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机属性选择

4. 结合策略

学习器结合可能会从三个方面带来好处:

1) 从统计的方面来看，由于学习任务的假设空间往往很大，可能有多个假设在训练集上达到同等性能，此时若使用单学习器可能因误选而导致泛化性能不佳，结合多个学习器则会减小这一风险

2) 从计算的方面来看，学习算法往往会陷入局部极小，有的局部极小点所对应的泛化性能可能很糟糕，而通过多次运行之后进行结合，可降低陷入糟糕局部极小点的风险

3) 从表示的方面来看，某些学习任务的真实假设可能不在当前学习算法所考虑的假设空间中，此时若使用单学习器则肯定无效，而通过结合多个学习器，由于相应的假设空间有所扩大, 有可能学得更好的近似

4.1 平均法

对数值型输出 $h_i(\boldsymbol{x}) \in \mathbb{R}$, 最常见的结合策略是使用平均法 (averaging).

- 简单平均法 (simple averaging)

$$H(\boldsymbol{x}) = \frac{1}{T}\sum_{i=1}^{T} h_i(\boldsymbol{x})$$

- 加权平均法 (weighted averaging)

$$H(\boldsymbol{x}) = \sum_{i=1}^{T} w_i h_i(\boldsymbol{x})$$

w_i 是个体学习器 h_i 的权重, 通常要求 $w_i \geq 0, \sum_{i=1}^{T} w_i = 1$

4.2 投票法

- 绝对多数投票法 (majority voting)
  若某标记得票过半数, 则预测为该标记；否则拒绝预测.
- 相对多数投票法 (plurality voting)
  预测为得票最多的标记, 若同时有多个标记获最高票，则从中随机选取一个.
- 加权投票法 (weighted voting)
  与加权平均法类似

4.3 学习法

通过另一个学习器来进行结合. Stacking 是学习法的典型代表, 多响应线性回归 (MLR) 作为次级学习器的学习算法, 效果较好.

贝叶斯模型平均 (Bayes Model Averaging, 简称 BMA) 基于后验概率来为不同模型赋予权重，可视为加权平均法的一种特殊实现.

5. 多样性

5.1 误差-分歧分解

定义个体学习器 h_i 与集成学习器 H 的分歧为

$$A(h_i|\boldsymbol{x}) = (h_i(\boldsymbol{x}) - H(\boldsymbol{x}))^2$$

经过一系列对回归问题的推导后,

$$E = \bar{E} - \bar{A}$$

$\bar{E}$ 表示个体学习器泛化误差的加权均值, $\bar{A}$ 表示个体学习器的加权分歧值

这个漂亮的式子显示: 个体学习器精确性越高、多样性越大，则集成效果越好。称为误差-分歧分解

5.2 多样性度量

多样性度量 (diversity measure) 是用于估算个体学习器的多样化程度.

常见的多样性度量:

- 不合度量 (Disagreement Measure)
- 相关系数 (Correlation Coefficient)
- Q-统计量 (Q-Statistic)
- κ-统计量 (Kappa-Statistic)

5.3 多样性增强

常见的增强个体学习器的多样性的方法:

- 数据样本扰动
- 输入属性扰动
- 输出表示扰动
- 算法参数扰动

IX 聚类

1. 聚类任务

目标：将数据样本划分为若干个通常不相交的“簇”(cluster)

2. 性能度量

聚类性能度量，亦称聚类“有效性指标”(validity index)

- 外部指标 (external index)
  将聚类结果与某个“参考模型”(reference model) 进行比较
- 内部指标 (internal index)
  直接考察聚类结果，无参考模型

3. 距离计算

聚类来自于分组，分组来自于合理度量，度量来自于距离, 因此距离对聚类很本质.

距离度量 (distance metric) 需满足的基本性质：

- 非负性: $dist(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$
- 同一性: $dist(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0 \iff \boldsymbol{x}_i = \boldsymbol{y}_i$
- 对称性: $dist(\boldsymbol{x}_i, \boldsymbol{x}_j) = dist(\boldsymbol{x}_j, \boldsymbol{x}_i)$
- 直递性: $dist(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq dist(\boldsymbol{x}_i, \boldsymbol{x}_k) + dist(\boldsymbol{x}_k, \boldsymbol{x}_j)$

常用距离形式:

- 闵可夫斯基距离 (Minkowski distance)

$$dist_{mk}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_{u=1}^{n} |x_{iu} - x_{ju}|^p \right)^{\frac{1}{p}}$$

  $p = 2$: 欧氏距离 (Euclidean distance)
  $p = 1$: 曼哈顿距离 (Manhattan distance)

- 对无序 (non-ordinal) 属性，可使用 VDM (Value Difference Metric)

  令 $m_{u,a}$ 表示属性 u 上取值为 a 的样本数，$m_{u,a,i}$ 表示在第 i 个样本簇中在属性 u 上取值为 a 的样本数，k 为样本簇数，则属性 u 上两个离散值 a 与 b 之间的 VDM 距离为

$$VMD_p(a, b) = \sum_{i=1}^{k} \left| \frac{m_{u,a,i}}{m_{u,a}} - \frac{m_{u,b,i}}{m_{u,b}} \right|^p$$

- 对混合属性，可使用 MinkovDM

$$MinkovDM_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_{u=1}^{n_c} |x_{iu} - x_{ju}|^p + \sum_{u=n_c+1}^{n} VDM_p(x_{iu}, x_{ju}) \right)^{\frac{1}{p}}$$

4. 原型聚类

原型 = 簇中心，有簇中心的聚类方法

4.1 k-均值聚类 (k-means)

每个簇中心以该簇中所有样本点的“均值”表示.
步骤:

1) 随机选取 k 个样本点作为簇中心
2) 将其他样本点根据其与簇中心的距离，划分给最近的簇
3) 更新各簇的均值向量，将其作为新的簇中心
4) 若所有簇中心未发生改变，则停止；否则执行 2)

4.2 学习向量量化 (LVQ)

Learning Vector Quantization. 也是试图找到一组原型向量来刻画聚类结构，但数据样本带有类别标记. 通过聚类来形成类别的“子类”结构；每个聚类对应于类别的一个子类.

4.3 高斯混合聚类 (GMM)

Gaussian Mixture Clustering. 采用高斯概率分布来表达聚类原型，簇中心 = 均值，簇半径 = 方差. 参数估计可采用极大似然法与 EM 算法.

5. 密度聚类

划成多个等价类，未必有簇中心

5.1 DBSCAN

通过邻域建立样本的可达路径，形成等价类 (连通分支).
DBSCAN 将“簇”定义为: 由密度可达关系导出的最大的密度相连样本集合

6. 层次聚类

聚类效果跟抽象的粒度有关，形成多层次聚类

6.1 AGNES (AGglomerative NESting)

从最细的粒度开始（一个样本一个簇），逐渐合并相似的簇，直到最粗的粒度（所有样本一个簇）
步骤:

1) 将每个样本点作为一个簇
2) 合并最近的两个簇
3) 若所有样本点都存在与一个簇中, 则停止; 否则 2)

X 降维与度量学习

1. k 近邻学习 (kNN)

k-Nearest Neighbor. 给定测试样本，基于某种距离度量找出训练集中与其最靠近的 k 个训练样本，然后基于这 k 个”邻居” 的信息来进行预测

没有显式的训练过程.

最近邻分类器虽简单，但它的泛化错误率不超过贝叶斯最优分类器的错误率的两倍.

2. 低维嵌入

维数灾难 (curse of dimensionality) ：在高维情形下出现的数据样本稀疏、距离计算困难等问题.

维数约简 (降维)：通过某种数学变换将原始高维属性空间转变为一个低维“子空间”(subspace) ，在这个子空间中样本密度大幅提高，距离计算也变得更为容易。

多维缩放：要求原始空间中样本之间的距离在低维空间中得以保持

对于降维后样本 $\boldsymbol{Z}$, 可通过降维前后保持不变的距离矩阵 $\boldsymbol{D}$ 求取内积矩阵 $\boldsymbol{B} = \boldsymbol{Z}^\top \boldsymbol{Z}$, 然后对 $\boldsymbol{B}$ 作特征值分解求 $\boldsymbol{Z}$.

一般来说，欲获得低维子空间，最简单的是对原始高维空间进行线性变换

$$\boldsymbol{Z} = \boldsymbol{W}^\top \boldsymbol{X}$$

基于线性变换来进行降维的方法称为线性降维方法.

3. 主成分分析 (PCA)

Principal Component Analysi. 对于正交属性空间中的样本点，用一个超平面 (直线的高维推广) 对所有样本进行恰当的表达.

- 最近重构性：样本点到这个超平面的距离都足够近
- 最大可分性：样本点在这个超平面上的投影能尽可能分开

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
低维空间维数 d'.
过程:
1: 对所有样本进行中心化: $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i$;
2: 计算样本的协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$;
3: 对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 做特征值分解;
4: 取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'}$.
输出: 投影矩阵 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$.

Figure X.1: PCA 算法

PCA 仅需保留 $\boldsymbol{W}$ 与样本的均值向量即可通过简单的向量减法和矩阵-向量乘法将新样本投影至低维空间中

4. 核化线性降维

非线性阵维的一种常用方法，是基于核技巧对线性降维方法进行 “核化”

$$\boldsymbol{z}_i = \phi(\boldsymbol{x}_i)$$

ϕ 不能显示表示, 于是引入核函数

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$$

反正最后计算开销较大

5. 流形学习

“流形”是在局部与欧氏空间同胚的空间，换言之，它在局部具有欧氏空间的性质，能用欧氏距离来进行距离计算。

若低维流形嵌入到高维空间中，则数据样本在高维空间的分布虽然看上去非常复杂，但在局部上仍具有欧氏空间的性质，因此，可以容易地在局部建立阵维映射关系，然后再设法将局部映射关系推广到全局。

两种著名的流形学习方法：

- 等度量映射 (Isometric Mapping，简称 Isomap)
- 局部线性嵌入 (Locally Linear Embedding，简称 LLE)

5.1 等度量映射 (Isomap)

计算测地线距离：可利用流形在局部上与欧氏空间同胚这个性质，对每个点基于欧氏距离找出其近邻点，然后就能建立一个近邻连接图，图中近邻点之间存在连接，而非近邻点之间不存在连接，于是，计算两点之间测地线距离的问题就转变为计算近邻连接图上两点之间的最短路径问题

5.2 局部线性嵌入 (LLE)

试图保持邻域内样本之间的线性关系

6. 度量学习

在机器学习中，对高维数据进行降维的主要目的是希望找一个合适的低维空间，在此空间中进行学习能比原始空间性能更好。事实上，每个空间对应了在样本属性上定义的一个距离度量，而寻找合适的壁间，实质上就是在寻找一个合适的距离度量。

改为” 学习” 出一个合适的距离度量

XI 特征选择与稀疏学习

特征: 描述物体的属性

特征的分类:

- 相关特征: 对当前学习任务有用的属性
- 无关特征: 与当前学习任务无关的属性
- 冗余特征 *: 其所包含信息能由其他特征推演出来

特征选择: 从给定的特征集合中选出任务相关特征子集, 必须确保不丢失重要特征.

1. 子集搜索与评价

用贪心策略选择包含重要信息的特征子集

- 前向搜索：逐渐增加相关特征
- 后向搜索：从完整的特征集合开始，逐渐减少特征
- 双向搜索：每一轮逐渐增加相关特征，同时减少无关特征

特征子集确定了对数据集的一个划分, 每个划分区域对应着特征子集的某种取值. 样本标记对应着对数据集的真实划分. 通过估算这两个划分的差异，就能对特征子集进行评价；与样本标记对应的划分的差异越小，则说明当前特征子集越好.

比如用信息熵进行子集评价.

2. 过滤式选择

先用特征选择过程过滤原始数据，再用过滤后的特征来训练模型；特征选择过程与后续学习器无关.

Relief (Relevant Features) 方法

3. 包裹式选择

包裹式选择直接把最终将要使用的学习器的性能作为特征子集的评价准则.

LVW(Las Vegas Wrapper) 在拉斯维加斯方法框架下使用随机策略来进行子集搜索，并以最终分类器的误差作为特征子集评价准则

4. 嵌入式选择与 L_1 正则化

嵌入式特征选择是将特征选择过程与学习器训练过程融为一体，两者在同一个优化过程中完成，在学习器训练过程中自动地进行特征选择

L_1 正则化: 采用 L_1 范数时交点常出现在坐标轴上，即容易产生值为 0 的稀疏解.

5. 稀疏表示与字典学习

5.1 稀疏表示

将数据集考虑成一个矩阵，每行对应一个样本，每列对应一个特征. 矩阵中有很多零元素，且非整行整列出现.

稀疏表达的优势:

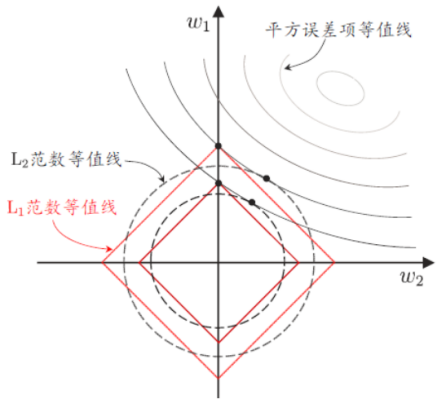

Figure XI.1: L_1 正则化

- 文本数据线性可分
- 存储高效

5.2 字典学习

为普通稠密表达的样本找到合适的字典，将样本转化为稀疏表示，这一过程称为字典学习

6. 压缩感知

利用部分数据恢复全部数据

XII 半监督学习

1. 未标记样本

半监督学习：让学习器不依赖外界交互、自动地利用未标记样本来提升学习性能。

- 纯 (pure) 半监督学习：假定训练数据中的未标记样本并非待预测的数据
- 直推学习：假定学习过程中所考虑的未标记样本恰是待预测数据，学习的目的就是在这些未标记样本上获得最优泛化性能

2. 生成式方法

生成式方法 (generative methods) 是直接基于生成式模型的方法. 此类方法假设所有数据 (无论是否有标记) 都是由同一个潜在的模型“生成”的. 然后使用诸如极大似然或 EM 的方法估计参数.

模型假设必须准确，即假设的生成式模型必须与真实数据分布吻合；否则利用未标记数据反倒会降低泛化性能。

3. 半监督 SVM

TSVM：尝试将每个未标记样本分别作为正例或反例，然后在所有这些结果中，寻求一个在所有样本 (包括有标记样本和进行了标记指派的未标记样本) 上间隔最大化的划分超平面.

在对未标记样本进行标记指派及调整的过程中，有可能出现类别不平衡问题，即某类的样本远多于另一类，这将对 SVM 的训练造成困扰。

4. 图半监督学习

给定一个数据集，可将其映射为一个图，数据集中每个样本对应于图中一个结点，若两个样本之间的相似度很高 (或相关性很强)，则对应的结点之间存在一条边，边的“强度”正比于样本之间的相似度 (或相关性)。

5. 基于分歧的方法

多视图数据：一个数据对象往往同时拥有多个” 属性集” (attribute set) ，每个属性集就构成了一个“视图”(view)。

- 相容性：不同视图所包含的关于输出空间的信息是一致的。
- 相容互补性：假设数据拥有两个充分且条件独立视图，“充分”是指每个视图都包含足以产生最优学习器的信息；“条件独立”则是指在给定类别标记条件下两个视图独立。

在此情形下，可用一个简单的办法来利用未标记数据：

1) 在每个视图上基于有标记样本分别训练出一个分类器

2) 让每个分类器分别去挑选自己” 最有把握的” 未标记样本赋予伪标记，并将伪标记样本提供给另一个分类器作为新增的有标记样本用于训练更新...

3) 这个” 互相学习、共同进步” 的过程不断迭代进行，直到两个分类器都不再发生变化，或达到预先设定的迭代轮数为止.

6. 半监督聚类

聚类任务中获得的监督信息：” 必连”(must-link) 与” 勿连” (cannot-link) 约束

约束 k 均值 (Constrained k - means) 算法是利用第一类监督信息的代表.

XIII 概率图模型

概率模型：提供了一种描述框架，将学习任务归结于计算变量的概率分布

推断：在概率模型中，利用已知变量推测未知变量的分布

1. 隐马尔可夫模型

概率图模型：一类用图来表达变量相关关系的概率模型。它以图为表示工具，最常见的是用一个结点表示一个或一组随机变量，结点之间的边表示变量间的概率相关关系，即“变量关系图”。

- 使用有向无环图表示变量间的依赖关系，称为有向圈模型或贝叶斯网
- 使用无向国表示变量间的相关关系，称为无向图模型或马尔可夫网

对于隐马尔可夫模型

- 状态变量: 假定状态变量是隐藏的、不可被观测的，因此状态变量亦称隐变量。
- 观测变量: 可以是离散型也可以是连续型。

马尔可夫链：(现在决定未来) 系统下一时刻的状态仅由当前状态决定，不依赖于以往的任何状态. 基于这种依赖关系，所有变量的联合概率分布为

$$P(x_1, y_1, \ldots, x_n, y_n) = P(y_1)P(x_1|y_1)\prod_{i=2}^{n} P(y_i|y_{i-1}P(x_i|y_i))$$

欲确定一个隐马尔可夫模型还需以下三组参数：

- 状态转移概率: 在各个状态间转换的概率
- 输出现测概率: 根据当前状态获得各个观测值的概率
- 初始状态概率

观测序列产生:

1) 设置 $t = 1$, 并根据初始状态概率 $\boldsymbol{\pi}$ 选择初始状态 y_1；
2) 根据状态 y_t 和输出观测概率 $\boldsymbol{B}$ 选择观测变量取值 x_t;
3) 根据状态 y_t 和状态转移矩阵 $\boldsymbol{A}$ 转移模型状态，即确定 y_{t+1}；
4) 若 $t < n$ 设置 $t = t + 1$, 并转到第 2) 步，否则停止.

2. 马尔可夫随机场

马尔可夫随机场 (Markov Random Field，简称 MRF)：典型的马尔可夫网，这是一种著名的无向图模型。图中每个结点表示一个或一组变量，结点之间的边表示两个变量之间的依赖关系。马尔可夫随机场有二组势函数，亦称“因子”，这是定义在变量子集上的非负实函数，主要用于定义概率分布函数。

可得到两个推论：

- 局部马尔可夫性：给定某变量的邻接变量，则该变量条件独立于其他变量
- 成对马尔可夫性：给定所有其他变量，两个非邻接变量条件独立

3. 条件随机场

条件随机场 (Conditional Random Field , 简称 CRF) 是一种判别式无向图模型, 试图对多个变量在给定观测值后的条件概率进行建模

4. 学习与推断

边际化：给定参数 Θ 求解某个变量 $\boldsymbol{x}$ 的分布，就变成对联合分布中其他无关变量进行积分的过程. 推断问题的目标就是计算边际概率或条件概率

两种代表性的精确推断方法:

- 变量消去
- 信念传播

5. 近似推断

- 采样 (sampling) ，通过使用随机化方法完成近似
- 使用确定性近似完成近似推断，典型代表为变分推断 (variational inference)

5.1 MCMC 采样

概率图模型中最常用的采样技术是马尔可夫链蒙特卡罗 (Markov Chain Monte Carlo , 简称 MCMC) 方法. MCMC 方法的关键就在于通过构造“平稳分布为 p 的马尔可夫链”来产生样本.

Metropolis-Hastings (简称 MH) 算法是 MCMC 的重要代表. . 它基于“拒绝采样”(reject sampling) 来逼近平稳分布 p. 吉布斯采样 (Gibbs sampling) 有时被视为 M H 算法的特例.

5.2 变分推断

变分推断通过使用已知简单分布来逼近需推断的复杂分布，并通过限制近似分布的类型，从而得到一种局部最优、但具有确定解的近似后验分布

6. 话题模型

话题模型 (topic model) 是一族生成式有向图模型，主要用于处理离散型的数据 (如文本集合)，在信息检索、自然语言处理等领域有广泛应用. 隐狄利克雷分配模型 (Latent Dirichlet Allocation, 简称 LDA) 是话题模型的典型代表.

XIV 强化学习

1. 任务与奖赏

强化学习常用马尔可夫决策过程 (Markov Decision Process, MDP) 描述.

强化学习对应了四元组

$$E = \langle X, A, P, R \rangle$$

其中 P 指定了状态转移概率, R 指定了奖赏, X 为状态空间, A 为动作空间. E 为机器所处环境.

强化学习的目标: 机器通过在环境中不断尝试从而学到一个策略 π，使得长期执行该策略后得到的累积奖赏最大

强化学习在某种意义上可以认为是具有“延迟标记信息”的监督学习.

2. K-摇臂赌博机

- 只有一个状态，K 个动作
- 每个摇臂的奖赏服从某个期望未知的分布
- 执行有限次动作
- 最大化累积奖赏

强化学习面临的主要困难：探索-利用窘境 (Exploration-Exploitation dilemma)

- 探索：估计不同摇臂的优劣 (奖赏期望的大小)
- 利用：选择当前最优的摇臂

在探索与利用之间进行折中:

- ϵ-贪心
- Softmax

2.1 ϵ 贪心

- 以 ϵ 的概率探索：均匀随机选择一个摇臂
- 以 $1-\epsilon$ 的概率利用：选择当前平均奖赏最高的摇臂

2.2 Softmax

基于当前已知的摇臂平均奖赏来对探索与利用折中, 若某个摇臂当前的平均奖赏越大，则它被选择的概率越高.

2.3 离散空间强化学习

离散空间状态空间、离散动作空间上的多步强化学习任务

方法：

- 每个状态上动作的选择看作一个 K-摇臂赌博机问题
- K-摇臂赌博机算法奖赏函数：强化学习任务的累积奖赏

局限：未考虑马尔科夫决策过程

3. 有模型学习

有模型学习 (model-based learning)

$$E = \langle X, A, P, R \rangle$$

X, A, P, R 均已知.

强化学习的目标：找到使累积奖赏最大的策略 π

策略评估：使用某策略所带来的累积奖赏

- 状态值函数：从状态 x 出发，使用策略 π 所带来的累积奖赏
- 状态-动作值函数：从状态 x 出发，执行动作 a 后再使用策略 π 所带来的累积奖赏

策略改进：将非最优策略改进为最优策略

策略迭代 (policy iteration): 求解最优策略的方法

1) 随机策略作为初始策略
2) 策略评估 + 策略改进 + 策略评估 + 策略改进 +...
3) 直到策略收敛

4. 免模型学习

免模型学习 (model-free learning)：

- 转移概率，奖赏函数未知
- 甚至环境中的状态数目也未知
- 假定状态空间有限

免模型学习所面临的困难: 策略无法评估, 无法通过值函数计算状态-动作值函数, 机器只能从一个起始状态开始探索环境.

解决困难的办法: 多次采样, 直接估计每一对状态-动作的值函数, 在探索过程中逐渐发现各个状态

两种著名的免模型学习方法:

- 蒙特卡罗强化学习
- 时序差分学习

4.1 蒙特卡罗强化学习

采样轨迹，用样本均值近似期望

策略评估：蒙特卡罗法

1) 从某状态出发，执行某策略
2) 对轨迹中出现的每对状态-动作，记录其后的奖赏之和
3) 采样多条轨迹，每个状态-动作对的累积奖赏取平均

策略改进：换入当前最优动作

蒙特卡罗强化学习可能遇到的问题：轨迹的单一性. 解决问题的办法: ϵ-贪心

- 同策略 (on-policy)：被评估与被改进的是同一个策略
- 异策略 (off-policy)：学习过程是基于行为策略产生的数据，但是学习的目标是优化目标策略 (用重要性采样技术)

4.2 时序差分学习

增量式地进行状态-动作值函数更新, 也是 ϵ-贪心

- 同策略：Sarsa 算法
- 异策略：Q-学习 (Q-learning)

Sarsa 算法 需要前一步的状态 (state)、前一步的动 (action)、奖赏值 (reward)、当前状态 (state)、将要执行的动作 (action)

Q-学习算法 该算法评估的是 ϵ-贪心策略, 而执行的是原始策略

5. 值函数近似

若状态空间是连续 (无限) 的, 需要使用值函数近似. 但为简便假定:

- 状态空间 $X = \mathbb{R}^n$
- 考虑线性近似
- 行为空间有限

值函数近似: 将值函数表达为状态的线性函数

$$V_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{x}$$

$\boldsymbol{\theta}$ 为参数向量, $\boldsymbol{x}$ 为状态向量.

用最小二乘误差来度量学到的值函数与真实的值函数 V^π 之间的近似程度

$$E_{\boldsymbol{\theta}} = \mathbb{E}_{\boldsymbol{x}\sim\pi}\left[\left(V^\pi(\boldsymbol{x}) - V_{\boldsymbol{\theta}}(\boldsymbol{x})\right)^2\right]$$

用梯度下降法更新参数向量，求解优化问题.

单个样本更新策略

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha(V^\pi(\boldsymbol{x}) - V_{\boldsymbol{\theta}}(\boldsymbol{x}))\boldsymbol{x}$$

借助时序差分学习，使用估计的值函数 $V^\pi(\boldsymbol{x}) = r + \gamma V^\pi(\boldsymbol{x}')$ 代替真实值函数

$$\begin{aligned}\boldsymbol{\theta} &= \boldsymbol{\theta} + \alpha(r + \gamma V^\pi(\boldsymbol{x}) - V_{\boldsymbol{\theta}}(\boldsymbol{x}))\boldsymbol{x} \\ &= \boldsymbol{\theta} + \alpha(r + \gamma\boldsymbol{\theta}^\top\boldsymbol{x}' - \boldsymbol{\theta}^\top\boldsymbol{x})\boldsymbol{x}\end{aligned}$$

线性值函数近似 Sarsa 算法. 可以通过引入核方法实现非线性值函数近似。

6. 模仿学习

强化学习任务中多步决策的搜索空间巨大，基于累积奖赏来学习很多步之前的合适决策非常困难. 直接模仿人类专家的状态-动作对来学习策略.

1) 利用专家的决策轨迹，构造数据集 D：状态作为特征，动作作为标记
2) 利用数据集 D，使用分类/回归算法即可学得策略
3) 将学得的策略作为初始策略
4) 策略改进，从而获得更好的策略

强化学习任务中，设计合理的符合应用场景的奖赏函数往往相当困难. 从人类专家提供的范例数据中反推出奖赏函数. i.e. 逆强化学习 (inverse reinforcement learning)