Artificial Intelligence Knowledge Base

一个人工智能博士的知识积累

陈若愚，chenruoyu@iie.ac.cn

2022 年 4 月 23 日——至今

目录

1 高等数学 **1**

2 矩阵论 & 线性代数 **2**

2.1 向量和向量空间 . . . 2

2.1.1 Euclid 空间 . . . 8

2.1.2 无限维内积空间（赋范空间） . . . 10

2.2 内积与范数 . . . 11

2.2.1 矩阵的内积与范数 . . . 11

2.2.2 矩阵范数与向量范数的相容性 . . . 11

2.2.3 从属范数 . . . 11

2.2.4 长方阵的范数 . . . 11

2.3 矩阵的重要性指标 . . . 11

2.4 逆矩阵与伪逆矩阵 . . . 11

2.5 Moore-Penrose 逆矩阵的计算方法 . . . 13

2.5.1 利用满秩分解求广义逆矩阵 . . . 13

2.5.2 计算 $\mathbf{A}^+$ 的 Zlobec 公式 . . . 13

2.5.3 Greville 方法 . . . 13

2.5.4 一些特殊分块矩阵的广义逆矩阵 . . . 13

2.6 . . . 13

3 Optimization **14**

3.1 优化基础 . . . 14

3.2 无约束优化算法 . . . 17

3.2.1 最速下降法 . . . 17

3.2.2 牛顿法 . . . 17

3.2.3 共轭梯度法 . . . 18

3.2.4 拟牛顿算法 . . . 18

3.2.5 SAG . . . 21

3.2.6 SAGA . . . 22
3.2.7 SVRG . . . 22
3.2.8 SARAH . . . 23
3.3 约束优化算法 . . . 24
3.3.1 对偶算法 . . . 24
3.3.2 对偶及鞍点问题 . . . 24
3.4 经验风险及其一般优化模型 . . . 24
3.5 结构风险及其一般优化模型 . . . 25
3.5.1 常见的正则化 . . . 25

Bibliography **25**

4 模式识别与机器学习 **26**
4.1 贝叶斯判别 . . . 26
4.1.1 正态分布模式的朴素贝叶斯分类器 . . . 26
4.1.2 朴素贝叶斯过程 . . . 27
4.1.3 Laplace 平滑 . . . 27
4.2 Fisher 线性判别 . . . 27
4.2.1 基本参数 . . . 28
4.2.2 Fisher 准则函数 . . . 28
4.2.3 计算最佳变换向量 w^* . . . 29
4.3 感知器算法 . . . 29
4.3.1 训练算法 . . . 29
4.3.2 收敛性证明 . . . 29
4.3.3 多类训练算法 . . . 30
4.4 K-L 变换 . . . 30
4.5 逻辑回归 LR . . . 33
4.5.1 判别式模型，二分类逻辑回归 . . . 33
4.5.2 多分类逻辑回归 . . . 34
4.5.3 LR（逻辑回归）与 NB（朴素贝叶斯） . . . 34
4.6 MLE 与 MAP . . . 34
4.6.1 MLP . . . 34
4.6.2 MAP . . . 35
4.7 高斯判别分析 . . . 35
4.8 SVM . . . 36
4.9 K-Means 聚类 . . . 38
4.9.1 算法流程 . . . 38
4.9.2 K 的选择问题 . . . 38
4.9.3 其他聚类方法（GMM，层次聚类，DBSCAN） . . . 38
4.10 降维 . . . 39
4.10.1 PCA . . . 39
4.10.2 流形学习：结构保持/距离保持 . . . 40

4.11 集成学习 . . . 42
4.11.1 Bagging 原理及降低 Variance 推导 . . . 42
4.11.2 Boosting 推导 . . . 43
4.12 半监督学习 . . . 44

5 图像处理与计算机视觉 45
5.1 目标跟踪 . . . 45
5.1.1 运动目标的表示方法 . . . 45
5.1.2 传统目标跟踪方法 . . . 45

6 Reinforcement Learning 46
6.1 马尔可夫决策过程 . . . 46
6.2 无模型控制学习 . . . 47
6.2.1 GLIE 蒙特卡洛控制学习 . . . 48
6.2.2 Sarsa . . . 48

Bibliography 50

7 Graph Neural Network 51
7.1 . . . 51

8 Attention Module 52
8.1 . . . 52

9 Contrastive Learning 53
9.1 . . . 53

10 Imbalanced & Long-Tailed Problem 54
10.1 Re-balancing . . . 54
10.1.1 Balanced MSE . . . 54

Bibliography 56

11 Face 57
11.1 Introduction . . . 57
11.2 Face Recognition . . . 57
11.2.1 ArcFace . . . 57
11.2.2 AdaFace . . . 58
11.3 Face Image Quality Assessment . . . 58
11.3.1 SER-FIQ . . . 59
11.3.2 MagFace . . . 59
11.4 Explainable Face Recognition . . . 59
11.4.1 PLQ . . . 60
11.5 Face Image Editing . . . 61

Bibliography **61**

12 Explainable AI **63**

12.1 Interpretability 63

12.2 可视化 63

12.3 特征重要性 63

12.4 反事实可解释性 63

Bibliography **63**

13 Few-shot Learning **65**

13.1 65

14 Federal Learning **66**

14.1 FedAvg 66

14.2 MOON 67

Bibliography **68**

1 高等数学

2 矩阵论 & 线性代数

2.1 向量和向量空间

定义 2.1: 数域

设 $\mathbb{K}$ 是一些数的集合，**定义了加法和乘法**两个运算，具有下述性质：

1) $\mathbb{K}$ 中任何两个元素运算的**结果还在 $\mathbb{K}$ 中**，即运算满足**封闭性**；

2) 加法和乘法运算都满足结合律、交换律；

3) 乘法对加法满足分配律；

4) 数 0 和 1 都在 $\mathbb{K}$ 中；

5) 如果 a 在 $\mathbb{K}$ 中，则在 $\mathbb{K}$ 中一定有另一个元素，记作 $-a$, 使得 $a+(-a)=0$；

6) 如果 $b\neq 0$ 在 $\mathbb{K}$ 中，则在 $\mathbb{K}$ 中一定有另一个元素，记作 b^{-1}, 使得 $b\cdot b^{-1}=1$；

则称 $\mathbb{K}$ 是一个数域。

定义 2.2: 向量

给定数域 $\mathbb{K}$，由数域 $\mathbb{K}$ 中的 n 个数组成的有序数组 $(a_1,a_2,\cdots,a_n)$ 称为数域 $\mathbb{K}$ 上的 n 维向量。记作：

$$\boldsymbol{\alpha}=\begin{pmatrix}a_1\\a_2\\\vdots\\a_n\end{pmatrix} \tag{2.1}$$

定义 2.3: 向量空间

数域 $\mathbb{K}$ 上所有 n 维向量的集合记为 V，与向量的数乘运算和加法运算一起构成了数域 $\mathbb{K}$ 上的 n 维向量空间。

注：在 V 中任何向量的线性组合仍然属于 V。

定义 2.4: 向量的线性无关

一个向量集合 $\{\boldsymbol{v_1},\boldsymbol{v_2},\cdots,\boldsymbol{v_n}\}$ 的向量线性无关，充要条件为如果线性组合 $a_1\boldsymbol{v_1}+a_2\boldsymbol{v_2}+\cdots+a_n\boldsymbol{v_n}=0$，则 $a_1=a_2=\cdots=a_k=0$。

定义 2.5: 线性空间

设 V 是一个非空集合，其元素用 $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ 等表示；$\mathbb{K}$ 是一个数域，其元素用 k, l, m 等表示。如果 V 满足：

(1) 在 V 中定义一个 **“加法”** 运算，即当 $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in V$ 时，有唯一的“和” $\boldsymbol{x}+\boldsymbol{y} \in V$（封闭性），且加法运算满足下列性质

- 结合律：$\boldsymbol{x}+(\boldsymbol{y}+\boldsymbol{z})=(\boldsymbol{x}+\boldsymbol{y})+\boldsymbol{z}$；
- 交换律：$\boldsymbol{x}+\boldsymbol{y}=\boldsymbol{y}+\boldsymbol{x}$；
- 零元律：存在零元素 0，使 $\boldsymbol{x}+0=\boldsymbol{x}$；
- 负元律：对于任一元素 $\boldsymbol{x} \in V$，存在一元素 $\boldsymbol{y} \in V$，使 $\boldsymbol{x}+\boldsymbol{y}=0$，且称 $\boldsymbol{y}$ 为 $\boldsymbol{x}$ 的负元素，记为 $(-\boldsymbol{x})$。即，$\boldsymbol{x}+(-\boldsymbol{x})=0$。

(2) 在 V 中定义一个**“数乘”**运算，即当 $x \in V$ 时，有唯一的“积” $k\boldsymbol{x} \in V$（封闭性），且数乘运算满足下列性质

- 数因子分配律：$l(\boldsymbol{x}+\boldsymbol{y})=l\boldsymbol{x}+l\boldsymbol{y}$；
- 分配律：$(k+l)\boldsymbol{x}=k\boldsymbol{x}+l\boldsymbol{x}$；
- 结合律：$(kl)\boldsymbol{x}=k(l\boldsymbol{x})$；
- 恒等律：$1\boldsymbol{x}=\boldsymbol{x}$；

则称 V 为数域 K 上的线性空间，V 中元素称为向量，K 中元素称为纯量（也可以叫数）。
通常我们把 V 中满足八条规律且为封闭的加法及数乘两种运算，统称线性运算。

定义 2.6: 线性空间的维数

向量空间中最大的线性无关向量组（极大线性无关组）包含向量的个数为向量空间的维数，记作 $dim(\cdot)$，$\mathbb{R}^n \to \mathbb{Z}^+$。

定义 2.7: 线性空间的维数

向量空间中最大的线性无关向量组（极大线性无关组）包含向量的个数为向量空间的维数，记作 $dim(\cdot)$，$\mathbb{R}^n \to \mathbb{Z}^+$。

定义 2.8: 线性空间的基

数域 K 上线性空间 V 中的一组向量 $\alpha_1, \alpha_2, \cdots, \alpha_n, \cdots$ 称为 V 的基，如果它具有下述两条性质：
(1) 向量组 $\alpha_1, \alpha_2, \cdots, \alpha_n, \cdots$ 线性无关；
(2) V 中任意向量 β 都由 $\alpha_1, \alpha_2, \cdots, \alpha_n, \cdots$ 线性表出，即 β 可由其中有限个向量线性表出。

例 1: 在线性空间 $\mathbb{R}[x]$ 中，对任意实数 c，多项式组 $1, (x-c), \cdots, (x-c)^n, \cdots$ 都是基，该空间是无限维的。
特殊向量：零向量 $\mathbf{0}$，单位向量 e，例如 i^{th} 单位向量记作 $e_i=(0, ..., 1, 0, ..., 0)$

示性向量"sign(·)", 定义如下:

$$\text{sign}(x)_i = \begin{cases} 1 & x_i > 0 \\ 0 & x_i = 0 \\ -1 & x_i < 0 \end{cases}$$

同维度向量之间的比较: $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 是两个包含相同元素个数的向量, 如果对应的 $x_i \geq y_i$, 则 $\boldsymbol{x} \geq \boldsymbol{y}$。

定义 2.9: 坐标

设 V 是域 K 上的 n 维线性空间, 向量组 $\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n$ 是 V 的基, 则任意向量 $\alpha \in V$ 都可由基唯一线性表出:

$$\alpha = a_1\varepsilon_1 + a_2\varepsilon_2 + \cdots + a_n\varepsilon_n \tag{2.2}$$

称表出系数为 α 在基 $\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n$ 下的坐标, 记作 $(a_1, a_2, \cdots, a_n)$, 也可形式化为:

$$\boldsymbol{\alpha} = (\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n) \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} \tag{2.3}$$

子空间: 设 V 是数域 K 上的 n 维线性空间, $\alpha_1, \alpha_2, \cdots, \alpha_s$ 是 V 中的一组向量, 考虑它们在域 K 上的所有线性组合 $W = \{c_1\alpha_1 + c_2\alpha_2 + \cdots + c_s\alpha_s \mid c_i \in \mathbb{K}, i = 1, 2, \cdots, s\}$, 则 W 是 V 的子空间, 称为向量组 $\alpha_1, \alpha_2, \cdots, \alpha_s$ 生成的子空间, 记作 $Span(\alpha_1, \alpha_2, \cdots, \alpha_s)$。

$W \subseteq V$ 表示 W 是 V 的子空间。

定义 2.10: 子空间的和

V_1, V_2 分别是两个向量空间, 向量空间 $V = \{v \mid v_1 + v_2, v_1 \in V_1, v_2 \in V_2\}$ 是空间 V_1 与 V_2 的和, 记作 $V = V_1 + V_2$。

定理 2.1: 子空间维度计算

如果 V_1, V_2 是 n 维线性空间 V 的两个子空间, 那么

$$dim(V_1) + dim(V_2) = dim(V_1 + V_2) + dim(V_1 \cap V_2) \tag{2.4}$$

定义 2.11: 子空间的直和

设 V_1, V_2 是线性空间 V 的两个子空间, 如果 $V_1 + V_2$ 中每个向量 的分解式

$$\alpha = \alpha_1 + \alpha_2, \alpha_1 \in V_1, \alpha_2 \in V_2 \tag{2.5}$$

都是唯一的, 这个和就称为直和, 记作 $V_1 \oplus V_2$。

From《矩阵分析与应用》: 若 A 和 B 是向量空间 V 的两个子空间, 并满足 $V = A + B$, 且 $A \cap B = \{0\}$, 则称 V 是子空间 A 和 B 的直接求和, 简称直和, 记作 $V = A \oplus B$。

定理 2.2: 子空间结论等价

设 V_1, V_2 是线性空间 V 的两个子空间，则下述结论等价：

- $V_1 + V_2$ 是直和。
- 零向量的分解式唯一。
- $V_1 \cap V_2 = \{0\}$。
- $dim(V_1 + V_2) = dim(V_1) + dim(V_2)$

定义 2.12: 矩阵

数域 $\mathbb{K}$ 上 $s \times n$ 型矩阵

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{s1} & a_{s2} & \cdots & a_{sn} \end{pmatrix} \tag{2.6}$$

矩阵包含向量，向量可以写成矩阵。

定义 2.13: 线性映射

设 V_1, V_2 分别是域 K 上的 n 维和 m 维线性空间，V_1 到 V_2 的映射 $\boldsymbol{A}$ 称为线性映射，如果它能保持线性空间的两种线性运算，即：

$$\begin{aligned} \forall \alpha, \beta \in V_1, \boldsymbol{A}(\alpha + \beta) &= \boldsymbol{A}(\alpha) + \boldsymbol{A}(\beta) \\ \forall c \in \mathbb{K}, \boldsymbol{A}(c\alpha) &= c\boldsymbol{A}(\alpha) \end{aligned} \tag{2.7}$$

其中 $\boldsymbol{A}(\alpha)$ 或 $\boldsymbol{A}\alpha$ 表示元素 在映射 $\boldsymbol{A}$ 下的像。

定义 2.14: 线性变换

线性变换是线性空间 V 到自身的映射通常称为 V 上的一个变换。

定理 2.3

数域 $\mathbb{K}$ 上的任意两个 n 维线性空间 V_1 和 V_2 总是同构的。

只要一个线性变换 $\mathrm{T} : V \to W$ 是可逆的，那么这个线性变换就称之为同构（英语：Isomorphism)。指的是一个保持结构的双射 (bijection)。双射既是单射也是满射，见图2.1。

设 $\boldsymbol{A}$ 是线性空间 V_1 到 V_2 的线性映射，$\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n$ 是 V_1 的基，则

- $\boldsymbol{A}$ 是单射当且仅当 $\forall \alpha \in V_1, \boldsymbol{A}\alpha = 0 \Rightarrow \alpha = 0$;
- $\boldsymbol{A}$ 是满射当且仅当 $V_2 = Span(\boldsymbol{A}\varepsilon_1, \boldsymbol{A}\varepsilon_2, \cdots, \boldsymbol{A}\varepsilon_n)$

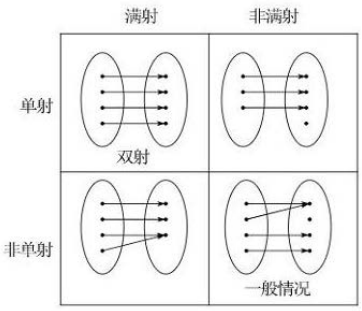


图 2.1: 单射，非单射，满射与非满射的关系。

定理 2.4

关于线性映射和它的矩阵有下述结论成立:

- 线性映射 $\boldsymbol{A}$ 是满射当且仅当它的矩阵 $\boldsymbol{A}_{m\times n}$ 是行满秩的。
- 线性映射 $\boldsymbol{A}$ 是单射当且仅当它的矩阵 $\boldsymbol{A}_{m\times n}$ 是列满秩的。
- 线性映射 $\boldsymbol{A}$ 是同构映射当且仅当它的矩阵 $\boldsymbol{A}_{m\times n}$ 是可逆的方阵。

定义 2.15: 值域与核

设 $\boldsymbol{A}$ 是域 K 上的两个向量空间 V_1 和 V_2 之间的线性映射，令 $\boldsymbol{A}^{-1}(0)=\{\alpha\in V_1\mid \boldsymbol{A}\alpha=0\}$，$\boldsymbol{A}V_1=\{\boldsymbol{A}\alpha\mid\alpha\in V_1\}$。则 $\boldsymbol{A}V_1$ 是 V_2 的子空间，成为线性映射 $\boldsymbol{A}$ 的值域，记作 $Im\boldsymbol{A}$，其维数等于矩阵 $\boldsymbol{A}$ 的秩 $rank(A)$。$\boldsymbol{A}^{-1}(0)$ 是 V_1 的子空间，称为线性映射 $\boldsymbol{A}$ 的核，记作 $Ker(\boldsymbol{A})$，其维数等于 $n-rank(\boldsymbol{A})$。

理解:

(1) $\boldsymbol{A}\alpha=0$ 的基础解系构成的线性空间，就是这个线性变换的核。值域是 $\boldsymbol{A}\alpha$ 取遍所有向量 后的解集。

(2) 核可以用来快速判断一个向量是否是线性方程组的解；值域可用来快速判断一个向量是否可以通过其它向量由这个线性变换得到。

定义 2.16: 矩阵的值域

矩阵 $A\in C^{m\times n}$ 的列向量 $a_1,a_2,\cdots,a_n$，则 A 的值域为

$$R(A)=Span(a_1,a_2,\cdots,a_n)=\{y\mid y=Ax\} \tag{2.8}$$

值域就是矩阵中的所有列向量张成的子空间。如果把 A 看作是一种线性变换，则矩阵的值域 $y=Ax$ 就是把线性空间中的原向量 x 经过线性变换后得到的像。

定义 2.17: 矩阵的核空间

$$N(A)=\{x\mid Ax=0\} \tag{2.9}$$

核空间又称为**零空间**，零空间的维数叫**零度**，记作 $n(A)$。矩阵的核空间就是使得 $Ax=0$ 成立的 x 所形成的空间。如果把 A 看成是一种线性变换，那么矩阵的核就是经过线性变换变成零向量的向量（原像）。

求导（微分）算子 在线性空间 $\mathbb{R}[x]_n$ 中，求导是一个线性变换，记作 $\boldsymbol{D}$：

$$\boldsymbol{D}f(x) = f'(x), (x) \in \mathbb{R}[x]_n$$

因为 $f^{(n)} = 0, \forall f(x) \in \mathbb{R}[x]_n$，有 $\boldsymbol{D}^n = \boldsymbol{0}$

平移变换 $\boldsymbol{S}_n$ 也是线性变换

$$\boldsymbol{S}_n f(x) = f(x+a), \forall f(x) \in \mathbb{R}[x]_n$$

投影算子 设 n 维线性空间 V 分解成 k 个非零子空间的直和，$V = W_1 \oplus W_2 \oplus \cdots W_k$。基 V 中的每个向量都能唯一地分解成各个子空间中向量的和，$\alpha = \alpha_1 + \alpha_2 + \cdots + \alpha_k, \alpha_i \in W_i, i = 1, 2, \cdots, k$，称 α_i 为向量 α 关于直和分解的第 i 个分量。称下面的变换 $\boldsymbol{P}_i : \alpha \to \alpha_i, \forall \alpha \in V$ 为直和分解的第 i 个**投影算子**。

可验证 $\boldsymbol{P}_i$ 是线性算子，而且具有如下性质：

1）$\boldsymbol{P}_i^2 = \boldsymbol{P}_i$；

2）$\boldsymbol{P}_i \boldsymbol{P}_j = 0$，当 $i \neq j$ 时；

3）$\boldsymbol{P}_1 + \boldsymbol{P}_2 + \cdots + \boldsymbol{P}_k = \boldsymbol{I}$

定义 2.18: 投影

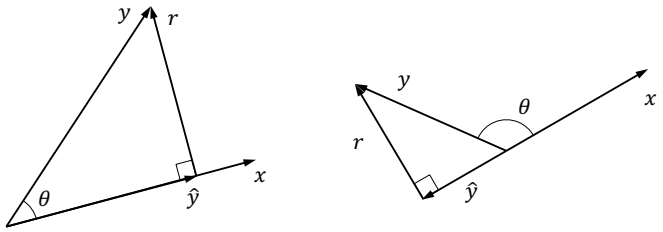
向量 y 在向量 x 上的**投影**为向量：

$$\hat{y} = \frac{\langle x, y \rangle}{\|x\|^2} x \tag{2.10}$$

定义 2.19: 剩余量或者残差

$$r = y - \hat{y} = y - \frac{\langle x, y \rangle}{\|x\|^2} x \tag{2.11}$$

重要性质： 残差向量 r 与投影向量 $\hat{y}$ 是正交的。



(a) 锐角 (b) 钝角

图 2.2: 投影的几何解释示意图。

正交变换 是线性变换的一种。如果对于任意向量 $\mathbf{u}$ 和 $\mathbf{v}$ 其内积等于正交转换后之向量 $T(\mathbf{u})$ 和 $T(\mathbf{v})$ 之内积，则称之为正交变换。$\langle \mathbf{u}, \mathbf{v} \rangle = \langle T(\mathbf{u}), T(\mathbf{v}) \rangle$。按照长度的定义，可知正交转换后的向量长度与转换前的长度相同 $\|T(\mathbf{x})\| = \|\mathbf{x}\|$。正交变换不会影响转换前后向量间的夹角和内积长度。

$$\begin{aligned}\widetilde{x}_1 &= \frac{1}{\|x_1\|}x_1 \\ \widetilde{x}_2 &= \frac{1}{\|x_2 - \langle \widetilde{x}_1, x_2\rangle \widetilde{x}_1\|}\left(x_2 - \langle \widetilde{x}_1, x_2\rangle \widetilde{x}_1\right)\end{aligned} \tag{2.12}$$

对于 $k = 2, 3, \cdots$ 我们可以给出一般式

$$\widetilde{x}_k = \left(x_k - \sum_{i=1}^{k-1} < \widetilde{x}_i, x_k > \widetilde{x}_i\right) \Big/ \left\|x_k - \sum_{i=1}^{k-1} < \widetilde{x}_i, x_k > \widetilde{x}_i\right\| \tag{2.13}$$

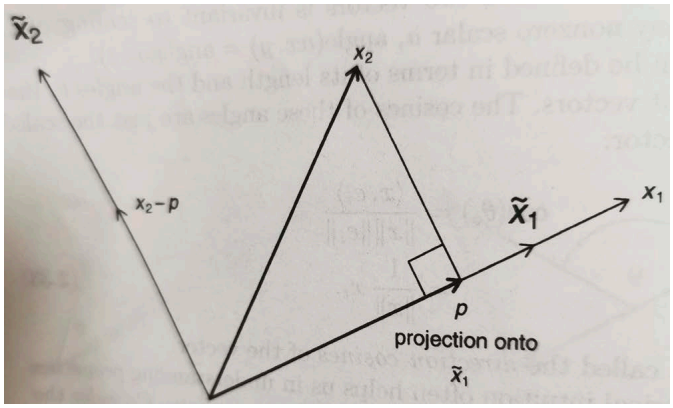


图 2.3: 向量正交化示意图。

定义 2.20: 内积空间

对所有向量 $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in V, \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{K}$, 映射函数 $\langle \cdot, \cdot \rangle$: $V \times V \to K$ 满足下列性质:

1) 共轭对称性 $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle^*$

2) 第一变元的线性性 $\langle \alpha \boldsymbol{x} + \beta \boldsymbol{y}, \boldsymbol{z} \rangle = \alpha \langle \boldsymbol{x}, \boldsymbol{z} \rangle + \beta \langle \boldsymbol{y}, \boldsymbol{z} \rangle$

3) 非负性 $\langle \boldsymbol{x}, \boldsymbol{x} \rangle \geq 0$，并且 $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0 \Leftrightarrow \boldsymbol{x} = 0$。

则称 $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ 为向量 $\boldsymbol{x}, \boldsymbol{y}$ 的内积，V 是内积向量空间。

2.1.1 Euclid 空间

定义 2.21: Euclid 空间

设 V 是实数域 $\mathbb{R}$ 上的线性空间，$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in V$，$c$ 是任意实数，在 V 上定义了一个二元实函数 $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{R}$, 称为内积, 具有如下性质:

1) $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle$

2) $\langle c\boldsymbol{x}, \boldsymbol{y} \rangle = c \langle \boldsymbol{x}, \boldsymbol{y} \rangle$

3) $\langle \boldsymbol{x} + \boldsymbol{y}, \boldsymbol{z} \rangle = \langle \boldsymbol{x}, \boldsymbol{z} \rangle + \langle \boldsymbol{y}, \boldsymbol{z} \rangle$

4) $\langle \boldsymbol{x}, \boldsymbol{x} \rangle \geq 0$，并且 $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0 \Leftrightarrow \boldsymbol{x} = 0$。

称定义了内积的实数域上的线性空间称为 **Euclid 空间**。

欧式空间中，定义向量的长度，$\sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle}$，记为 $\|\boldsymbol{x}\|$，零向量的长度为 0，长度的定义具有性质:

$$\|c\boldsymbol{x}\| = c\|\boldsymbol{x}\|, \forall \boldsymbol{x} \in V, c \in \mathbb{R}$$

且 $\dfrac{x}{\|x\|}$ 是单位向量。

定理 2.5: Cauchy-ByHHkobckHH 不等式

对欧氏空间 V 中任意两个向量 $\boldsymbol{x}, \boldsymbol{y}$，有 $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \neq \|\boldsymbol{x}\|\|\boldsymbol{y}\|$ 当且仅当 $\boldsymbol{x}, \boldsymbol{y}$ 线性相关时，等号成立。

定义 2.22: 正交变换

在 Euclid 空间 V 中，若线性变换 $\boldsymbol{A}$ 保持向量内积不变，即：

$$\langle \boldsymbol{A\alpha}, \boldsymbol{A\beta} \rangle = \langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle \tag{2.14}$$

则称 $\boldsymbol{A}$ 为正交变换。

三角不等式 $\|\boldsymbol{x} \pm \boldsymbol{y}\| \leq \|\boldsymbol{x}\|\|\boldsymbol{y}\|$

欧式空间中，由正交向量组构成的基称为**正交基**，由标准正交向量组构成的基称为**标准正交基**（归一化后）。

假设 $\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n$，是 n 维欧氏空间的基，则矩阵

$$\mathbf{M} = \begin{pmatrix} \langle \varepsilon_1, \varepsilon_1 \rangle & \langle \varepsilon_1, \varepsilon_2 \rangle & \cdots & \langle \varepsilon_1, \varepsilon_n \rangle \\ \langle \varepsilon_2, \varepsilon_1 \rangle & \langle \varepsilon_2, \varepsilon_2 \rangle & \cdots & \langle \varepsilon_2, \varepsilon_n \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varepsilon_n, \varepsilon_1 \rangle & \langle \varepsilon_n, \varepsilon_2 \rangle & \cdots & \langle \varepsilon_n, \varepsilon_n \rangle \end{pmatrix}$$

$\mathbf{M}$ 矩阵是实对称矩阵。由于对任意非零向量 $\boldsymbol{x} \pm \boldsymbol{y}$，内积 $\langle \varepsilon, \varepsilon \rangle > 0$，这个矩阵是正定的（对任意向量 X 都有 $X^\top MX > 0$，则 M 矩阵是正定的）。

向量与空间正交 在欧氏空间 V 中，如果一个向量 $\boldsymbol{\alpha}$ 与子空间 V_1 的每个向量都正交，则说 $\boldsymbol{\alpha}$ 与子空间 V_1 正交，记作：$\boldsymbol{\alpha} \perp V_1$

子空间正交 如果子空间 V_1 中每个向量都与子空间 V_2 的每个向量正交，则子空间 V_1 与子空间 V_2 正交，记作 $V_1 \perp V_2$。

子空间的正交补 若子空间 V_1 和子空间 V_2 正交，而且 $V = V_1 + V_2$，则说子空间 V_2 是子空间 V_1 的正交补，记作 $V_2 = V_1^\perp$

定义 2.23: Hermite 空间

在复数域 $\mathbb{C}$ 上的线性空间 V 上定义一个二元复函数，称为内积，记作 $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$
1) $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle = \overline{\langle \boldsymbol{\beta}, \boldsymbol{\alpha} \rangle}$，其中 $\overline{\langle \boldsymbol{\beta}, \boldsymbol{\alpha} \rangle}$ 是 $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$ 的共轭复数；
2) $\langle c\boldsymbol{\alpha}, \boldsymbol{\beta} \rangle = c \langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$
3) $\langle \boldsymbol{\alpha} + \boldsymbol{\beta}, \boldsymbol{\gamma} \rangle = \langle \boldsymbol{\alpha}, \boldsymbol{\gamma} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{\gamma} \rangle$
4) $\langle \boldsymbol{\alpha}, \boldsymbol{\alpha} \rangle$ 是非负实数，且 $\langle \boldsymbol{\alpha}, \boldsymbol{\alpha} \rangle = 0$ 当且仅当 $\boldsymbol{\alpha} = \mathbf{0}$
这样的空间被称为酉空间，也就是 Hermite 空间。

酉变换 在 Hermite 空间中，对 $\forall \boldsymbol{\alpha}, \boldsymbol{\beta} \in V$ 保持向量内积不变 $\langle \boldsymbol{A\alpha}, \boldsymbol{A\beta} \rangle = \langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$ 的线性变换（算子）$\boldsymbol{A}$ 称为酉变换，它在标准正交基下的矩阵是酉矩阵。（欧式空间里的正交变换就是酉变换）

Hermite 变换 在 Hermite 空间中，对 $\forall \boldsymbol{\alpha}, \boldsymbol{\beta} \in V$ 满足关系式 $\langle \boldsymbol{A\alpha}, \boldsymbol{\beta} \rangle = \langle \boldsymbol{\alpha}, \boldsymbol{A\beta} \rangle$ 的线性变换（算子）$\boldsymbol{A}$ 称为 Hermite 变换，它在标准正交基下的矩阵是 Hermite 矩阵。

伴随算子 对于 n 维内积空间 V 的线性算子 $\boldsymbol{A}$，存在唯一的线性算子 $\boldsymbol{A}^*$，使得

$$\langle \boldsymbol{A\alpha}, \boldsymbol{\beta} \rangle = \langle \boldsymbol{\alpha}, \boldsymbol{A}^* \boldsymbol{\beta} \rangle \tag{2.15}$$

其中，算子 $\boldsymbol{A}^*$ 称为 $\boldsymbol{A}$ 的**伴随算子（也叫共轭算子）**。若满足 $\boldsymbol{A}^* = \boldsymbol{A}$，则称为自伴随算子。如果线性算子 $\boldsymbol{A}$ 满足 $\boldsymbol{A}^*\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^*$，称为正规算子（其中 Euclid 空间中的对称算子，正交算子，Hermite 空间中的 Hermite 算子、酉算子都是正规算子）。

2.1.2 无限维内积空间（赋范空间）

定义 2.24: 赋范空间

赋范向量空间（英语：Normed vector space）是具有“长度”概念的向量空间。称函数 $\|\cdot\|$ 为 V 的一个范数。定义了范数的线性空间称为赋范空间。
1) 零向量的长度是零，并且任意向量的长度是非负实数。
2) 一个向量 $\mathbf{v}$ 乘以一个标量 a 时，长度应变为原向量 $\mathbf{v}$ 的 $|a|$（a 的绝对值）倍。即 $a\|\mathbf{v}\| = |a|\|\mathbf{v}\|$
3) 三角不等式成立。也就是说，对于两个向量 $\mathbf{v}$ 和 $\mathbf{u}$，它们的长度和（“三角形”的两边）大于 $\mathbf{v}+\mathbf{u}$（第三边）的长度：$\|\mathbf{v}+\mathbf{u}\| \leq \|\mathbf{v}\| + \|\mathbf{u}\|$。

定义 2.25: 度量空间

设 V 是非空集合，如果二元函数 $d(\cdot,\cdot) : V \times V \to \mathbb{R}$ 满足以下条件：
1) 非负性与同一性原则：$d(\alpha, \beta) \geq 0$ 且 $d(\alpha, \beta) = 0 \Rightarrow \alpha = \beta$
2) 对称性：$d(\alpha, \beta) = d(\beta, \alpha)$
3) 三角不等式：$d(\alpha, \gamma) \leq d(\alpha, \beta) + d(\beta, \gamma)$

2.2 内积与范数

常用范数:

- L_0 范数 $\|x\|_0$，表示非零元素的个数
- L_1 范数 $\|x\|_1 = \sum_{i=1}^{n} |x_i|$
- L_2 范数 $\|x\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$
- L_∞ 范数 $\|x\|_\infty = \max\{|x_1|, |x_2|, \cdots, |x_n|\}$
- L_p 范数 $\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}, p \geq 1$

2.2.1 矩阵的内积与范数

2.2.2 矩阵范数与向量范数的相容性

2.2.3 从属范数

2.2.4 长方阵的范数

2.3 矩阵的重要性指标

2.4 逆矩阵与伪逆矩阵

定义 2.26: 逆矩阵的定义与性质

$n \times n$ 矩阵，如果行列式不为零，或者具有 n 个线性无关的列向量或者 n 个线性无关的行向量，或者行满秩，或者列满秩，则这个矩阵就是非奇异矩阵。

若矩阵 $A \in \mathbb{C}^{n \times n}$ 的逆矩阵存在，则称矩阵 A 是非奇异的或是可逆的。下列性质等价:

(1) A 非奇异; (2) A^{-1} 存在; (3)$rank(A) = n$; (4)A 的行线性无关; (5)A 的列线性无关; (6)$\det(A) \neq 0$; (7)A 的值域的维数为 n; (8)A 的零空间的维数是 0; (9) $AX = b$ 对每一个 $b \in \mathbb{C}$ 都是一致方程; (10) $AX = b$ 对每一个 $b \in C$ 有唯一解; (11) $AX = 0$ 只有平凡解 $X = 0$。

定义 2.27: 左逆矩阵与右逆矩阵

一个矩阵 $A^{m \times n}$，若满足 $LA = I$，但不满足 $AL = I$ 的矩阵 L 称为 A 的左逆矩阵（left inverse)。类似，满足 $AR = I$，但不满足 $RA = I$ 的矩阵称为 A 的右逆矩阵。

注:

(1) 仅当 $m \geq n$ 时，矩阵 $A^{m \times n}$ 可能有左逆矩阵。

(2) 仅当 $m \leq n$ 时，矩阵 $A^{m \times n}$ 可能有右逆矩阵。

(3) 一个矩阵的左逆矩阵或者右逆矩阵往往不唯一。

定义 2.28: 广义（伪）逆矩阵

- 该矩阵对于奇异矩阵甚至长方矩阵都存在；
- 它具有通常逆矩阵的一些性质；
- 当矩阵非奇异时，它还原到通常的逆矩阵。

定义 2.29: 左伪逆矩阵

我们考虑若 $m > n$，并且 $rank(A) = n$，是列满秩，则 $n \times n$ 矩阵 $A^H A$ 是可逆的，可以验证

$$L = \left(A^H A\right)^{-1} A^H \tag{2.16}$$

满足左逆矩阵的定义 $LA = I$，这种左逆矩阵是唯一的，称为**左伪逆矩阵**。

定义 2.30: 右伪逆矩阵

我们考虑若 $m < n$，并且 $rank(A) = m$，是行满秩，则 $m \times m$ 矩阵 AA^H 是可逆的，可以验证

$$L = A^H \left(AA^H\right)^{-1} \tag{2.17}$$

满足左逆矩阵的定义 $LA = I$，这种右逆矩阵是唯一的，称为**右伪逆矩阵**。

定义 2.31: Moore-Penrose 逆矩阵

设矩阵 $\boldsymbol{A} \in \mathbb{C}^{m \times n}$，若矩阵 $\boldsymbol{X} \in \mathbb{C}^{n \times m}$ 满足一下 4 个 Penrose 方程

(1) $\boldsymbol{AXA} = \boldsymbol{A}$

(2) $\boldsymbol{XAX} = \boldsymbol{X}$

(3) $(\boldsymbol{AX})^H = \boldsymbol{AX}$

(4) $(\boldsymbol{XA})^H = \boldsymbol{XA}$

则称 $\boldsymbol{X}$ 为 $\boldsymbol{A}$ 的 Moore-Penrose 逆, 记为 $\boldsymbol{A}^+$。

特例：

- 若 $\boldsymbol{A}$ 是可逆矩阵，则 $\boldsymbol{A}^+ = \boldsymbol{A}^{-1}$；
- 若 $\boldsymbol{A} = \boldsymbol{O}_{m \times n}$，则 $\boldsymbol{A}^+ = \boldsymbol{O}_{n \times m}$；
- 若 $\boldsymbol{A} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$，则 $\boldsymbol{A}^+ = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$。

2.5 Moore-Penrose 逆矩阵的计算方法

2.5.1 利用满秩分解求广义逆矩阵

2.5.2 计算 A^+ 的 Zlobec 公式

2.5.3 Greville 方法

2.5.4 一些特殊分块矩阵的广义逆矩阵

2.6

3 Optimization

3.1 优化基础

$\arg\min_x f(x)$ 表示的是使得 $f(x)$ 取得最小值时的决策变量 x 值，而 $\min_x f(x)$ 指的是决策变量为 x 时的最小函数值。

导数，偏导数，极值点，鞍点（一个方向极小，但另一个方向却是极大点），方向导数（标量）

梯度 ，如果函数 $f(x,y)$ 在 $p_0(x_0,y_0)$ 点的两个偏导数 $f_x^{'}(x_0,y_0)$ 和 $f_y^{'}(x_0,y_0)$ 都存在，则把 $f_x^{'}(x_0,y_0)\mathbf{i}+f_y^{'}(x_0,y_0)\mathbf{j}$ 称为函数 $f(x,y)$ 在 $p_0(x_0,y_0)$ 点的梯度。记作 $\mathbf{grad}f(x_0,y_0)$ 和 $\nabla f(x_0,y_0)$。

定义 3.1: 凸集

设 x,y 为欧式空间 E^n 中相异的两个点，则点集 $P=\{\lambda x+(1-\lambda)y \mid \lambda\in R\}$，称为通过 x 和 y 的直线。

设 $S\subseteq E^n$，若对 $\forall x^{(1)},x^{(2)}\in S$ 以及 $\forall\lambda\in[0,1]$，都有 $\lambda x^{(1)}+(1-\lambda)x^{(2)}\in S$，则称 S 为**凸集**。

设 $x^{(1)},x^{(2)},\cdots,x^{(k)}\in S$，称

$$\lambda_1 x^{(1)}+\lambda_2 x^{(2)}+\cdots+\lambda_k x^{(k)}$$

（其中 $\lambda_1+\lambda_2+\cdots+\lambda_k=1$）为 $x^{(1)},x^{(2)},\cdots,x^{(k)}$ 的**凸组合**。

在点集拓扑学与欧几里得空间中，凸集（Convex set）是一个点集合，其中每两点之间的直线点都落在该点集合中，如图3.1所示。

超平面（hyper-plane） ，$H=\{x \mid p^Tx=a\}$

（闭）半空间 ，$H^-=\{x \mid p^Tx\le a\}$

射线 ，$L=\{x \mid x=x^{(0)}+\lambda d,\lambda\ge 0\}$

定理 3.1: 凸集的性质

设 S_1 和 S_2 为 E^n 中的两个凸集，β 是实数，则

(1) $\beta S_1=\{\beta x \mid x\in S_1\}$

(2) $S_1\cap S_2$，如图3.2所示。

(3) $S_1+S_2=\{x^{(1)}+x^{(2)} \mid x^{(1)}\in S_1,x^{(2)}\in S_2\}$

(4) $S_1-S_2=\{x^{(1)}-x^{(2)} \mid x^{(1)}\in S_1,x^{(2)}\in S_2\}$

都是**凸集**。

定义 3.2: 凸锥

设有集合 $C\subset E^n$，若对 C 中的每一点 x，及任意的 $\lambda\le 0$，都有 $\lambda x\in C$，则称 C 为锥；若 C 为凸集，则称 C 为凸锥。

(a) 凸集 (b) 非凸集

图 3.1: 凸集与非凸集。

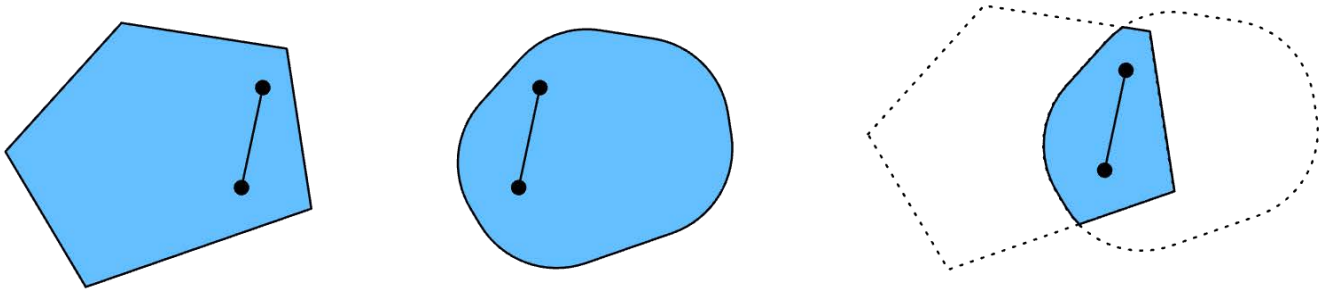


图 3.2: 两个凸集的并集也是凸集。

定义 3.3: 多面体

有限个闭半空间的交 $\{x \mid Ax \leq b\}$ 称为多面体。
非空有界的多面体称为多胞形（polytope）。

定义 3.4: 极点

设 S 是非空凸集，$x \in S$，若由 $x = \lambda x^{(1)} + (1-\lambda)x^{(2)}$，其中 $\lambda \in (0,1), x^{(1)}, x^{(2)} \in S$，必推出 $x = x^{(1)} = x^{(2)}$，则称 x 是 S 的极点。如图3.3所示。即，不能通过不同元素组成的真凸组合表示的点。

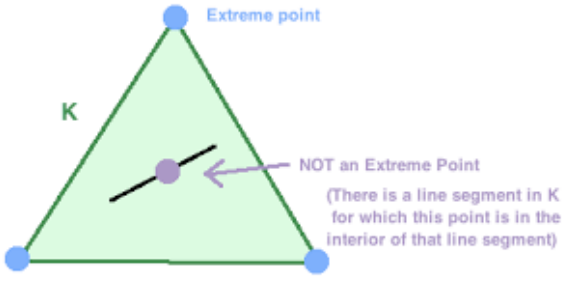


图 3.3: 凸集的极点与内点。

定义 3.5: 极方向

如图3.4所示，设 S 为 E^n 中的闭凸集，$d \in E^n, d \neq 0$，如果对 $\forall x \in S$，有

$$\{x + \lambda d \mid \lambda \geq 0\} \subset S$$

则称向量 d 为 S 的方向。设 $d^{(1)}, d^{(2)}$ 为 S 的主方向，若对任意的 $\lambda > 0$，有 $d^{(1)} \neq \lambda d^{(2)}$，则称 $d^{(1)}, d^{(2)}$ 是两个不同的方向。若 S 的方向 d 不能表示为该集合的两个不同方向的正的线性组合，则称 d 为 S 的极方向。

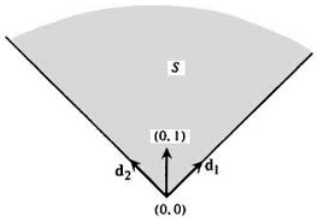


图 3.4: 凸集的极方向，$d^{(1)}$ 与 $d^{(2)}$。

定理 3.2: 凸集分离定理

设 S_1 和 S_2 是 E^n 中两个非空集合，$H = \{x \mid p^T x = \alpha\}$ 为超平面，如果 $\forall x \in S_1$，都有 $p^T x \geq \alpha$，对 $\forall x \in S_2$，都有 $p^T x \leq \alpha$（或情况恰好相反），则称超平面 H 分离集合 S_1 和 S_2。典型的案例如 SVM。

定义 3.6: 凸函数

设 S 是 E^n 中的非空凸集，$f(x)$ 是定义在 S 上的实函数，如果对于每一对 $x_1, x_2 \in S$ 及每一个 $a, 0 \leq a \leq 1$ 都有

$$f(ax_1 + (1-a)x_2) \leq af(x_1) + (1-a)f(x_2)$$

则称函数 $f(x)$ 为 S 上的凸函数。上式中，若 $\leq$ 变为 $<$，则称为严格凸函数。若 $-f(x)$ 为 S 的凸函数，则称 $f(x)$ 为 S 上的凹函数。如图3.5所示。

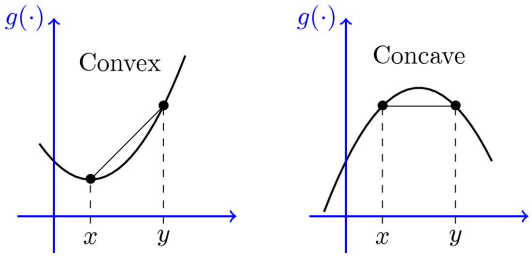


图 3.5: 凸函数与凹函数。

3.2 无约束优化算法

3.2.1 最速下降法

最速下降法 (Steepest Descent Method) [1,2]，解决的问题是无约束优化问题，而所谓的无约束优化问题就是对目标函数的求解，没有任何的约束限制的优化问题。

算法 3.1: 最速下降法

1. 选取初始点 x^0，置 $k := 0$
2. 计算梯度 $g^k = g(x^k) = \nabla f(x^k)$
3. 若 $\nabla f(x^k) = 0$，或小于一定数值，则停止计算；否则从 x^k，沿 $d^k = -g(x^k)$ 进行一维搜索，即求 α_k，使得

$$f(x^k + \alpha_k d^k) = \min\{f(x^k + \alpha d^k) \mid \alpha \geq 0\}$$

4. 置 $x^{k+1} = x^k + \alpha_k d^k$，置 $k = k + 1$，转第 2 步。

3.2.2 牛顿法

牛顿迭代法（Newton's method）又称为牛顿-拉夫逊（拉弗森）方法（Newton-Raphson method），它是牛顿在 17 世纪提出的一种在实数域和复数域上近似求解方程的方法。

算法 3.2: 牛顿法

1. 选取初始点 x^0，置 $k := 0$
2. 计算梯度 $g^k = g(x^k) = \nabla f(x^k)$
3. 若 $\nabla f(x^k) = 0$，或小于一定数值，则停止计算；否则计算 Hessian 矩阵 $G_k = G(x^k)$，并求出搜索方向 $d^k : G_k d^k = -g^k$，即 $d^k = -G_k^{-1} g^k$
4. 置 $x^{k+1} = x^k + d^k$，置 $k = k + 1$，转第 2 步。

Hessian 矩阵，假设有一实值函数 $f(x_1, x_2, \ldots, x_n)$，如果 f 的所有二阶偏导数都存在并在定义域内连续，那么函数 f 的 Hessian 矩阵 $\mathbf{H}$ 为：

$$\mathbf{H} = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

算法 3.3: 阻尼牛顿法

1. 选取初始点 x^0，置 $k := 0$
2. 计算梯度 $g^k = g(x^k) = \nabla f(x^k)$
3. 若 $\nabla f(x^k) = 0$，或小于一定数值，则停止计算；否则计算 Hessian 矩阵 $G_k = G(x^k)$，并求出搜索方向 $d^k = -G_k^{-1} g^k$，从 d^k 方向进行一维搜索，求 α_k，使得 $f(x^k + \alpha_k d^k) = \min\{f(x^k + \alpha d^k) \mid \alpha \geq 0\}$
4. 置 $x^{k+1} = x^k + \alpha_k d^k$，置 $k = k + 1$，转第 2 步。

阻尼牛顿法能够保证函数值在迭代过程中不会上升。

3.2.3 共轭梯度法

共轭梯度法（Conjugate Gradient）是介于最速下降法与牛顿法之间的一个方法，它仅需利用**一阶导数**信息，但克服了最速下降法收敛慢的缺点，又避免了牛顿法需要存储和计算 Hessian 阵并求逆的缺点，共轭梯度法不仅是解决大型线性方程组最有用的方法之一，也是解**大型非线性最优化**最有效的算法之一。

3.2.4 拟牛顿算法

针对大规模优化问题，Hessian 阵的计算非常复杂。这种情况下，最好用 Hessian 阵的近似值替代。核心思想也是泰勒展开。

拟牛顿条件 假设初始点 x^0，下一个迭代点 $x^1 = x^0 + \alpha_0 d^0$。将函数 $f(x)$ 进行泰勒展开得：

$$f(x) \approx f(x^1) + (x - x^1)\nabla f(x^1) + \frac{1}{2}(x - x^1)^T \nabla^2 f(x^1)(x - x^1) + \cdots$$

而在 x^1 附近 (上式对 x 求导)：

$$\nabla f(x) \approx \nabla f(x^1) + \nabla^2 f(x^1)(x - x^1)$$

如果此时取 $x = x^0$，则：

$$\nabla f(x^0) \approx \nabla f(x^1) + \nabla^2 f(x^1)(x^0 - x^1)$$

$$\nabla f(x^1) - \nabla f(x^0) \approx +\nabla^2 f(x^1)(x^1 - x^0)$$

此时我们记 $s^0 = x^1 - x^0$，$y^0 = \nabla f(x^1) - \nabla f(x^0)$，即前后两迭代点数值之差与梯度之差。则我们希望 $y^0 \approx G_1 s^0$，故**拟牛顿方程**：

$$H_{k+1} y^k = s^k \tag{3.1}$$

而**拟牛顿方向**：

$$d^{k+1} = -H_{k+1} \nabla f(x^k) \tag{3.2}$$

从拟牛顿方程3.1中计算得到 H_k，当 H_k 已知时，可以通过计算修正矩阵 δH_k 从而得到 H_{k+1}，进而直接获取下一个搜索方向 $d^{k+1} = -H_{k+1}\nabla f(x^k)$。

$$H_{k+1} = H_k + \Delta H_k \tag{3.3}$$

拟牛顿法的修正公式有秩-1 修正和秩-2 修正等较为简单的情况。其中比较著名的有 DFP 修正公式、BFGS 修正公式、Broyden 算法族和 Huang 算法族。

秩-1 修正公式 当 ΔH_k 满足其秩为 1 时，最简的 H_{k+1} 构造方式。此时可以令

$$\Delta H_k = uv^T$$

其中 $u, v \in \mathbb{R}^n$，ΔH_k 的各行成比例，其秩为 1。秩为 1 的矩阵有如下特点，因为 Hessian 矩阵也是对称矩阵，所以各行成比例，矩阵的秩为 1。

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} a_1 b_1 & a_1 b_2 & a_1 b_3 \\ a_2 b_1 & a_2 b_2 & a_2 b_3 \\ a_3 b_1 & a_3 b_2 & a_3 b_3 \end{bmatrix}$$

故我们可以将 $H_{k+1} = H_k + uv^T$ 带入拟牛顿方程3.1得 $(H_k + uv^T)y^k = s^k$ 即：

$$u = \frac{1}{v^T y^k}(s^k - H_k y^k) \tag{3.4}$$

$$H_{k+1} = H_k + \frac{1}{v^T y^k}(s^k - H_k y^k)v^T \tag{3.5}$$

而 H_{k+1} 具有成对性，则向量 v 与 $s^k - H_k y^k$ 成比例，即

$$v = \alpha(s^k - H_k y^k) \tag{3.6}$$

联立可得：

$$H_{k+1} = H_k + \frac{1}{(s^k - H_k y^k)^T y^k}(s^k - H_k y^k)(s^k - H_k y^k)^T \tag{3.7}$$

公式3.7就是秩-1 修正公式。

算法 3.4: 秩-1 修正算法

1. 选取初始点 x^0，计算 $g^0 = \nabla f(x^0)$，如果 g^0 小于设定值则停止计算，否则转 2.
2. 置 $k := 1$，$H_0 = I$（I 为单位阵）
3. 置搜索方向 $d^k = -H_k g^k$
4. 一维线性搜索，从 d^k 方向进行一维搜索，求 α_k，使得 $f(x^k + \alpha_k d^k) = \min\{f(x^k + \alpha d^k) \mid \alpha \geq 0\}$
5. 置 $x^{k+1} = x^k + \alpha_k d^k$
6. 计算 $g^0 = \nabla f(x^0)$，如果 g^0 小于设定值则停止计算，否则置

$$H_{k+1} = H_k + \frac{1}{(s^k - H_k y^k)^T y^k}(s^k - H_k y^k)(s^k - H_k y^k)^T$$

其中 $s^k = x^{k+1} - x^k, y^k = g^{k+1} - g^k$

7. 置 $k : k + 1$ 转步骤 3

DFP (Davidon-Fletcher-Powell) 由拟牛顿公式3.1与3.3联立可计算新公式：

$$\Delta H_k y^k = s^k - H_k y^k, \quad s^k = x^{k+1} - x^k, \quad y^k = \nabla f(x^{k+1}) - \nabla f(x^k) \tag{3.8}$$

而此方程为一个不确定性方程，因此我们可以令 $\Delta H_k = -H_k y^k (q^k)^T + s^k (w^k)^T$，即用公式3.8的两项加权表示，则：

$$\Delta H_k y^k = -H_k y^k (q^k)^T y^k + s^k (w^k)^T y^k$$

保证与公式3.8一致，可以使 $(q^k)^T y^k = (w^k)^T y^k = 1$，但不能设 $y^k (q^k)^T = y^k (w^k)^T = 1$。此时我们可以使：

$$H_{k+1} = H_k - H_k y^k (q^k)^T + s^k (w^k)^T \tag{3.9}$$

此时修正矩阵 $\Delta H_k = -H_k y^k (q^k)^T + s^k (w^k)^T$ 是一个秩不超过 2 的矩阵，故公式3.9也称为秩-2 修正矩阵。这里我们需要保证 H_{k+1} 为对称矩阵，则：

$$H_{k+1}^T = H_k^T - q^k (y^k)^T H_k^T + w^k (s^k)^T = H_k - q^k (y^k)^T H_k + w^k (s^k)^T = H_{k+1} \tag{3.10}$$

联立公式3.9与3.10，以及之前提到的条件 $(q^k)^T y^k = (w^k)^T y^k = 1$，我们可以计算得到

$$q^k = \frac{H_k y^k}{(y^k)^T H_k y^k} \tag{3.11}$$

$$w^k = \frac{s^k}{(s^k)^T y^k} \tag{3.12}$$

故可以得到 H_{k+1} 的秩-2 修正公式：

$$H_{k+1} = H_k + \frac{s^k (s^k)^T}{(s^k)^T y^k} - \frac{H_k y^k (y^k)^T H_k}{(y^k)^T H_k y^k} \tag{3.13}$$

算法 3.5: DFP 算法

1. 选取初始点 x^0，计算 $g^0 = \nabla f(x^0)$，如果 g^0 小于设定值则停止计算，否则转 2.
2. 置 $k := 1$，$H_0 = I$（I 为单位阵）
3. 置搜索方向 $d^k = -H_k g^k$
4. 一维线性搜索，从 d^k 方向进行一维搜索，求 α_k，使得 $f(x^k + \alpha_k d^k) = \min\{f(x^k + \alpha d^k) \mid \alpha \geq 0\}$
5. 置 $x^{k+1} = x^k + \alpha_k d^k$
6. 计算 $g^0 = \nabla f(x^0)$，如果 g^0 小于设定值则停止计算，否则置

$$H_{k+1} = H_k + \frac{s^k (s^k)^T}{(s^k)^T y^k} - \frac{H_k y^k (y^k)^T H_k}{(y^k)^T H_k y^k}$$

其中 $s^k = x^{k+1} - x^k, y^k = g^{k+1} - g^k$

7. 置 $k : k + 1$ 转步骤 3

Broyden 族和 BFGS (Broyden-Fletcher-Goldfarb-Shanno) 针对 DFP 的公式3.11与公式3.12，可以表示为：

$$q^k = \frac{H_k y^k}{(y^k)^T H_k y^k} = \alpha_k H_k y^k$$

$$w^k = \frac{s^k}{(s^k)^T y^k} = \beta_k s^k$$

这里我们将其改写为：

$$q^k = \frac{H_k y^k}{(y^k)^T H_k y^k} = \alpha_k H_k y^k + \beta_k s^k$$

$$w^k = \frac{s^k}{(s^k)^T y^k} = -\beta_k H_k y^k + \alpha_k s^k$$

由条件 $(q^k)^T y^k = (w^k)^T y^k = 1$（见上一节 DFP 算法）可得：

$$q^k = \frac{\left(1 - \beta_k \left(s^k\right)^T y^k\right)}{\left(y^k\right)^T H_k y^k} H_k y^k + \beta_k s^k$$

$$w^k = \frac{\left(1 - \beta_k \left(s^k\right)^T y^k\right)}{\left(y^k\right)^T H_k y^k} s^k - \beta_k H_k y^k$$

带入公式3.9中可得：

$$H_{k+1} = H_k + \frac{s^k \left(s^k\right)^T}{\left(s^k\right)^T y^k} - \frac{H_k y^k \left(y^k\right)^T H_k}{\left(y^k\right)^T H_k y^k} + \beta_k \left(\left(s^k\right)^T y^k\right) \left(\left(y^k\right)^T H_k y^k\right) v^k \left(v^k\right)^T \tag{3.14}$$

其中 $v^k = \frac{s^k}{(s^k)^T y^k} - \frac{H_k y^k}{(y^k)^T H_k y^k}$，公式3.14称为 Broyden 族秩-2 修正公式。由于参数 β_k 可以任意取值，不妨设 $\beta_k = \frac{\theta}{(s^k)^T y^k}$，且令 $\tilde{v}^k = \left((y^k)^T H_k y^k\right)^{\frac{1}{2}} \left(\frac{s^k}{(s^k)^T y^k} - \frac{H_k y^k}{(y^k)^T H_k y^k}\right)$，用于取代 v^k，则可以将公式3.14改写为：

$$\begin{aligned} H_{k+1} &= H_k + \frac{s^k (s^k)^T}{(s^k)^T y^k} - \frac{H_k y^k (y^k)^T H_k}{(y^k)^T H_k y^k} + \theta_k \tilde{v}^k (\tilde{v}^k)^T \\ &= H_{k+1}^{DFP} + \theta_k \tilde{v}^k (\tilde{v}^k)^T \end{aligned} \tag{3.15}$$

当 $\theta_k = 0$ 时就是 DFP 公式，而当 $\theta_k = 1$ 时即为 BFGS 修正公式。

$$H_{k+1} = \left(I - \frac{s^k (y^k)^T}{(s^k)^T y^k}\right) H_k \left(I - \frac{s^k (y^k)^T}{(s^k)^T y^k}\right)^T + \frac{s^k (s^k)^T}{(s^k)^T y^k} \tag{3.16}$$

LBFGS LBFGS 算法通过存储 k 个最近的低秩-1 修正的 BFGS 更新来保持一个低秩的近似 Hessian 矩阵的逆矩阵，代替全 Hessian 矩阵的逆矩阵。当连续的完全梯度 $\nabla F(w_t), \nabla F(w_{t-1})$ 可用时，标准秩-1 修正可用于直接估计 Hessian 矩阵的逆矩阵 G^{-1}。

在 BFGS 算法中，需要用到 $H_k \in \mathbb{R}^{n\times n}$，当 n 很大时，存储该矩阵需要很大的内存。对 BFGS 算法进行近似，不存完整的矩阵 $H_k \in \mathbb{R}^{n\times n}$，而是存储计算过程中的向量序列 $\{s^k\}$ 和 $\{y^k\}$，需要矩阵 H_k 时，利用向量 $\{s^k\}$ 和 $\{y^k\}$ 计算得到。而且，向量 $\{s^k\}$ 和 $\{y^k\}$ 也不是所有的都存储，而是固定存储最新的 m（一般是 3-20）个。

由 H_k 的 BFGS 公式

$$H_{k+1} = \left(I - \frac{s^k (y^k)^T}{(s^k)^T y^k}\right) H_k \left(I - \frac{s^k (y^k)^T}{(s^k)^T y^k}\right)^T + \frac{s^k (s^k)^T}{(s^k)^T y^k}$$

记 $\rho_k = \frac{1}{(s^k)^T y^k}, V_k = I - \rho_k (s^k) (y^k)^T$，则 ρ_k, V_k 只与 s^k, y^k 有关，且 $H_{k+1} = V_k^T H_k V_k + \rho_k s^k (s^k)^T$，可以得到递推关系：

$$\begin{aligned} H_2 &= V_1^T H_1 V_1 + \rho_1 s^1 (s^1)^T \\ H_3 &= V_2^T H_2 V_2 + \rho_2 s^2 (s^2)^T = V_2^T \left(V_1^T H_1 V_1 + \rho_1 s^1 (s^1)^T\right) V_2 + \rho_2 s^2 (s^2)^T \\ &= V_2^T V_1^T H_1 V_1 V_2 + V_2^T \rho_1 s^1 (s^1)^T V_2 + \rho_2 s^2 (s^2)^T \\ H_4 &= V_3^T H_3 V_3 + \rho_3 s^3 (s^3)^T = V_3^T \left(V_2^T V_1^T H_1 V_1 V_2 + V_2^T \rho_1 s^1 (s^1)^T V_2 + \rho_2 s^2 (s^2)^T\right) V_3 + \rho_3 s^3 (s^3)^T \\ &= V_3^T V_2^T V_1^T H_1 V_1 V_2 V_3 + V_3^T V_2^T \rho_1 s^1 (s^1)^T V_2 V_3 + V_3^T \rho_2 s^2 (s^2)^T V_3 + \rho_3 s^3 (s^3)^T \end{aligned}$$

一般情况为：

$$\begin{aligned} H_{k+1} =& V_k^T V_{k-1}^T \cdots V_1^T H_1 V_1 V_2 \cdots V_k + V_k^T V_{k-1}^T \cdots V_2^T \rho_1 s^1 (s^1)^T V_2 V_3 \cdots V_k \\ &+ \cdots + V_k^T \rho_{k-1} s^{k-1} (s^{k-1})^T V_k + \rho_k s^k (s^k)^T \end{aligned} \tag{3.17}$$

3.2.5 SAG

SAG 算法 [4] 算法在内存中为每个样本都维护一个旧的梯度 y_i，随机选择一个样本 i 来更新 d，并用 d 来更新参数 w。具体得说，更新的项 d 来自于用新的梯度 $\nabla f_i(w)$ 替换掉 d 中的旧梯度 y_i，这也就是 $d = d + y_i$ 表达的意

思。如此，每次更新的时候仅仅需要计算一个样本的梯度，而不是所有样本的梯度。计算开销与 SGD 无异，但是内存开销要大得多。

算法 3.6: SAG 算法

对 $i = 1, 2, \cdots, n,$，$d = 0$，$y_i = 0$

for $k = 1, 2, \cdots$, **do**

从 $\{1, 2, \cdots, n\}$ 中抽取样本 i;

$d = d - y_i$;

$y_i = \nabla f_i(w)$

$d = d + y_i$

$w = w - \frac{\alpha}{n} d$

End for

3.2.6 SAGA

SAGA 算法在每次迭代计算一个随机向量 g_k 作为之前迭代计算的随机梯度的平均。具体来说，在第 k 次迭代，SAGA 对所有 $i \in \{1, \cdots, n\}$ 储存 $\nabla f_i(w_{[i]})$，其中 $w_{[i]}$ 代表在最新的迭代中计算的是 ∇f_i。然后随机选取 $j \in \{1, \cdots, n\}$，则随机向量如下：

$$g_k = \nabla f_j(w_k) - \nabla f_j(w_{[j]}) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_{[i]}) \tag{3.18}$$

算法 3.7: SAGA 算法

初始化：$w_1 \in \mathbb{R}^d$，步长 $\alpha > 0$;

for $i = 1, 2, \cdots, n$, **do**

计算 $\nabla f_i(w_1)$，并且 $\nabla f_i(w_{[i]}) \leftarrow \nabla f_i(w_1)$;

End for

for $k = 1, 2, \cdots$, **do**

按均匀分布从 $\{1, 2, \cdots, n\}$ 中选择 j

计算 $\nabla f_i(w_k)$

$g_k \leftarrow \nabla f_j(w_k) - \nabla f_j(w_{[j]}) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_{[i]})$

$\nabla f_j(w_{[j]}) \leftarrow \nabla f_j(w_k)$

$w_{k+1} \leftarrow w_k - \alpha g_k$

End for

3.2.7 SVRG

SVRG 算法 [3] 是在一定次数更新后，计算一次全梯度去作为偏置项控制随机梯度的方差。每一次外循环计算全梯度 $\nabla R_n(w_k) = \frac{1}{n} \sum_i \nabla f_i(w_k)$，然后初始化 $\tilde{w}_1 = w_k$，进行内循环 $\tilde{w}_{j+1} = \tilde{w}_j - \alpha \tilde{g}_i$，其中

$$\tilde{g}_j = \nabla f_{i_j}(\tilde{w}_j) - \left(\nabla f_{i_j}(w_k) - \nabla R_n(w_k)\right)$$

其中，最后两项可以看做是对于梯度估计 $\nabla f_{i_j}(\tilde{w}_j)$ 的偏置项。SVRG 方法是通过不断减少方差上界来实现方差缩减的。

算法 3.8: SVRG 算法

初始化：$w_1 \in \mathbb{R}^d$，步长 $\alpha > 0$，内循环次数 m；
for $k = 1, 2, \cdots$, **do**
　计算批样本梯度 $\nabla R_n(w_k)$；
　$\tilde{w}_1 \leftarrow w_k$；
　for $j = 1, 2, \cdots, m$ **do**
　　按照均匀分布从 $\{1, 2, \cdots, n\}$ 中选择 i_j；
　　令：$\tilde{g}_j \leftarrow \nabla f_{i_j}(\tilde{w}_j) - \left(\nabla f_{i_j}(w_k) - \nabla R_n(w_k)\right)$
　　$\tilde{w}_{j+1} \leftarrow \tilde{w}_j - \alpha \tilde{g}_j$；
　End for
　选择 (a): 令 $w_{k+1} \leftarrow \tilde{w}_{m+1}$
　选择 (b): 令 $w_{k+1} \leftarrow \dfrac{1}{m}\sum_{j=1}^{m} \tilde{w}_{j+1}$
　选择 (c): 按照均匀分布从 $\{1, 2, \cdots, m\}$ 中选择 j，并且令 $w_{k+1} \leftarrow \tilde{w}_{j+1}$
End for

3.2.8 SARAH

SARAH 算法也在不断修正偏置项。

$$v_t = \nabla f_{i_t}\left(w^t\right) - \nabla f_{i_t}\left(w^{t-1}\right) + v_{t-1}$$
$$w^{t+1} = w^t - \eta v_t$$

具体的算法流程如下

算法 3.9: SARAH 算法

初始化：$\tilde{w}_0$，步长 $\eta > 0$，内循环次数 m；
for $s = 1, 2, \cdots$, **do**
　$w_0 \leftarrow \tilde{w}_{s-1}$；
　$v_0 \leftarrow \dfrac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_0)$
　$w_1 \leftarrow w_0 - \eta v_0$
　for $t = 1, 2, \cdots, m-1$ **do**
　　按照均匀分布从 n 个样本中选择样本 i_t；
　　$v_t \leftarrow \nabla f_{i_t}(w_t) - \nabla f_{i_t}(w_{t-1}) + v_{t-1}$
　　$w_{t+1} \leftarrow w_t - \eta v_t$；
　End for
　按照均匀分布从 $\{1, 2, \cdots, m\}$ 中选择 t，并且令 $\tilde{w}_s \leftarrow \tilde{w}_t$
End for

3.3 约束优化算法

3.3.1 对偶算法

对偶理论是研究线性规划中原始问题与对偶问题之间关系的理论。在线性规划早期发展中最重要的发现是对偶问题，即每一个线性规划问题（称为原始问题）有一个与它对应的对偶线性规划问题（称为对偶问题）。对偶将原始问题变为凸问题。对偶问题的解提供了原始问题（假设是最小化问题）的下限 [5]。

一般形式的约束规划问题：

$$\begin{aligned} &\min f(x) \\ \text{s.t. } &g_i(x) \le 0, i \in I = \{1, 2, \cdots, m\} \\ &h_j(x) = 0, j \in E = \{1, 2, \cdots, p\} \\ &x \in \mathbf{D} \subset \mathbb{R}^n \end{aligned} \tag{3.19}$$

则，拉格朗日函数 $\mathcal{L}(x, \lambda, \mu)$

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{i \in I} \lambda_i g_i(x) + \sum_{j \in E} \mu_j h_i(x) \tag{3.20}$$

定义 3.7: 鞍点定理

鞍点对于优化问题（公式3.19），若存在 $x^* \in \mathbb{R}^n$ 和 $\lambda^* \in \mathbb{R}_+^{|I|}$ 和 $\mu^* \in \mathbb{R}^{|E|}$ 满足

$$\mathcal{L}(x^*, \lambda, \mu) \le \mathcal{L}(x^*, \lambda^*, \mu^*) \le \mathcal{L}(x, \lambda^*, \mu^*)$$

则 $\mathcal{L}(x^*, \lambda^*, \mu^*)$ 成为该优化问题的 Language 函数的鞍点。也将 x^* 成为公式3.19的鞍点。

x^* 是 $\mathcal{L}(x, \lambda, \mu)$ 在 $\lambda = \lambda^*, \mu = \mu^*$ 时的极小值，(λ^*, μ^*) 是 $\mathcal{L}(x, \lambda, \mu)$ 在 $x = x^*$ 时的极大值点。

3.3.2 对偶及鞍点问题

3.4 经验风险及其一般优化模型

定义 3.8: 损失函数

损失函数使针对单个样本而言的，表示的是模型预测的值与样本真实值之间的差距。

定义 3.9: 过拟合与欠拟合

过拟合：如果算法在样本集上效果好，而在全局数据集上效果不好，那么认为这个算法能力过强了，称为过拟合（Overfitting）。

欠拟合：如果算法在样本集和数据集上效果都不好，则认为这个算法的能力不行，称为欠拟合（Underfitting）

数据集常被分为：

训练集 (Train Set)：训练不同的算法模型。

验证集 (Validation Set)：验证不同的模型，选择最合适的模型。

测试集 (Test Set)：在验证的模型上得到的测试准确度。

3.5 结构风险及其一般优化模型

最佳模型是对经验风险和泛化误差的整体最佳。

3.5.1 常见的正则化

Bibliography

[1] Haskell B Curry. The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics*, 2(3):258–261, 1944.

[2] Juan C Meza. Steepest descent. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(6):719–722, 2010.

[3] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

[4] Nicolas Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. *Advances in neural information processing systems*, 25, 2012.

[5] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

4 模式识别与机器学习

4.1 贝叶斯判别

定义 4.1: 基本概念

条件概率:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \tag{4.1}$$

全概率公式:

$$P(A) = P(A \cap B) + P(A \cap C) = P(A \mid B) \times P(B) + P(A \mid C) \times P(C) \tag{4.2}$$

为了要确认 x 是属于 w_1 类还是 w_2 类，通常要看 x 是来自 w_1 类的概率大还是来自 w_2 类的概率大。

似然函数:

$$p(w_i \mid x) = \frac{p(x \mid w_i) \cdot p(w_i)}{p(x)} = \frac{p(x \mid w_i) \cdot p(w_i)}{\sum_j p(x \mid w_j) \cdot p(w_j)} \tag{4.3}$$

似然比:

$$l_{12}(x) = \frac{p(x \mid w_1)}{p(x \mid w_2)} \tag{4.4}$$

判决阈值: $p(w_2)/p(w_1)$

4.1.1 正态分布模式的朴素贝叶斯分类器

通常这种情况是在给定的数据是连续而非离散的情况，假设在 w_i 类样本的第 i 个属性上取均值和方差，则有:

$$p(x \mid w_i) = \frac{1}{\sqrt{2\pi}\sigma_{c,i}} \exp\left(-\frac{(x-\mu_{c,i})^2}{2\sigma_{c,i}^2}\right) \tag{4.5}$$

已知类别 w_i 的判别函数可以写成如下:

$$d_i(x) = P(x \mid w_i)P(w_i) \tag{4.6}$$

或者: $d_i(x) = \ln P(x \mid w_i) + \ln P(w_i)$

若两类都满足正态分布:

$$d_1(x) - d_2(x) = \ln P(w_1) - \ln P(w_2) + (m_1 - m_2)^\top C^{-1}x - \frac{1}{2}m_1^\top C^{-1}m_1 + \frac{1}{2}m_2^\top C^{-1}m_2 \tag{4.7}$$

矩阵格式的正态类概率密度函数:

$$p(x \mid w_i) = \frac{1}{\sqrt{2\pi}\,|C_i|^{1/2}} \exp\left(-\frac{1}{2}(x-m_i)^\top C_i^{-1}(x-m_i)\right) \tag{4.8}$$

其中 m_i 为均值向量，C_i 为协方差矩阵:

$$\begin{aligned} m_i &= E_i\{x\} \\ C_i &= E_i\{(x-m_i)(x-m_i)^\top\} \end{aligned} \tag{4.9}$$

$|C_i|$ 表示 C_i 的行列式的值，矩阵的逆一般求法:

$$[A \mid I_n] \overset{\square\square\square}{\rightarrow} [U \mid B] \overset{\square\square\square}{\rightarrow} [I_n \mid A^{-1}] \tag{4.10}$$

4.1.2 朴素贝叶斯过程

首先估算概率与条件概率：

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},\ k=1,2,\cdots,K \tag{4.11}$$

离散条件：

$$P(X^{(j)}=a_{jl}\mid Y=c_k)=\frac{\sum_{i=1}^{N}I\left(x_i^{(j)}=a_{jl},y_i=c_k\right)}{\sum_{i=1}^{N}I(y_i=c_k)} \tag{4.12}$$

连续条件：

$$p(x\mid w_i)=\frac{1}{\sqrt{2\pi}\,|C_i|^{1/2}}\exp\left(-\frac{1}{2}(x-m_i)^{\top}C_i^{-1}(x-m_i)\right) \tag{4.13}$$

对于给定的实例 $x=(x^{(1)},x^{(2)},\cdots,x^{(n)})^{\top}$：

$$P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}\mid Y=c_k),\ k=1,2,\cdots,K \tag{4.14}$$

确定实例 x 的类别：

$$y=\arg\max_{c_k}P(Y=c_k)\prod_{j=1}^{n}P(X^{(j)}=x^{(j)}\mid Y=c_k) \tag{4.15}$$

也可以通过最小风险选择。

4.1.3 Laplace 平滑

$$p(x=j)=\frac{\sum_{i=1}^{N}I_{x^i=j}+1}{N+K},j=1,...,K \tag{4.16}$$

4.2 Fisher 线性判别

定理 4.1: Fisher 基本思想

发点：降低维数

考虑把 d 维空间的样本投影到一条直线上，形成一维空间。如何选择投影方向使得类别能够被分开?

假设一集合包含 N 个 d 维样本 $x^1,x^2,\cdots,x^N$，其中 N_1 个属于 w_1 类，记为 Γ_1，而 N_2 个属于 w_2 类，记为 Γ_2。对 x^n 做投影可得：

$$y_n=w^{\top}x^n,\quad n=1,2,\cdots,N \tag{4.17}$$

通常 $|w|$ 的值不重要，因为只是大小，而其导致的方向是最为重要的。

4.2.1 基本参数

各样本的均值:

$$m = \frac{1}{N_i} \sum_{x \in \Gamma_i} x \tag{4.18}$$

样本内的离散度矩阵 S_i 和总样本类内离散度矩阵 S_w:

$$S_i = \sum_{x \in \Gamma_i} (x - m_i)(x - m_i)^\top S_w = \sum_i S_i \tag{4.19}$$

其中 S_w 是半正定矩阵，当 $N > d$ 时是非奇异的。
样本类间离散度矩阵 S_b:

$$S_b = (m_1 - m_2)(m_1 - m_2)^\top \tag{4.20}$$

在一维空间:
各类样本的均值 $\tilde{m}_i$:

$$\tilde{m}_i = \frac{1}{N_i} \sum_{y \in \Gamma_i'} y \tag{4.21}$$

样本类内离散度 $\tilde{S}_i^2$ 和总样本类内离散度 $\tilde{S}_w$:

$$\tilde{S}_i^2 = \sum_{y \in \Gamma_i'} (y - \tilde{m}_i)^2 \tilde{S}_w = \sum_i \tilde{S}_i^2 \tag{4.22}$$

4.2.2 Fisher 准则函数

$$J_F(w) = \frac{(\tilde{m}_1 - \tilde{m}_2)^2}{\tilde{S}_1^2 + \tilde{S}_2^2} \tag{4.23}$$

而

$$\tilde{m}_i = w^\top m_i \tag{4.24}$$

$$\begin{aligned} \tilde{S}_i^2 &= \sum_{y \in \Gamma_i'} (y - \tilde{m}_i)^2 = \sum_{x \in \Gamma_i} \left(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{m}_i \right)^2 \\ &= \boldsymbol{w}^{\mathrm{T}} \left[\sum_{x \in \Gamma_i} (\boldsymbol{x} - \boldsymbol{m}_i) (\boldsymbol{x} - \boldsymbol{m}_i)^{\mathrm{T}} \right] \boldsymbol{w} \\ &= \boldsymbol{w}^{\mathrm{T}} \left[\sum_{x \in \Gamma_i} (\boldsymbol{x} - \boldsymbol{m}_i) (\boldsymbol{x} - \boldsymbol{m}_i)^{\mathrm{T}} \right] \boldsymbol{w} \\ &= \boldsymbol{w}^{\mathrm{T}} \boldsymbol{S}_i \boldsymbol{w} \end{aligned} \tag{4.25}$$

故:

$$J_F(w) = \frac{w^\top S_b w}{w^\top S_w w} \tag{4.26}$$

4.2.3 计算最佳变换向量 w^*

为了求使 $J_F(w)$ 取最大值的 w^*，可采用拉格朗日乘数法进行求解。

定义拉格朗日函数：$L(w,\lambda)=w^\top S_b w-\lambda(w^\top S_w w-c)$

然后对 w 求偏导：

$$\frac{\partial L(\boldsymbol{w},\lambda)}{\partial \boldsymbol{w}}=\boldsymbol{S}_b\boldsymbol{w}+\boldsymbol{S}_b^T\boldsymbol{w}-\lambda\left(\boldsymbol{S}_w\boldsymbol{w}+\boldsymbol{S}_w^T\boldsymbol{w}\right)=2\boldsymbol{S}_b\boldsymbol{w}-\lambda 2\boldsymbol{S}_w\boldsymbol{w} \tag{4.27}$$

令偏导数为 0：

$$S_b w^*=\lambda S_w w^* \tag{4.28}$$

由于 $S_b w^*=(m_1-m_2)(m_1-m_2)^\top w^*$，

而 $(m_1-m_2)^\top w^*$ 为一个标量，不影响 m_1-m_2 这个方向，用 R 表示。

故：

$$w^*=\frac{R}{\lambda}S_w^{-1}(m_1-m_2) \tag{4.29}$$

其中 $\dfrac{R}{\lambda}$ 取任何值都没有问题。多类问题的推导基本一致。

注： 矩阵运算常用的三个求导问题：

$$\frac{\partial Ax}{\partial x}=A^\top\frac{\partial x^\top x}{\partial x}=2x\frac{\partial x^\top Ax}{\partial x}=Ax+A^\top x \tag{4.30}$$

4.3 感知器算法

4.3.1 训练算法

算法 4.1: 感知器训练算法

两类问题分别属于 c_1 类和 c_2 类问题，初始权重为 $w(1)$，若 $x^k\in c_1$ 则 $w^\top(k)x^k>0$，否则若 $x^k\in c_2$ 则 $w^\top(k)x^k\le 0$。

第 k 步训练：

若 $x^k\in c_1$，而 $w^\top(k)x^k\le 0$，则对第 k 个模式 x^k 做惩罚，$w(k+1)=w(k)+Cx^k$，其中 C 为一个校正增量。

若 $x^k\in c_2$，而 $w^\top(k)x^k>0$，则对第 k 个模式 x^k 做惩罚，$w(k+1)=w(k)-Cx^k$，其中 C 为一个校正增量。

若分类正确，则 $w(k+1)=w(k)$

4.3.2 收敛性证明

假设模式类别是线性可分的，感知器算法可以在有限的迭代步骤里面求出权向量。

思路：第 $k+1$ 次迭代权重比第 k 次更接近解矢量，则收敛。由于线性可分，则存在 w^* 使 $(w^*)^\top x>0$，即证明 $\|w(k+1)-\alpha w^*\|^2\le\|w(k)-\alpha w^*\|^2$

证明 4.1: 感知器算法收敛性

由于 $w(k+1)-\alpha w^*=w(k)+x_k-\alpha w^*$

则：

$$\|w(k+1)-\alpha w^*\|^2=\|w(k)-\alpha w^*\|^2+2(w(k)-\alpha w^*)^\top x_k+\|x\|^2\leq\|w(k)-\alpha w^*\|^2-2\alpha(w^*)^\top x_k+\|x\|^2$$

设 $\beta^2=\max\limits_k\|x_k\|^2$ 而 $\gamma=\min\limits_k(w^*)^\top x_k>0$，则：

$$\|w(k+1)-\alpha w^*\|^2<\|w(k)-\alpha w^*\|^2-2\alpha\gamma+\beta^2$$

若 $\alpha=\beta^2/\gamma$ 则：

$$\|w(k+1)-\alpha w^*\|^2<\|w(k)-\alpha w^*\|^2-\beta^2$$

经过 k 步后，权系数：

$$\|w(k+1)-\alpha w^*\|^2<\|w(1)-\alpha w^*\|^2-k\beta^2$$

由于不能为负数，故经过不超过 $k_0=\dfrac{\|w(1)-\alpha w^*\|^2}{\beta^2}$ 步即终止。

4.3.3 多类训练算法

算法 4.2: 感知器多类训练算法

存在 M 类的判别函数 $\{d_i,i=1,2,...,M\}$，若 $x_k\in c_i$，则 $d_i>d_j,\forall j\neq i$。

设有 M 个模式类别 $c_1,c_2,...,c_M$，则在第 k 次迭代时，若一个属于 c_i 的模式样本 x，先计算 M 个模式的判别函数：

$d_j(k)=w_j(k)x,j=1,2,...M$

若第 j 个权向量使 $d_i(k)<d_j(k)$，则调整权向量：

$w_i(k+1)=w_i(k)+Cx$

$w_j(k+1)=w_j(k)-Cx$

其他权向量的数值保持不变。

如果分类正确也保持权向量不变 $w_j(k+1)=w_j(k)$。

4.4 K-L 变换

适用于任意概率密度函数的正交变换。

设有一连续的随机实函数 $x(t)$，其中 $T_1\leq t\leq T_2$，则 $x(t)$ 可用正交函数集 $\{\varphi_j(i),j=1,2,...\}$ 的线性组合展开：

$$\begin{aligned}x(t)&=a_1\varphi_1(t)+a_2\varphi_2(t)+...+a_j\varphi_j(t)+...\\&=\sum_{j=1}^{\infty}a_j\varphi_j(t),\quad T_1\leq t\leq T_2\end{aligned}$$

其中 a_j 为展开式的随机系数，$\varphi_j(t)$ 为一组连续的正交函数，满足：

$$\int_{T_1}^{T2} \varphi_n^{(t)} \tilde{\varphi}_m(t) \mathrm{d}t = \begin{cases} 1, & m = n \\ 0, & m \neq n \end{cases} \tag{4.31}$$

其中 $\tilde{\varphi}_m(t)$ 为 $\varphi_n^{(t)}$ 的共轭复数式。

写成离散形式的正交函数形式，使连续随机函数 $x(t)$ 和连续正交函数 $\varphi_j(t)$ 被等间隔采样为 n 个离散点，即：

$$x(t) \to \{x(1), x(2), ..., x(n)\} \varphi_j(t) \to \{\varphi_j(1), \varphi_j(2), ..., \varphi_j(n)\} \tag{4.32}$$

向量：

$$x = (x(1), x(2), ..., x(n))^\top \varphi_j = (\varphi_j(1), \varphi_j(2), ..., \varphi_j(n)),\ j = 1, 2, ..., n \tag{4.33}$$

写为离散展开式：

$$x = \sum_{j=1}^{n} a_j \varphi_j = \Phi a,\ T_1 \leq t \leq T_2 \tag{4.34}$$

其中，a 为展开式的随机系数的向量形式：

$$a = (a_1, a_2, ..., a_j, ..., a_n)^\top \tag{4.35}$$

Φ 为 $n \times n$ 的矩阵，即：

$$\Phi = (\varphi_1, \varphi_2, \cdots, \varphi_n) = \begin{bmatrix} \varphi_1(1) & \varphi_2(1) & \cdots & \varphi_n(1) \\ \varphi_1(2) & \varphi_2(2) & \cdots & \varphi_n(2) \\ \cdots & \cdots & \cdots & \cdots \\ \varphi_1(n) & \varphi_2(n) & \cdots & \varphi_n(n) \end{bmatrix} \tag{4.36}$$

Φ 将 x 变换为 a。

算法 4.3: K-L 展开式系数计算过程

1. 给定一系列随机向量（样本）x，首先保证 $E[x] = 0$，否则对其归一化，就是减均值。
2. 求随机向量的自相关矩阵：$R = E\{xx^\top\}$
3. 然后，求出矩阵 R 的特征值 λ_j 与特征向量 φ_j, $j = 1, 2, ..., n$，这样可以满足上述的正交函数形式，然后列出矩阵 $\Phi = (\varphi_1, \varphi_2, ..., \varphi_n)$
4. 计算变换后的展开式：$a = \Phi^\top x$

结论 4.1: K-L 变换小结

K-L 变换前需要将 $E[x] = 0$，因此应先将均值作为新坐标轴的原点，然后再采用协方差矩阵和自相关矩阵计算特征值。

采用 K-L 变换作为模式分类的特征提取时，需要特别注意保留不同类别的模式分类鉴别信息，仅单纯考虑尽可能代表原来模式的主成分，有时并不一定有利于分类的鉴别。

证明 4.2: K-L 特征选择问题

注：可以不选 n 个，也可以只选 m 个，$m<n$，选择特征值最大的前 m 个特征。

$$\Phi=(\varphi_1,\varphi_2,...,\varphi_m)$$

由于我们在选择了变换矩阵后，我们要使得降维后的向量在最小均方差条件下接近原来的向量 x，对于 $x=\sum_{j=1}^{n}a_j\varphi_j$，现只取 m 项，对于省略部分用常数 b 代替，则：

$$\hat{x}=\sum_{j=1}^{m}a_j\varphi_j+\sum_{j=m+1}^{n}b\varphi_j$$

误差：

$$\triangle x=x-\hat{x}=\sum_{j=m+1}^{n}(a_j-b)\varphi_j$$

均方误差：

$$E\{\|\triangle x\|\}^2=\sum_{j=m+1}^{n}E(a_j-b)^2$$

为了让误差最小，对应 b 应满足：

$$\frac{\partial}{\partial b}[E(a_j-b)^2]=\frac{\partial}{\partial b}[E(a_j^2-2a_jb+b^2)]=2[E(a_j)-b]=0$$

则应该使 $b=E[a_j]$，即对省略了的 a 的分量，此时应该满足：

$$E\{\|\triangle x\|\}^2=\sum_{j=m+1}^{n}E(a_j-E[a_j])^2=\sum_{j=m+1}^{n}[\varphi_j^{\top}(x-E[x])(x-E[x])^{\top}\varphi_j]=\sum_{j=m+1}^{n}\varphi_j^{\top}C_x\varphi_j$$

设 λ_j 为 C_x 的第 j 个特征值（因为我们计算特征值时候就是基于这个协方差矩阵或者自相关矩阵计算的），φ_j 为 λ_j 对应的特征向量，则：

$$C_x\varphi_j=\lambda_j\varphi_j$$

由于 $\varphi_j^{\top}\varphi_j=1$ 故 $\varphi_j^{\top}C_x\varphi_j=\lambda_j$

因此：

$$E\{\|\triangle x\|\}^2=\sum_{j=m+1}^{n}\varphi_j^{\top}C_x\varphi_j=\sum_{j=m+1}^{n}\lambda_j$$

故被遗弃的特征的特征值应越小越好。

4.5 逻辑回归 LR

最小二乘法 最优化问题：$\min_w J(w) = \sum_{i=1}^{N}(w^\top x^i - y^i)^2$

梯度下降：$\dfrac{\partial J(w)}{\partial w} = 2\sum_{i=1}^{N} x_j^i(w^\top x^i - y^i)$

更新规则：

批量梯度下降：$w_j = w_j - 2\alpha\sum_{i=1}^{N} x_j^i(w^\top x^i - y^i),\ \alpha > 0$

随机梯度下降：

$$w_j = w_j - 2\alpha x_j^i(w^\top x^i - y^i),\ \alpha > 0$$

4.5.1 判别式模型，二分类逻辑回归

估计后验概率 $p(y \mid x)$

$$P(y=1 \mid x) = f(x,w) = \text{Sigmoid}(w^\top x) = \frac{1}{1+\exp -w^\top x} \tag{4.37}$$

概率分布（伯努利分布）：

$$P(y \mid x,w) = (f(x,w))^y(1-f(x,w))^{1-y} \tag{4.38}$$

似然：

$$L(w) = \prod_{i=1}^{N} P(y^i \mid x^i,w) = \prod_{i=1}^{N}(f(x^i,w))^{y^i}(1-f(x^i,w))^{1-y^i} \tag{4.39}$$

最大化 log 似然：

$$l(w) = \log L(w) = \sum_{i=1}^{N}(y^i \log f(x^i,w) + (1-y^i)\log(1-f(x^i,w))) \tag{4.40}$$

梯度：

$$\frac{\partial l(w)}{\partial w_j} = (y^i - f(x^i,w))x_j^i \tag{4.41}$$

SGD：

$$w_j = w_j + \alpha(y^i - f(x^i,w))x_j^i \tag{4.42}$$

注： sigmoid 函数 $f(x)$ 求导：$f(x)(1-f(x))$，log 函数求导：$\dfrac{d\log_a x}{dx} = \dfrac{1}{x\ln a}$。

4.5.2 多分类逻辑回归

这里用 softmax 代替 sigmoid

$$P(C_k \mid x, w) = \frac{\exp\left(w_k^\top x\right)}{\sum_{j=1}^{K} \exp\left(w_k^\top x\right)} \tag{4.43}$$

概率分布：

$$\mu_i = P(y_i = 1 \mid w, x) P(y, \mu) = \prod_{i=1}^{K} \mu_i^{y_i} \tag{4.44}$$

则：

$$P(y^1, y^2, ..., y^N \mid w_1, w_2, ..w_k, x^1, x^2, ..., x^N) = \prod_{i=1}^{N} \prod_{j=1}^{K} P(C_k \mid w_k, x^i) \tag{4.45}$$

优化（交叉熵损失函数）：

$$\min E = -\ln P(y^1, y^2, ..., y^N \mid w_1, w_2, ..w_k, x^1, x^2, ..., x^N) \min - \sum_{i=1}^{N} \sum_{k=1}^{K} y_k^i \ln \mu_{ik} \tag{4.46}$$

梯度：

$$\nabla_{w_j} E = \sum_{i=1}^{N} \left(\sum_{k=1}^{K} y_k^i \frac{\mu_{ij}^{'}}{\mu_{ij}} \right) = \sum_{i=1}^{N} (\mu_{ij} - y_j^i) x^i \tag{4.47}$$

4.5.3 LR（逻辑回归）与 NB（朴素贝叶斯）

当模型假设正确，NB 和 LR 产生相似的分类器。
假设不争取时候，LR 偏差较小，预期 LR 优于 NB。
NB 收敛速度一般比 LR 快。

4.6 MLE 与 MAP

4.6.1 MLP

假设给定函数，输入给了高斯噪声 ε，则 $y = f(x, w) + \varepsilon$，均值是 0，方差为 β^{-1} 的高斯噪声。
似然函数：
$\prod_{i=1}^{N} N(y^i \mid w^\top x^i, \beta^{-1})$ 满足正太分布。
其对数似然：

$$\begin{aligned}\sum_{i=1}^{N}\ln N(y^i \mid w^\top x^i,\beta^{-1}) &= \sum_{i=1}^{N}\ln\left[\frac{1}{\sqrt{2\pi}\sqrt{\beta^{-1}}}\exp\left(-\frac{(f(x^i,w)-y^i)^2}{2\beta^{-1}}\right)\right]\\ &= \sum_{i=1}^{N}\left[\frac{1}{2}\ln\beta-\frac{1}{2}\ln 2\pi-\frac{1}{2}\beta(f(x^i,w)-y^i)^2\right]\\ &= \frac{N}{2}\ln\beta-\frac{N}{2}\ln 2\pi-\frac{1}{2}\beta\sum_{i=1}^{N}(f(x^i,w)-y^i)^2\\ &= \frac{N}{2}\ln\beta-\frac{N}{2}\ln 2\pi-\frac{1}{2}\beta J(w)\end{aligned} \tag{4.48}$$

4.6.2 MAP

贝叶斯：$p(w \mid y)=p(y \mid w)p(w)/p(y)$

似然：$p(y \mid X,w,\beta)=\prod_{i=1}^{N}N(y^i \mid w^\top x^i,\beta^{-1})$

先验：$p(w)=N(0,\lambda^{-1}I)$

后验概率：

$$\ln(p(w \mid y))=-\beta\sum_{i=1}^{N}(y^i-w^\top x^i)^2-\lambda w^\top w+constant \tag{4.49}$$

最大化后验概率，等同于最小化带有正则项的平方和误差：

$$\min_w\sum_{i=1}^{N}(w^\top x^i-y^i)^2+\alpha w^\top w,\ \alpha=\frac{\lambda}{\beta} \tag{4.50}$$

结论 4.2: MLP 与 MAP 总结

1. MLE：$\hat{\theta}_{MLE}=\arg\max_\theta P(D \mid \theta)$
2. MAP：$\hat{\theta}_{MAP}=\arg\max_\theta P(\theta \mid D)=\arg\max_\theta P(D \mid \theta)P(\theta)$
   就是最大似然估计其实不需要知道先验 $P(\theta)$，但是最大后验估计是需要知道先验 $P(\theta)$

4.7 高斯判别分析

伯努利分布：$x\in\{0,1\}$ 的分布由连续参数 $\beta\in[0,1]$ 控制：

$$P(x \mid \beta)=\beta^x(1-\beta)^{1-x} \tag{4.51}$$

而给出 N 个服从伯努利分布的样本中，观察到 m 次 x=1 的概率：

$$P(m \mid N,\beta)=C_N^m\cdot\beta^m(1-\beta)^{N-m} \tag{4.52}$$

多项式分布，即取 K 个状态，第 k 个状态被观测到 m_k 次的概率：

$$P(m_1,...,m_K \mid N,\beta)=\binom{N}{m_1,...,m_K}\prod_{k=1}^{K}\beta_k^{m_k} \tag{4.53}$$

GDA 多变量正态分布建模:

$$y \sim Bernoulli(\beta); x \mid y=0 \sim N(\mu_0, \Sigma); x \mid y=1 \sim N(\mu_1, \Sigma) \tag{4.54}$$

Log 似然:

$$L(\beta, \mu_0, \mu_1, \Sigma) = \log \prod_{i=1}^{N} p(x^i, y^i; \beta, \mu_0, \mu_1, \Sigma) = \log \prod_{i=1}^{N} p(y^i \mid \beta) p(x^i \mid y^i; \beta, \mu_0, \mu_1, \Sigma) \tag{4.55}$$

MLE:

$$\beta = \frac{1}{N} \sum_{i=1}^{N} I_{y^i=1}; \mu_k = \frac{\sum_{i=1}^{N} I_{y^i=k} x_i}{\sum_{i=1}^{N} I_{y^i=k}}, k = \{0,1\}; \Sigma = \frac{1}{N} \sum_{i=1}^{N} (x^i - \mu_{y^i})(x^i - \mu_{y^i})^\top \tag{4.56}$$

结论 4.3: 高斯判别分析小结

1. GDA 有很强的模型假设，当假设正确时，处理数据的效率更高
2. LR 假设很弱，因此对偏离假设时具有鲁棒性
3. 实际中 LR 更常用

4.8 SVM

函数间隔: 给定一个样本 (x_i, y_i)，它到 (w, b) 确定的超平面的函数间隔为:

$$\hat{\gamma}_i = y_i(w^\top x_i + b) \tag{4.57}$$

则定义超平面 (w, b) 的所有样本点 (x_i, y_i) 的函数间隔最小值:

$$\hat{\gamma} = \min_{i=1,\dots,N} \hat{\gamma}_i \tag{4.58}$$

几何间隔: 为了使函数间隔具有不变性

$$\gamma_i = y_i \left(\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|} \right) \tag{4.59}$$

样本中的几何间隔最小值:

$$\gamma = \min_{i=1,\dots,N} \gamma_i \tag{4.60}$$

几何间隔与函数间隔的关系:

$$\gamma = \frac{\hat{\gamma}}{\|w\|} \tag{4.61}$$

间隔最大化：

$$\max_{\gamma} \frac{\gamma}{\|w\|} \tag{4.62}$$

约束条件：

$$\gamma(w^\top x + b) \geq \gamma \tag{4.63}$$

硬间隔中，令 $\gamma = 1$
优化：$\max_{\gamma} \frac{\gamma}{\|w\|} = \min_{w} \frac{1}{2}\|w\|^2$
约束：$y(w^\top x + b) \geq 1$

对偶算法：

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (w^\top x_i + b) + \sum_{i=1}^{N} \alpha_i \tag{4.64}$$

本质是在优化：

$$\max_{\alpha} \min_{w,b} L(w, b, \alpha) \tag{4.65}$$

先对 w, b 取极小：

$$\min_{w,b} L(w, b, \alpha) \nabla_w L = w - \sum_{i=1}^{N} \alpha_i y_i x_i \nabla_b L = -\sum_{i=1}^{N} \alpha_i y_i \tag{4.66}$$

设最优为 w^*：

$$w^* = \sum_{i=1}^{N} \alpha_i y_i x_i \sum_{i=1}^{N} \alpha_i y_i = 0 \alpha_i \geq 0 \tag{4.67}$$

此时：

$$\begin{aligned} L(w, b, \alpha) &= \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j - \sum_{i=1}^{N} \alpha_i y_i \left(\sum_{j=1}^{N} \alpha_i y_i x_i + b \right) + \sum_{i=1}^{N} \alpha_i \\ &= -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j + \sum_{i=1}^{N} \alpha_i \end{aligned} \tag{4.68}$$

即优化：

$$\begin{aligned} &\max_{\alpha} -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j + \sum_{i=1}^{N} \alpha_i \\ \longrightarrow &\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j - \sum_{i=1}^{N} \alpha_i \\ &s.t. \sum_{i=1}^{N} \alpha_i = 0,\ \alpha_i \geq 0 \end{aligned} \tag{4.69}$$

之后围绕着这个方程求解即可，可能比较麻烦，需要启发式计算。

4.9 K-Means 聚类

4.9.1 算法流程

算法 4.4: K-means 算法

问题： 给定 N 个样本点 $X = \{x_i\}_{i=1}^N$ 进行聚类

输入： 数据 $\mathcal{D} = \{x_1, x_2, ..., x_N\}$，聚类簇数目为 K。

随机选择 K 个种子数据点作为 K 个簇的中心

repeat

for each $x_i \in \mathcal{D}$ do

计算 x_i 与每一个簇中心的距离 $\text{dist}(x_i, \mu_{k'})$

将 x_i 指配到距离簇中心最近的簇中心：$z_i = \arg\min_{k'} \text{dist}(x_i, \mu_{k'})$

end for

用当前的簇内点，重新计算 K 个簇中心的位置

until 当前簇中心未更新

备注：

欧式距离作为距离度量

每个节点都划分到距离最近的那个簇中心，用 $r_{j,k} \in \{0, 1\}$ 表示。

目标函数：

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} r_{i,k} \|x_i - \mu_k\|^2 \tag{4.70}$$

优化： 固定 μ_k，优化 $r_{i,k}$

固定 $r_{i,k}$，优化 μ_k，就是求为 $r_{i,k}$ 所有样本点的均值

4.9.2 K 的选择问题

寻找一个 K，使 $J_K - J_{K+1}$ 的改进很小。

$$K_\alpha = \min \left\{ k : \frac{J_k - J_{k+1}}{\sigma^2} \leq \alpha \right\} \tag{4.71}$$

其中 $\sigma^2 = \mathbb{E}[\|X - \mu\|^2]$，$\mu = \mathbb{E}[X]$

4.9.3 其他聚类方法（GMM，层次聚类，DBSCAN）

GMM 假设 K 个簇，每个簇服从高斯分布，以概率 π_k 随机选择一个簇，然后观测数据。

概率密度函数：

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k) \tag{4.72}$$

π_k 为混合比例，$\mathcal{N}(x \mid \mu_k, \Sigma_k)$ 为混合的成分。

参数估计：极大似然估计 +EM 算法，E 步求期望，M 步重新估计参数

与 K-means 异同：Kmeans 采用虽小平方距离和损失函数，而 GMM 是最小化负对数的似然函数。Kmeans 是硬划分到簇，而 GMM 是软划分。Kmeans 样本属于每个簇概率相同，球形簇，而 GMM 簇概率不同，可以被用于椭球形簇。

层次聚类 不需要提前假定簇的数目，选择树状图某一层就可以获取任意簇数量的聚类结构。

自底向上：递归合并相似度最高/距离最近的两个簇

自顶向下：递归分裂不一致的簇，例如拥有最大直径的簇

DBSCAN，基于密度的聚类 基于密度的聚类，将临近的密度高的区域练成一片形成簇

优点：各种大小或形状的簇，且具有一定的抗噪声能力

优势：不需要簇的数目，可以对任意形状簇进行聚类，对离群点比较鲁棒

劣势：参数选择困难，不适合密度差异大的数据集，计算复杂度比较大。

4.10 降维

矩阵的秩：最大线性无关行或列的数目

矩阵的迹：方阵 A 的 $tr(A)$ 是对角线元素之和

奇异值分解 SVD:

输入 $A_{M\times N}$ 矩阵，SVD：$A = U\Sigma V^\top$

其中 $\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_R \end{bmatrix}$ 奇异值，$R = \min(M, N)$

u_k、$v_k^\top$：奇异值 σ_k 对应的奇异向量

$U^\top U = I, V^\top V = I$：$U$ 和 V 是正交矩阵

4.10.1 PCA

目标：最小平方重建误差和：

$$\min_{W\in\mathbb{R}^{D\times D'}} \sum_{i=1}^{N} \|x_i - Wz_i\|^2 \tag{4.73}$$

目标函数 1 推导：

$$\begin{aligned} \sum_{i=1}^{N} \|x_i - Wz_i\|^2 &= \sum_{i=1}^{N} \left\| \sum_{j=1}^{D'} z_{i,j} w_j - x_i \right\|_2^2 \\ &= \sum_{i=1}^{N} z_i^\top z_i - 2\sum_{i=1}^{N} z_i^\top W^\top x_i + constant \\ \because z_i = W^\top x_i, \therefore \sum_{i=1}^{N} \|x_i - Wz_i\|^2 &\propto -\text{tr}(W^\top (\sum_{i=1}^{N} x_i x_i^\top) W) = -\text{tr}(W^\top X X^\top W) \\ &= -\text{tr}(ZZ^\top) = -\sum_{i=1}^{N} z_i^\top z_i \end{aligned} \tag{4.74}$$

即寻找一个方向向量，使数据投影到其上方后方差最大。
最大化样本的投影误差：

$$\max \sum_i z_i^\top z_i = \max_W \text{tr}(W^\top X X^\top W) = \max_W \text{tr} W^\top S W \tag{4.75}$$

寻找 $W \in \mathbb{R}^{D \times D'}$ 使上述投影误差最大，$W^\top W = I$
由于带约束的优化，可以用拉格朗日乘子：
针对第一个分量 w_1：

$$\begin{aligned} L &= w_1^\top S w_1 - \lambda_1 (w_1^\top w_1 - 1) \\ \frac{\partial L}{\partial w_1} &= 2 S w_1 - 2\lambda_1 w_1 = 0, \\ S w_1 &= \lambda_1 w_1 \\ J &= w_1^\top S w_1 = w_1^\top \lambda_1 w_1 = \lambda_1 \end{aligned} \tag{4.76}$$

而 λ_1 是最大特征值。
接着计算第二个方向投影，约束为：$w_2^\top w_2 = 1, w_2^\top w_1 = 0$
拉格朗日函数：$L = w_1^\top S w_1 + w_2^\top S w_2 - \lambda_2 (w_2^\top w_2 - 1) - \lambda_{2,1} w_2^\top w_1$
$\frac{\partial L}{\partial w_2} = 2 S w_2 - 2\lambda_2 w_2 - \lambda_{2,1} w_1 = 0$
同时乘 $w_1^\top$：
$2 w_1^\top S w_2 - 2\lambda_2 w_1^\top w_2 - \lambda_{2,1} w_1^\top w_1 = 0$，由于 $w_1^\top w_1 = 1$ 所以 $\lambda_{2,1} = 0$
$S w_2 = \lambda_2 w_2$ 且 λ_2 为第二大特征值
此时 $L = \lambda_1 + \lambda_2$
然后以此类推。

算法 4.5: PCA 算法

输入： $X = (x_1, x_2, ..., x_N)$

过程： $x_i = x_i - \frac{1}{N} \sum_{j=1}^{N} x_j$

计算 $S = X X^\top$

对 S 做特征分解

D' 最大特征值对应的特征向量 $w_1, w_2, ..., w_{D'}$

输出： $W = (w_1, w_2, .., w_{D'})$

4.10.2 流形学习：结构保持/距离保持

度量型 MDS 的算法 测地距离：

- 邻近的点：输入空间的欧氏距离提供一个测地线距离的近似。
- 较远的点：测点距离能够通过一系列邻域点之间的欧式距离的累加近似。

算法流程如下

算法 4.6: 度量型 MDS 算法

将 n 个 D 维原始数据降维到 D'

1. 计算样本的距离矩阵 $D=[d_{i,j}]=[dist(x_i,x_j)]$，其中 $D\times\mathbb{R}^{N\times N}$

2. 中心矫正方法构造矩阵 $B=JDJ, J=I-\frac{1}{n}O$，其中 I 为 $n\times n$ 的单位阵，O 为 $n\times n$ 的值全为 1 的矩阵。

3. 计算矩阵 B 的特征向量 $e_1,e_2,...,e_m$ 以及对应的特征值 $\lambda_1,\lambda_2,...,\lambda_m$

4. 确定维度 D' 重构数据 $Z=E\Lambda^{1/2}$，$Z\in\mathbb{R}^{D'\times N}$，其中 E 为 D' 个最大特征值组成的特征向量，而 Λ 为 D' 个特征值构成的对角矩阵。

全局距离保持 ISOMAP 算法

- 构建邻接图
- 计算最短路径：$\min(d_G(i,j),d_G(i,k)+d_G(k,j))$，通常用 Floyd 算法（复杂度 $O(N^3)$）和 Dijkstra 算法（复杂度 $O(KN^2\log N)$）。
- 构建低维嵌入：保持点对之间的测地距离，用 MDS 方法。

局部距离保持 拉普拉斯特征映射 (Laplacian Eigenmaps)：

- 令 $x_1,x_2,...,x_N\in\mathbb{R}^D$
- 构造相似性图，表示节点之间的邻接关系：ε 邻域，K 邻域
- 通过对图的拉普拉斯矩阵进行特征值分解，得到映射。

局部优先，兼顾全局 T-SNE：

SNE 将欧氏距离转换为用概率表示的相似度，原始空间中的相似度由高斯联合概率表示，嵌入空间的相似度由“学生 T 分布”表示。

T-SNE 的目标是对所有数据点对的概率 p_{ij} 与 q_{ij}，最小化 KL 散度 $C=KL(p\|q)=\sum_i\sum_j p_{i,j}\log\frac{p_{i,j}}{q_{i,j}}$。$p_{i|j}=\frac{\exp(-\|x_j-x_i\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|x_k-x_i\|^2/2\sigma_i^2)}$，$p_{i,j}=\frac{p_{i|j}+p_{j|i}}{2N}$，其中 σ_i 参数是以数据点 x_i 为中心的高斯分布的标准差，但是实际很难评估，故一般用困惑度表示，原始论文困惑度建议设为 5-50，为固定数。

优化可用随机梯度下降法。

为什么用 T 分布？T 分布与正太分布很类似，中间比正态分布低，但是两侧比正态分布高。当高维空间中距离较小的点对，概率一致时候，T 分布的在低维空间距离更小，相似样本更紧致，距离较大的点对距离更大，不相似样本更远。

T-SNE 缺点：计算复杂，$O(N^2)$；全局结构没有明确保留；主要用于可视化。

UMAP 步骤：学习高维空间中的流形结构，找到该流形的低维表示。

算法 4.7: UMAP

1. 寻找最近邻，例如 KNN 算法
2. 根据 K 近邻，构建相似图（变化距离，局部连接，概率 & 边的权重 $p_{j|i}$，合并边的权重 $p_{i,j}$）
3. 优化，目标函数为交叉熵：

$$J = \sum_{e_{i,j} \in \varepsilon} p_{i,j} \log \frac{p_{i,j}}{q_{i,j}} + (1 - p_{i,j}) \log \frac{(1 - p_{i,j})}{(1 - q_{i,j})} \tag{4.77}$$

优点：不仅可以可视化，还可以降维；更快，用随机梯度下降法；更好全局结构

4.11 集成学习

无免费午餐定理： 模型的选取要以问题的特点为根据。

丑小鸭定理： 世界上不存在分类的客观标准，一切分类的标准都是主观的。丑小鸭与白天鹅之间的区别和两只白天鹅之间的区别一样大。

奥卡姆剃刀： 在性能相同的情况下，应该选取更加简单的模型。

过于简单的模型会导致欠拟合，过于复杂的模型会导致过拟合。

从误差分解的角度看，欠拟合模型的偏差较大，过拟合模型的方差较大。

4.11.1 Bagging 原理及降低 Variance 推导

对给定的 N 个样本的数据集 $\mathcal{D}$ 进行 Bootstrap 采样，得到 $\mathcal{D}^1$，在 $\mathcal{D}^1$ 上训练模型 f_1

上述过程重复 M 次，得到 M 个模型，则 M 个模型的平均回归/投票为：

$$f_{avg}(x) = \frac{1}{M} \sum_{m=1}^{M} f_m(x) \tag{4.78}$$

bootstrap 样本：通过对原始的 N 个样本数据 $\mathcal{D} = \{x_1, ..., x_N\}$ 进行 N 次有放回的采样 N 个数据 $\mathcal{D}'$，则称为一个 bootstrap 样本。

对原始数据进行有放回的随机采样，抽取的样本数目同原始样本数目一样。

一个样本不在采样集中概率：$(1 - \frac{1}{N})^N$，约有 63.2

为什么 Bagging 可降低模型的方差： 令随机变量 X 的均值为 μ，方差为 σ^2

则 N 个独立同分布的样本的样本均值 $\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i$

则样本均值 $\bar{X}$ 的期望为：$\mathbb{E}(\bar{X}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}(X_i) = \mu$，故 $\bar{X}$ 是 X 的无偏估计。

而样本均值 $\bar{X}$ 的方差为：$Var(\bar{X}) = \frac{1}{N^2} \sum_{i=1}^{N} Var(X_i) = \frac{\sigma^2}{N}$

4.11.2 Boosting 推导

算法 4.8: Boosting 算法描述

给定训练集：$(x_1, y_1), ..., (x_n, y_n)$，其中 $y_i \in \{-1, 1\}$ 表示 x_i 的类别标签

对 $m = 1 : M$：

对训练样本采用权重 $w_{m,i}$，计算弱分类器 $\phi_m(x)$

计算该弱分类器 $\phi_m(x)$ 在分布 w_m 上的误差：$\varepsilon_m = \sum_{i=1}^{N} w_{m,i} \mathbb{I}(\phi_m(x_i) \neq y_i)$

计算 $d_m = \sqrt{(1-\varepsilon_m)/\varepsilon_m}, \alpha = \log d_m = \frac{1}{2} \log \frac{1-\varepsilon_m}{\varepsilon_m}$

更新训练样本的分布：$w_{m+1,i} = \begin{cases} \frac{w_{m,i}/d_m}{z_m} & y_i \phi_m(x_i) = 1 \\ \frac{w_{m,i} d_m}{z_m} & y_i \phi_m(x_i) = -1 \end{cases}$

$= \frac{w_{m,i} d_m^{-y_i \phi_m(x_i)}}{Z_m} = \frac{w_{m,i} \exp(-\alpha_m y_i \phi_m(x_i))}{Z_m}$，其中 Z_m 为归一化常数，使 w_{m+1} 是个分布。

强分类器：$f(x) = sgn(\sum_{m=1}^{M} \alpha_m \phi_m(x))$

证明 4.3: 证明 Boosting 可减少偏差

$$\because w_{M+1,i} = w_{M,i} \frac{\exp(-\alpha_M y_i \phi_M(x_i))}{Z_M} = w_{1,i} \frac{\exp(-y_i \sum_{m=1}^{M} \alpha_m \phi_m(x)))}{\prod_{m=1}^{M} Z_m} \tag{4.79}$$

其中 $\sum_{i=1}^{N} w_{M+1,i} = 1$，所以 $\prod_{m=1}^{M} Z_m = \sum_{i=1}^{N} w_{1,i} \exp(-y_i f(x_i)) = \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i f(x_i))$。

训练误差：

$$ERR_{train}(f(x)) = \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i f(x_i)) \tag{4.80}$$

而：

$$\begin{aligned} Z_m = \sum_{i=1}^{N} w_{m,i} &= \sum_{i=1}^{N} w_{m,i} d_m \mathbb{I}(y_i \neq \phi_m(x_i)) + \sum_{i=1}^{N} w_{m,i}/d_m \mathbb{I}(y_i = \phi_m(x_i)) \\ &= d_m \varepsilon_m + 1/d_m (1-\varepsilon_m) \end{aligned} \tag{4.81}$$

$$\because d_m = \sqrt{(1-\varepsilon_m)/\varepsilon_m}$$

$$\therefore = 2\sqrt{(1-\varepsilon_m)\varepsilon_m}$$

训练误差：

$$ERR_{train}(f(x)) = \frac{1}{N} \sum_{i=1}^{N} \exp(-y_i f(x_i)) = \prod_{m=1}^{M} Z_m = 2^M \prod_{m=1}^{M} \sqrt{(1-\varepsilon_m)\varepsilon_m} \tag{4.82}$$

由于 $0 < \varepsilon_m < 0.5, 0 < 1 - \varepsilon_m < 1$，所以误差 ERR_{train} 越来越小。

Boosting 是弱学习器按顺序学习的，第 $n-1$ 个学习器 $\phi_{n-1}(x)$ 要能帮助第 n 个学习器 $\phi_n(x)$，然后组合所有的

弱学习器 $f(x)=\sum_{m=1}^{M}\alpha_m\phi_m(x)$。

分对的样本减少权重，分错的样本增加权重。

4.12 半监督学习

定理 4.2: 基本假设

聚类假设：如果两个点都在同一个簇，那么很可能属于同一个类别。

低密度分割：决策边界应该在低密度区域

平滑假设：如果高密度区域有两个点 x_1,x_2 距离较近，那么对应的输出 y_1□y_2 也应该比较近

流形假设：高维数据大致会分布在一个低维的流形上，流形上临近的样本拥有相似的输出（平滑性假设相似）

自学习算法

- 假设：输出高置信度的预测是正确的
- 缺点：早期的错误会被强化，收敛方面没有保障。

多视角学习（协同学习） 训练两个分类器 f_1,f_2，他们针对的特征不同（特征分割），把 f_1 的 k 个最高置信结果给 f_2，f_2 的 k 个最高置信结果给 f_1 做标签。

优点：对于错误没有那么敏感，且可以用到已有的各种分类器

缺点：自然的特征分裂不存在，使用全部特征的模型可能效果更好。

本质是搜索最好的分类器

生成式半监督模型,GMM GMM 使用 MLE 计算参数 θ(频率,样本均值,协方差):$\ln p(X_L,y_L\mid\theta)=\sum_{i=1}^{L}\ln\left(p(y_i\mid\theta)p(x_i\mid y_i,\theta)\right)$

MLE 通常计算困难，因此可以用 EM 算法寻找局部最优解。

核心是最大化 $p(X_L,y_L,X_U\mid\theta)$

优点：清晰；如果模型接近真实分布会比较有效

缺点：验证模型比较困难；EM 局部最优；如果生成式错误的，无监督数据会加重错误。

S^3VMs，即 TSVMs 基本假设：来自不同类别的无标记数据之间会被较大间隔隔开

基本思想：

- 遍历所有 2^U 种可能性的标注 X_U
- 为每一种标注构建一个标准的 SVM，包括 X_L
- 选择间隔最大的 SVM

5 图像处理与计算机视觉

5.1 目标跟踪

5.1.1 运动目标的表示方法

传统方法中基于运动目标的表示方法包括基于点、基于区域、基于轮廓和基于模型的跟踪，他们方法依次由简到难。

5.1.2 传统目标跟踪方法

传统目标跟踪的两种处理思路包括

- 自底向上 (Bottom-up) 的处理方法，依靠数据驱动的方法，不依赖先验知识。
- 自顶向下 (Top-down) 的处理方法，依靠模型驱动，依赖于所构建的模型或者先验知识。

模板匹配法 (Template Matching) 在前一帧图像中的目标位置（或模板 T 的位置）为：(x, y)，并在当前帧搜索位置 $(x', y') = (x + dx, y + dy)$，使得

$$\arg\max_{dx,dy} \operatorname{cov}\left(T(x, y), I(x + dx, y + dy)\right)$$

该方法在概念上相对比较简单，但是进行穷尽的搜索计算量非常大。

基于卡尔曼滤波器的跟踪方法 首先介绍卡尔曼滤波相关的理论

定义 5.1: 卡尔曼滤波

卡尔曼滤波（Kalman filter）是一种高效率的递归滤波器（自回归滤波器），它能够从一系列的不完全及包含噪声的测量中，估计动态系统的状态。卡尔曼滤波会根据各测量量在不同时间下的值，考虑各时间下的联合分布，再产生对未知变数的估计，因此会比只以单一测量量为基础的估计方式要准。

6 Reinforcement Learning

定义 6.1: 强化学习

强化学习（英语：Reinforcement learning，简称 RL）是机器学习中的一个领域，强调如何基于环境而行动，以取得最大化的预期利益 [1]。

强化学习是一种优化智能体在环境中行为的一种方法。根据环境反馈的**奖励**，调整智能体的行为策略，提升智能体实现**目标**的能力。

6.1 马尔可夫决策过程

定义 6.2: 马尔可夫性

在给定现在状态以及所有过去状态下，智能体未来状态的条件概率分布，**仅依赖于当前状态**。即给定现在状态时，未来状态与过去状态是**条件独立**的。

$$\mathbb{P}[s_{t+1} \mid s_1, \cdots, s_t] = \mathbb{P}[s_{t+1} \mid s_t]$$

一个马尔可夫过程可以用一组 $\langle \mathcal{S}, \mathcal{P} \rangle$ 表示，其中 $\mathcal{S}$ 是有限的状态集，而 $\mathcal{P}$ 是状态转移概率矩阵 $\mathcal{P}_{ss'} = \mathbb{P}[s_{t+1} = s' \mid s_t = s]$。而一个马尔可夫奖励过程由一组 $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ 构成。$\mathcal{R}$ 是奖励函数，$\mathcal{R}_s = \mathcal{R}(s) = \mathbb{E}[r_{t+1} \mid s_t = s]$，$\gamma$ 是折扣因子，$\gamma \in [0, 1]$。

回报 G_t 代表在 t 时刻后轨迹的累加奖励，对应某一具体轨迹

$$G_t = r_{t+1} + \gamma r_{t+2} + \cdots = \sum_{k=1}^{\infty} \gamma^k r_{t+k+1} \tag{6.1}$$

价值，马尔可夫奖励过程的价值 V 等于从状态 s 出发的期望回报，价值代表所有轨迹的期望

$$V(s) = \mathbb{E}[G_t \mid s_t = s] \tag{6.2}$$

定义 6.3: 马尔可夫决策过程

一个马尔科夫决策过程由 $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ 组成

- $\mathcal{S}$ 是有限的状态集
- $\mathcal{A}$ 是有限的动作集
- $\mathcal{P}$ 是状态转移概率矩阵

$$\mathcal{P}^a_{ss'} = \mathbb{P}[s_{t+1} = s' \mid s_t = s, a_t = a]$$

- $\mathcal{R}$ 是奖励函数，$\mathcal{R}^a_s = \mathbb{E}[r_{t+1} \mid s_t = s, a_t = a]$
- γ 是折扣因子 $\gamma \in [0, 1]$

策略 π 是从状态到动作的一种分布

$$\pi(a \mid s) = \mathbb{P}[a_t = a \mid s_t = s] \tag{6.3}$$

状态价值 $V_\pi(s)$ 定义为从状态 s 出发，在策略 π 作用下的期望回报

$$V_\pi(s) = \mathbb{E}_\pi[G_t \mid s_t = s] \tag{6.4}$$

动作价值 $Q_\pi(s,a)$ 是智能体从状态 s 出发，首先执行动作 a，然后按照策略 π 的期望回报

$$Q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid s_t = s, a_t = a] \tag{6.5}$$

$$V_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a \mid s) Q_\pi(s,a) \tag{6.6}$$

$$Q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a V_\pi(s') \tag{6.7}$$

定义 6.4: 贝尔曼期望方程

状态价值的贝尔曼期望方程

$$V_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a \mid s) \left(\mathcal{R}_s^a + \gamma \sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a V_\pi(s') \right) \tag{6.8}$$

其中 $\mathcal{R}_s^a$ 为在该状态 s 下执行动作 a 的奖励，$\sum_{s'\in\mathcal{S}} \mathcal{P}_{ss'}^a V_\pi(s')$ 为后续期望回报。

动作价值的贝尔曼期望方程

$$Q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s'\in S} \mathcal{P}_{ss'}^a V_\pi(s') \tag{6.9}$$

(a) 状态价值 $V_\pi(s)$ (b) 动作价值 $Q_\pi(s,a)$

图 6.1: 状态价值与动作价值的轨迹，其中 s 为当前时刻状态，a 为当前时刻动作，s' 与 a' 分别为下一时刻状态与动作。

6.2 无模型控制学习

预测问题与控制问题区别：

预测问题计算 MDP 某一策略的价值函数。而控制问题计算 MDP 的最优策略/最优价值函数。

6.2.1 GLIE 蒙特卡洛控制学习

算法 6.1: GLIE 蒙特卡洛控制学习

初始化 $Q(s,a)=0, N(s,a)=0, \epsilon=1, k=1$

$\pi_k = \epsilon$-greedy(Q)

loop

使用策略 π_k 采集第 k 个轨迹 $\{s_0, a_0, r_1, s_1, \cdots, s_T\}$

for $t=0,\cdots,T$ **do**

if (s_t, a_t) 是 k 次轨迹上首次访问 **then**

$$G_t = r_{t+1} + \gamma r_{t+2} + \cdots$$

$$N(s_t,a_t) = N(s_t,a_t) + 1$$

$$Q(s_t,a_t) = Q(s_t,a_t) + \frac{1}{N(s_t,a_t)}\left(G_t - Q(s_t,a_t)\right)$$

end if

end for

$k = k+1, \epsilon = 1/k$

$\pi_k = \epsilon$-greedy(Q)

end loop

GLIE 蒙特卡洛控制是收敛到最优动作-价值函数，$Q(s,a) \to Q_*(s,a)$

6.2.2 Sarsa

Q 函数的贝尔曼期望方程

$$Q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a' \mid s') Q_\pi(s', a')$$

其中 $\mathcal{R}_s^a$ 和 $\mathcal{P}_{ss'}^a$ 是未知的，需要利用**样本**代表奖励 $\mathcal{R}$ 和模型 $\mathcal{P}$ 函数。

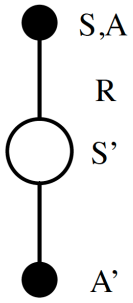


图 6.2: 样本轨迹示意图

如图6.2所示，智能体在线学习时每个时刻都能观察到的一段轨迹 (s,a,r,s',a')

TD 学习 Q 函数

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma Q(s',a') - Q(s,a)\right)$$

Sarsa 控制在每一时刻：

- 策略评估：使用 TD 学习 $Q \approx Q_\pi$
- 策略提升：$\epsilon-$ 贪心策略提升

算法 6.2: Sarsa 算法

初始化 $Q(s,a), \pi = \epsilon - \text{greedy}(Q), t = 0$，初始状态 $s_t = s_0$

采样动作 $a_t \sim \pi(s_t)$

loop

执行动作 a_t 后获得观测量 (r_{t+1}, s_{t+1})，通过 s_t 与 a_t 计算奖励 r_{t+1}

采样动作 $a_{t+1} \sim \pi(s_{t+1})$

根据 $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ 更新 Q

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma Q(s',a') - Q(s,a)\right)$$

策略更新

$$\pi = \epsilon - \text{greedy}(Q)$$

$t = t + 1$

end loop

图 6.3: Sarsa 与 Q-learning 流程图。

Bibliography

[1] Junyan Hu, Hanlin Niu, Joaquin Carrasco, Barry Lennox, and Farshad Arvin. Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning. *IEEE Transactions on Vehicular Technology*, 69(12):14413–14423, 2020.

7 Graph Neural Network

7.1

8 Attention Module

8.1

9 Contrastive Learning

9.1

10 Imbalanced & Long-Tailed Problem

许多人工智能系统依赖于监督学习方法，在这种方法中，神经网络根据标记数据进行训练。通常，在模型训练的过程中，随着特定类别的样本的频率下降，会使该类别的平均模型性能也会下降。这种情况是人类在采集数据的过程中经常遇到的，如图10.1所示。正是由于人类在采集数据的过程中，不能保证每一个类别的样本数目的数量都是充足的，会存在有一部分类别的样本数目非常充足，而有一部分类别的样本数目稀缺，这种数据分布被称为长尾数据分布，也是标签分布不均衡的一种常见形式。目前针对标签不均衡问题的一些解决方案包括三种 [1]：Re-balancing，Information Augmentation 和 Module Improvement。

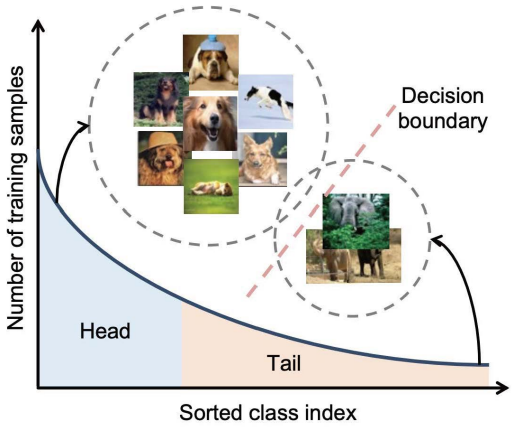


图 10.1: 长尾数据集的标签分布。[1]

10.1 Re-balancing

Re-balancing 的方法大致分为 3 种，第一种类似 Re-Sampling 的方法，即在训练过程中通过重采样的方法已达到不同类标签数量的均衡 [2]；第二种被称为 Re-weighting，旨在通过在训练期间调整不同类别的损失值来重新平衡类别，例如有名的 Focal Loss [3]；第三种被称之为 Logit adjustment [1]，即基于标签频率的模型 logit 事后移动，是在类不平衡问题中获得类之间较大的相对余量的经典思想 [4]。

Re-balancing 在二分类问题的解决中最为常见，例如一个二分类问题损失函数我们可以定义为

$$\mathcal{L} = -[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log (1 - \sigma(x_n))]$$

其中 x_n, y_n 分别代表模型输出的 logit 和真实标签，σ 为 sigmoid 函数。这里我们可以为此损失函数加上一个系数 p 以平衡单个二分类问题的正负样本，

$$\mathcal{L} = -[p \cdot y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log (1 - \sigma(x_n))] \tag{10.1}$$

其中 p = negative/positive，例如一个二分类样本中我有 100 个正例 (positive) 和 300 个负例，则我们可以使 $p = \frac{300}{100} = 3$ 以平衡单个类的正负样本的均衡。

10.1.1 Balanced MSE

Balanced MSE [5] 来自 CVPR 2022 (oral) 的一项技术，原文在https://arxiv.org/abs/2203.16427。

代码见https://github.com/jiawei-ren/BalancedMSE

假设输入 $\boldsymbol{x} \in X$，标签 $\boldsymbol{y} \in Y = \mathbb{R}^d$，其中 $d > 1$。不平衡回归假设训练集和测试集来自不同的联合分布，即训练集的标签分布 $p_{\text{train}}(\boldsymbol{x}, \boldsymbol{y})$ 与测试集标签分布 $p_{\text{bal}}(\boldsymbol{x}, \boldsymbol{y})$ 不一样。通常的理想情况下，我们希望训练与测试中标签条件概率 $p(\boldsymbol{x} \mid \boldsymbol{y})$ 相同。

在预测分布上，MSE 等价于 NLL (Negative Log Likelihood, $-\log p(\boldsymbol{y} \mid \boldsymbol{x})$)。如图10.2所示，当训练数据不平衡时，MSE (Mean Square Error) 会被标签分布所影响而倾向于预测常见的标签。当测试集是平衡的或衡量指标是平衡的时候，MSE 的这一特点会导致模型在整体标签上的平均表现变差。相比之下，Balanced MSE 利用 $p_{\text{train}}(\boldsymbol{y})$ 进行从 $p_{\text{bal}}(\boldsymbol{y} \mid \boldsymbol{x})$ 到 $p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x})$ 的统计转换，从而允许回归器通过仍然最小化 $p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x})$ 的 NLL 来建模所需的 $p_{\text{bal}}(\boldsymbol{y} \mid \boldsymbol{x})$。

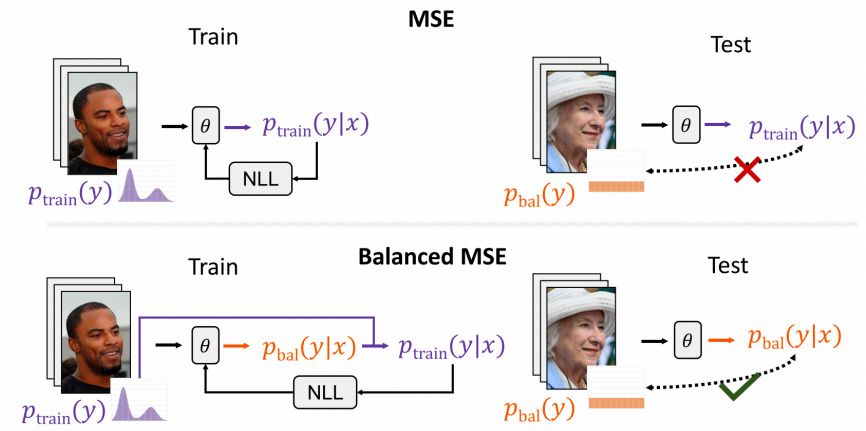


图 10.2: MSE 与 Balanced MSE 的比较。[5]

首先通过联立贝叶斯公式有 $\frac{p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x})}{p_{\text{bal}}(\boldsymbol{y} \mid \boldsymbol{x})} \propto \frac{p_{\text{train}}(\boldsymbol{y})}{p_{\text{bal}}(\boldsymbol{y})}$，我们的目标仍然是优化 $p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x})$，但是我们希望网络的参数 θ 学习到的分布不是 $p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x})$ 而是 $p_{\text{bal}}(\boldsymbol{y} \mid \boldsymbol{x})$，则可以通过先估算 p_{bal} 再转换 p_{train} 的方式优化 θ，Balanced MSE 定义为：

$$\begin{aligned} L &= -\log p_{\text{train}}(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta}) = -\log \frac{p_{\text{bal}}(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{\theta}) \cdot p_{\text{train}}(\boldsymbol{y})}{\int_Y p_{\text{bal}}(\boldsymbol{y}' \mid \boldsymbol{x}; \boldsymbol{\theta}) \cdot p_{\text{train}}(\boldsymbol{y}') \, d\boldsymbol{y}'} \\ &\cong -\log \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}_{\text{pred}}, \sigma_{\text{noise}}^2 \mathrm{I}) + \log \int_Y \mathcal{N}(\boldsymbol{y}'; \boldsymbol{y}_{\text{pred}}, \sigma_{\text{noise}}^2 \mathrm{I}) \cdot p_{\text{train}}(\boldsymbol{y}') \, d\boldsymbol{y}' \end{aligned} \tag{10.2}$$

其中 $\cong$ 隐藏了一个常数项 $-\log p_{\text{train}}(\boldsymbol{y})$，而实际计算中，因为数值都是离散化的，可写作：

$$p_{\text{train}}(y \mid \boldsymbol{x}; \boldsymbol{\theta}) = \frac{\exp(\eta[y]) \cdot p_{\text{train}}(y)}{\sum_{y' \in Y} \exp(\eta[y']) \cdot p_{\text{train}}(y')} \tag{10.3}$$

其中，$\eta[y]$ 为模型在类别 y 上的输出。大致思想便是如此，$p_{\text{train}}(y)$ 作者给了一些计算思路，例如 GMM 方法 $p_{\text{train}}(\boldsymbol{y}) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$，蒙特卡洛模拟或者基于 Batch 的蒙特卡洛（这个不需要知道标签先验分布）

$$L = -\log \frac{\exp\left(-\|\boldsymbol{y}_{\text{pred}} - \boldsymbol{y}\|_2^2 / \tau\right)}{\sum_{\boldsymbol{y}' \in B_y} \exp\left(-\|\boldsymbol{y}_{\text{pred}} - \boldsymbol{y}'\|_2^2 / \tau\right)} \tag{10.4}$$

其中 $\tau = 2\sigma_{\text{noise}}^2$，等等。总而言之这些方法除了基于 Batch 的蒙特卡洛方法外都是用有先验的噪声模拟出来的数值，而不是直接通过公式10.2中复杂积分计算而来。

Bibliography

[1] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. Deep long-tailed learning: A survey. *arXiv preprint arXiv:2110.04596*, 2021.

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[3] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[4] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *International Conference on Learning Representations*, 2020.

[5] Jiawei Ren, Mingyuan Zhang, Cunjun Yu, and Ziwei Liu. Balanced mse for imbalanced visual regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

11 Face

11.1 Introduction

人脸识别系统已经在我们的生活中被广泛的使用，他对我们的生活是非常重要的。

人脸识别系统需要面对复杂的环境，例如光照，背景，以及人可能存在的姿态，遮挡以及不同的表情，都将会影响到人脸识别系统的决策问题 [1]。

11.2 Face Recognition

11.2.1 ArcFace

ArcFace [2]，或 Additive Angular Margin Loss，是在人脸识别任务中使用的损失函数。softmax 传统上用于这些任务。然而，softmax 损失函数并没有明确优化特征嵌入以增强类内样本的相似性和类间样本的多样性，这导致在类内外观变化较大的情况下，深度人脸识别的性能存在差距。原始的论文在https://arxiv.org/abs/1801.07698中，其损失函数的公式如下

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}, \tag{11.1}$$

其中 y_i 为标签，θ_{y_i} 为分类层权重 W_i 与特征 x 的角度，角度是用余弦相似度计算的。s 为缩放因子，m 为角度边界间隔，类似 SVM 的间隔。其损失函数计算流程如图11.1所示。

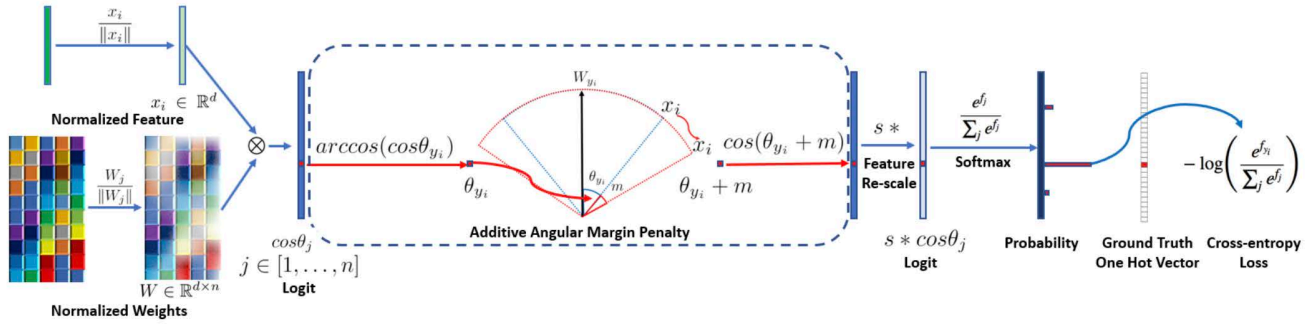


图 11.1: ArcFace 损失函数的计算流程。

一段基于 Pytorch 的代码示例如下，实例请见
https://github.com/RuoyuChen10/FaceTechnologyTool/tree/master/FaceRecognition

```
class ArcFace(nn.Module):
    def __init__(self, s=64.0, m=0.5):
        super(ArcFace, self).__init__()
        self.s = s
        self.m = m

    def forward(self, cosine: torch.Tensor, label):
        index = torch.where(label != -1)[0]
        m_hot = torch.zeros(index.size()[0], cosine.size()[1], device=cosine.device)
        m_hot.scatter_(1, label[index, None], self.m)
```

```
        cosine.acos_()
        cosine[index] += m_hot
        cosine.cos_().mul_(self.s)
        return cosine
```

11.2.2 AdaFace

AdaFace [3] 来自 CVPR 2022 (oral) 的一项技术，原文在https://arxiv.org/pdf/2204.00964.pdf。

代码在https://github.com/mk-minchul/AdaFace

在这项工作中，在损失函数中引入了另一个因素，即**图像质量**。作者认为，强调错误分类样本的策略应根据其图像质量进行调整。具体来说，简单或困难样本的相对重要性应该基于样本的图像质量来给定。据此作者提出了一种新的损失函数来通过图像质量强调不同的困难样本的重要性。

如图11.2所示，与传统的基于 Margin Loss 的损失函数相比，AdaFace 首先对特征评估质量，并根据质量自适应调整边界函数。

其中质量评估 (Image Quality Indicator) 利用了类似 Batch Normalization 的方案，批量输入一堆图像，并计算均值与方差，将图像分布模拟为一组高斯分布，

$$\widehat{\|\boldsymbol{z}_i\|} = \left\lfloor \frac{\|\boldsymbol{z}_i\| - \mu_z}{\sigma_z / h} \right\rceil_{-1}^{1} \tag{11.2}$$

h 为超参数，论文中建议 0.33。随后根据这一参数引导 Adaptive Margin Function，

$$f(\theta_j, m)_{\text{AdaFace}} = \begin{cases} s\cos(\theta_j + g_{\text{angle}}) - g_{\text{add}} & j = y_i \\ s\cos\theta_j & j \neq y_i \end{cases} \tag{11.3}$$

其中 $g_{\text{angle}} = -m \cdot \widehat{\|\boldsymbol{z}_i\|}, g_{\text{add}} = m \cdot \widehat{\|\boldsymbol{z}_i\|} + m$。这里损失函数 $f(\theta_j, m)_{\text{AdaFace}}$ 在 $\widehat{\|\boldsymbol{z}_i\|} = -1, 0, 1$ 时分别相当于 ArcFace [2]，CosFace [4] 与有位移的负的 angular margin。

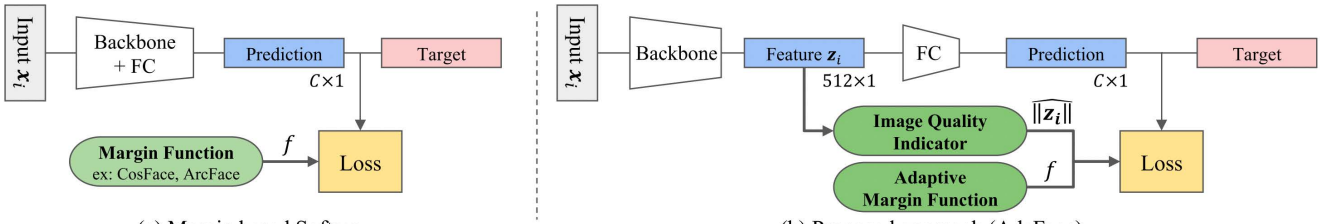


图 11.2: AdaFace 与传统基于 Margin Loss 的损失函数对比。[3]

11.3 Face Image Quality Assessment

人脸图像的质量会影响到人脸识别系统的性能。Terhörst 等人 [1] 将人脸质量评估 (Face Image Quality Assessment, FIQA) 技术的发展描述为三个阶段。在第一个阶段的 FIQA 中，人们为人脸图像的质量提出了一些标准，例如 ICAO 9303 的 Machine Readable Travel Documents [5,6] 和 ISO/IEC 19794-5 [7]，将标准质量大致分为光照，遮挡，姿态，表情等等。故有部分工作是基于这些标准而进行的人脸质量评估 [8–13]。而在第二个阶段的 FIQA 中，由于机器学习性能的提高，评估人脸图像质量的方法更倾向于通过学习的方式得到 [14–19]。基于学习的方法需要人为的标签，这

些标签要么是通过人为判断并标定，要么是通过基于比较的分数分布 [1]。然而人为的判断是存在主观性的，人类可能不知道面部识别系统所倾向的质量分数，人为的标注也容易出现错误标记。第三个阶段的 FIQA 中，人脸质量评估避开了使用质量标签训练的方式。如 Terhörst 等人 [20] 通过对人脸识别网络提取的 embedding 加入了 dropout [21] 并观察输出的扰动以评估 embedding 的鲁棒性，以此来衡量图像质量。

11.3.1 SER-FIQ

SER-FIQ [20] 来自 CVPR 2020 的一项技术，是在没有 Quality 标签的情况下计算 Face Image Quality 的，如图 11.3 所示。将输入 I 输入到一个人脸识别的模型 $\mathcal{M}$ 中，其中 subnetwork 为全连接层，将 embedding 映射为另一个 embedding x_s。经过 m 次的 dropout [21] (在 subnetwork 上进行) 生成了 m 个随机的 embeddings $X(I) = \{x_s\}_{s\in\{1,2,\cdots,m\}}$。这时通过计算 embedding 之间两两欧氏距离，观察输出的扰动以评估 embedding 的鲁棒性，以此来衡量图像质量。人脸图像质量被定义为：

$$q(X(I)) = 2\sigma\left(-\frac{2}{m^2}\sum_{i<j} d(x_i, x_j)\right) \tag{11.4}$$

其中 σ 为 Sigmoid 函数，目的是将人脸质量限制在 $[0, 1]$ 的范围内，d 为欧式距离。

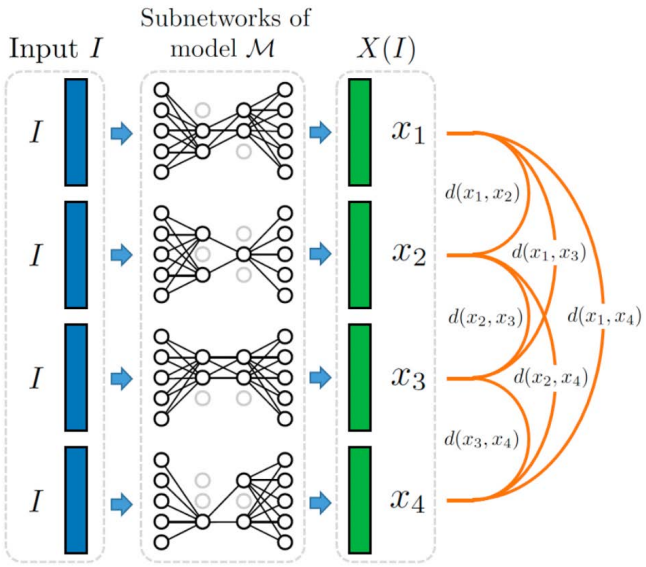


图 11.3: SER-FIQ，人脸图像质量评估方法。[20]

11.3.2 MagFace

MagFace [22] 是来自 CVPR 2021 (oral) 的一项技术[1]

11.4 Explainable Face Recognition

可解释性人脸识别 (Explainable Face Recognition, XFR) 这个概念是近些年提出的，为一个比较新的概念。尽管深度学习模型取得了重大的进步，但是模型黑盒般的信息决策总是令人不安的，尤其是在医学或者生物特征这些领

[1] AdaFace: https://arxiv.org/abs/2103.06627

域，我们需要一个透明可释的模型，以使人们信任模型所做出的决策。通常与解释神经网络的传播过程不同，可解释性的目的是网络在做出决策时，提供一个它们所做出这样决策的依据，这个依据是能被人类很好的理解的，尽管这个依据可能是由其他神经网络预测得来的，例如通过语言模型为当前识别模型生成一些文本信息以使人类很好理解为什么模型做出这样的决策，但是这个语言模型将是另一个黑盒 [23]。XFR 旨在为模型的结果或者模型的架构提供一个可令人理解的内容。通常他们会可视化一些人脸匹配或者识别中一些重要的区域 [24]，或者设计一些特殊的框架来解释人脸模型 [25]。

11.4.1 PLQ

Terhörst 等人 [1] 提出了一个针对图像评估分数所做的模型可解释性的工作。他们的先前关于人脸图像质量评估的工作见章节11.3.1。通常，人脸识别不像分类，有一个 classification head，而是输出一个 embedding，人脸图像质量评估也是基于这个 embedding 方法。作者通过首先计算一个图像 I 的质量 $\hat{Q}_I$，然后将 embedding e_I 与质量值 $\hat{Q}_I$ 通过一层全连接层 $w_{N_{\hat{Q}_I}}$ 连起来，即一个向量 e_I 映射到一个值 $\hat{Q}_I$。作者假设 embedding e_I 的每一个节点对图像质量评估的贡献相同，因此 $w_{N_{\hat{Q}_I}}$ 的计算方式为

$$w_{N_{\hat{Q}_I}} = \frac{\hat{Q}_I}{\|e_I\|_1} \tag{11.5}$$

因为 $w_{N_{\hat{Q}_I}}$ 的作用，图像 I 到这个人脸图像质量 $\hat{Q}_I$ 有一个直接的映射 $\hat{Q}_I = \mathcal{M}_Q(I)$，这时我们根据梯度计算显著图 $S(I) = \frac{\partial \mathcal{M}_Q(I)}{\partial I}$。由于输入图像通常是 RGB 三通道图像，回传到原始图像的梯度也是三通道，我们需要沿着通道方向将其求平均以获得单通道显著图 $\hat{S}(I) = \frac{1}{3}\sum_{c=1}^{3}|g_{i,j,c}|$，注意这里数值不需要像 Grad-CAM 一样将显著图归一化。这时显著图为像素级的，但是由于数值分布问题，需要采取一个可视化方程 v:

$$v(\hat{S}) = 1 - \frac{1}{1 + (10^{\gamma} \times \hat{g}^2_{i,j,c})} \tag{11.6}$$

将 $\hat{S}$ 映射到 $[0, 1]$ 的值域。整体的流传如图11.4所示。

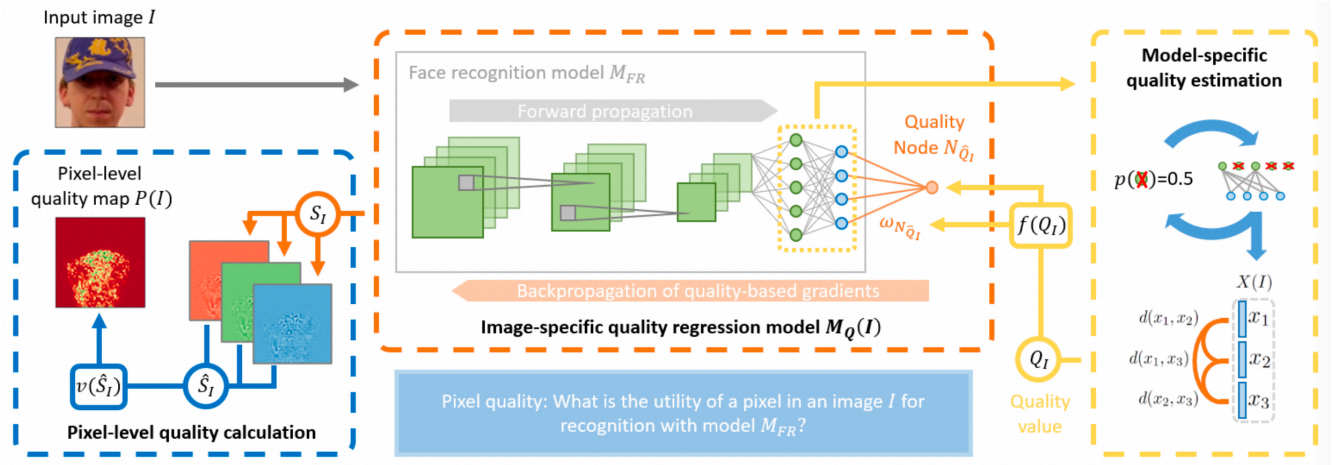


图 11.4: Pixel-level quality estimation approach，人脸图像质量评估方法的可解释性。[1]

11.5 Face Image Editing

Bibliography

[1] Philipp Terhörst, Marco Huber, Naser Damer, Florian Kirchbuchner, Kiran Raja, and Arjan Kuijper. Pixel-level face image quality assessment for explainable face recognition. *arXiv preprint arXiv:2110.11001*, 2021.

[2] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

[3] Minchul Kim, Anil K Jain, and Xiaoming Liu. Adaface: Quality adaptive margin for face recognition. 2022.

[4] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018.

[5] Machine readable travel documents. https://www.icao.int/publications/pages/publication.aspx?docnum=9303.

[6] Jean Monnerat, Serge Vaudenay, and Martin Vuagnoux. About machine-readable travel documents. Technical report, Springer, 2007.

[7] Information technology —biometric data interchange formats —part 5: Face image data. https://www.iso.org/standard/50867.html.

[8] Rein-Lien Vincent Hsu, Jidnya Shah, and Brian Martin. Quality assessment of facial images. In *2006 Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference*, pages 1–6. IEEE, 2006.

[9] Xiufeng Gao, Stan Z Li, Rong Liu, and Peiren Zhang. Standardization of face image sample quality. In *International Conference on Biometrics*, pages 242–251. Springer, 2007.

[10] Matteo Ferrara, Annalisa Franco, Dario Maio, and Davide Maltoni. Face image conformance to iso/icao standards in machine readable travel documents. *IEEE Transactions on Information Forensics and Security*, 7(4):1204–1213, 2012.

[11] Ayman Abaza, Mary Ann Harrison, and Thirimachos Bourlai. Quality metrics for practical face recognition. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3103–3107. IEEE, 2012.

[12] Ayman Abaza, Mary Ann Harrison, Thirimachos Bourlai, and Arun Ross. Design and evaluation of photometric image quality measures for effective face recognition. *IET Biometrics*, 3(4):314–324, 2014.

[13] Abhishek Dutta, Raymond Veldhuis, and Luuk Spreeuwers. A bayesian model for predicting face recognition performance using image quality. In *IEEE International Joint Conference on Biometrics*, pages 1–8. IEEE, 2014.

[14] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C Lovell. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *CVPR 2011 WORKSHOPS*, pages 74–81. IEEE, 2011.

[15] Gaurav Aggarwal, Soma Biswas, Patrick J Flynn, and Kevin W Bowyer. Predicting performance of face recognition systems: An image characterization approach. In *CVPR 2011 WORKSHOPS*, pages 52–59. IEEE, 2011.

[16] Jiansheng Chen, Yu Deng, Gaocheng Bai, and Guangda Su. Face image quality assessment based on learning to rank. *IEEE signal processing letters*, 22(1):90–94, 2014.

[17] Hyung-Il Kim, Seung Ho Lee, and Man Ro Yong. Face image assessment learned with objective and relative face image qualities for improved face recognition. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 4027–4031. IEEE, 2015.

[18] Lacey Best-Rowden and Anil K Jain. Learning face image quality from human assessments. *IEEE Transactions on Information Forensics and Security*, 13(12):3064–3077, 2018.

[19] Javier Hernandez-Ortega, Javier Galbally, Julian Fierrez, Rudolf Haraksim, and Laurent Beslay. Faceqnet: Quality assessment for face recognition based on deep learning. In *2019 International Conference on Biometrics (ICB)*, pages 1–8. IEEE, 2019.

[20] Philipp Terhorst, Jan Niklas Kolf, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. Ser-fiq: Unsupervised estimation of face image quality based on stochastic embedding robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5651–5660, 2020.

[21] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[22] Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. Magface: A universal representation for face recognition and quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14225–14234, 2021.

[23] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European conference on computer vision*, pages 3–19. Springer, 2016.

[24] Jonathan R Williford, Brandon B May, and Jeffrey Byrne. Explainable face recognition. In *European Conference on Computer Vision*, pages 248–263. Springer, 2020.

[25] Yu-Sheng Lin, Zhe-Yu Liu, Yu-An Chen, Yu-Siang Wang, Ya-Liang Chang, and Winston H Hsu. xcos: An explainable cosine metric for face verification task. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 17(3s):1–16, 2021.

12 Explainable AI

Explainable AI (XAI)，或 Interpretable AI 的概念指的是人工智能 (AI) 的解决方案的结果可以被人类理解 [1]。

人工智能中使用的算法可以分为白盒和黑盒机器学习算法。白盒模型是可提供领域专家可以理解的结果。另一方面，黑盒模型极难解释，甚至领域专家也难以理解 [2]。XAI 算法被认为遵循透明性、可解释性和可解释性三个原则。“如果方法设计者可以描述和激励从训练数据中提取模型参数并从测试数据中生成标签的过程”，则可以提供透明度 [3]。可解释性描述了理解 ML 模型并以人类可以理解的方式呈现决策制定的基本基础的可能性 [4, 5, 5]。可解释性是一个被认为很重要的概念，但还没有一个联合定义 [3]。有人建议，ML 中的可解释性可以被视为“可解释域的特征的集合，这些特征有助于给定示例产生决策（例如，分类或回归）”[6]。如果算法满足这些要求，它们就为证明决策、跟踪和验证决策、改进算法和探索新事实提供了基础 [7]。

关于更多的可解释性机器学习的资源，我强烈建议关注于以下几个部分，Christoph Molnar 所著《Interpretable machine learning》[2] [8]，上海交通大学 Quanshi Zhang 老师的工作[3]。

12.1 Interpretability

12.2 可视化

12.3 特征重要性

12.4 反事实可解释性

Bibliography

[1] Femke M Janssen, Katja KH Aben, Berdine L Heesterman, Quirinus JM Voorham, Paul A Seegers, and Arturo Moncada-Torres. Using explainable machine learning to explore the impact of synoptic reporting on prostate cancer. *Algorithms*, 15(2):49, 2022.

[2] Octavio Loyola-Gonzalez. Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. *IEEE Access*, 7:154096–154113, 2019.

[3] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *Ieee Access*, 8:42200–42216, 2020.

[4] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*, 2019.

[5] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[6] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[7] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

[2] Interpretable machine learning, A Guide for Making Black Box Models Explainable: https://christophm.github.io/interpretable-ml-book/

[3] Quanshi Zhang's homepage: http://qszhang.com/

[8] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

13 Few-shot Learning

13.1

14 Federal Learning

定义 14.1: 联邦机器学习

联邦学习是一种机器学习技术，具体来说就是人们在多个拥有本地数据样本的分散式边缘设备或服务器上训练算法。这种方法与传统的集中式机器学习技术有显著不同，传统的集中式机器学习技术将所有的本地数据集上传到一个服务器上，而更经典的分散式方法则通常假设本地数据样本都是相同分布的，如图14.1所示。

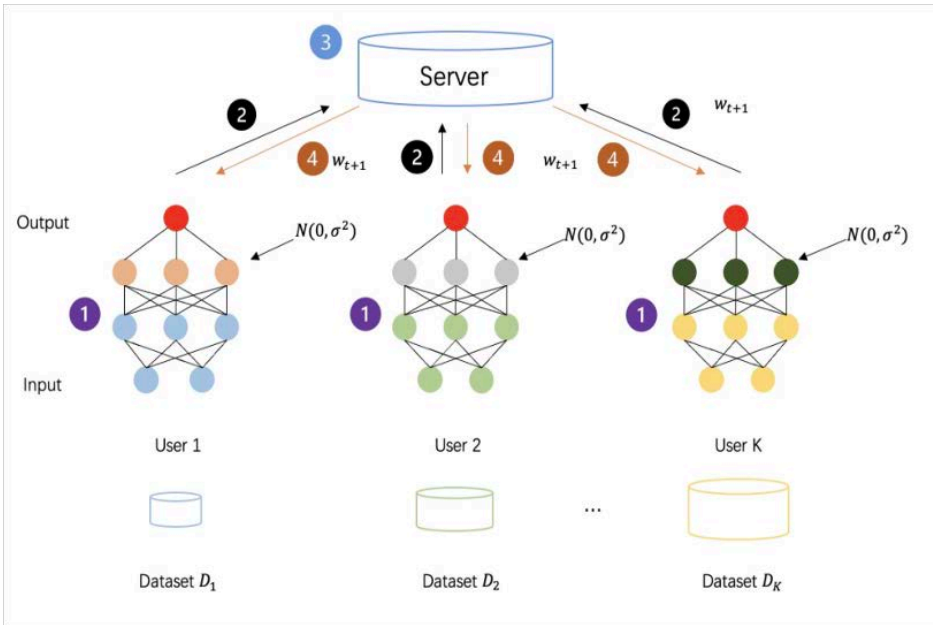


图 14.1: 联邦学习示意图。

联邦学习的主要问题 [1]

- 隐私问题：联邦学习只共享模型更新信息，比如梯度，而不是原始数据。然而，信息将被泄露给第三方或中央服务器。
- 系统不一致：联邦网络的存储、计算和交互能力因硬件、网络和能源的不同而不同。由于网络规模或设备系统的限制，每次只激活部分设备。
- 数据不一致：网络中不断产生分布差异较大的数据，导致设备之间存在非 IID 数据。
- 交互开销：由于带宽、能量等问题，网络上的交互比本地计算要慢得多。

14.1 FedAvg

FedAvg [2] 是 2017 年提出的，以往的联邦机器学习 FL 存在**通信速率不稳定，且可能不可靠**，以及**聚合服务器的容量有限，同时与 server 通信的 client 的数量受限的问题**。FedAvg 算法采取增加客户端计算，限制通信频率的策略。其算法流程如图14.2所示。

Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

Server executes:
initialize w_0
for each round $t = 1, 2, \dots$ **do**
  $m \leftarrow \max(C \cdot K, 1)$
  $S_t \leftarrow$ (random set of m clients)
  for each client $k \in S_t$ **in parallel do**
    $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
  $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate(k, w): *// Run on client k*
$\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size B)
for each local epoch i from 1 to E **do**
  for batch $b \in \mathcal{B}$ **do**
    $w \leftarrow w - \eta \nabla \ell(w; b)$
return w to server

图 14.2: FedAvg 算法流程。

14.2 MOON

MOON [3] 来自 CVPR 2021，是一篇将 Contrastive Learning 引入联邦学习的 Paper，其核心如图14.3所示。

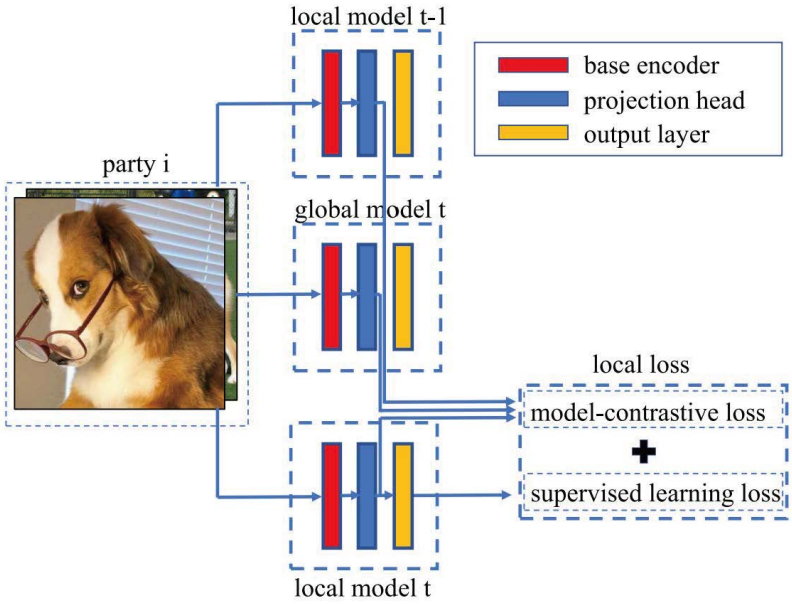


图 14.3: MOON 算法损失函数。

$$\ell_{\text{con}} = -\log \frac{\exp\left(\text{sim}\left(z, z_{\text{glob}}\right)/\tau\right)}{\exp\left(\text{sim}\left(z, z_{\text{glob}}\right)/\tau\right) + \exp\left(\text{sim}\left(z, z_{\text{prev}}\right)/\tau\right)} \tag{14.1}$$

其中 $z_{\text{glob}} = R_{w^t}(x)$ 为在 t 时刻从全局模型 w^t 中提取的特征，$z_{\text{prev}} = R_{w_i^{t-1}}(x)$ 为从上一时刻 $t-1$ 时局部模型 w_i^{t-1} 提取的特征，而 $z = R_{w_i^t(x)}$ 为 t 时刻从局部模型提取的特征。该损失函数 $\ell_{\text{con}}\left(w_i^t; w_i^{t-1}; w^t; x\right)$ 的目的为减少 z 与 z_{glob} 之间的距离，而增加 z 与 z_{prev} 之间的距离。

Bibliography

[1] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[2] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[3] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021.

参考文献

[1] Haskell B Curry. The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics*, 2(3):258–261, 1944.

[2] Juan C Meza. Steepest descent. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(6):719–722, 2010.

[3] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

[4] Nicolas Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. *Advances in neural information processing systems*, 25, 2012.

[5] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.