# 模式识别与计算机视觉：第三次作业

人工智能学院 221300066 季千焜

2025 年 4 月 21 日

## 1 习题一

### (a)

```
import numpy as np

# 生成10维样本
d = 10
samples = np.random.randn(20, d)
norms = np.linalg.norm(samples, axis=1)

mean_norm = np.mean(norms)
min_norm = np.min(norms)
max_norm = np.max(norms)

print(f"10维： 均值={mean_norm:.4f}， 最小值={min_norm:.4f}， 最大值={max_norm:
```

示例输出：

```
10维： 均值=2.9395， 最小值=1.4004， 最大值=4.5317
```

### (b)

不同维度的结果：

```
dims = [100, 1000, 10000, 100000]
```

```
for d in dims:
    samples = np.random.randn(20, d)
    norms = np.linalg.norm(samples, axis=1)
    print(f"维度 {d}: 均值={norms.mean():.4f}, 最小值={norms.min():.4f}, 最大
```

示例输出:

维度 100: 均值=9.9377, 最小值=9.1908, 最大值=11.0601
维度 1000: 均值=31.6023, 最小值=30.7002, 最大值=32.3765
维度 10000: 均值=100.4276, 最小值=99.0455, 最大值=101.9868
维度 100000: 均值=316.3742, 最小值=315.1172, 最大值=318.1817

### (c)

猜想: 高维标准高斯分布的样本 $\ell_2$-范数 的均值趋近于其维度的平方根 $\sqrt{d}$, 且随着维度增加, 样本范数越来越集中在该值附近, 相对波动减小。

### (d)

数学描述:
设随机向量 $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, 其中 $\mathbf{I}_d$ 为 d 维单位矩阵。当 $d \to \infty$ 时, 有:

$$\frac{\|\mathbf{X}\|_2}{\sqrt{d}} \xrightarrow{P} 1$$

即对于任意 $\epsilon > 0$, 有:

$$\lim_{d\to\infty} P\left(\left|\frac{\|\mathbf{X}\|_2}{\sqrt{d}} - 1\right| \geq \epsilon\right) = 0$$

该性质由大数定律保证, 因 $\|\mathbf{X}\|_2^2 = \sum_{i=1}^{d} X_i^2$ 的均值与方差分别为 $d$ 和 $2d$, 其归一化后依概率收敛到 1。

## 2 习题三

### (a)

$\kappa$ 是合法的核函数.
证明:

对于任意 $n$ 和 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$,
因为 $\kappa_1$ 和 $\kappa_2$ 是核函数, 因此其对应的核矩阵 $\boldsymbol{K}_1$ 和 $\boldsymbol{K}_2$ 是半正定矩阵,
即有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{y} \geq 0$ 与 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_2\boldsymbol{y} \geq 0$, 对于任何 $n$ 维向量 $\boldsymbol{y}$ 成立.
因为 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j) + \kappa_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$,
所以有 $\boldsymbol{K}_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j) + \kappa_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$
因此有 $\boldsymbol{K} = \boldsymbol{K}_1 + \boldsymbol{K}_2$.
则我们有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{y} = \boldsymbol{y}^{\mathrm{T}}(\boldsymbol{K}_1 + \boldsymbol{K}_2)\boldsymbol{y} = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{y} + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_2\boldsymbol{y} \geq 0$
所以可知 $\boldsymbol{K}$ 也是半正定矩阵, $\kappa$ 是合法的核函数.

## (b)

$\kappa$ 不是合法的核函数.

证明: 只需要举出一个反例

令 $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) = 0, \kappa_2(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$, 易知两者均为合法的核函数, 则我们有 $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa_1(\boldsymbol{x}, \boldsymbol{y}) - \kappa_2(\boldsymbol{x}, \boldsymbol{y}) = -\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$, 不妨令 $d = 2$.

则存在两个样本 $\boldsymbol{x}_1 = (1, 0)^{\mathrm{T}}, \boldsymbol{x}_2 = (0, 1)^{\mathrm{T}}$, 其对应的核矩阵为

$$\boldsymbol{K} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$

令 $\boldsymbol{y} = (1, 0)^{\mathrm{T}}$, 我们有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{y} = -1 < 0$

因此 $\boldsymbol{K}$ 不是一个半正定矩阵, 也即 $\kappa$ 不是合法的核函数.

## (c)

$\kappa$ 是合法的核函数.

证明:

对于任意 $n$ 和 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$,

因为 $\kappa_1$ 是核函数, 因此其对应的核矩阵 $\boldsymbol{K}_1$ 是半正定矩阵,

即有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{y} \geq 0$, 对于任何 $n$ 维向量 $\boldsymbol{y}$ 成立.

因为 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha\kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j)$,

所以有 $\boldsymbol{K}_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \alpha\kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j)$

因此有 $\boldsymbol{K} = \alpha\boldsymbol{K}_1$.

因为 $\alpha \in \mathbb{R}^+$ 是一个正实数,

则我们有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{y} = \boldsymbol{y}^{\mathrm{T}}(\alpha\boldsymbol{K}_1)\boldsymbol{y} = \alpha\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{y} \geq 0$

所以可知 $\boldsymbol{K}$ 也是半正定矩阵, $\kappa$ 是合法的核函数.

## (d)

$\kappa$ 不是合法的核函数.

证明: 只需要举出一个反例.

令 $\alpha = 1$ 以及 $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$, 易知其为合法的核函数, 则我们有 $\kappa(\boldsymbol{x}, \boldsymbol{y}) = -\alpha\kappa_1(\boldsymbol{x}, \boldsymbol{y}) = -\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$, 不妨令 $d = 2$.

则存在两个样本 $\boldsymbol{x}_1 = (1, 0)^{\mathrm{T}}, \boldsymbol{x}_2 = (0, 1)^{\mathrm{T}}$, 其对应的核矩阵为

$\boldsymbol{K} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$

令 $\boldsymbol{y} = (1, 0)^{\mathrm{T}}$, 我们有 $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{y} = -1 < 0$

因此 $\boldsymbol{K}$ 不是一个半正定矩阵, 也即 $\kappa$ 不是合法的核函数.

## (e)

$\kappa$ 是合法的核函数.

证明:

对于任意 $n$ 和 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$,

因为 $\kappa_1$ 和 $\kappa_2$ 是核函数, 因此其对应的核矩阵 $\boldsymbol{K}_1$ 和 $\boldsymbol{K}_2$ 是半正定矩阵,

因为 $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j)\kappa_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$,

所以有 $\boldsymbol{K}_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \kappa_1(\boldsymbol{x}_i, \boldsymbol{x}_j)\kappa_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$

因此有 $\boldsymbol{K} = \boldsymbol{K}_1 \circ \boldsymbol{K}_2$, 即 $\boldsymbol{K}_1$ 与 $\boldsymbol{K}_2$ 的逐元素相乘.

则可证明 Schur 乘积定理:

对于 $\mathbb{R}^{n\times n}$ 上的半正定矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$, 对于 $\forall \boldsymbol{x} \in \mathbb{R}^n$, 我们可以证明 $\boldsymbol{C} = \boldsymbol{A} \circ \boldsymbol{B}$ 半正定.

由于 $\boldsymbol{A}$ 为半正定矩阵, 因此存在 $\boldsymbol{U} \in \mathbb{R}^n$ 使得 $\boldsymbol{A} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}$.

$$
\begin{aligned}
\boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{x} &= \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ij}x_i x_j \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} u_{ki}u_{kj}b_{ij}x_i x_j \\
&= \sum_{k=1}^{n}[\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(u_{ki}x_i)(u_{kj}x_j)] \\
&\geq 0
\end{aligned}
$$

即 $\boldsymbol{C} = \boldsymbol{A} \circ \boldsymbol{B}$ 半正定.

所以同理可知 $\boldsymbol{K}$ 也是半正定矩阵, $\kappa$ 是合法的核函数.

### (f)

$\kappa$ 是合法的核函数.

证明:

对于任意 $n$ 和 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$, 通过核函数 $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa_3(\phi(\boldsymbol{x}), \phi(\boldsymbol{y}))$ 得到的 $\boldsymbol{K}$ 为

$[\boldsymbol{K}]_{ij} = \kappa_3(\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j))$

我们使用函数 $\phi(\cdot)$ 处理得到 $\{\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \cdots, \phi(\boldsymbol{x}_n)\}$, 这也是一组样本集合, 因此通过核函数 $\kappa_3$ 得到的 $\boldsymbol{K}_3$ 是半正定矩阵, 其中

$[\boldsymbol{K}_3]_{ij} = \kappa_3(\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j))$

可见 $\boldsymbol{K} = \boldsymbol{K}_3$.

所以可知 $\boldsymbol{K}$ 也是半正定矩阵, $\kappa$ 是合法的核函数.

## 3 习题四

### (a)

我们对每个样例 $(\boldsymbol{x}_i, y_i)$ 附上一个代价系数 $k_i$,

其中 $k_i = \begin{cases} 1, & y_i = +1 \\ k, & y_i = -1 \end{cases}$ 也即 $k_i = 1 - \frac{1}{2}(k-1)(y_i - 1)$.

然后相应的 SVM 优化问题即可改为

$$\begin{aligned} \min_{\boldsymbol{w}, b, \xi_i} \quad & \frac{1}{2}\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + C\sum_{i=1}^{n} k_i \xi_i \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad 1 \leq i \leq n \\ & \xi_i \geq 0, \qquad\qquad\qquad\quad 1 \leq i \leq n \end{aligned}$$

### (b)

通过拉格朗日乘子法得到拉格朗日函数

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

其中 $\alpha_i \geq 0, \mu_i \geq 0$ 是拉格朗日乘子.
令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\mu})$ 对 $\boldsymbol{w}, b, \xi_i$ 的偏导等于零可得

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

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

$$k_i C = \alpha_i + \mu_i$$

将上三式逐步带入有

$$\begin{aligned}
&L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\mu}) \\
&= \frac{1}{2}\|\boldsymbol{w}\|_2^2 + C\sum_{i=1}^{n} k_i\xi_i + \sum_{i=1}^{n}\alpha_i(1 - \xi_i - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)) - \sum_{i=1}^{n}\mu_i\xi_i \\
&= \frac{1}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i=1}^{n}\alpha_i(1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)) + C\sum_{i=1}^{n} k_i\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\mu_i\xi_i \\
&= -\frac{1}{2}\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}}\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n} k_i C\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i - \sum_{i=1}^{n}\mu_i\xi_i \\
&= -\frac{1}{2}\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}}\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}(k_i C - \alpha_i - \mu_i)\xi_i \\
&= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j
\end{aligned}$$

又因为 $\alpha_i \geq 0, \mu_i \geq 0, k_i C = \alpha_i + \mu_i$,
消去 $\mu_i$ 即可得到约束条件 $0 \leq \alpha_i \leq k_i C$.
因此对偶问题为

$$
\begin{aligned}
\max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x}_j \\
\text{s.t.} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, && 1 \le i \le n \\
& 0 \le \alpha_i \le k_i C, && 1 \le i \le n
\end{aligned}
$$

其中 $k_i = \begin{cases} 1, & y_i = +1 \\ k, & y_i = -1 \end{cases}$ 也即 $k_i = 1 - \frac{1}{2}(k-1)(y_i - 1)$.

根据 (a) 中的原优化问题所需满足的条件 $y_i(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i + b) \ge 1 - \xi_i$ 可知 KKT 条件 $\alpha_i(y_i f(\boldsymbol{x}_i) - 1 + \xi_i) = 0$, 根据 $\xi_i \ge 0$ 可知 $\mu_i \xi_i = 0$.

因此, KKT 条件要求

$$
\begin{cases}
\alpha_i \ge 0, \mu_i \ge 0, \\
y_i f(\boldsymbol{x}_i) - 1 + \xi_i \ge 0, \\
\alpha_i(y_i f(\boldsymbol{x}_i) - 1 + \xi_i) = 0, \\
\xi_i \ge 0, \mu_i \xi_i = 0.
\end{cases}
$$

# 4 习题五

## (a)

朴素贝叶斯的基本假设是" 属性条件独立性假设": 对已知类别, 假设所有属性相互独立, 即假设每个属性独立地对分类结果发生影响.

优点是能够缓解计算联合概率时的组合爆炸问题或数据上的样本稀疏问题, 能够用较少的样本就计算出置信度较高的结果; 局限是由于使用了样本属性独立性的假设, 而这个假设在实际应用中往往是不成立的, 所以如果样本属性有关联时, 朴素贝叶斯的效果不好.

朴素贝叶斯是参数化的, 因为朴素贝叶斯的参数个数在分类类数, 样本维度等给定的情况下, 参数个数不会随着样本个数的增长而增长, 也即参数空间的维度不是无限的.

## (b)

先验概率: $P(A) = \frac{4}{12} = \frac{1}{3}, P(B) = \frac{4}{12} = \frac{1}{3}, P(C) = \frac{4}{12} = \frac{1}{3}$

类别 $A$ 属性 1:

均值 $\mu_{A,1} = \frac{1}{4} \cdot (1+2+3+4) = 2.5$

方差 $\sigma^2_{A,1} = \frac{1}{4} \cdot [(1-2.5)^2 + (2-2.5)^2 + (3-2.5)^2 + (4-2.5)^2] = 1.25$

类别 $A$ 属性 2:

均值 $\mu_{A,2} = \frac{1}{4} \cdot (2+3+4+5) = 3.5$

方差 $\sigma^2_{A,1} = \frac{1}{4} \cdot [(2-3.5)^2 + (3-3.5)^2 + (4-3.5)^2 + (5-3.5)^2] = 1.25$

类别 $B$ 属性 1:

均值 $\mu_{B,1} = \frac{1}{4} \cdot (1+2+3+4) = 2.5$

方差 $\sigma^2_{B,1} = \frac{1}{4} \cdot [(1-2.5)^2 + (2-2.5)^2 + (3-2.5)^2 + (4-2.5)^2] = 1.25$

类别 $B$ 属性 2:

均值 $\mu_{B,2} = \frac{1}{4} \cdot (4+5+6+7) = 5.5$

方差 $\sigma^2_{B,2} = \frac{1}{4} \cdot [(4-5.5)^2 + (5-5.5)^2 + (6-5.5)^2 + (7-5.5)^2] = 1.25$

类别 $C$ 属性 1:

均值 $\mu_{C,1} = \frac{1}{4} \cdot (4+5+6+7) = 5.5$

方差 $\sigma^2_{C,1} = \frac{1}{4} \cdot [(4-5.5)^2 + (5-5.5)^2 + (6-5.5)^2 + (7-5.5)^2] = 1.25$

类别 $C$ 属性 2:

均值 $\mu_{C,2} = \frac{1}{4} \cdot (1+2+3+4) = 2.5$

方差 $\sigma^2_{C,2} = \frac{1}{4} \cdot [(1-2.5)^2 + (2-2.5)^2 + (3-2.5)^2 + (4-2.5)^2] = 1.25$

因此对于样本 $\boldsymbol{x} = (2,2)$ 和 $\boldsymbol{y} = (6,1)$ 有

$p_{x_1|A} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-2.5)^2}{2\times 1.25}) \approx 0.323$

$p_{x_1|B} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-2.5)^2}{2\times 1.25}) \approx 0.323$

$p_{x_1|C} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-5.5)^2}{2\times 1.25}) \approx 0.003$

$p_{x_2|A} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-3.5)^2}{2\times 1.25}) \approx 0.145$

$p_{x_2|B} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-5.5)^2}{2\times 1.25}) \approx 0.003$

$p_{x_2|C} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(2-2.5)^2}{2\times 1.25}) \approx 0.323$

$p_{y_1|A} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(6-2.5)^2}{2\times 1.25}) \approx 0.003$

$p_{y_1|B} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(6-2.5)^2}{2\times 1.25}) \approx 0.003$

$p_{y_1|C} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(6-5.5)^2}{2\times 1.25}) \approx 0.323$

$p_{y_2|A} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(1-3.5)^2}{2\times 1.25}) \approx 0.029$

$p_{y_2|B} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(1-5.5)^2}{2\times 1.25}) \approx 0.0001$

$p_{y_2|C} = \frac{1}{\sqrt{2\pi}\sqrt{1.25}} \exp(-\frac{(1-2.5)^2}{2\times 1.25}) \approx 0.145$

于是我们有

$P(A) \times p_{x_1|A} \times p_{x_2|A} \approx 0.016$

$P(B) \times p_{x_1|B} \times p_{x_2|B} \approx 0.00029$
$P(C) \times p_{x_1|C} \times p_{x_2|C} \approx 0.00029$
$P(A) \times p_{y_1|A} \times p_{y_2|A} \approx 2.59 \cdot 10^{-5}$
$P(B) \times p_{y_1|B} \times p_{y_2|B} \approx 9.59 \cdot 10^{-8}$
$P(C) \times p_{y_1|C} \times p_{y_2|C} \approx 0.016$
所以 $\boldsymbol{x} = (2, 2)$ 被分类为 $A$ 类, $\boldsymbol{y} = (6, 1)$ 被分类为 $C$ 类.

# 5 习题六

## (a)

$x$ 的信息熵: $H_x = -(0.5 \log_2 0.5 + 0.25 \log_2 0.25 + 0.25 \log_2 0.25) = 1.5$
$y$ 的信息熵: $H_y = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$
$\hat{x}$ 的信息熵: $H_{\hat{x}} = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$

可以看出, $x$ 变为 $\hat{x}$ 的过程中, 信息熵减小了, 并且存在 $x = 3$ 而 $\hat{x} = 2$ 这种情况使得系统无法恢复原信息, 因此系统是有损的.

系统会在 $x = 3$ 时产生信息损失, 当 $x = 3$ 时, 恢复得到的 $\hat{x} = 2$. 其他情况没有信息损失.

## (b)

这里我们考虑的是 $x$ 是实数, $y$ 是有限范围内的整数的情况.

记 $x \sim p(x)$, $y$ 遵循的分布列为 $Q(y = y_i) = Q_i, i = 1, 2, \cdots, m$, 则有性能指标

$$I = D + \lambda R = \int_{-\infty}^{\infty} p(x)[x - g(f(x))]^2 \mathrm{d}x - \lambda \sum_{j=1}^{m} Q_j \log_2 Q_j$$

若 $x$ 是一个有限范围内的整数, 遵循的分布列为 $P(x = x_i) = P_i, i = 1, 2, \cdots, n$, 则同理有性能指标

$$I = D + \lambda R = \sum_{i=1}^{n} P_i[x_i - g(f(x_i))]^2 - \lambda \sum_{j=1}^{m} Q_j \log_2 Q_j$$

为了让码率最小, 则信息熵最小有 $R = 0$, 即我们可以令编码器 $y = f(x) = 0$, 而 $\hat{x} = g(y) = c$, 其中 $c$ 为一个常数.

当 $c=1$ 时, 重构误差 $D=0.5\times(1-1)^2+0.25\times(2-1)^2+0.25\times(3-1)^2=1.25$

当 $c=2$ 时, 重构误差 $D=0.5\times(1-2)^2+0.25\times(2-2)^2+0.25\times(3-2)^2=0.75$

因此我们选择 $c=2$, 也即 $g(y)=2$.

我们使用公式计算得到 (a) 中的系统性能指标

$I_a=[0.5\times(1-1)^2+0.25\times(2-2)^2+0.25\times(3-2)^2]-\lambda[0.5\times\log_2 0.5+0.5\times\log_2 0.5]=0.25+\lambda$

我们再有 (b) 中的系统性能指标

$I_b=0.75-\lambda\times 0=0.75$

我们分别带入 $\lambda$ 的不同取值可得

| $I$ $\lambda$ | 0.1 | 1 | 10 |
|---|---|---|---|
| $I_a$ | 0.35 | 1.25 | 10.25 |
| $I_b$ | 0.75 | 0.75 | 0.75 |

而性能指标越小, 系统性能越好.

因此在 $\lambda=0.1$ 时 (a) 的系统性能好; 当 $\lambda=1,10$ 时 (b) 的系统性能好.

## (c)

当 $y\in\{0,1\}$ 时, 我们从信息熵的角度分析, 可知 $y$ 最大的信息熵为 $-(0.5\log_2 0.5+0.5\log_2 0.5)=1$, 一旦 $x\sim p(x)$ 的信息熵大于 1 便无法达到无损压缩了.

或者从重构 $x$ 的角度出发, 对于一个确定性的解码器 $g$, $\hat{x}=g(y)$ 只有两个取值, 因此一旦 $x$ 可能的取值大于两个 (基本所有情形都是大于两个的), 就一定会存在信息损失, 无法达到无损压缩了. (a) 中的例子中 $x$ 有三种可能的取值, 因此对于 $y\in\{0,1\}$ 无法达到无损压缩.

如果我们让 $y\in\mathbb{Z}$ 或 $y\in\mathbb{R}$, 我们有新的性能指标,

当 $y\in\mathbb{Z}$ 时, 我们仿照 (b) 中的结果, 但是 $y$ 的可能取值个数 $m$ 可以达到无穷:

$$I=D+\lambda R=\int_{-\infty}^{\infty}p(x)[x-g(f(x))]^2\mathrm{d}x-\lambda\sum_{j=1}^{\infty}Q_j\log_2 Q_j$$

当 $y\in\mathbb{R}$ 时, 我们仿照 (b) 中的结果, 而有 $y\sim q(y)$:

$$I = D + \lambda R = \int_{-\infty}^{\infty} p(x)[x - g(f(x))]^2 \mathrm{d}x - \lambda \int_{-\infty}^{\infty} q(y) \ln q(y) \mathrm{d}y$$

从表达能力上来说, $y \in \mathbb{R}$ 相较于 $y \in \mathbb{Z}$ 更强, $y \in \mathbb{R}$ 可以取连续值, 而 $y \in \mathbb{Z}$ 只能取离散值.

从编码难度上来说, $y \in \mathbb{R}$ 是连续值, 因此编码难度更高, 想要将其存储在离散状态的计算机中, 很多情况下需要用近似的方式来编码存储. 而 $y \in \mathbb{Z}$ 是离散值, 一般来说可以直接编码存储.

从编解码器设计难度上来说, 由于 $y \in \mathbb{R}$ 是连续值, 难以编码, 进而也难以设计出较为通用的编解码器, 设计起来更为复杂, 而 $y \in \mathbb{Z}$ 的编解码器设计起来就较为简单, 也存在许多较为通用的编码方法, 例如 Huffman 编码.

## (d)

可以通过机器学习自动地学习出编解码器. 我们可以将 $\hat{x} = g(f(x))$ 当成一个整体的模型, 而 $y = f(x)$ 是模型中的一层宽度为 1 的" 瓶颈", 如此我们便可以使用神经网络训练的方式同时训练编码器和解码器.

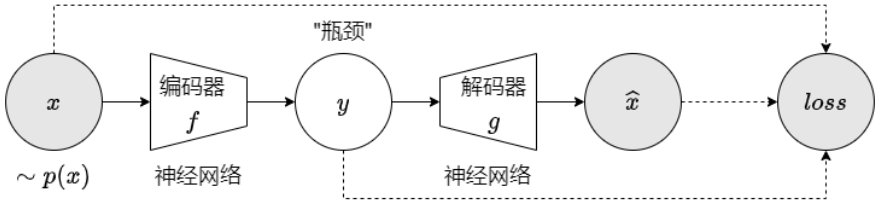


图 1: 模型

如图所示, 其中损失函数 $loss$ 需要用到 $x, y, \hat{x}$ 三者共同计算, 而损失函数的计算方法为

$$loss = MSE(x, \hat{x}) + \lambda R(y)$$

其中 $R(y)$ 为对 $y$ 码率的计算, 而 $\lambda$ 是权衡 $MSE(x, \hat{x})$ 和 $R(y)$ 的超参数.

这里我们依然可以使用信息熵的定义来计算 $y$ 的码率, 我们要求最终得到所有样本的平均误差 $\bar{R}(y)$ 等于信息熵 $\int_{-\infty}^{\infty} -q(y)\ln q(y)\mathrm{d}y$, 即要求 $-\ln q(y)$ 在 $q(y)$ 分布下的期望, 我们即应有

$$R(y) = -\ln q(y)$$

其中 $q(y)$ 为 $y$ 的连续分布密度函数. 这样一来, 问题就转换为了如何求得 $q(y)$. 我们可以假定 $q(y)$ 遵循某个分布, 例如高斯混合模型, 然后在训练过程中逐批次地迭代更新 $q(y)$ 的参数 (例如 EM 算法), 这样一来我们就可以通过信息熵来求出 $y$ 的码率, 进而优化这里的神经网络.

除了使用信息熵的方式计算 $y$ 的码率 $R(y)$, 我们也可以从码率的概念出发: 如果 $y \in \mathbb{Z}$ 的话, 那么为了让码率最小, 我们可以令

$$R(y) = \|y\|_p$$

即优化 $y$ 的 Lp 范数, 进而让 $y$ 的取值范围尽可能小, 以达到减小码率的效果. 这样一来, $\lambda R(y)$ 也就相当于一个正则化项.

常用的码率定义有 L2 范数

$$R(y) = \|y\|_2 = y^2$$

和 L1 范数

$$R(y) = \|y\|_1 = |y|$$

## (e)

分析训练过程中的 $\hat{y} = y + \epsilon$ 的分布有概率密度函数:

$f_{\hat{y}}(y) = f_{y+\epsilon}(y) = \int_{-\infty}^{\infty} f_y(y-x)f_\epsilon(x)\mathrm{d}x = \int_{-0.5}^{0.5} f_y(y-x)\mathrm{d}x$

而我们在训练过程中的 $\hat{y} = [y]$ 的分布根据取整操作 $[\cdot]$ 的定义 (即四舍五入到整数) 即有分布列

$$P(\hat{y} = y) = P([y] = y) = \begin{cases} \int_{-0.5}^{0.5} f_y(y-x)\mathrm{d}x, & y \in \mathbb{Z} \\ 0, & \text{otherwise} \end{cases}$$

因此可以看出, 在 $\hat{y}$ 为整数的情况下, 训练过程通过加性噪声得到的概率密度函数值和测试过程中的概率在数值上是一致的, 因此我们可以通过加性噪声来解决取整操作的有效梯度产生问题.

## (f)

这里我选择使用 PyTorch 来完成我们的基于神经网络的数据压缩系统.

我们先是通过构建了一个类 gmm_dataset, 用于生成 12000 个遵循 $0.25N(x;0,1)+0.75N(x;6,4)$ 的样本点, 其中 10000 个样本用于训练, 1000 个样本用于验证 (学习超参数), 1000 个样本用于测试最终性能.

神经网络使用 $x == encoder ==> y ==> yhat == decoder ==> xhat$ 架构, 而编码器和解码器使用相同的架构, 即网络是对称的, 中间存在着" 瓶颈" yhat. 编码器和解码器的神经网络使用的是通过 relu 函数激活的全连接层, hidden_size 超参数控制每一个隐层的参数个数, hidden_layers 超参数控制隐层的个数 (每个隐层包含一个全连接层和一个 relu 激活层).

基于 (e) 的结论, 在训练过程和测试过程中使用不同的方式从 y 生成 yhat, 训练过程中使用一个分布为 $U(-0.5,0.5)$ 的加性噪声 $\epsilon$, 而测试过程使用取整函数 $yhat = torch.round(y)$.

为了训练神经网络, 这里选择使用 Adam 优化器, 而损失函数为

$$loss(x) = MSE(x, g(f(x))) + \lambda|f(x)| = MSE(x,\hat{x}) + \lambda|y|$$

即超参数 lmbda 控制码率 (正则化项) 所占比例, epochs 超参数控制迭代次数, lr 超参数控制学习率.

为了确定 lmbda 的大小, 我们首先得明确我们需要什么: 我们希望知道 $MSE$ 与 $|y|$ 在数量级上占比互相平衡时的 lmbda 大小. 由不等式

$$\frac{1}{\sigma}A + \sigma B \geq 2\sqrt{AB}$$

当且仅当 $\frac{1}{\sigma}A = \sigma B$ 取得等号 (也即有最小值) 可知, 我们可以通过超参数学习优化

$$loss(x) = \frac{1}{\sigma}MSE(x,\hat{x}) + \sigma|y|$$

最后得到 $\sigma^*$ 值之后, 则有 $MSE$ 与 $|y|$ 平衡时的

$$\lambda^* = \sigma^{*2}$$

通过代码优化我们得到

$$\lambda^* = 10^{-4.4}$$

则我们可以令 $\lambda = 10 \times 10^{-4.4}$, 让码率的影响对损失函数的影响为 $MSE$ 的 10 倍.

然后我们在以下超参数选择范围内进行网格搜索:

```
config = {
    'hidden_size': [5, 10, 20, 50],
    'hidden_layers': [1, 2, 3],
    'lmbda': [10 * 10 ** -4.4],
    'checkpoint': 10,
    'epochs': 200,
    'lr': [0.001, 0.005, 0.01],
}
```

根据验证集 valid_dataset 的数据跑出来的最佳超参数为:

```
best_hidden_size: 5
best_hidden_layers: 3
best_lmbda: 0.00039810717055349697
best_epochs: 190
best_lr: 0.01
best_valid_loss: 0.008435608819127083
```

在测试集上的误差为:

```
test loss: 0.009454215876758099
```

最后的网络结构为:

```
autoencoder(
  (encoder): Sequential(
```

```
    (input): Linear(in_features=1, out_features=5, bias=True)
    (relu_input): ReLU()
    (hidden_layer_0): Linear(in_features=5, out_features=5, bias=True)
    (relu_0): ReLU()
    (hidden_layer_1): Linear(in_features=5, out_features=5, bias=True)
    (relu_1): ReLU()
    (hidden_layer_2): Linear(in_features=5, out_features=5, bias=True)
    (relu_2): ReLU()
    (output): Linear(in_features=5, out_features=1, bias=True)
  )
  (decoder): Sequential(
    (input): Linear(in_features=1, out_features=5, bias=True)
    (relu_input): ReLU()
    (hidden_layer_0): Linear(in_features=5, out_features=5, bias=True)
    (relu_0): ReLU()
    (hidden_layer_1): Linear(in_features=5, out_features=5, bias=True)
    (relu_1): ReLU()
    (hidden_layer_2): Linear(in_features=5, out_features=5, bias=True)
    (relu_2): ReLU()
    (output): Linear(in_features=5, out_features=1, bias=True)
  )
)
```

我们尝试使用一些常见数据, 可以看出在 $\hat{y}$ 为整数的情况下, 最后拟合的结果也仍然不错:

```
x = -1.0, yhat = 29.0, xhat = -1.0349719524383545
x = 0.5, yhat = 18.0, xhat = 0.48566997051239014
x = 1.25, yhat = 13.0, xhat = 1.1768709421157837
x = 4.5, yhat = -0.0, xhat = 4.4466633796691895
x = 6.0, yhat = -9.0, xhat = 6.080208778381348
x = 6.5, yhat = -11.0, xhat = 6.443219184875488
```

即使是非常偏离原始数据分布的数据, 也仍然能有不错的拟合效果:

```
x = -100.0, yhat = 263.0, xhat = -6.937225818634033
```

x = −50.0, yhat = 152.0, xhat = −5.142919063568115
x = 50.0, yhat = −189.0, xhat = 38.75682830810547
x = 100.0, yhat = −381.0, xhat = 73.6153793334961

而我们如果令 $\lambda = 0$, 则会得到如下结果:

x = −1.0, yhat = 7.0, xhat = −0.9560794234275818
x = 0.5, yhat = −12.0, xhat = 0.5415831804275513
x = 1.25, yhat = −21.0, xhat = 1.2605472803115845
x = 4.5, yhat = −87.0, xhat = 4.493512153625488
x = 6.0, yhat = −120.0, xhat = 5.989358425140381
x = 6.5, yhat = −131.0, xhat = 6.487973213195801
x = −100.0, yhat = 229.0, xhat = −4.962228775024414
x = −50.0, yhat = 131.0, xhat = −4.036450386047363
x = 50.0, yhat = −1010.0, xhat = 46.33588790893555
x = 100.0, yhat = −2007.0, xhat = 91.53396606445312

可以看出 $\lambda|y|$ 对码率存在着不错的限制效果, 如果不加入正则化项则会导致 yhat 的取值范围较大.