

2025 模式识别

作业二

人工智能学院 221300079 王俊童

2025.3.25

221300079 王俊童, 人工智能学院

1 问题一

- a. 对于这个形式化问题, 根据题目的描述如下: γ_{ij} 表示第 j 个样本 (共 M 个) 被分到了第 i 类别 (共 K 类)。而 μ_i 代表这个类别的均值。那很显然的一个事情是, 我们的分类标准应该是类内是要尽量小的, 意思是离均值点的距离应当近。因此我们的目标是, 对于每一个类别都要做到这一点, 那对应样本应该被分进正确的类别以达到最小, 问题形式化如下:

$$\mathcal{L} = \sum_{i=1}^K \sum_{j=1}^M \gamma_{ij} \|x_j - \mu_i\|$$

那么优化函数就可以写为题目说的那个样子:

$$\arg \min_{\gamma_{ij}, \mu_i} \sum_{i=1}^K \sum_{j=1}^M \gamma_{ij} \|x_j - \mu_i\| \quad (1)$$

QED, Eqn. 1

- b. 根据题目描述, 第一步应该是 μ_i 被固定, 那么优化问题就变成了对于 M 个样本, 我们要做到优化对于每个样本来说:

$$\mathcal{L}_j = \sum_{i=1}^K \gamma_{ij} \|x_j - \mu_i\|^2$$

要做到上述的 j 个 (共 M 个式子) 最小, 意思是属于某一个类别。那么对于这个东西, 相当于求一个

$$\arg \min_i \|x_j - \mu_i\|^2$$

只要找到了最小的, 其他的标识变量都是 0 了。所以 γ_{ij} 可以重写为:

$$\gamma_{ij} = \begin{cases} 1, & \text{if } dist = \min_i \|x_j - \mu_i\|^2 \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

对于第二步骤, 其实是固定了类别之后, 需要重新找一个分类的最小均值。此时问题变为:

$$\mathcal{L}_i = \sum_{j=1}^M \gamma_{ij} \|x_j - \mu_i\|^2$$

可以求导:

$$\frac{\partial \mathcal{L}}{\partial \mu_i} = 0$$

可以解得:

$$\mu_i = \frac{\sum_{j=1}^M \gamma_{ij} x_j}{\sum_{j=1}^M \gamma_{ij}} \quad (3)$$

所以这个题目的更新规则如 Eqn. b.,Eqn. 3所示。

c. 证明敛散性, 相当于证明, 新得到的 $\mathcal{L}' - \mathcal{L}$ 跟 0 的关系:

对于步骤 1, 样本找到了一个新的类别, 意思是找到了一个更小的均值点和他的距离, 那么问题可以写为:

$$\begin{aligned} \mathcal{L}' - \mathcal{L} &= \sum_{i=1}^K \sum_{j=1}^M (\gamma'_{ij} - \gamma_{ij}) \|x_j - \mu'_i\|^2 \\ \mathcal{L}' - \mathcal{L} &= \sum_{i=1}^K \sum_{j=1}^M (\|x_j - \mu'_i\|^2 - \|x_j - \mu_i\|^2) \leq 0 \end{aligned}$$

显然, 根据刚才的证明, 他会更小, 所以一定成立。

对于步骤 2, 只针对每一个类别来说, 切换新的均值点, 不增加计算成本, 那么考虑一个类就行了:

$$\begin{aligned} \mathcal{L}' - \mathcal{L} &\sim \sum_{j=1}^M (\|x_j - \mu'_i\|^2 - \|x_j - \mu_i\|^2) \\ \mathcal{L}' - \mathcal{L} &\sim \sum_{j=1}^M (\mu_i - \mu'_i)^\top (2x_j - \mu_i - \mu'_i) \end{aligned}$$

将 Eqn. 3代入可以化简得到:

$$\mathcal{L}' - \mathcal{L} \sim -(\mu_i - \mu'_i)^\top (\mu_i - \mu'_i) \leq 0$$

那么, 根据证明, 确实收敛。

2 问题二

a. 把题目说的形式带进去可以了:

$$\begin{aligned} \sum_{i=1}^n \epsilon_i^2 &= \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \beta)^2 \\ \arg \min_{\beta} \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \beta)^2 \end{aligned}$$

b. 可以写成矩阵形式的优化问题:

$$\arg \min_{\beta} (\mathbf{y} - \mathbf{X}\beta)^\top (\mathbf{y} - \mathbf{X}\beta)$$

矩阵形式的优化问题就是这样。

c. 可逆的话, 那么用求导等于 0 就好了:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \beta} &= 2\mathbf{X}^\top (\mathbf{X}\beta - \mathbf{y}) = 0 \\ \beta^* &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} \end{aligned}$$

d. 因为维度 d 比 n 大, 第二问说 X 是 n*d 的, 那么 $\text{rank}(\mathbf{X}) \leq n < d$ 可以得到, 那么 $\text{rank}(\mathbf{X}^\top \mathbf{X}) \leq n < d$ 可以得到。但是, $\mathbf{X}^\top \mathbf{X}$ 这个东西是 d*d 的, 所以一定不满秩。那就不可逆了

e. 在正则化项加入了之后，这个东西一般不可逆的，就可以求解了。而且一般模型都没有闭式解，这种情况下引入正则化项既降低了复杂度，也同时使得式子可以求解，这样就有闭式解了，一般为复杂度最低的，感觉也算是一种归纳偏好吧。

f. 引入正则化之后的优化目标如下：

$$\arg \min_{\beta} (\mathbf{y} - \mathbf{X}\beta)^{\top} (\mathbf{y} - \mathbf{X}\beta) + \lambda \beta^{\top} \beta$$

求导化简：

$$\frac{\partial \mathcal{L}}{\partial \beta} = 2\mathbf{X}^{\top} (\mathbf{X}\beta - \mathbf{y}) + 2\lambda \beta = 0$$

$$\beta^* = (\mathbf{X}^{\top} \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^{\top} \mathbf{y}$$

g. 上面说了，如果不可逆的时候，这样就解决不了了，因为没有唯一的闭式解了。但是这个东西加入之后，那个 $\lambda \mathbf{I}$ 可以让基本所有情况都可以解。这个时候就可以选择了。

h. $\lambda = 0$, 就变成之前的普通线性回归, $\lambda = \infty$, 这个只能解后面的 $\beta^{\top} \beta = 0$, 答案就是 0.

i. 我觉得不行，假如我们可以找到最优解 λ^*, β^* ，那么原来的运算可以写作：

$$(\mathbf{y} - \mathbf{X}\beta^*)^{\top} (\mathbf{y} - \mathbf{X}\beta^*) + \lambda^* \beta^{*\top} \beta^*$$

但是你会发现一个非常严肃的问题，当没有正则化的时候，一定存在：

$$(\mathbf{y} - \mathbf{X}\beta^*)^{\top} (\mathbf{y} - \mathbf{X}\beta^*) < (\mathbf{y} - \mathbf{X}\beta^*)^{\top} (\mathbf{y} - \mathbf{X}\beta^*) + \lambda^* \beta^{*\top} \beta^*$$

那不是，我令为 0 不就行了，所以没啥用。

3 问题三

a. 表格如下：

Table 1: Performance Metrics Table						
下标	类别标记	得分	查准率	查全率	AUC-PR	AP
0	-	-	1.0000	0.0000	-	-
1	1	1.0	1.0000	0.2000	0.2000	0.2000
2	2	0.9	0.5000	0.2000	0.0000	0.0000
3	1	0.8	0.6667	0.4000	0.1167	0.1333
4	1	0.7	0.7500	0.6000	0.1417	0.1500
5	2	0.6	0.6000	0.6000	0.0000	0.0000
6	1	0.5	0.6667	0.8000	0.1267	0.1333
7	2	0.4	0.5714	0.8000	0.0000	0.0000
8	2	0.3	0.5000	0.8000	0.0000	0.0000
9	1	0.2	0.5556	1.0000	0.1056	0.1111
10	2	0.1	0.5000	1.0000	0.0000	0.0000
-	-	-	-	-	0.6907	0.7277

b. 由于 AP 的算法为

$$AP = \sum_{i=1}^n (r_i - r_{i-1}) * p_i$$

而 AUC-PR 为：

$$AUC - PR = \sum_{i=1}^n (r_i - r_{i-1}) * \frac{p_i + p_{i-1}}{2}$$

做个减法:

$$AP - AUC - PR = \sum_{i=1}^n (r_i - r_{i-1}) * \frac{p_i - p_{i-1}}{2}$$

一般情况下 $p_i \geq p_{i-1}$ 所以确实 AP 会比 AUC 大一些。

c. 我们通过计算可以得到新的 AUC 为 0.6794, 新的 AP 为 0.7167. 其实就是跟先进入正样本还是负样本的顺序有区别罢了。

d. 计算得到:AUC-PR: 0.6906, AP: 0.7278. 说明我们算对了。图像如下: 代码如下:

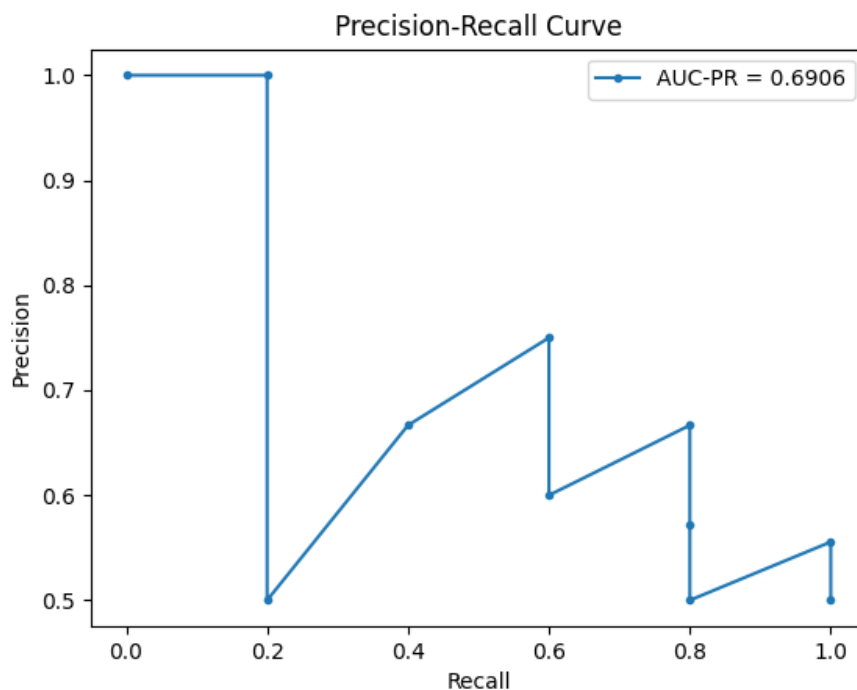


Figure 1: auc

Listing 1: Python 代码示例

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import precision_recall_curve, auc, average_precision_score

data = [
    (0, 1, 1.0),
    (1, 2, 0.9),
    (2, 1, 0.8),
    (3, 1, 0.7),
    (4, 2, 0.6),
    (5, 1, 0.5),
    (6, 2, 0.4),
    (7, 2, 0.3),
    (8, 1, 0.2),
]
```

```
(9, 2, 0.1)
]

true_labels = np.array([label for _, label, _ in data])
scores = np.array([score for _, _, score in data])

binary_labels = (true_labels == 1).astype(int)
precision, recall, _ = precision_recall_curve(binary_labels, scores)
auc_pr = auc(recall, precision)
ap = average_precision_score(binary_labels, scores)

print(f"AUC-PR: {auc_pr:.4f}")
print(f"AP: {ap:.4f}")

plt.figure()
plt.plot(recall, precision, marker='.', label=f'AUC-PR={auc_pr:.4f}')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title('Precision-Recall Curve')
plt.legend()
plt.savefig('pr.png')
plt.show()
```

4 问题四

a. 1