《大数据机器学习》第 4 次作业

姓名：刘培源　　学号：2023214278

题目 1：用贝叶斯估计法推出朴素贝叶斯法中的概率估计公式1及公式2。

$$P_\lambda(X^{(j)} = a_{jl}|Y = c_k) = \frac{\sum\limits_{i=1}^{N} I(x_i^{(j)} = a_{jl}, y_i = c_k) + \lambda}{\sum\limits_{i=1}^{N} I(y_i = c_k) + S_j\lambda} \tag{1}$$

$$P_\lambda(Y = c_k) = \frac{\sum\limits_{i=1}^{N} I(y_i = c_k) + \lambda}{N + K\lambda} \tag{2}$$

答：

公式2。给定数据集 $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$，我们可以按照以下步骤推导和润色模型。

1. **观测模型**：对于一个观测 y，其服从多项式分布。其概率质量函数为：

   $$P(Y = y) = \frac{N!}{m_1!m_2!...m_K!} \prod_{k=1}^{K} u_k^{m_k}$$

   其中，m_k 表示 Y 取值为 c_k 的次数，$\sum_{k=1}^{K} m_k = N$。

2. **先验概率**：对于多项式分布的参数 $u = (u_1, u_2, ..., u_K)^T$，其先验分布服从 Dirichlet 分布，表示为：

   $$P(u) = P(u_1, u_2, ..., u_K) = C(\lambda) \prod_{k=1}^{K} u_k^{\lambda_k - 1}$$

   其中，$C(\lambda)$ 是归一化常数。

3. **似然函数**：给定样本 $m = (m_1, m_2, ..., m_K)^T$，其似然函数为：

   $$P(m|u) = u_1^{m_1} \cdot u_2^{m_2} \cdot ... \cdot u_K^{m_K} = \prod_{k=1}^{K} u_k^{m_k}$$

4. **后验概率**：结合先验概率和似然函数，我们可以得到后验概率：

   $$P(u|m) = \frac{P(m|u)P(u)}{P(m)} \propto P(m|u)P(u) \propto \prod_{k=1}^{K} u_k^{\lambda_k + m_k - 1}$$

5. **期望**：最后，我们可以计算参数 u_k 的期望值。使用贝叶斯公式，我们得到：

$$E(u_k) = \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} = \frac{\lambda + m_k}{K\lambda + N} = \frac{\lambda + m_k}{K\lambda + \sum_{i=1}^{N} I(y_i = c_k)}$$

其中，$\alpha_k = \lambda + m_k$。

6. **结论**：观测 Y 的后验概率为：

$$P(Y = c_k) = \frac{\sum_{i=1}^{N} I(y_i = c_k) + \lambda}{N + K\lambda}$$

公式1。给定数据集 $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$，我们可以按照以下步骤推导和润色模型。

1. **观测模型**：观测到的 y 遵循多项式分布，概率质量函数为：

$$P(Y = y) = \frac{N!}{m_1!m_2!...m_K!} \prod_{k=1}^{K} u_k^{m_k}$$

其中，m_k 表示 Y 取值为 c_k 的次数，且有 $\sum_{k=1}^{K} m_k = N$。

2. **先验概率**：多项式分布的参数向量 $u = (u_1, u_2, ..., u_K)^T$ 的先验分布为 Dirichlet 分布，表示为：

$$P(u) = P(u_1, u_2, ..., u_K) = C(\lambda) \prod_{k=1}^{K} u_k^{\lambda_k - 1}$$

其中，$C(\lambda)$ 是归一化常数。

3. **似然函数**：给定样本集合 $m = (m_1, m_2, ..., m_K)^T$，似然函数可表示为：

$$P(m|u) = u_1^{m_1} \cdot u_2^{m_2} \cdot ... \cdot u_K^{m_K} = \prod_{k=1}^{K} u_k^{m_k}$$

4. **后验概率**：结合先验概率与似然函数，我们可以得到后验概率：

$$P(u|m) = \frac{P(m|u)P(u)}{P(m)} \propto P(m|u)P(u) \propto \prod_{k=1}^{K} u_k^{\lambda_k + m_k - 1}$$

5. **期望**：计算参数 u_k 的期望值，使用贝叶斯定理得到：

$$E(u_k) = \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k} = \frac{\lambda + m_k}{K\lambda + N} = \frac{\lambda + m_k}{K\lambda + \sum_{i=1}^{N} I(y_i = c_k)}$$

这里，$\alpha_k = \lambda + m_k$。

6. **结论**：观测 Y 的后验概率为：

$$P(Y = c_k) = \frac{\sum_{i=1}^{N} I(y_i = c_k) + \lambda}{N + K\lambda}$$

题目 2： 已知如表1所示的训练数据，试用平方误差损失准则生成一个二叉回归树。

x_i	1	2	3	4	5	6	7	8	9	10
y_i	4.50	4.75	4.91	5.34	5.80	7.05	7.90	8.23	8.70	9.00

表 1: 训练数据表

答： 基于书中的算法，写出平方误差损失准则的二叉回归数代码，并带入数据，如下：

```
import numpy as np
class LeastSquaresRegressionTree:
    def __init__(self, train_X, y, epsilon):
        self.train_X = train_X
        self.y = y
        self.num_features = train_X.shape[1]
        self.epsilon = epsilon
        self.tree = None

    def _fit(self, X, y, num_features, epsilon):
        j, s, min_loss, c1, c2 = self._find_optimal_split(X, y, num_features)
        node = {"feature": j, "split_point": X[s, j], "left": None, "right": None}

        if min_loss < epsilon or len(y[X[:, j] <= X[s, j]]) <= 1:
            node["left"] = c1
        else:
            node["left"] = self._fit(X[X[:, j] <= X[s, j]], y[X[:, j] <= X[s, j]], num_features, epsilon)

        if min_loss < epsilon or len(y[X[:, j] > X[s, j]]) <= 1:
            node["right"] = c2
        else:
            node["right"] = self._fit(X[X[:, j] > X[s, j]], y[X[:, j] > X[s, j]], num_features, epsilon)
        return node

    def fit(self):
        self.tree = self._fit(self.train_X, self.y, self.num_features, self.epsilon)

    @staticmethod
    def _find_optimal_split(X, y, num_features):
        cost = np.zeros((num_features, len(X)))

        for i in range(num_features):
            for k in range(len(X)):
                value = X[k, i]
                y_left, y_right = y[X[:, i] <= value], y[X[:, i] > value]
                c1, c2 = np.mean(y_left), np.mean(y_right)
                y_left, y_right = y_left - c1, y_right - c2
                cost[i, k] = np.sum(y_left**2) + np.sum(y_right**2)
        i, k = np.unravel_index(cost.argmin(), cost.shape)
        c1 = np.mean(y[X[:, i] <= X[k, i]])
        c2 = np.mean(y[X[:, i] > X[k, i]])
        return i, k, cost[i, k], c1, c2

train_X = np.array([[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]]).T
y = np.array([4.50, 4.75, 4.91, 5.34, 5.80, 7.05, 7.90, 8.23, 8.70, 9.00])

model = LeastSquaresRegressionTree(train_X, y, .2)
model.fit()
print(model.tree)
```

最终得到结果如下：

$$f(x)=\begin{cases} 4.72 & \text{if } x\leq 3 \\ 5.57 & \text{if } 3<x\leq 5 \\ 7.05 & \text{if } 5<x\leq 6 \\ 7.9 & \text{if } 6<x\leq 7 \\ 8.23 & \text{if } 7<x\leq 8 \\ 8.85 & \text{if } x>8 \end{cases} \tag{3}$$

题目 3： 证明 CART 剪枝算法中，当 α 确定时，存在唯一的最小子树 T_α 使损失函数 $C_\alpha(T)$ 最小。

答： 对于任何内部节点，其是否进行剪枝仅与以该节点为根的子树有关。为了理解剪枝的过程，定义子树的损失函数为：

$$C_\alpha(T) = C(T) + \alpha \times |T|$$

其中：$C(T) = \sum_{t=1}^{|T|} N_t \left(1 - \sum_{k=1}^{K} \left(\frac{N_{tk}}{N_t}\right)^2\right)$ 是子树的经验损失，$|T|$ 是叶节点的数量，K 是数据的类别数量。对于给定的 α，如果剪枝前后的子树分别为 T_0 和 T_1，当满足 $C_\alpha(T_1) \le C_\alpha(T_0)$ 时，我们会选择剪枝。

采用反证法。假设当 α 确定时，存在两颗不同的子树，分别为 T_1 和 T_2，它们都使得损失函数 C_α 达到最小值。我们会有以下两种情况：

1 假定两颗子树被剪枝的部分位于相同的方向。在这种情况下，其中一颗子树将是另一颗子树剪枝后得到的结果。因此，它们不可能同时是最优的，这与我们的假设矛盾。

2 假定两颗子树被剪枝的部分位于不同的方向。然而，由于它们都使损失函数 C_α 达到最小值，这意味着它们都可以继续进行剪枝操作。这与存在两个最优子树的假设相矛盾。

基于上述的反证法，可以得出结论：当 α 确定时，只能存在唯一的最小子树 T_α 使得损失函数 $C_\alpha(T)$ 最小。

题目 4： 已知正例点 $x_1=(1,2)^T$，$x_2=(2,3)^T$，$x_3=(3,3)^T$，负例点 $x_4=(2,1)^T$，$x_5=(3,2)^T$，试求最大间隔分离超平面和分类决策函数，并在图上画出分离超平面、间隔边界及支持向量。

答：

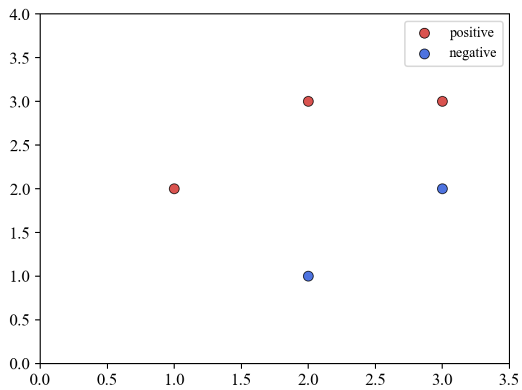

图 1: 可视化训练数据。其中，红色点是正例，蓝色点是反例。

由图可知，正例点的支持向量可能是 $x_1=(1,2)^T$ 或 $x_3=(3,3)^T$，负点的支持向量可能是 $x_5=(3,2)^T$，因此分类讨论：

1. 假设支持向量分别是 $x_1=(1,2)^T$ 和 $x_5=(3,2)^T$。考虑支持向量 $x_1=(1,2)^T$ 和 $x_5=(3,2)^T$，正例和负例满足：

$$\mathbf{w}\cdot(1,2)^T+b=1,$$
$$\mathbf{w}\cdot(3,2)^T+b=-1.$$

   可以解得决策函数为：$f(x)=\mathrm{sign}(-x_1+2x_2-2)$

2. 假设支持向量分别是 $x_3=(3,3)^T$ 和 $x_5=(3,2)^T$。考虑支持向量 $x_1=(1,2)^T$ 和 $x_5=(3,2)^T$，正例和负例满足：

$$\mathbf{w}\cdot(3,3)^T+b=1,$$
$$\mathbf{w}\cdot(3,2)^T+b=-1.$$

   可以解得决策函数为：$f(x)=\mathrm{sign}(-x_1+2x_2-2)$

所以支持向量为是 $x_1=(1,2)^T$，$x_3=(3,3)^T$ 和 $x_5=(3,2)^T$，决策函数为 $f(x)=\mathrm{sign}(-x_1+2x_2-2)$。

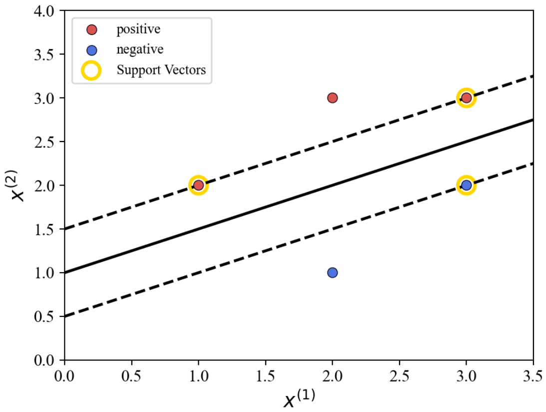

图 2: 可视化结果。其中，红色点是正例，蓝色点是反例，黑线是决策函数，黄色圈是支持向量。

题目 5： 证明内积的正整数幂函数:

$$K(x, z) = (x \cdot z)^p$$

是正定核函数，这里 p 是正整数，$x, z \in \mathbf{R}^n$。

答： 考虑书中的定理（7.5），如图3所示，我们的目标是证明核函数 $K(x, z) = (x \cdot z)^p$ 对应的 Gram 矩阵 $K = [K(x_i, x_j)]_{m \times m}$ 是半正定矩阵。

定理 7.5（正定核的充要条件） 设 $K : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$ 是对称函数，则 $K(x, z)$ 为正定核函数的充要条件是对任意 $x_i \in \mathcal{X}$，$i = 1, 2, \cdots, m$，$K(x, z)$ 对应的 Gram 矩阵：

$$K = [K(x_i, x_j)]_{m \times m} \tag{7.85}$$

是半正定矩阵。

图 3: 正定核的充要条件

假设 $\mathbf{c} = \{c_1, c_2, \cdots, c_m\} \in \mathbf{R}^m$，可以得到：

$$\begin{aligned}
\sum_{i,j=1}^m c_i c_j K(x_i, x_j) &= \sum_{i,j=1}^m c_i c_j (x_i \cdot x_j)^p \\
&= (\sum_{i=1}^m c_i x_i)(\sum_{j=1}^m c_j x_j)(x_i \cdot x_j)^{p-1} \\
&= \|\sum_{i=1}^m c_i x_i\|^2 (x_i \cdot x_j)^{p-1}
\end{aligned}$$

注意到，对于任意向量 v，有 $\|v\|^2 \geq 0$。由于 p 是正整数，$p - 1 \geq 0$。进而可以得知，$(x_i \cdot x_j)^{p-1} \geq 0$。因此，$\sum\limits_{i,j=1}^m c_i c_j K(x_i, x_j)$ 的值为非负数，即 $K(x, z) = (x \cdot z)^p$ 对应的 Gram 矩阵是半正定的，得证。