ML Lab 3

XGBoost

王世炟

PB20151796

November 20, 2022

Part 1: 实验要求

本次实验要求手动实现 XGBoost（eXtreme Gradient Boosting），并在给定数据集上进行训练和验证/测试。具体需要完成以下部分：

- 读取训练数据集并自行划分验证集
- 读取训练数据集并自行划分验证集
- 设置模型停止运行的标准，决策树节点停止划分的标准
- 在训练集上训练模型，进行参数优化，精度达到某一程度停止，并得到损失函数曲线
- 在数据集上进行模型的优化与参数选择

Part 2: 数据集介绍

本次实验的数据集每个样本有 40 个特征，特征都是连续的，预测值也是连续的。

	0	1	2	3	4	5	6	7	8	9	...	31	32	33	34	35	36	37	38	39	40
0	2	-56	-0.33	-0.09	0.90	0.2	-11	12	0.004	-0.1	...	0.0	0.0	0.0	0.0	0.0	0.000	0.0	0.9	0.032	-0.0009
1	470	-39	0.02	0.12	0.39	-0.6	-12	8	0.009	-1.6	...	0.0	0.0	0.0	0.0	0.0	0.000	0.0	0.9	0.034	-0.0011
2	165	4	0.14	0.14	0.78	0.4	-11	-9	-0.003	-0.2	...	0.0	0.0	0.0	0.0	0.0	0.000	0.0	1.0	0.034	-0.0012
3	-113	5	-0.12	0.11	1.06	0.6	-10	-7	-0.008	0.0	...	0.0	0.0	0.0	0.0	0.0	0.000	0.0	0.9	0.033	-0.0011
4	-411	-21	-0.17	0.07	1.33	-0.6	-11	0	0.002	0.1	...	0.0	0.0	0.0	0.0	0.0	-0.002	0.0	0.9	0.032	-0.0008

图 1: Features of the data set

Part 3: **理论基础**

3.1 XGBoost

XGBoost 是 boosting 族中的算法，遵从前向分步加法，是由多个基模型组成的一个加法模型，假设第 k 个基本模型是 $f_k(x)$, 那么前 t 个模型组成的模型的输出为

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

其中 x_i 为第表示第 i 个训练样本，y_i 表示第 i 个样本的真实标签; $\hat{y}_i^{(t)}$ 表示前 t 个模型对第 i 个样本的标签最终预测值。
在学习第 t 个基模型时，XGBoost 要优化的目标函数为:

$$\begin{aligned} Obj^{(t)} &= \sum_{i=1}^{n} loss(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{t} penalty(f_k) \\ &= \sum_{i=1}^{n} loss(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \sum_{k=1}^{t} penalty(f_k) \\ &= \sum_{i=1}^{n} loss(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + penalty(f_t) + constant \end{aligned}$$

其中 n 表示训练样本的数量, $penalty(f_k)$ 表示对第 k 个模型的复杂度的惩罚项, $loss(y_i, \hat{y}_i^{(t)})$ 表示损失函数,
例如二分类问题的

$$loss(y_i, \hat{y}_i^{(t)}) = -y_i \cdot \log\ p(\hat{y}_i^{(t)} = 1|x_i) - (1-y_i) \log\ (1 - p(\hat{y}_i^{(t)} = 1|x_i))$$

回归问题

$$loss(y_i, \hat{y}_i^{(t)}) = (y_i - \hat{y}_i^{(t)})^2$$

将 $loss(y_i, y_i^{(t-1)} + f_t(x_i))$ 在 $y_i^{(t-1)}$ 处泰勒展开可得

$$loss(y_i, y_i^{(t-1)} + f_t(x_i))\ loss(y_i, y_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)$$

其中 $g_i = \frac{\partial\ loss(y_i, y_i^{(t-1)})}{\partial\ y_i^{(t-1)}}$, $h_i = \frac{\partial^2 loss(y_i, y_i^{(t-1)})}{\partial\ (y_i^{(t-1)})^2}$
，即 g_i 为一阶导数，h_i 为二阶导数。
此时的优化目标变为

$$Obj^{(t)} = {}_{i=1}^{n}[loss(y_i, y_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + penalty(f_t) + constant$$

去掉常数项 $loss(y_i, y_i^{(t-1)})$ (学习第 t 个模型时候，$loss(y_i, y_i^{(t-1)})$ 也是一个固定值) 和 constant，可得目标函数为

$$Obj^{(t)} = \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + penalty(f_t)$$

下面解决回归问题，即用基模型 $f(x_i)$ 拟合标签数据。那么

$$loss(y_i, \hat{y}_i^{(t-1)}) = (y_i - \hat{y}_i^{(t-1)})^2$$

则

$$g_i = \frac{\partial loss(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}} = -2(y_i - \hat{y}_i^{(t-1)}) \qquad h_i = h_i = \frac{\partial^2 loss(y_i, y_i^{(t-1)})}{\partial\ (y_i^{(t-1)})^2} = 2$$

所以我们只需要求出每一步损失函数的一阶导和二阶导的值（由于前一步的是已知的，所以这两个值是常数)，然后最优化目标函数，就可以得到每一步的，最后根据加法模型得到一个整体模型。

3.2 回归树

本实验中，我们以决策树（回归树）为基，因此还需要写出决策树的算法。

假设决策树有 T 个叶子节点，每个叶子节点对应有一个权重。决策树模型就是将输入 x_i 映射到某个叶子节点，决策树模型的输出就是这个叶子节点的权重，即 $f(x_i) = w_{q(x_i)}$ ，w 是一个要学的 T 维的向量其中 $q(x_i)$ 表示把输入 x_i 映射到的叶子节点的索引。例如：$q(x_i) = 3$，那么模型输出第三个叶子节点的权重，即 $f(x_i) = w_3$。

我们对于某一棵决策树，他的惩罚为

$$penalty(f) = \gamma \cdot T + \frac{1}{2}\lambda \cdot \|w\|^2$$

其中 γ, λ 为我们可调整的超参数，T 为叶子数，w 为权重向量. 由于显示问题，$\|w\|$ 实际上为 w 的范数，且 $\|w\|^2 = \sum_{i=1}^{dim} w_i^2$

我们将分配到第 j 个叶子节点的样本用 I_j 表示，即 $I_j = \{i|q(x_i) = j\}(1\ j\ T)$。

综上，我们在树结构确定（你可以自行确定）时，可以进行如下优化：

$$\begin{aligned}
Obj^{(t)} &= \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + penalty(f_t) \\
&= \sum_{i=1}^{n}[g_i w_{q(x_i)} + \frac{1}{2}h_i w_{q(x_i)}^2] + \gamma \cdot T + \frac{1}{2}\lambda \cdot \|w\|^2 \\
&= \sum_{j=1}^{T}[(\sum_{i \in I_j} g_i)\ w_j + \frac{1}{2} \cdot (\sum_{i \in I_j} h_i + \lambda) \cdot w_j^2] + \gamma \cdot T
\end{aligned}$$

简单起见，我们简记 $G_j = \sum_{i \in I_j} g_i, H_j = \sum_{i \in I_j} h_i$

$$Obj^{(t)} = \sum_{j=1}^{T}[G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2] + \gamma T$$

在上述推导之后，可以推导出最优的权重（对 w_j 优化）这里注意 G_j 和 H_j 是前 $t-1$ 步得到的结果，其值为常数，只有最后一棵树的叶子节点 w_j 不确定。
令 $\frac{\partial Obj^{(t)}}{w_j} = 0$ 可以得到 w_j 的闭式解 $w_j^* = -\frac{-G_j}{H_j+\lambda}$
将 w_j^* 带回 $Obj^{(t)}$ 可得：

$$Obj^{(t)} = -\frac{1}{2}\frac{G^2}{H+\lambda} + \gamma T$$

Part 4: 实验步骤

4.1 回归树

求出了每个叶子节点的权重和整颗树对应的目标值后，可以度量树的好坏程度。根据划分前后的收益确定节点的划分。假设划分前，该节点包含了若干个训练样本，要将训练样本划分为两部分，分别形成左孩子和右孩子。

构造过程

对于每一棵决策树，即每一个基的训练，我们可以按照以下步骤划分结点
1. 从根节点开始递归划分，初始情况下，所有的训练样本 x_i 都分配给根节点。
2. 根据划分前后的收益划分结点，收益为

$$Gain = Obj_P - Obj_L - Obj_R = (\frac{1}{2}\frac{G_L^2}{H_L+\lambda} + \frac{1}{2}\frac{G_R^2}{H_R+\lambda}) - \frac{1}{2}\frac{G^2}{H+\lambda} - \gamma$$

其中 Obj_P 为父结点的得分，Obj_L, Obj_R 为左右孩子的得分.
3. 选择最大增益进行划分
选择最大增益的过程如下：
1. 选出所有可以用来划分的特征集合 $\mathcal{F}$；
2. For feature in $\mathcal{F}$:
3. 将节点分配到的样本的特征 feature 提取出来并升序排列，记作 sorted_f_value_list；
4. For f_value in sorted_f_value_list ：
5. 在特征 feature 上按照 f_value 为临界点将样本划分为左右两个集合；
6. 计算划分后的增益；
7. 返回最大的增益所对应的 feature 和 f_value。

停止策略：

- 若划分后增益小于某个阈值则停止划分；（本次实验选择的阈值为 0）
- 划分后树的深度大于某个阈值停止划分；（对于该参数进行了调整，见下文）
- 该节点分配到的样本数目小于某个阈值停止分化。（本次实验选择的阈值为 50）
由于后两个停止策略大体上相关，所以仅对于第二个参数进行调整。

4.2 XGBoost

训练第 k+1 颗树时，将前 k 颗树的结果作为输入传递给决策树。
停止策略：
- 学习 M 个颗决策树后停下来；（对于该参数进行了调整，见下文）

Part 5: 实验结果

数据划分 8:2

5.1 参数对比

模型中可调参数有：- γ: 树节点个数的惩罚系数
- λ: 权重 w 的系数
- max_epoch: 最大的树的颗数
- max_depth: 树的最大深度

在控制其他参数不变的条件下，对于不同的 γ 进行测试，得到如下结果:

表 1: $\lambda = 1$, $max\_epoch = 3$, $max\_depth = 3$ 条件下不同 γ 对于测试集 MSE 的影响

γ	10	1	0.1	0.01	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
测试集 MSE(10^{-7})	1.815	1.815	1.815	1.815	1.815	1.099	0.565	0.504	0.503	0.503

绘出折线图如下：

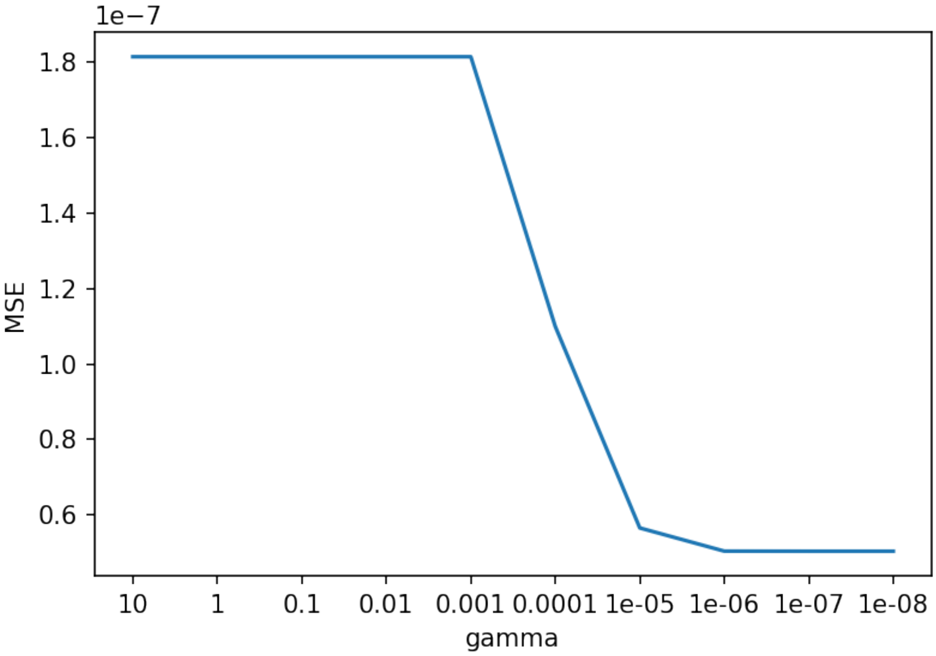

图 2: γ –MSE 折线图

对于上述现象的解释：

$$Gain = Obj_P - Obj_L - Obj_R = (\frac{1}{2}\frac{G_L^2}{H_L+\lambda} + \frac{1}{2}\frac{G_R^2}{H_R+\lambda}) - \frac{1}{2}\frac{G^2}{H+\lambda} - \gamma$$

由于结点划分时的增益为上式，最后减去了 γ , 所以若 γ 过大，则增益几乎不会大于 0，这样就会导致结点不会继续划分，即欠拟合，所以 γ 要取合适的值，大概与

$$\frac{1}{2}\frac{G_L^2}{H_L+\lambda} + \frac{1}{2}\frac{G_R^2}{H_R+\lambda}) - \frac{1}{2}\frac{G^2}{H+\lambda}$$

同量级即可。

由上图可看出 γ 取 10^{-7} 较为合适。

在 $\gamma = 10^{-7}$ 其他参数不变的情况下，对于不同 max_depth 进行测试，得到以下结果：

表 2: $\lambda = 1$，$max\_epoch = 3$，$\gamma = 10^{-7}$ 条件下不同 max_depth 对于测试集 MSE 的影响

max_depth	1	2	3	4	5	6	7	8	99	10
测试集 MSE(10^{-8})	9.507	6.190	5.033	4.864	4.743	4.738	4.909	4.971	4.836	4.884

绘出折线图如下：

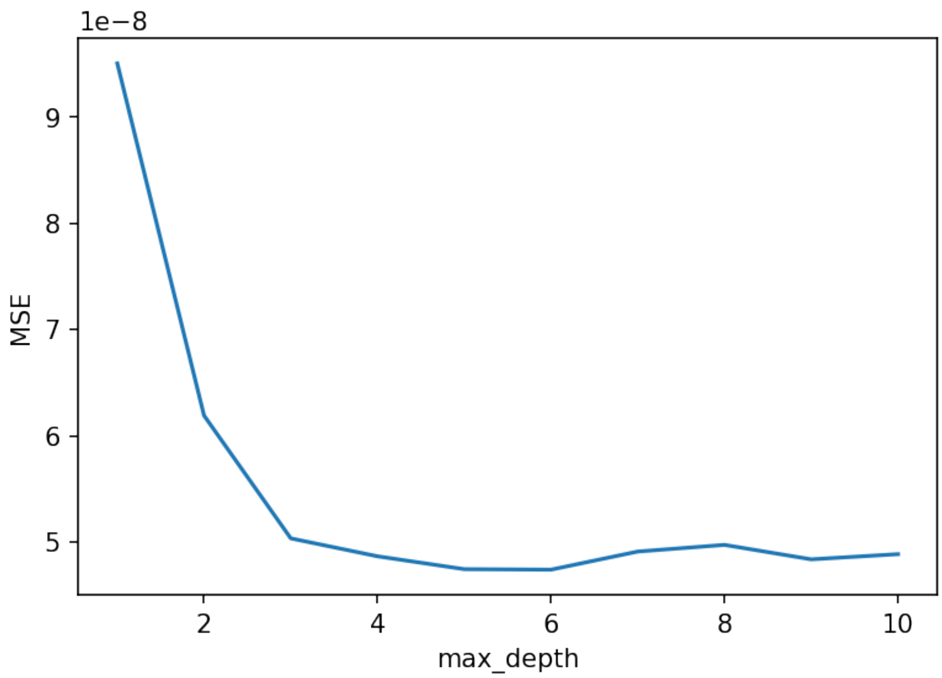

图 3: max_depth –MSE 折线图

对于上述现象的解释：
深度太小，可能会欠拟合；深度太大，可能会过拟合。
从图中可看出，最大深度为 6 的时候，测试集 MSE 最小，之后测试集 MSE 就会上升，说明过拟合，所以 max_depth 取值为 6，效果最好。
在 $\gamma = 10^{-7}$ $max\_depth = 6$ 其他参数不变 $(\lambda = 1)$ 的情况下，对于不同 max_epoch 进行测试，得到以下结果：
绘出折线图如**图** 4：

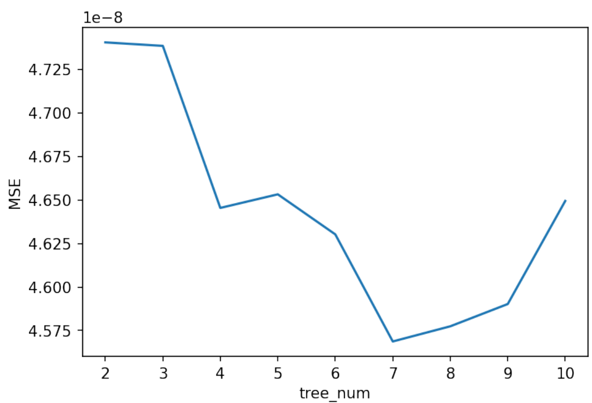

图 4: max_epoch 折线图 (from 2 to 10)

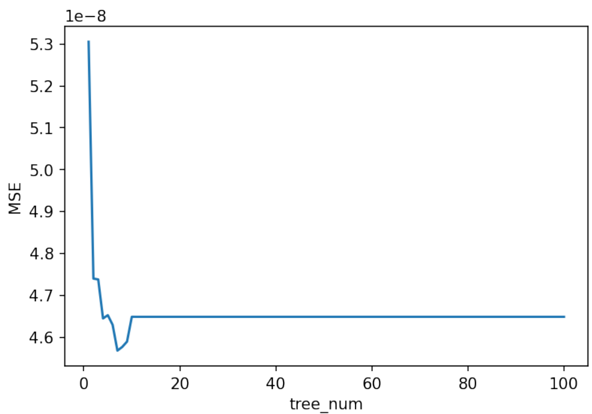

图 5: max_epoch 折线图 (from 1 to 100)

由**图** 4 并不能得到明显的结论（即没有收敛），所以将数的颗数加到了 100，画出**图** 5

对于上述现象的解释：

由图可见，最开始当树的颗数增大的时候，测试集 MSE 会减小，这是从欠拟合到拟合的过程，当树的颗数增加到一定程度的时候，MSE 反而增大，说明过拟合，最后收敛。图中最佳取值为 11.

对于参数 λ, 其变化的范围太大，又由于程序运行时间过长，于是便用 XGBoost 的库作为参考，最终选择了 $\lambda = 65$ 作为其最佳取值, 得到以下结果。

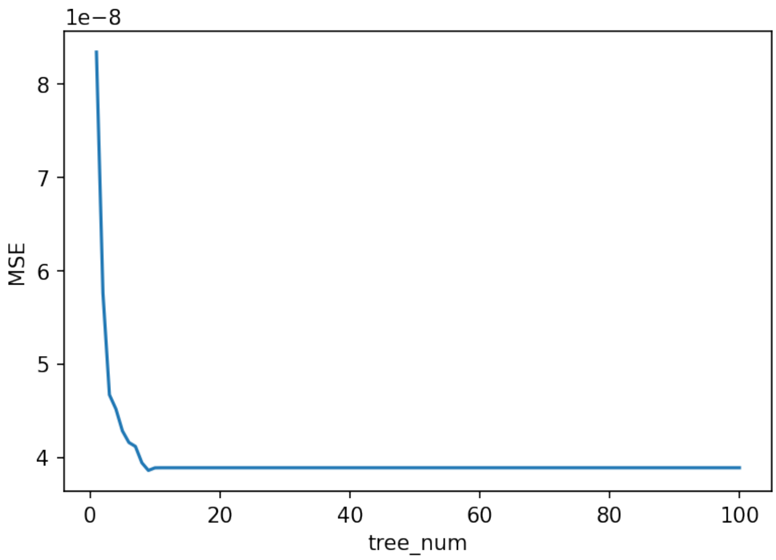

图 6: $max\_depth = 6, \lambda = 65, \gamma = 10^{-7}, max\_depth = 100$ 折线图

综上，参数调整完毕，最佳参数分别为：

- gamma: 10^{-7}
- Lambda: 65
- max_epoch: 11
- max_depth: 6

5.2 Loss 曲线

对于上述调整过后的最佳参数，画出 Loss 曲线（取 R^2 作为评估标准）：

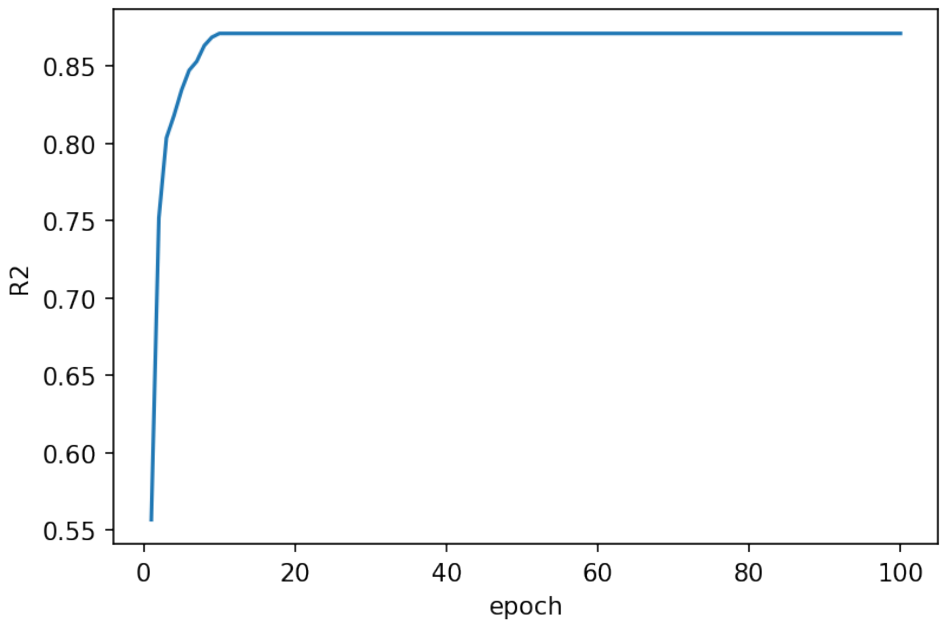

图 7: tree_num–Loss 曲线

由上图可见：best_epoch=11

5.3 实验总结

通过理解并实现 XGBoost 模型，学习到了集成学习的思想，并对于回归树有了进一步的认识。实现的 XGBoost 模型在测试集上表现良好。