决策数确定最佳参数算法：

后剪枝算法（prune）和采样暴力搜索法（tune）

2017年10月17号

申小虎

一、开发的对象GBP
DTGetBestParas
简称GBP

```
from sklearn.datasets import load_boston
boston = load_boston()
X = boston.data
y = boston.target

#init
op = DTGetBestParas(method='tune',figtitle='boston')

#fit
op.fit(X,y)
#print(op.clf)

#predict
op.clf.predict(X)

#get result
print(op.max_depth,
      op.min_samples_leaf,
      op.min_samples_split,
      op.max_leaf_nodes)
```

获取决定树的复杂度的4个参数：

树的深度：max_depth
子节点下最小样本数：min_sample_leaf
父节点下最小样本数：min_sample_split
树的最大叶子节点数：max_leaf_nodes

二、GBP特色

- 支持5种类型方法获取最优参数：
  - ‘none’: sklearn 默认设置
  - ‘cal’：通过样本量换算相关参数
  - ‘tune’：暴力搜索获取树的最佳深度
  - ‘prune’：通过后剪枝算法获取最优参数
  - ‘both’：先通过暴力搜索获取最佳深度后再后剪枝
- 剪枝优化和暴力搜索优化都支持多线程并行运算（如：n_jobs = 4），大幅加快运算速度
- 支持返回所选取的方法下最优的树对象clf，可以用来直接预测，如GBP.fit(X,y).clf.predict(X1)

三、GBP后剪枝算法

GBP后剪枝算法简介：

1. 对于原始的CART树A0，先剪去一棵子树，生成子树A1，然后再从A1剪去一棵子树生成A2，直到最后剪到只剩一个根结点的子树An。于是得到了A0-AN一共n+1棵子树。然后再用n+1棵子树预测独立的验证数据集，谁的误差最小就选谁

2. 对于每次剪的时候到底取哪个节点来剪，取决于误差增益alpha,每次剪最小alpha对应的叶节点:

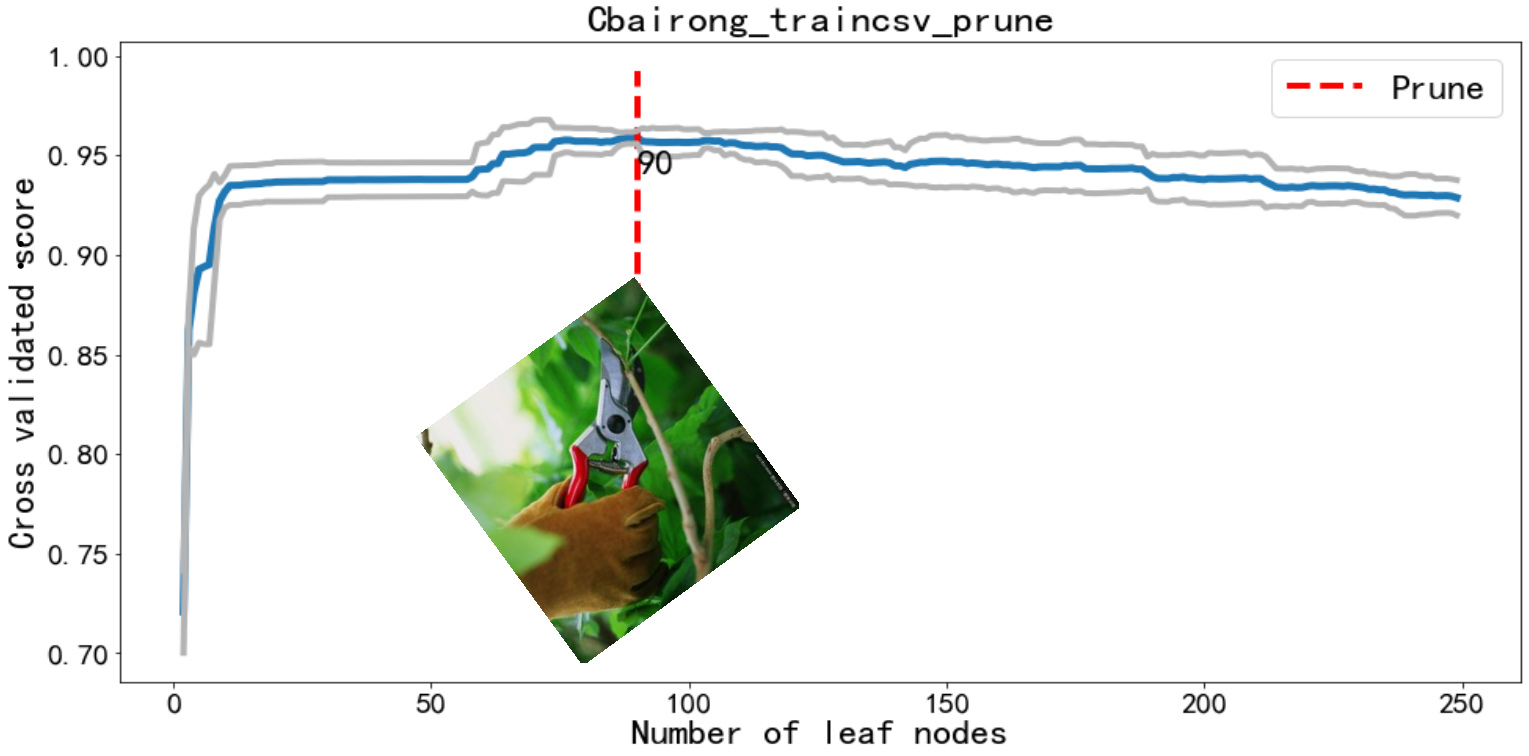
$$\alpha = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

3. 采用固定比例的洗牌抽样法选取测试集，迭代一定的次数，然后检验测试集合的效果。选取测试集合打分最高对应的叶子节点数

打分函数（越高越好）

分类：ROC_AUC

回归：neg_MSE

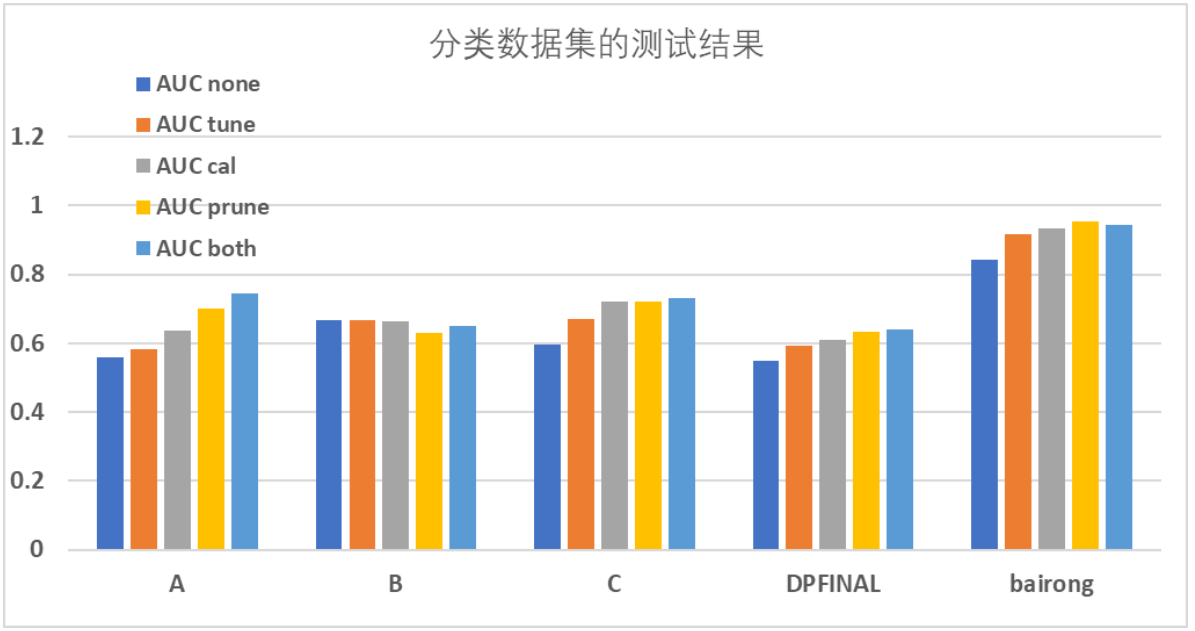


四、结果测试

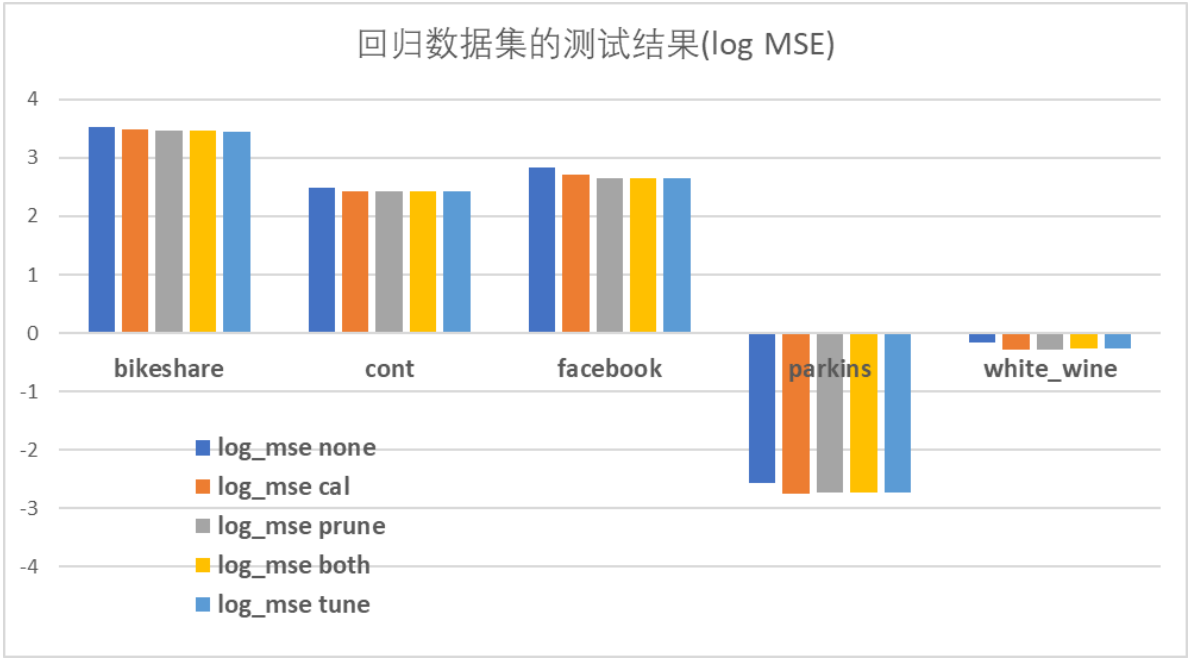
数据集（5个分类，5个回归，测试集合数量占总数30%）

数据集名称	数据描述	布尔型特征	连续性特征	样本总数	测试样本	训练样本	特征数目	目标变量	类型	来源
white_wine	白酒质量	0	11	4898	1470	3428	11	quality	回归	https://archive.ics.uci.edu/ml/datasets/Wine+Quality
parkins	帕金森CT	1	20	5875	1763	4112	21	PPE	回归	https://archive.ics.uci.edu/ml/datasets/Parkinsons+Telemonitoring
bikeshare	共享单车	3	10	17379	5214	12165	13	cnt	回归	https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset
facebook	Facebook的post	15	38	41049	12315	28734	53	53	回归	https://archive.ics.uci.edu/ml/datasets/Facebook+Comment+Volume+Dataset
cont	金融	7	20	96366	28910	67456	27	TargetD	回归	刘帅
C	活性化合物分析	113	31	4279	1284	2995	144	Outcome	分类	https://www.kaggle.com/uciml/bioassay-datasets
A	活性化合物分析	124	31	59788	17937	41851	155	Outcome	分类	https://www.kaggle.com/uciml/bioassay-datasets
B	活性化合物分析	123	31	59795	17939	41856	154	Outcome	分类	https://www.kaggle.com/uciml/bioassay-datasets
DPFINAL	金融	57	124	90000	27001	62999	181	TARGET	分类	刘帅
bairong	金融	74	69	100800	30240	70560	143	flag	分类	刘帅

测试集性能上的表现

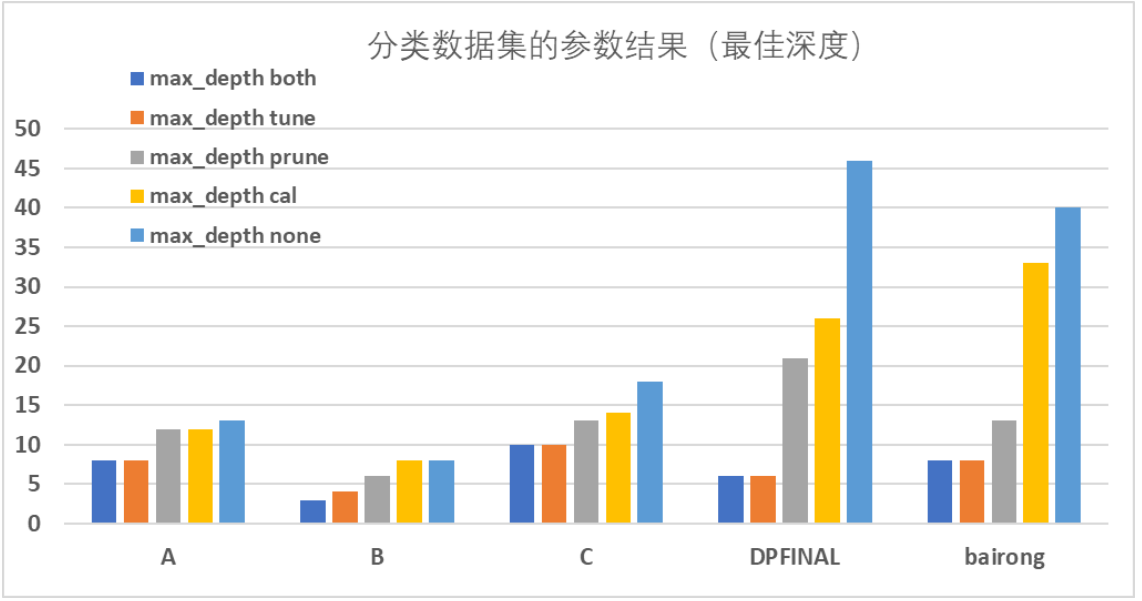


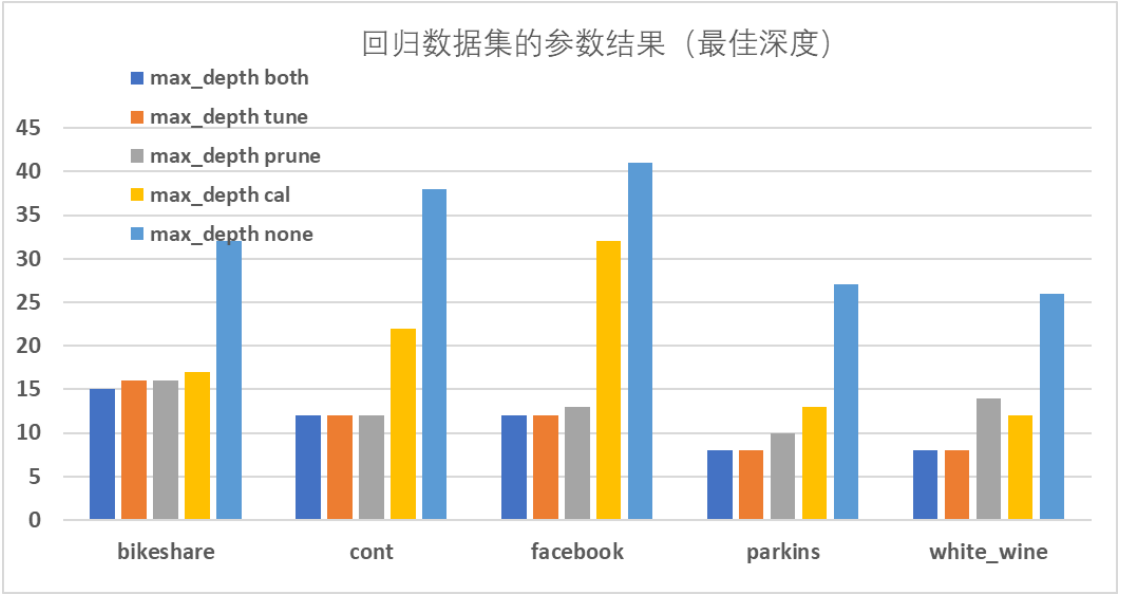
性能：
both > prune
> cal > tune > none



性能：
tune > both
> prune > cal > none

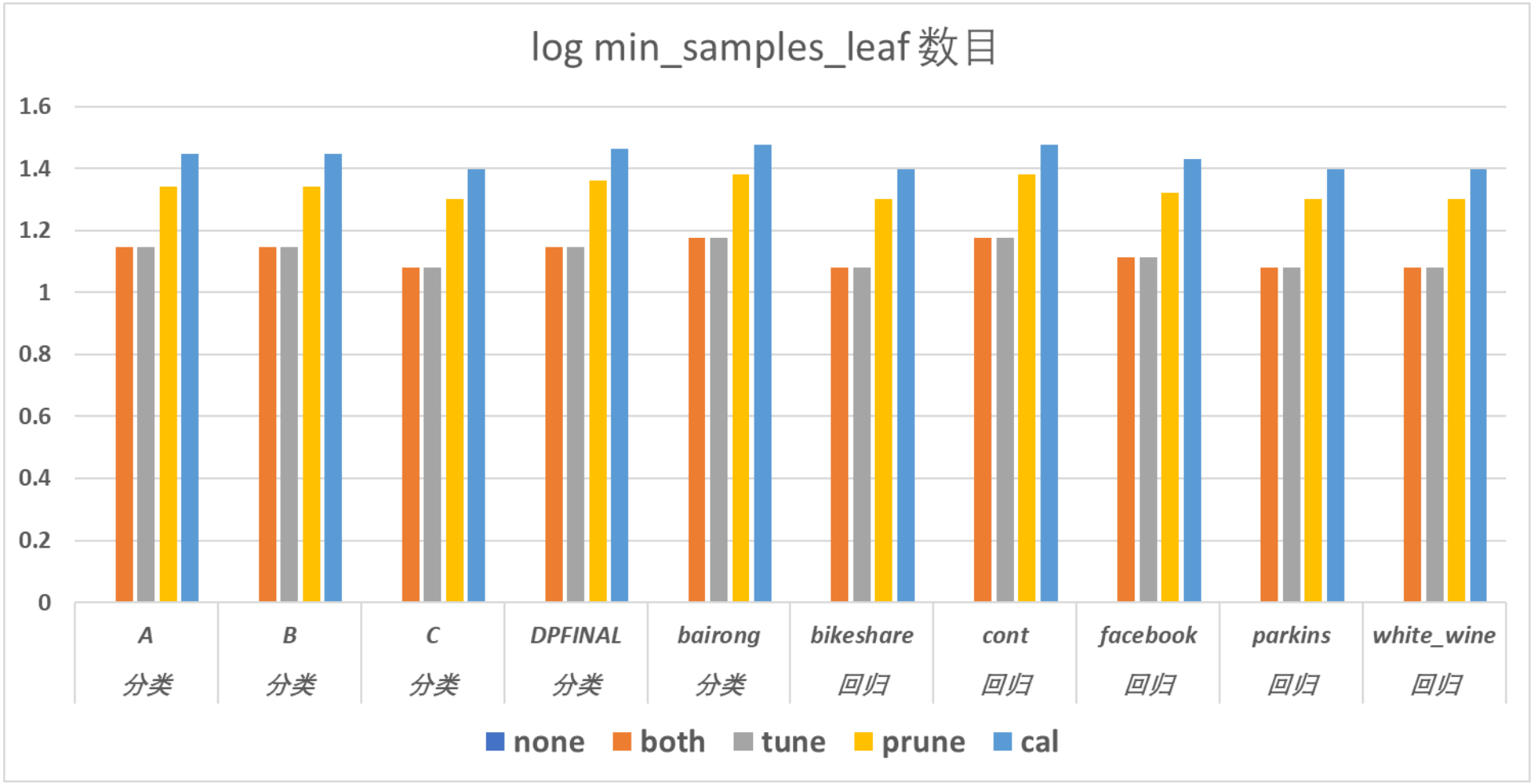
最佳深度表现





深度：
none > cal
> prune > tune > both

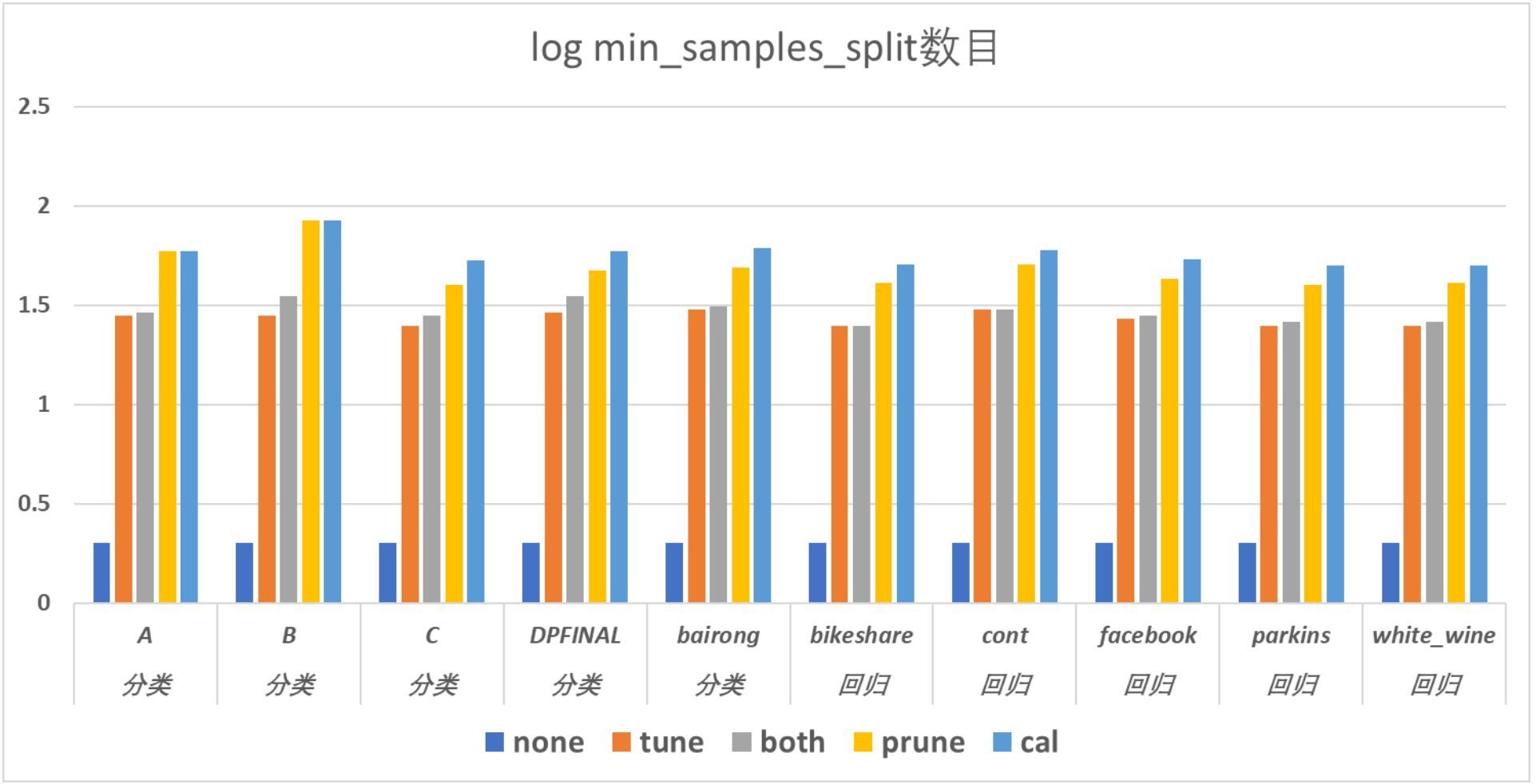
子节点下最小样本数



min_sample_leaf：cal > prune > tune > both > none

Prune 总是大于tune

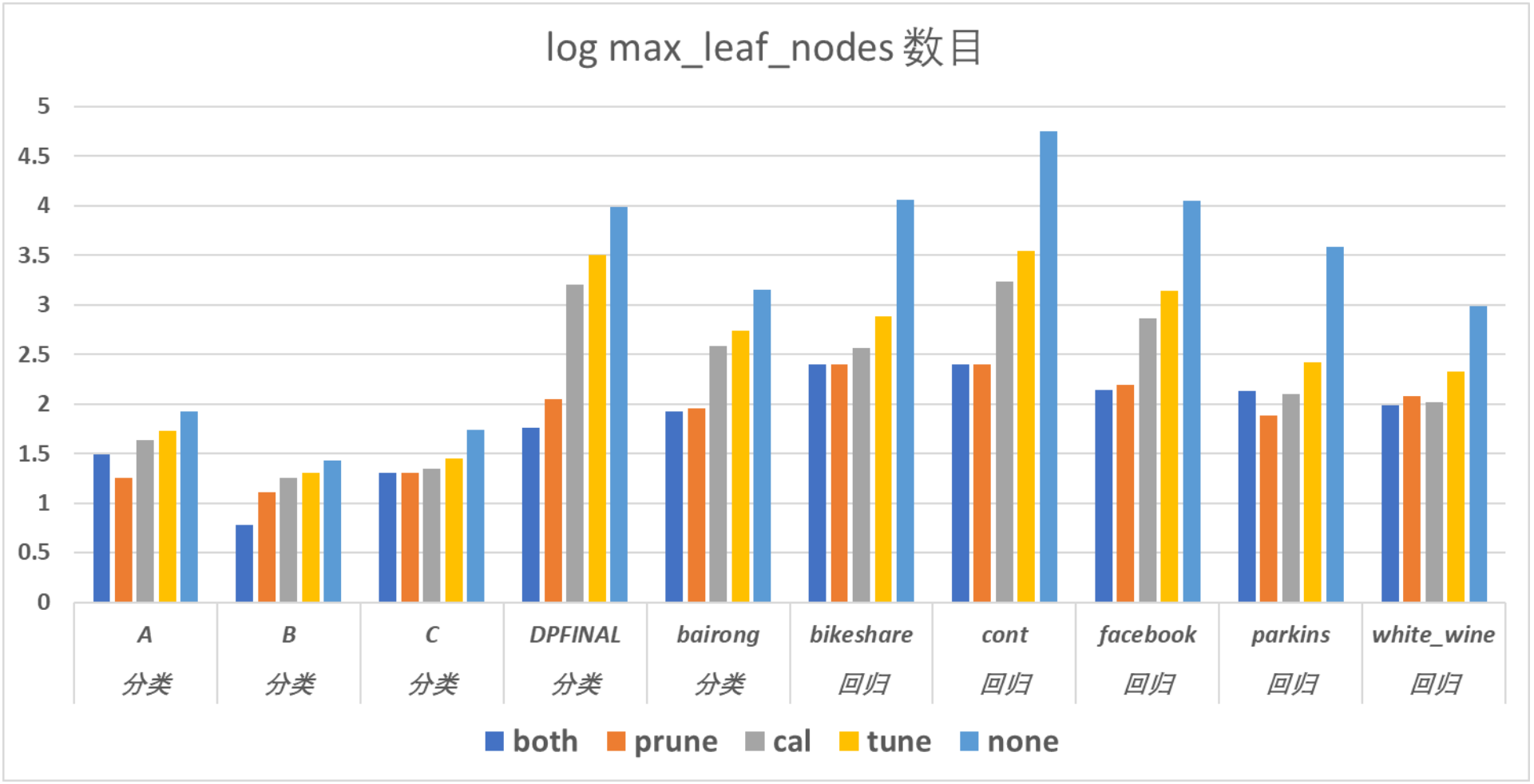
父节点下最小样本数



min_sample_split：cal > prune > both > tune > none

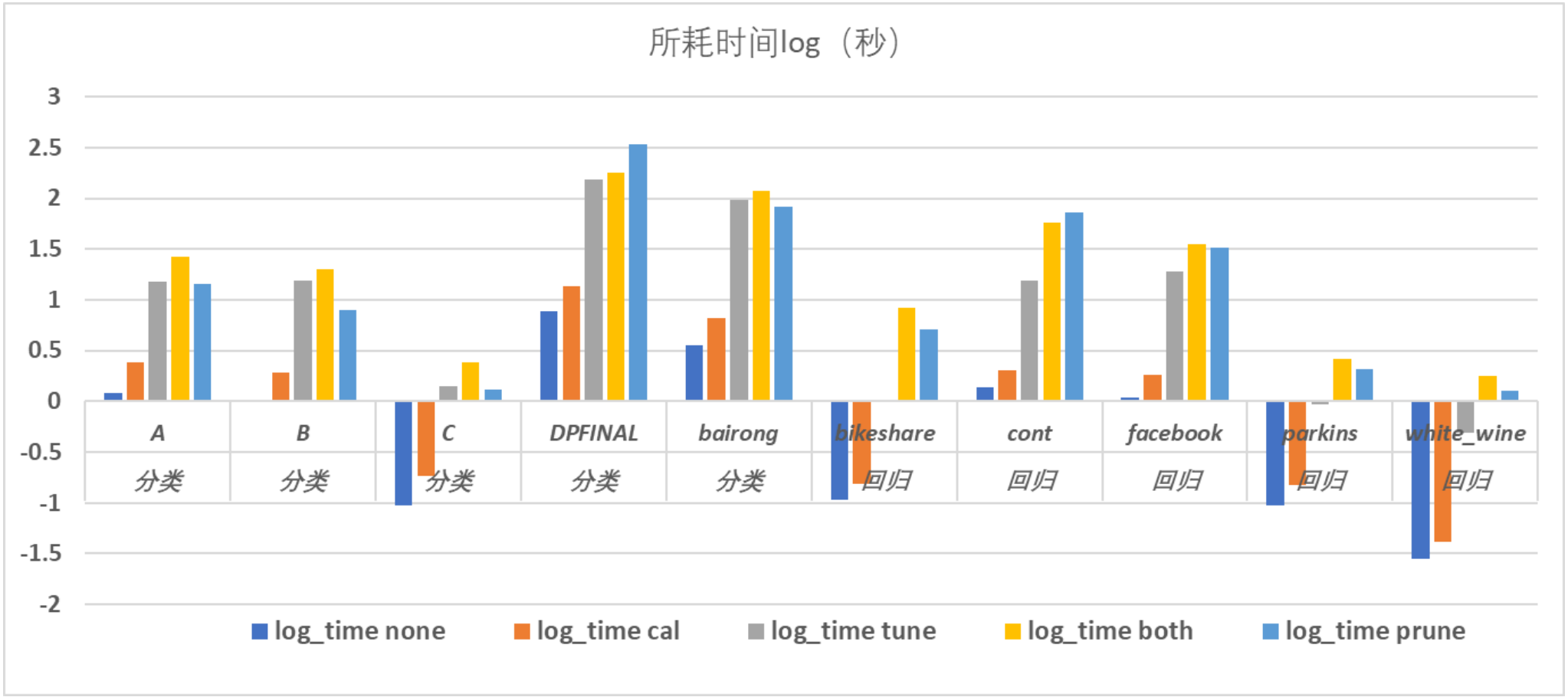
Prune 总是大于tune

树的最大叶子节点数



max_leaf_nodes：none > tune > cal > prune > both

所耗时间表现



耗时长短：prune > both > tune > cal > none

五、结论

1. Both的综合性能在单棵决策树中表现最好，在测试集的表现最佳
2. Both方法比prune的速度还快，原因是先使得树变得小一些，降低了后面的剪枝的复杂度
3. Both方法继承了tune的深度，并且通过剪枝方法，深度小于tune
4. Both的子节点下最小样本数和tune的接近，父节点下最小样本数**总是**介于tune和prune之间
5. Both的最大叶子节点数和prune接近，相比于其他方法最小