机器学习第七章集成学习实验报告

162050127 颜劭铭

2022 年 6 月 2 日

1 原理

这次的编程是集成学习（ensemble learning）部分的内容，顾名思义，就是使用多个子模型集成预测，从而提升整体的性能。

集成方式主要分为两大类：并行和串行。其中并行主要有使用随机子样本的 Bagging 和随机子空间的随机森林，串行主要使用 Boosting，代表算法有 AdaBoost 等。

本次实验主要选取了 UCI 数据集 Adult，用 AUC 作为评价分类器性能的评价指标，实现了 AdaBoost 和随机森林两种算法。

AdaBoost：

首先先将分布的权重设为相等的 $\frac{1}{m}$，然后根据设计的训练轮数 T，在每一轮的训练中，根据之前的分布从 D 中训练出基模型，计算出加权错误率，并依据加权错误率计算出样本的权重，错误的样本的权重更大，并依据样本的权重更新分布，进入下一轮训练。

随机森林：

随机森林的子模型是决策树。①在每一次训练的过程中有放回地随机选择 N 个样本训练一个决策树.

②随机选取 m 个属性，根据属性选取的策略（如信息增益、基尼指数）选择一个属性，对于决策树的结点进行分裂.

③重复步骤 2 直到不能分裂.

④重复步骤 1、2、3 建立大量决策树，构建随机森林.

2 实现代码过程

2.1 处理数据

将数据从 txt 文件中读出来以后，由于很多算法是在 array 上进行，所以进行一个类型转换为 numpy.array 型，同时对于数据的标签，在数据集上是 0-1 分布，而为了方便我们后续的算法编写，将其转化为-1 和 1 分布.

2.2 AdaBoost

编写过程参考 scikit-learn 的编写方式，建立了 TreeClassifier_weight 和 Adaboost 类。

对于 AdaBoost 的基学习器采用的是加权决策树桩，先对于权重矩阵初始化为相同权重，之后依次设定每一列属性为特征属性，并从小到大排列，依次选定阈值，计算出最低错误率，并选择此时的属性作为划分属性建立决策树桩，同时编写 predict 函数以及对于模型准确度进行预测的 score 函数。

对于 Adaboost 类的主要编写过程和伪代码基本一致，主要要注意一个数据类型的转化和计算，要使用好 Numpy 库，可以提高计算效率，同时由于评价指标选定的是 AUC，所以我们除了正常的 predict 函数返回标签值以外，还需要编写一个 predict_proba 预测类概率，这个问题一开始困扰了我很久，因为用正常的 predict 函数，我们返回的是样本的标签，这并不适合 AUC 计算，计算出来的 AUC 值是非常低的。

在每一个基学习器训练完以后计算对应的 AUC 值和准确度 Score，并存储到文件中，方便后续画图。

2.3 随机森林

随机森林的编写主要就是调库实现了，同时利用了 KFold 进行交叉验证，用两个 for 循环，在训练基学习器数量逐渐增加的情况下，对每种数量的基学习器进行 5 折交叉验证，并计算出平均的 AUC 和 score 并存储到文件中，方便后续画图。

2.4 画图

画图过程中为了保证图上的点不会超出范围，利用 xlim 和 ylim 函数重新确定横纵轴坐标范围。

3 实验结果

3.1 AdaBoost

从图 1中我们可以发现：

①AdaBoost 在测试集上的 AUC 在基学习器数量 50 以内时增加较为快速，从 0.591 增加到了 0.908.

②在 50 个基学习器到 100 个基学习器这段过程中，AUC 增长到了 0.911，可以看出增长是非常小的了.

③在后续的改变更是非常非常小的，直到 500 个基学习器时，AUC 值为 0.917，而这段时间的时间花销和计算花销是非常大的。

因此我认为对于这个数据集和 AdaBoost 来说最好的基学习器数目为 50 到 100 之内，甚至可以认为在这个数据集和 AdaBoost 算法上选取 50 个基学习器是一个最佳的基学习器数量。

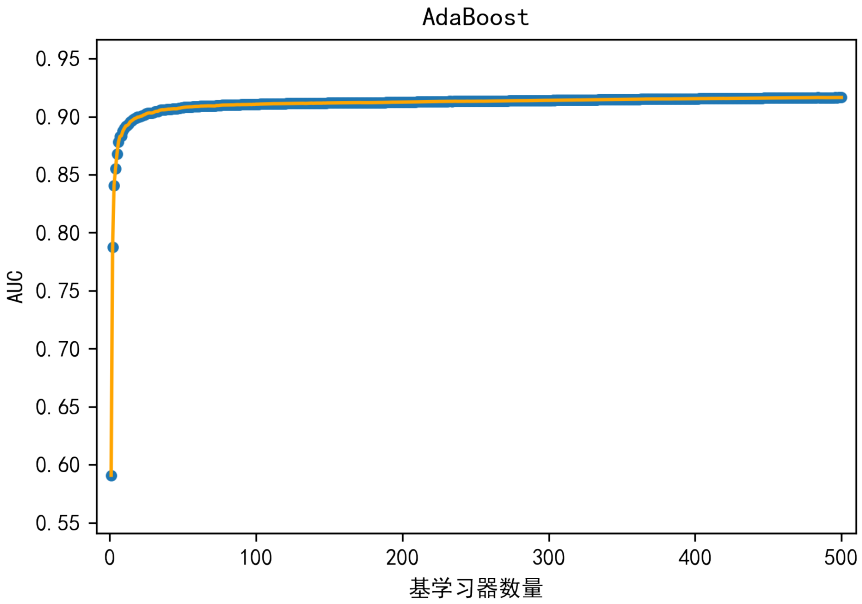

图 1: AdaBoost 结果

3.2 随机森林

同样的，从图 2中我们可以发现：

①随机森林在测试集上的 AUC 在基学习器数量 25 以内时增加较为快速，从 0.739 增加到了 0.897.

②在 25 个基学习器到 75 个基学习器这段过程中，AUC 增长到了 0.905，可以看出增长是比较小的了.

③在后续的改变更是非常非常小的，直到 300 个基学习器时，AUC 值为 0.907，甚至在 125 个以后他就基本没有什么增长了.

④通过比较可以发现在训练到 75 个基学习器以后的这段时间的时间花销和计算花销是非常大的，一开始只使用一个基学习器训练随机森林只需要 0.15s，使用 25 个基学习器时需要 3.23s，使用 300 个时需要 31.93s。

因此我认为对于这个数据集和随机森林来说最好的基学习器数目为 25 到 75 之内，甚至可以认为在这个数据集和随机森林算法上选取 50 个基学习器是一个最佳的基学习器数量。

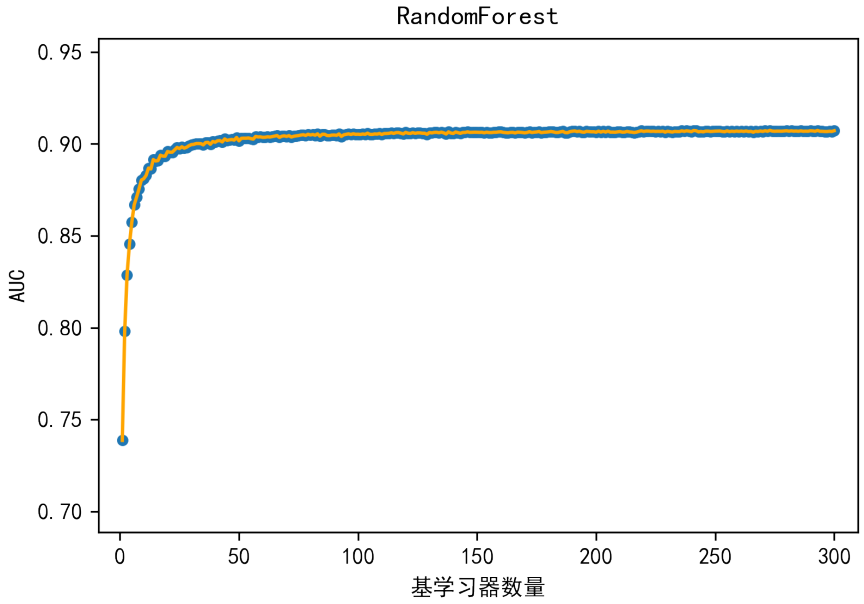

图 2: 随机森林结果

3.3 对比

通过在图 3上的对比我们可以发现，随机森林的初始效果是要好于 AdaBoost，不过我认为这个主要是因为随机森林是调库实现，而 AdaBoost 的基学习器是自己编写的决策树桩，但是可以发现两者的增长速度基本接近一致，并且最后效果差不多好，AdaBoost 还是要稍微优

于随机森林一点点的，这个原因还不是很了解。

但是这也从侧面反映出了集成学习的有效性，将一个效果可以说是挺差的基学习器通过集成预测，提升了非常多的整体性能。

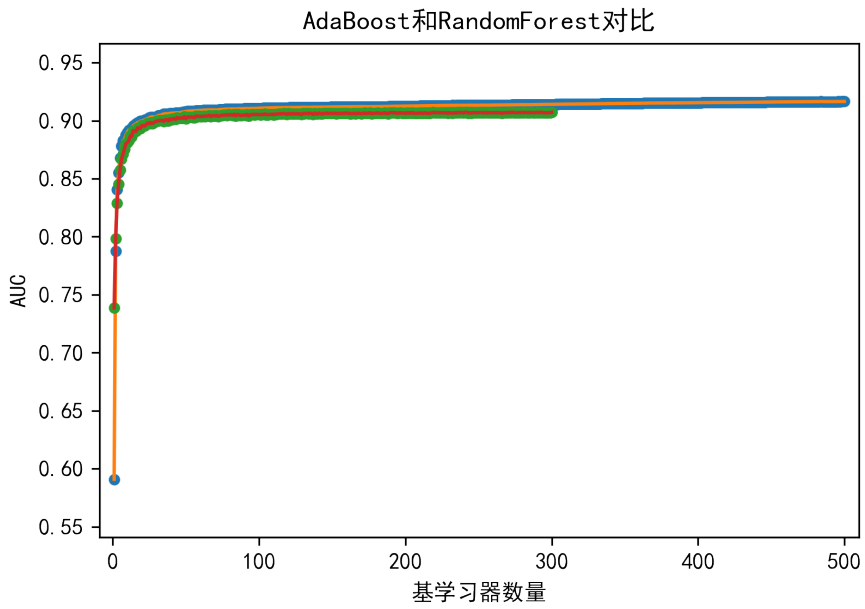

图 3: 随机森林和 AdaBoost 对比结果，蓝色点和黄色线是 AdaBoost，绿色点和红色线是随机森林