1. 随机森林、XGBoost、LightGBM

在数据分析的初始阶段，决策树对于探索多个因素与目标变量（即AQI值）之间的关系非常有用且易于解释。为了进一步确保结果更加准确可靠，我们首先介绍三种基于决策树的集合方法，包括随机森林、XGBoost和轻梯度提升机（Light Gradient Boosting Machine，简称轻GBM）。

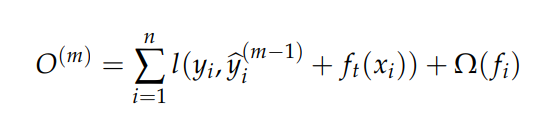
**随机森林的基本原理:**

随机森林是一种使用多棵决策树对样本进行训练、分类和预测的方法。在数据分类过程中，可以通过每个变量的重要性来衡量其在分类中的地位。随机森林中的 "随机 "有两层含义：首先，样本选择是基于带撤回的抽样，这意味着每个样本都有可能被多次选中或不被选中。这种抽样方法可以有效地增加决策树之间的差异，进一步降低过度筛选的风险。其次，在构建决策树的过程中，随机森林并不是使用所有特征来构建每一棵决策树，而是从所有特征中随机抽取一部分特征来构建决策树。这种方法可以有效降低特征之间的相关性，提高模型的性能。在随机森林中，每棵决策树都是通过对数据进行迭代分区来构建的。在构建决策树的过程中，数据会根据特定的指标进行分割，直到达到预定的停止条件。为了避免过度分裂问题，随机森林还可以通过控制决策树的深度和节点停止分裂的最小样本数等参数来限制决策树的生长。

**XGBoost的基本原理:**

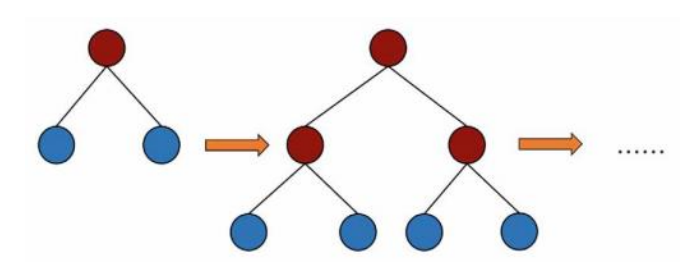
XGBoost 算法属于Boosting框架，其本质区别在于优化残差树所需的增益不同，XGBoost 使用的增益是分割前后的结构得分之差。XGBoost的一个重要特点是引入了新的分割标准，在最佳分割点将分割损失降至最低。XGBoost算法的核心思想分为三个步骤。首先，采用特征分割法不断增加树，每增加一棵树实质上都是学习一个新函数来修正上次预测的残差。其次，在完成训练并获得 k 棵树后，应预测样本的得分。第三，样本的预测值是每棵树对应分数相加的结果。XGBoost 模型迭代 m 次后，目标函数的定义如下式所示。

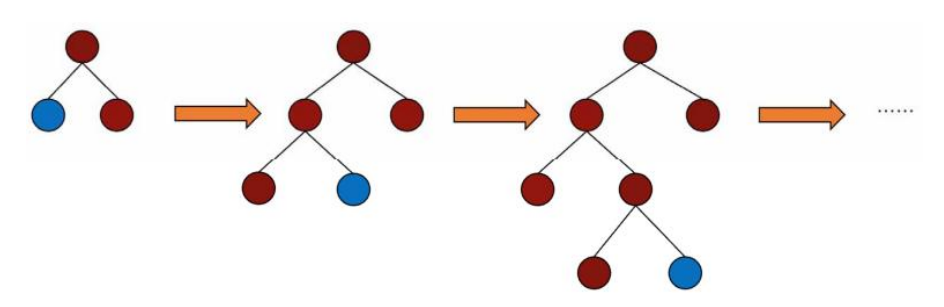
O (m) = n ∑ i=1 l(yi , yb (m−1) i + ft(xi)) + Ω(fi)

**(13)**

式（13）中，l 和 Ω 分别为损失函数和正则项，yi 和 ybi 分别代表结果对应的真实值和模型的预测值。

**LightGBM的基本原理:**

LightGBM旨在解决GBDT在处理大规模数据时所面临的挑战，使GBDT能更好、更快地应用于工业实践。LightGBM 通过引入直方图算法和采用限制性分叶策略，克服了 XGBoost 算法内存消耗大、训练时间长等缺点。直方图算法利用直方图找出最佳分割点，处理连续变量，减少特征中的特征值数量，并减少叶节点分割时需要处理的特征值数量。其基本思想分为三个步骤。首先，将连续波动点特征值离散为 k 个整数，并构建宽度为 k 的直方图。然后，在遍历数据时，将离散值作为指数并累积到直方图中。在此基础上，进行遍历并找到最佳分割点。如下图所示，XGBoost 算法采用 Level-wise 作为增长策略。这种策略对数据进行一次遍历，可以同时分割同一层的叶子，有利于控制模型的复杂度，达到控制迭代的效果。但在实际应用中，大多数叶子的分割增益相对较小，因此没有必要对叶子进行搜索和分割，从而避免了不必要的计算。

LightGBM 算法使用 " Leaf-wise "作为生长策略。如图 13 所示，Leaf-wise 算法每次从当前的所有叶子中进行分割，找出分割增益最大的叶子，然后重复这一过程。与分层算法相比，分叶算法具有以下优点。在分割数量相同的情况下，叶式算法能有效减少误差，提高算法的准确性。但叶式算法的缺点是会形成一棵更深的决策树，从而导致算法过度。因此，LightGBM 算法在 " Leaf-wise "算法的基础上增加了最大深度限制，以避免过密，提高计算效率。

使用三种算法进行分析预测，得到R方值并绘制拟合效果图：

放个表：

随机森林：

Train R^2 Score: 0.9757477919112507

Test R^2 Score: 0.9208221928727248

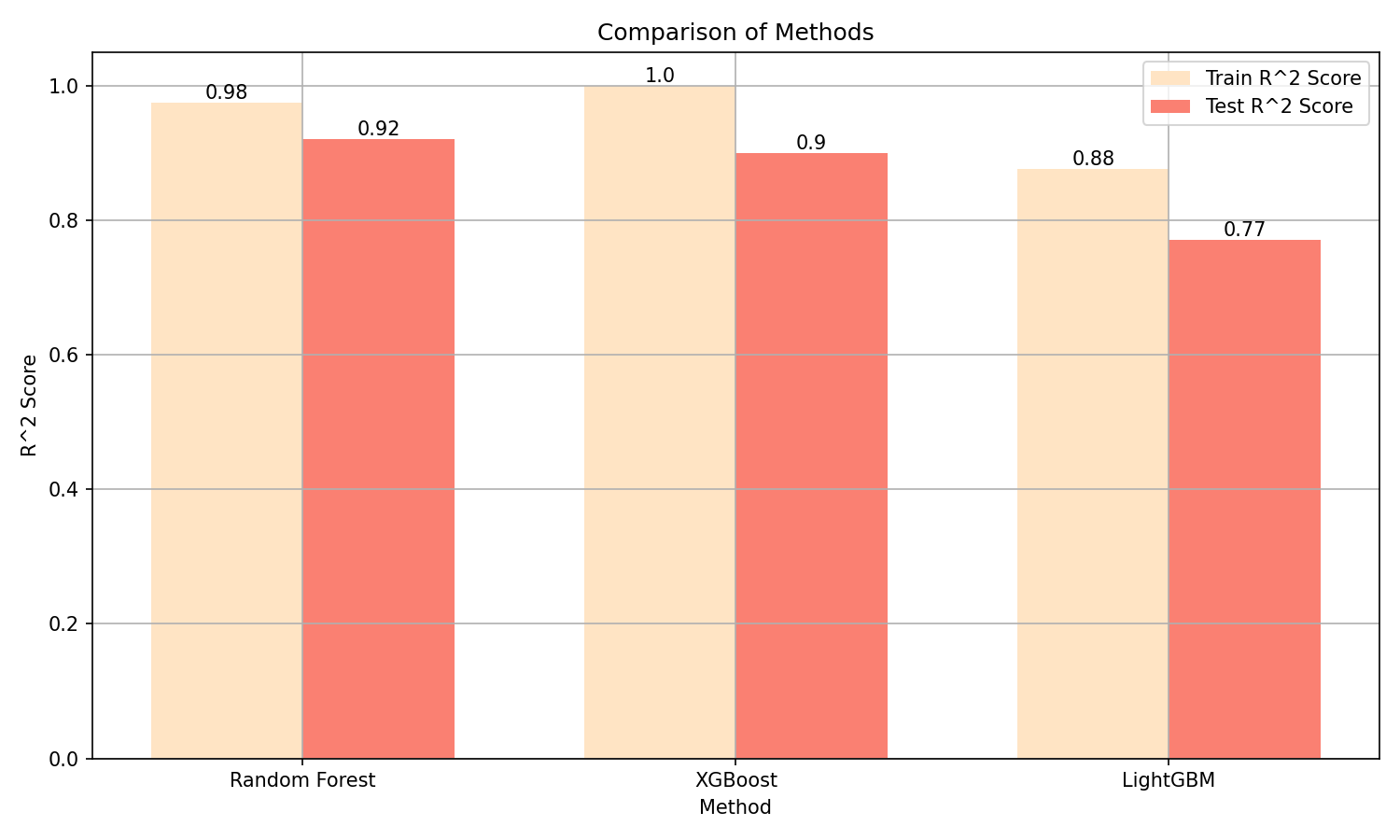
LightGBM:

Train R^2 Score: 0.8766610200363871

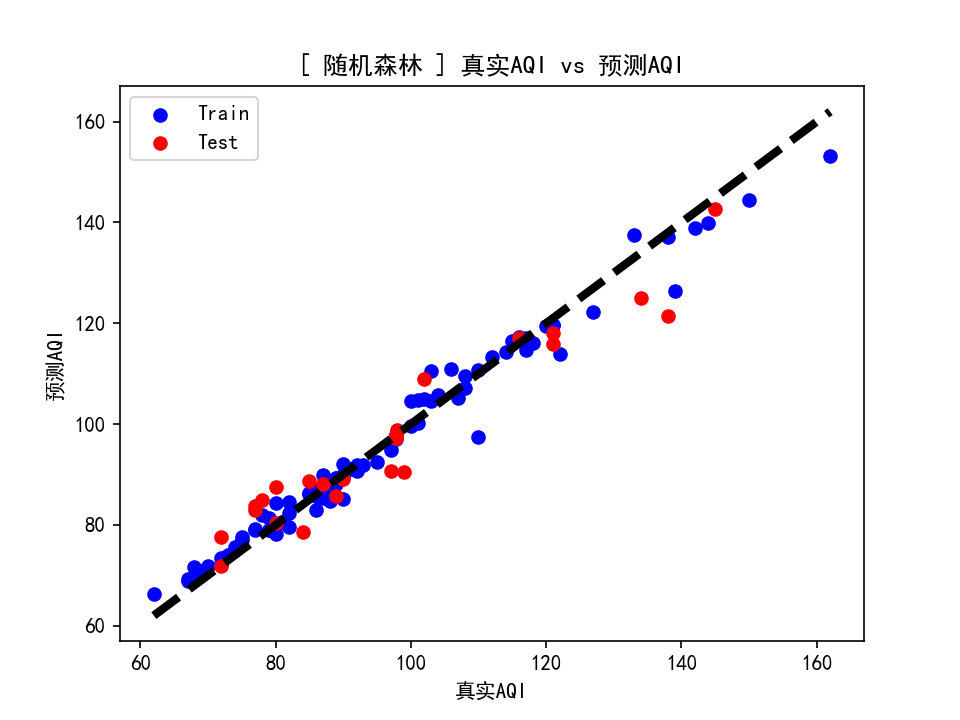
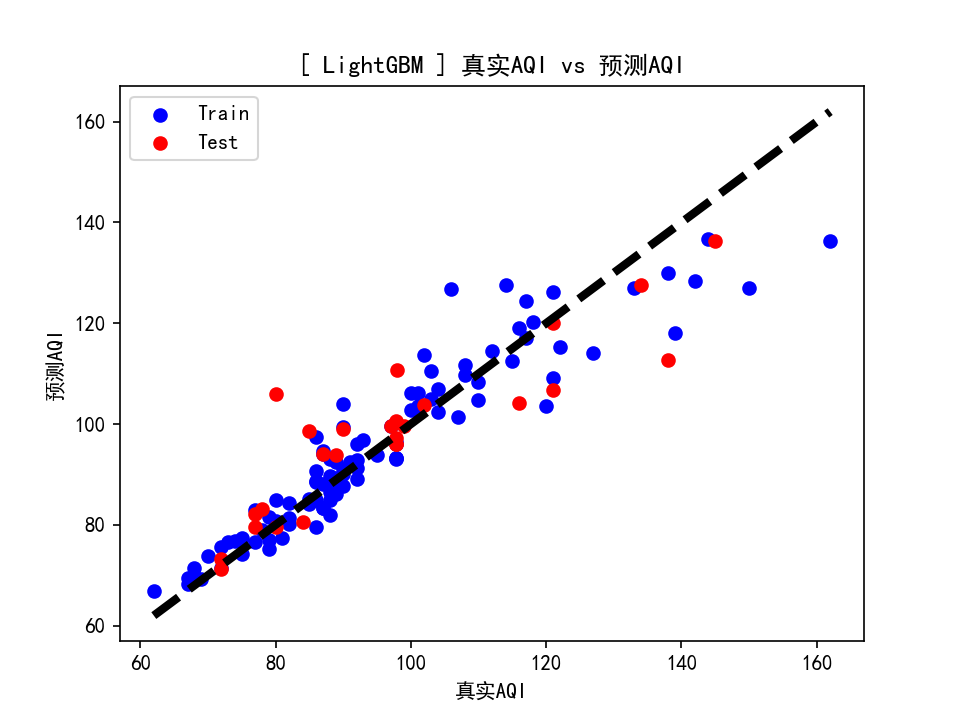
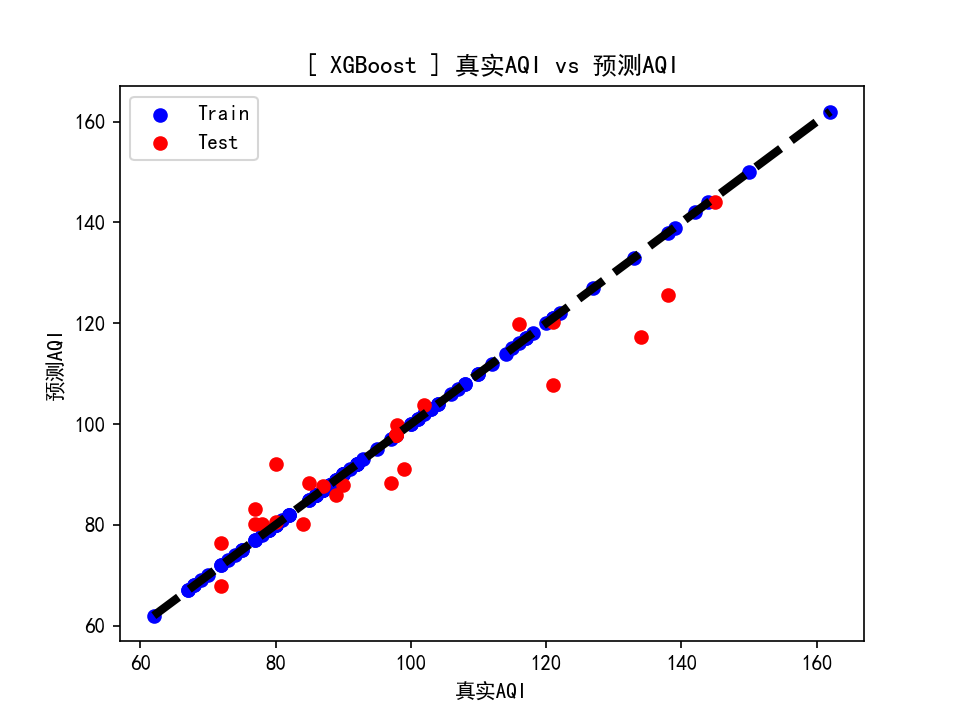
Test R^2 Score: 0.7713688256175901

XGBoost：

Train R^2 Score: 0.9999999971717937

Test R^2 Score: 0.8999337510900779

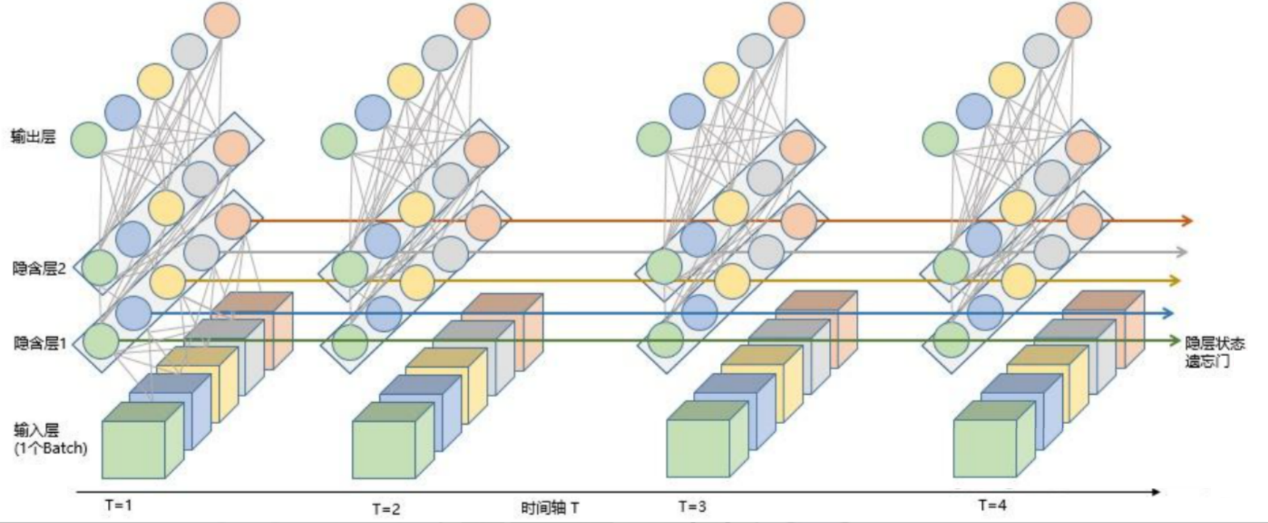
拟合效果图如下：

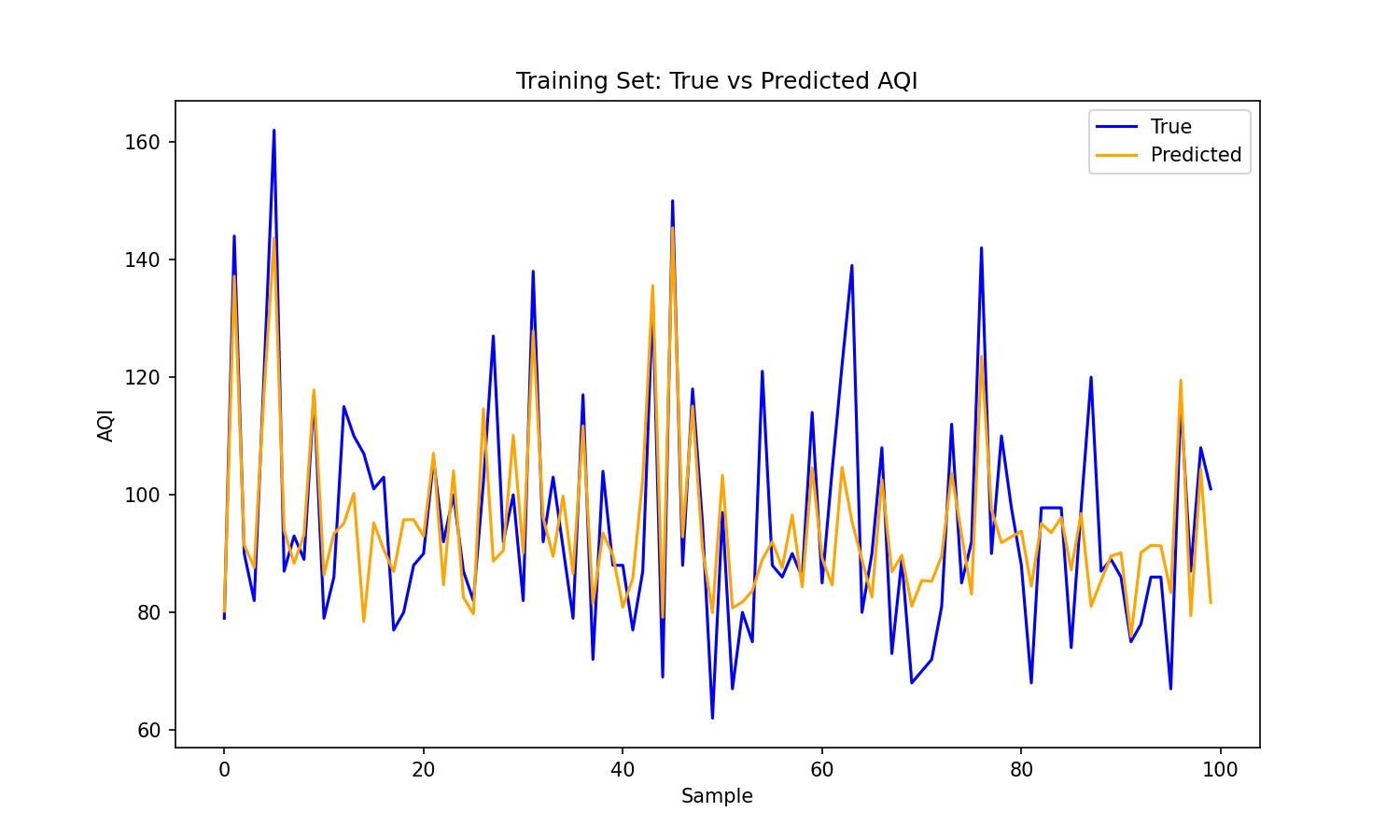
根据R方值和拟合图，可知随机森林的训练和预测效果均较好，LightGBM效果相对较差，XGBoost对于训练集有很好的拟合度，但是对于预测集的效果不好。

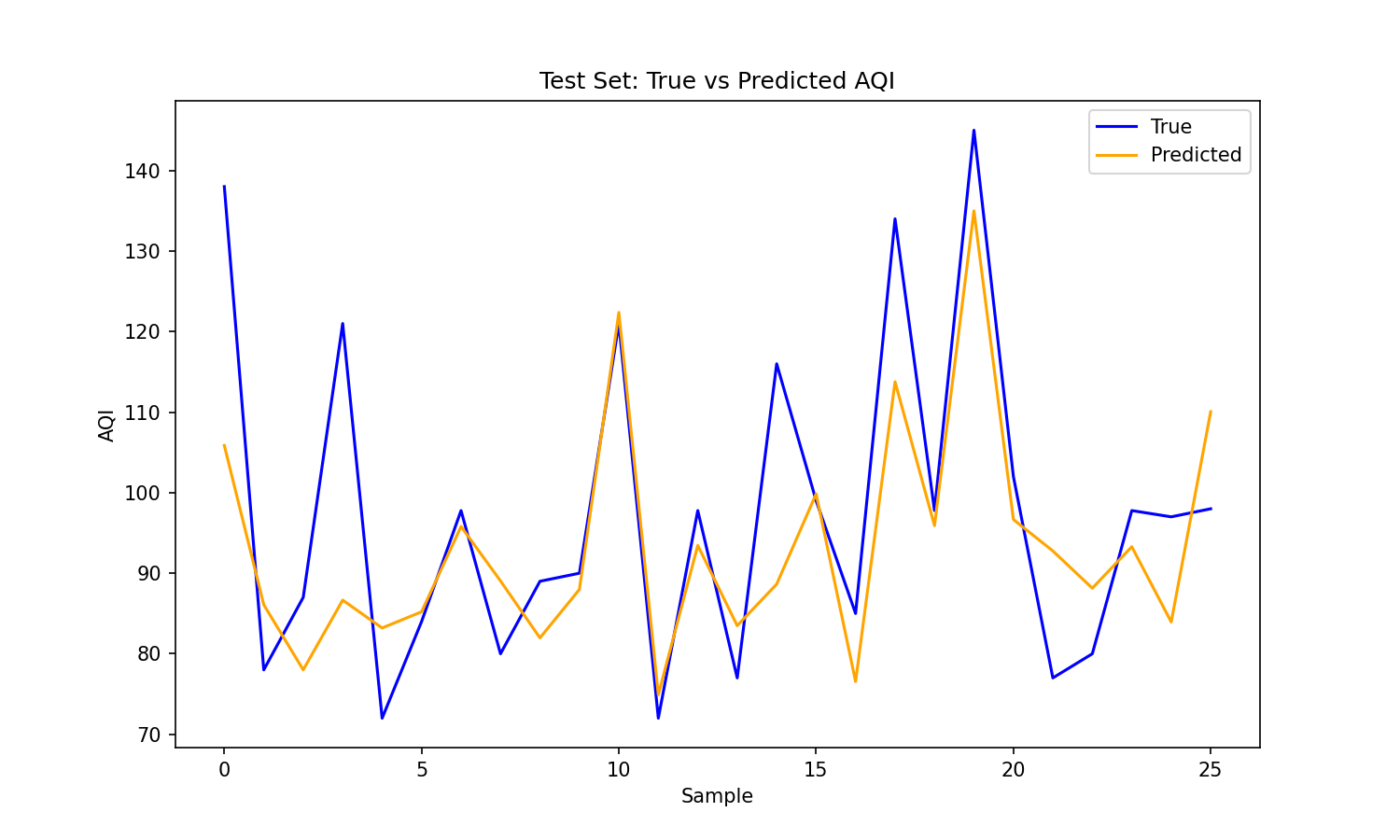
基于随机森林良好和稳定的分析预测效果，我们选择其进行进一步的优化算法。

LSTM

LSTM（Long Short-Term Memory）是一种循环神经网络（RNN）的变种。RNN是一类用于处理序列数据的神经网络，它在每个时间步接受一个输入，并在下一个时间步产生一个输出，同时还会保存一些内部状态以处理序列信息。

LSTM是RNN的一种特殊类型，旨在解决传统RNN在处理长期依赖性时容易出现的梯度消失或梯度爆炸问题。它通过引入门控机制（如输入门、遗忘门、输出门）来控制信息的流动，有效地捕捉和利用长期依赖关系。

直接通过LSTM算法进行时间序列预测得到MSE值并绘制时间序列效果图：

 其中：

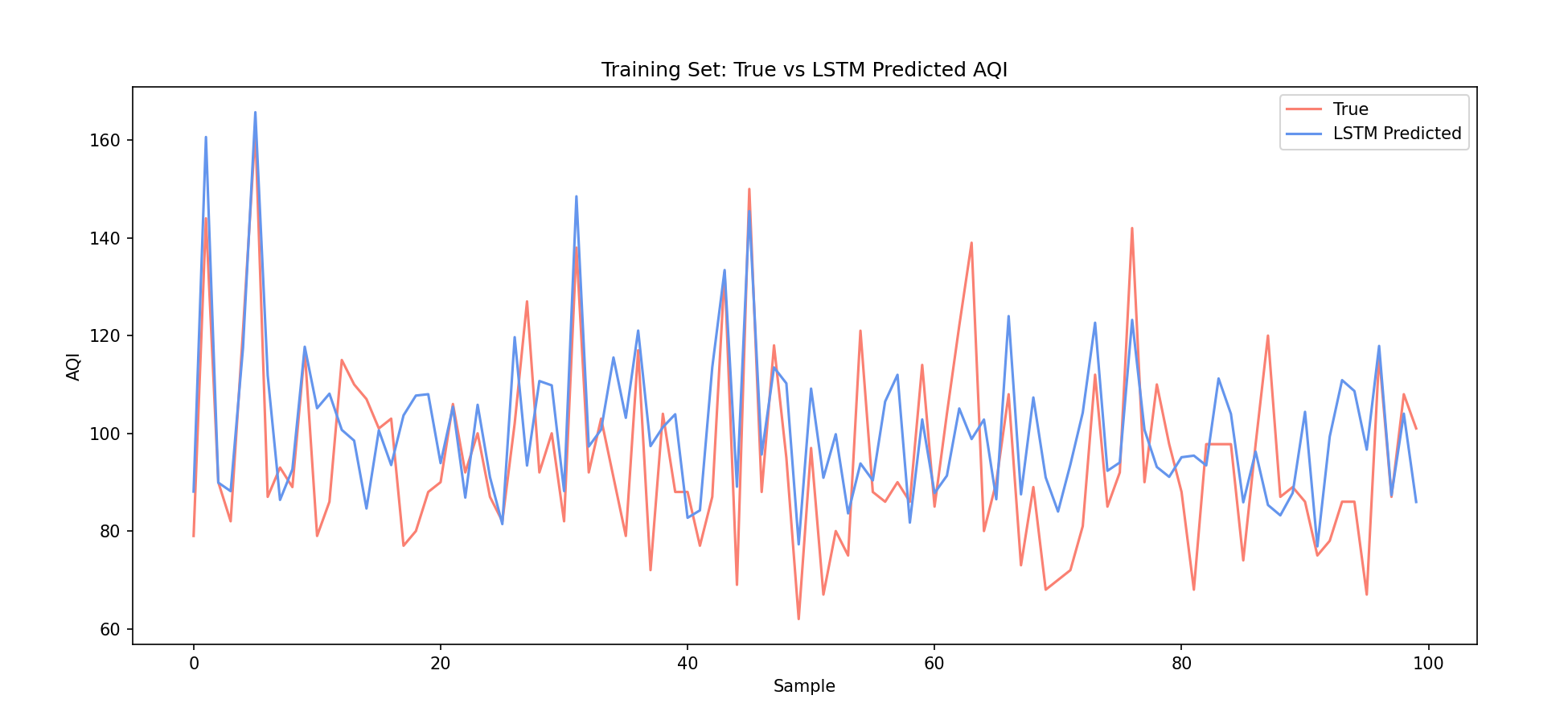
LSTM Train MSE Score: 202.85757446289062

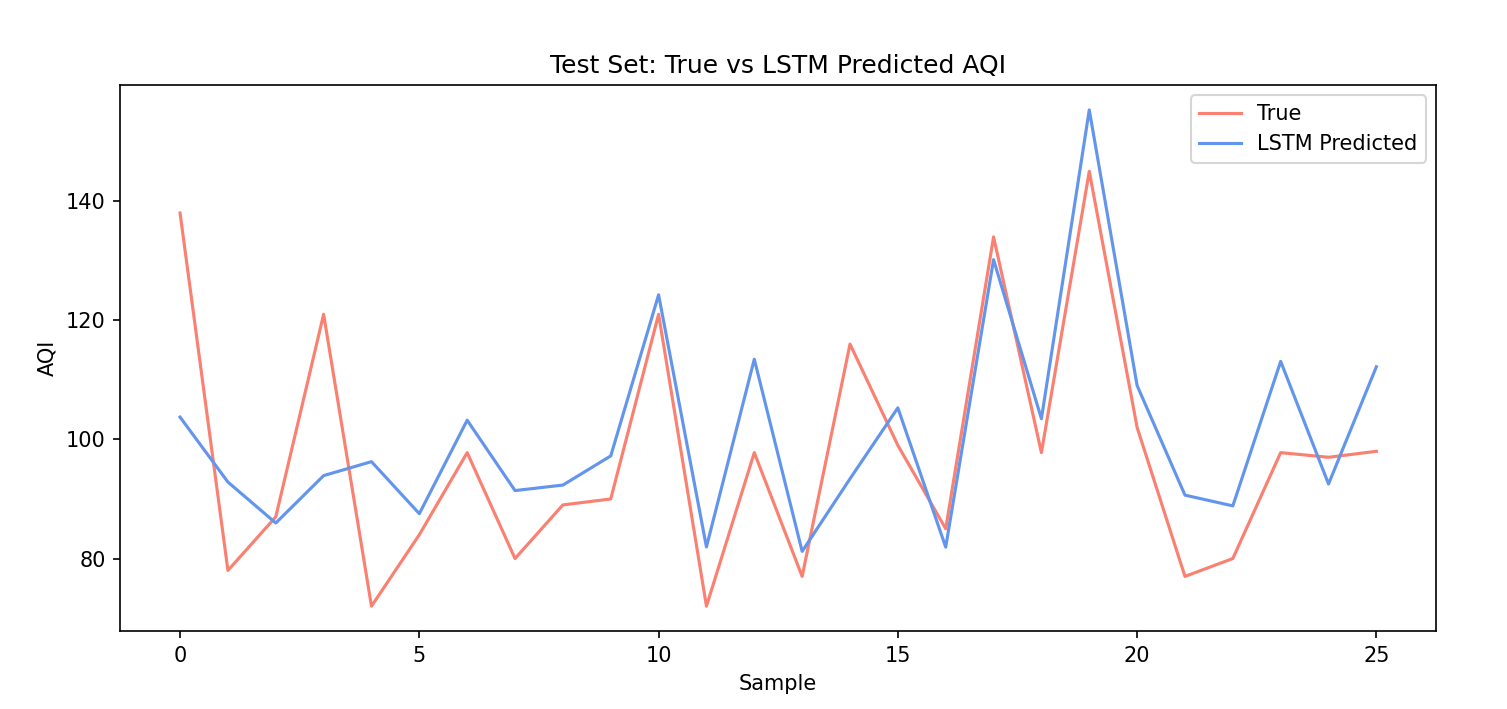
LSTM Test MSE Score: 198.930908203125

LSTM能进行基本的拟合和预测，但是不能达到最优的效果。

1. 基于随机森林的LSTM时间序列分析

首先使用随机森林模型对数据进行训练和预测，然后将随机森林的预测结果作为特征加入到LSTM模型中





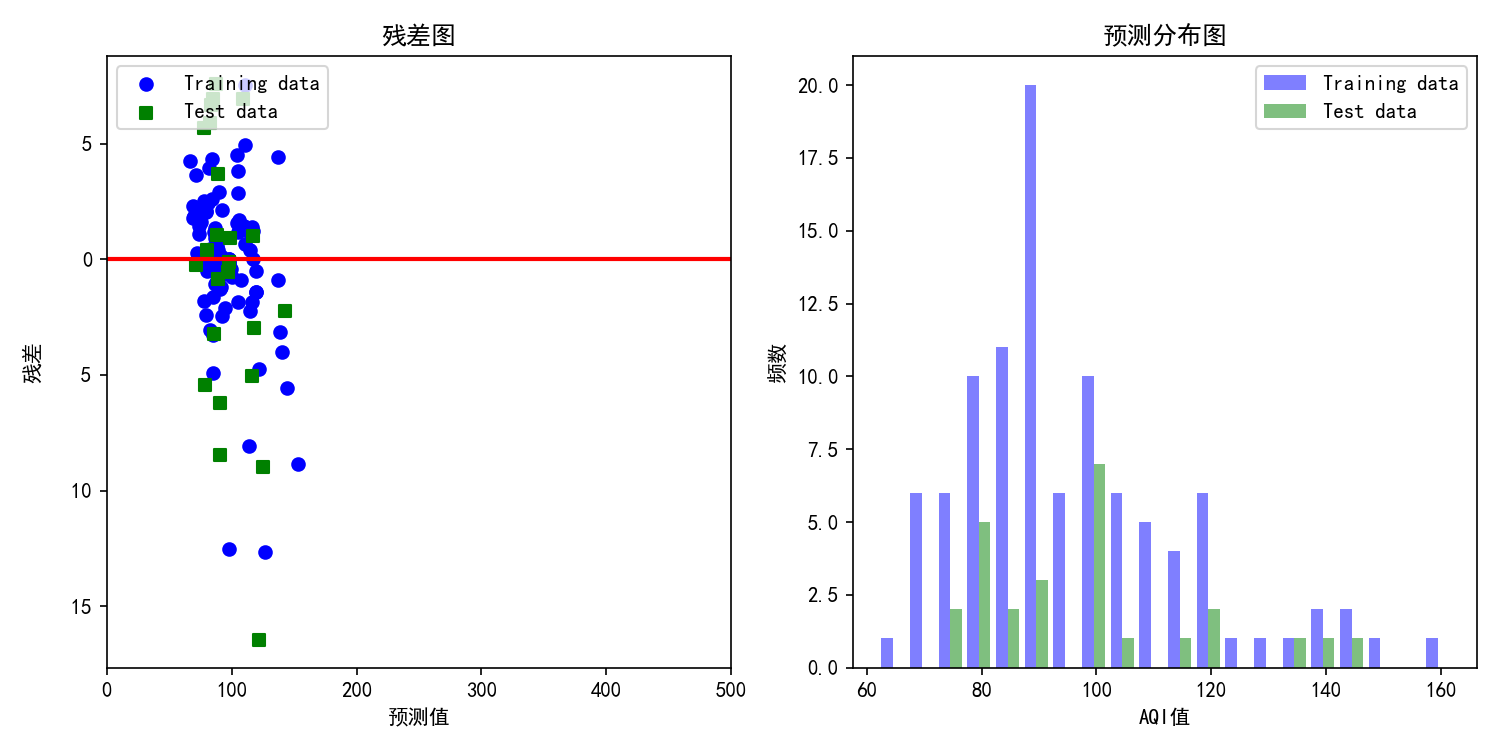
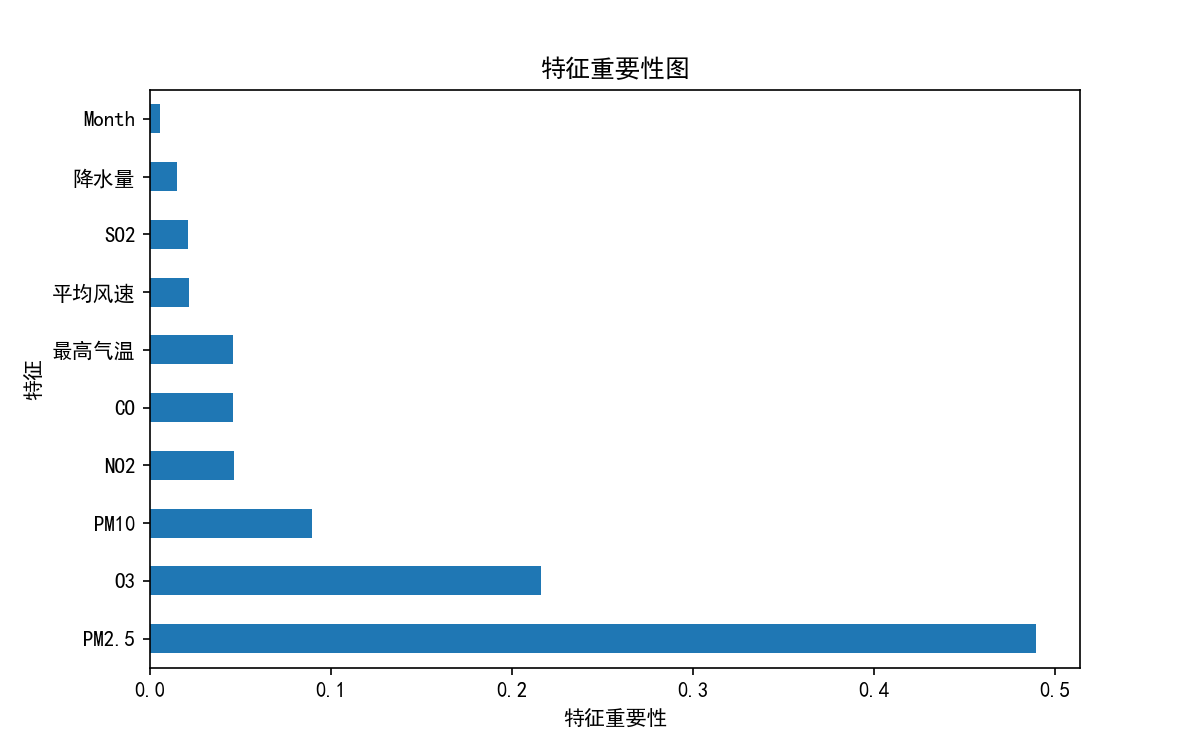
Random Forest Train R^2 Score: 0.977653914328056

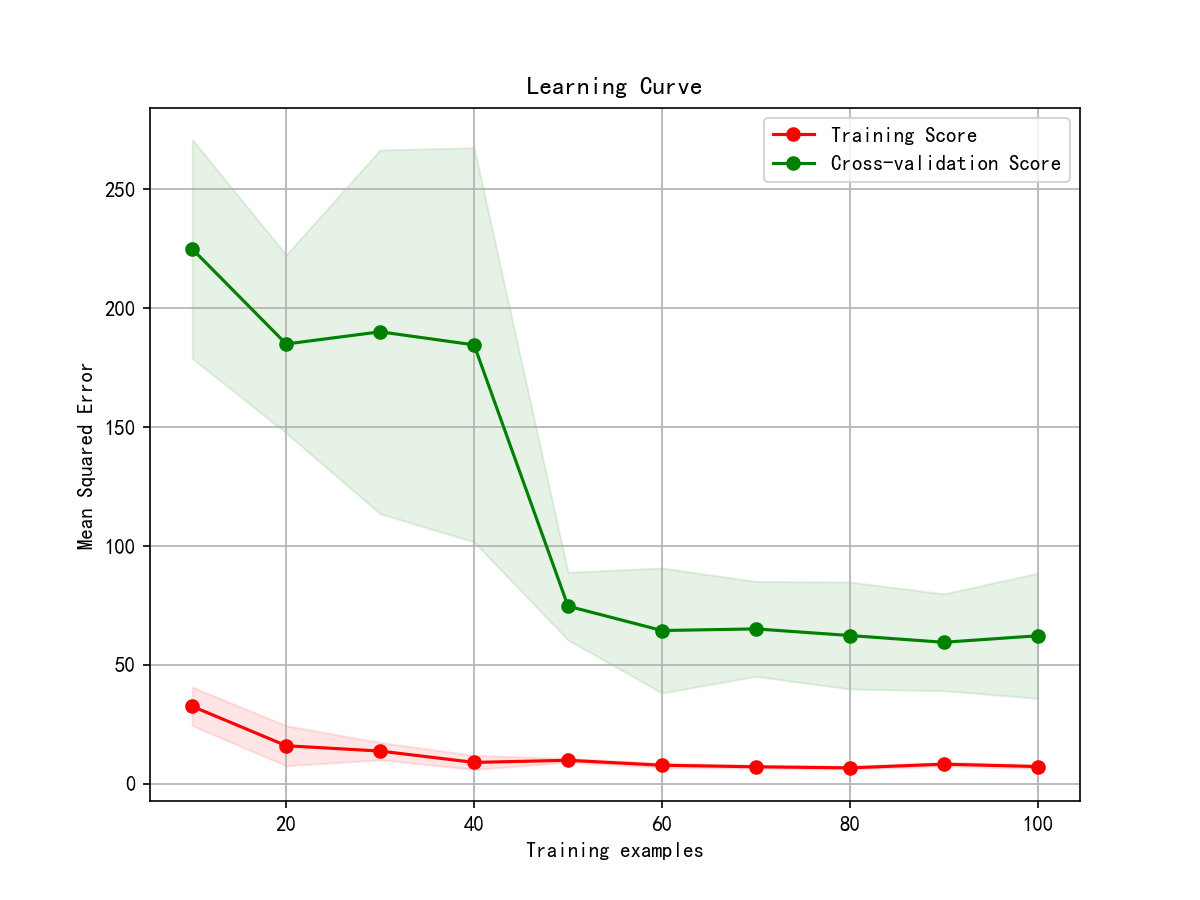
Random Forest Test R^2 Score: 0.9115190859292949

LSTM Train MSE Score: 150.00856018066406

LSTM Test MSE Score: 181.08969116210938

模型评估：





（对其他可视化的分析）

学习曲线：Training Score逐渐减小且趋于稳定，可见随着训练集大小的增加，模型在训练集上的性能逐渐提高，但随着数据量的增加，提高的幅度逐渐减小，最终达到一个稳定的水平。这表示模型能够很好地拟合训练数据。

但是Cross-validation Score并没有逐渐增大且趋于稳定，可见随着训练集大小的增加，模型在交叉验证集上的性能并未逐渐提高，反而在不断下降。