**基于规则引导的半监督日志异常检测系统**

我想尝试做一个web日志分析系统，在查阅相关资料后我了解到web日志每一条都包含了ip/time/method/path/status/size/referrer/users\_agent这几个数据

Ip和time就不说了

Method主要是看请求方式是get还是post

Path是访问路径，如果涉及到很多../这种跳往上级目录的也说明很有问题

Status是状态，就是访问状态，像我们平常出错的404、403这类的,如果同一个ip访问失败次数特别多基本上说明它非常可疑

referer告诉服务器你是从哪个页面点进来的，攻击者/脚本程序/扫描工具可能直接请求接口，不带referer，或者伪造一个奇怪的referer。

user\_agent告诉服务器你是谁、你是什么设备或浏览器,如果太短大概率也有问题

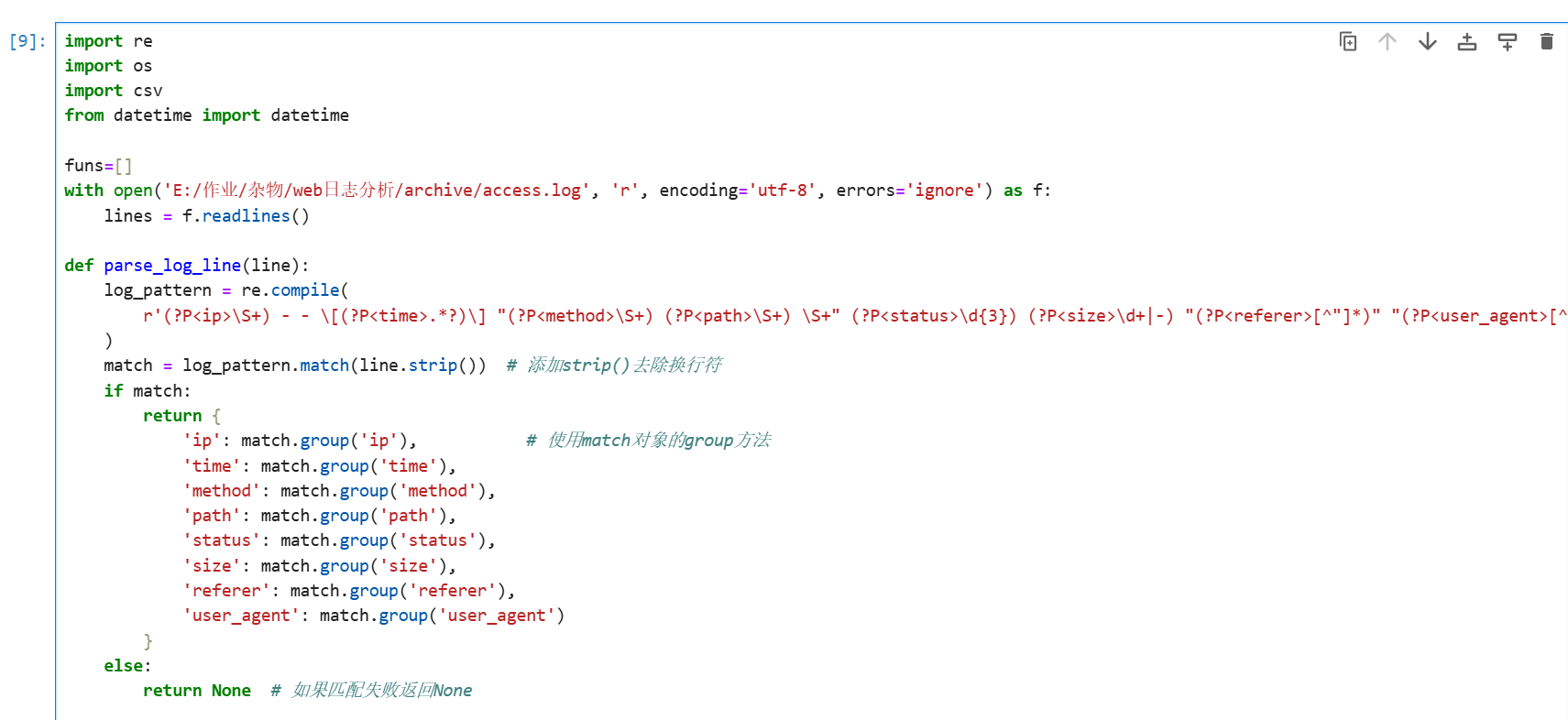
声明：使用数据集来源

<https://www.kaggle.com/datasets/eliasdabbas/web-server-access-logs>

下载压缩包解压后里面有个.log扩展名文件，我们要是用的就是这个

首先我们提取数据变成csv文件，数据提取需要用到正则表达式：

数据提取





正则表达式提取数据代码：

log\_pattern = re.compile(

r'(?P<ip>\S+) - - \[(?P<time>.\*?)\] "(?P<method>\S+) (?P<path>\S+) \S+" (?P<status>\d{3}) (?P<size>\d+|-) "(?P<referer>[^"]\*)" "(?P<user\_agent>[^"]\*)"'

)

参照一下正则表达式规则就能看懂，之前学习爬虫的时候对这些也略有涉猎，感觉还挺亲切的

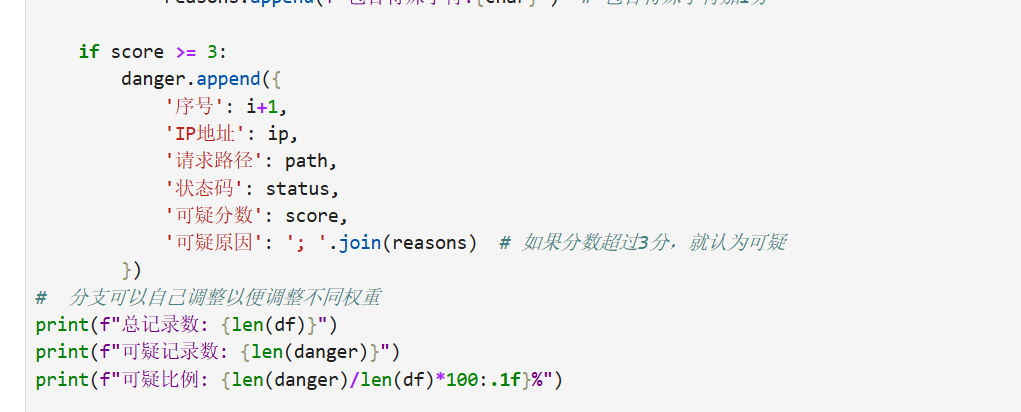
数据提取完成之后数据有点太多了，所以我就随机挑选20万使用。

检测数据

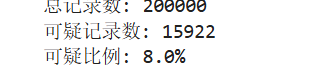
这一步其实我想了好久好久，我们自己怎么找出异常数据呢？最后使用的是打分的方式。路径太长加几分，有危险词汇（比如sql注入）也加几分，状态错误码的话在加几分，User-Agent太长或者太短、出现特殊字符（比如<\>%$@等）也会加分，分越高越可疑。

好处就是也方便调整权重，哪些主要特征，无关紧要的特征可以设置得分少一些。





结果是：



思路

首先数据是没法自己来一点点标记的，索性就直接思考无监督模型吧。我的思路是给每个数据评分后就能通过无监督学习去划定一个范围，正常数据肯定占大多数，得分低比较密集，恶意数据会比较分散。通过训练来划定一定范围然后据此范围来预测数据是正常的还是恶意的。但是无监督模型不好测试，因为我们谁也不知道测试数据的真正结果没法据此参照测试模型。

然后我又在想先每个数据跟据评分判断出是否是恶意的或者正常的，打上标签，然后将原本的字符串数据喂给模型让他根据日志本来的样子来判断。

用规则系统（你就是这样做的）给一小部分样本打标签（伪标签），训练一个初步模型用模型预测未标记样本，把置信度高的预测结果当成新标签加入训练集，再训练模型（迭代），这是伪标签+ 监督学习是半监督学习得方法。

说干就干，我用上边的代码制作伪标签，



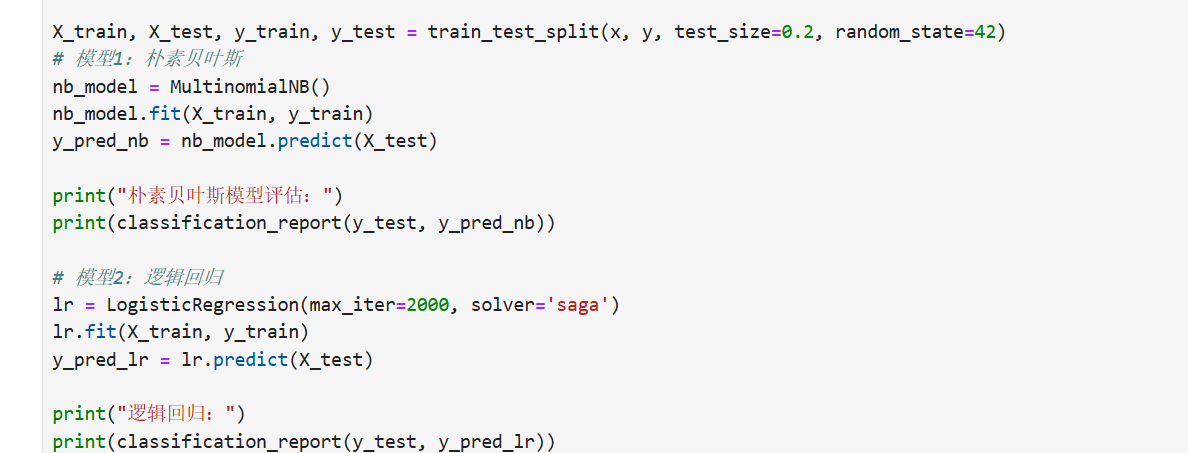
差不多和上边的代码一模一样，然后就有了新的问题，在平常的日志中可疑访问总归是少数的，我的20万数据中只有不到1.6万数据时可疑数据，样本极度不平衡。

但是我在想我们的数据本身是足够的而且非常多，我们仅仅只是使用了20万数据，我本来的数据是特别大的（有上千万条），那我抽300万数据打标签从中提取出20万可疑数据，然后将这20万可疑数据和20万普通数据混合不就够了？

成功了，现在数据平衡了，20万普通20万可疑，保证数据仍然真实有效。

监督学习

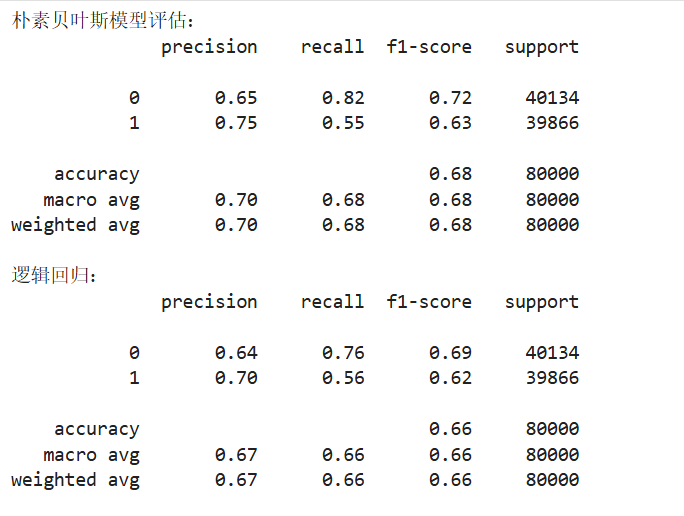




因为我们和之前的恶意url识别系统不一样，他是一个url对应一个label，这次我们日志分析使用的是多个数据path/method/user\_agent等，数据太多，所以我们需要一一处理然后合并成一个，所以中间处理数据有点麻烦。

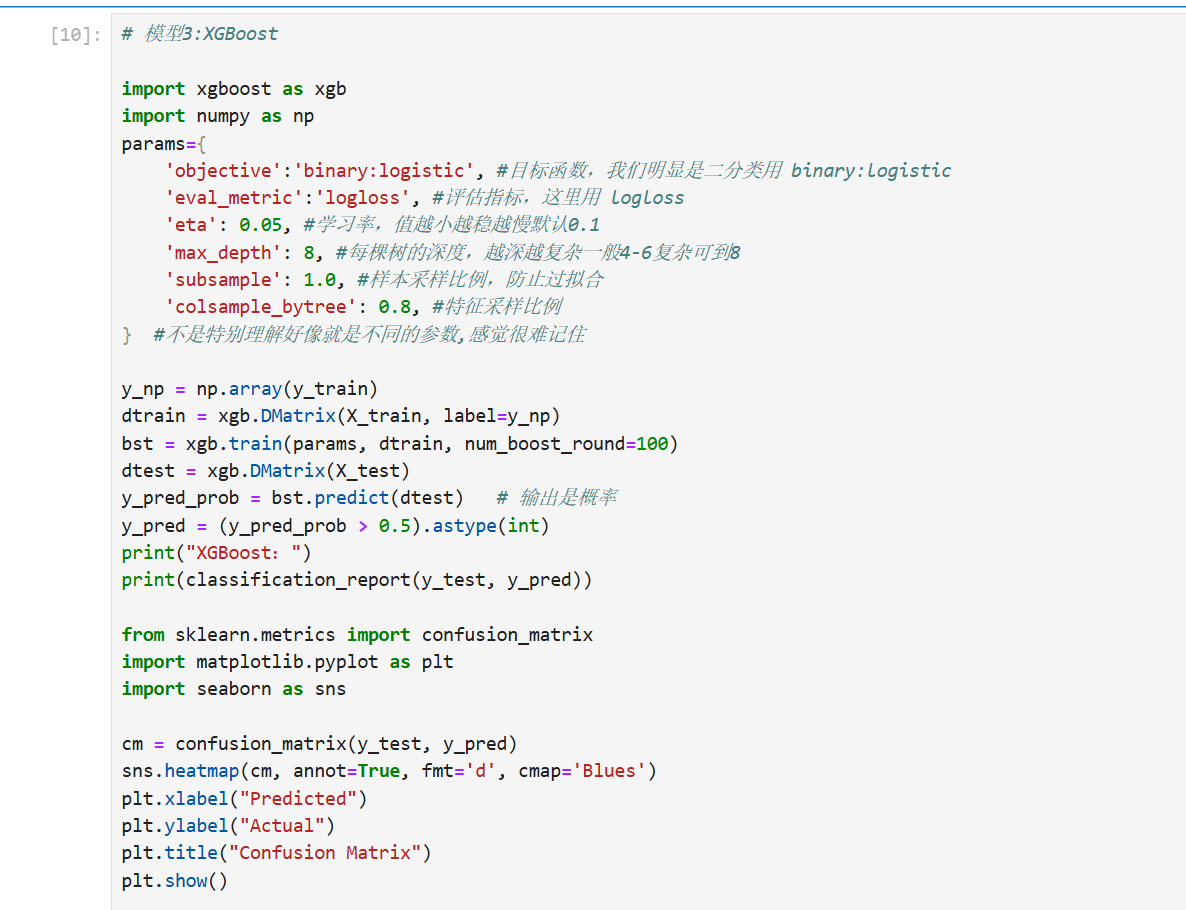
我们对比了朴素贝叶斯和逻辑回归两种。但是逻辑回归并未收敛所以我加大了迭代次数到2000次，仍未收敛。于是就saga求解器迭代2000次，终于收敛了。

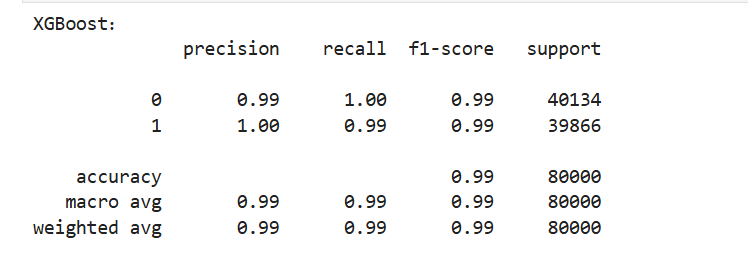
但是结果是这样的



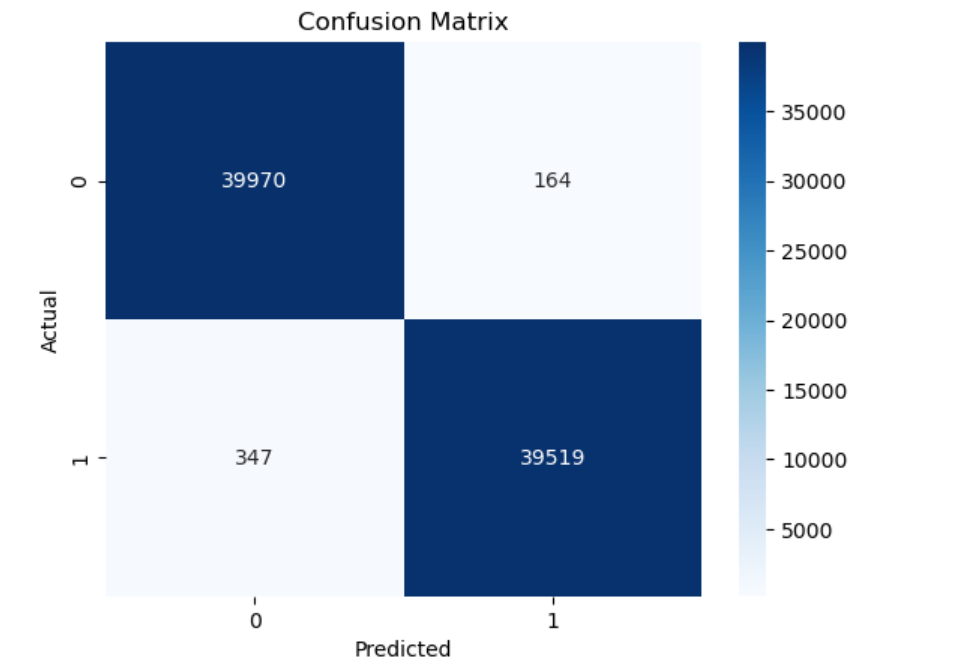
可以看到准确率一个在0.68一个在0.66，两者召回率都不算高（都只有 0.55-0.56），说明漏判很多恶意网址，结果都不尽如意。失败。

我又尝了一下XGBoost，但是结果有点离谱

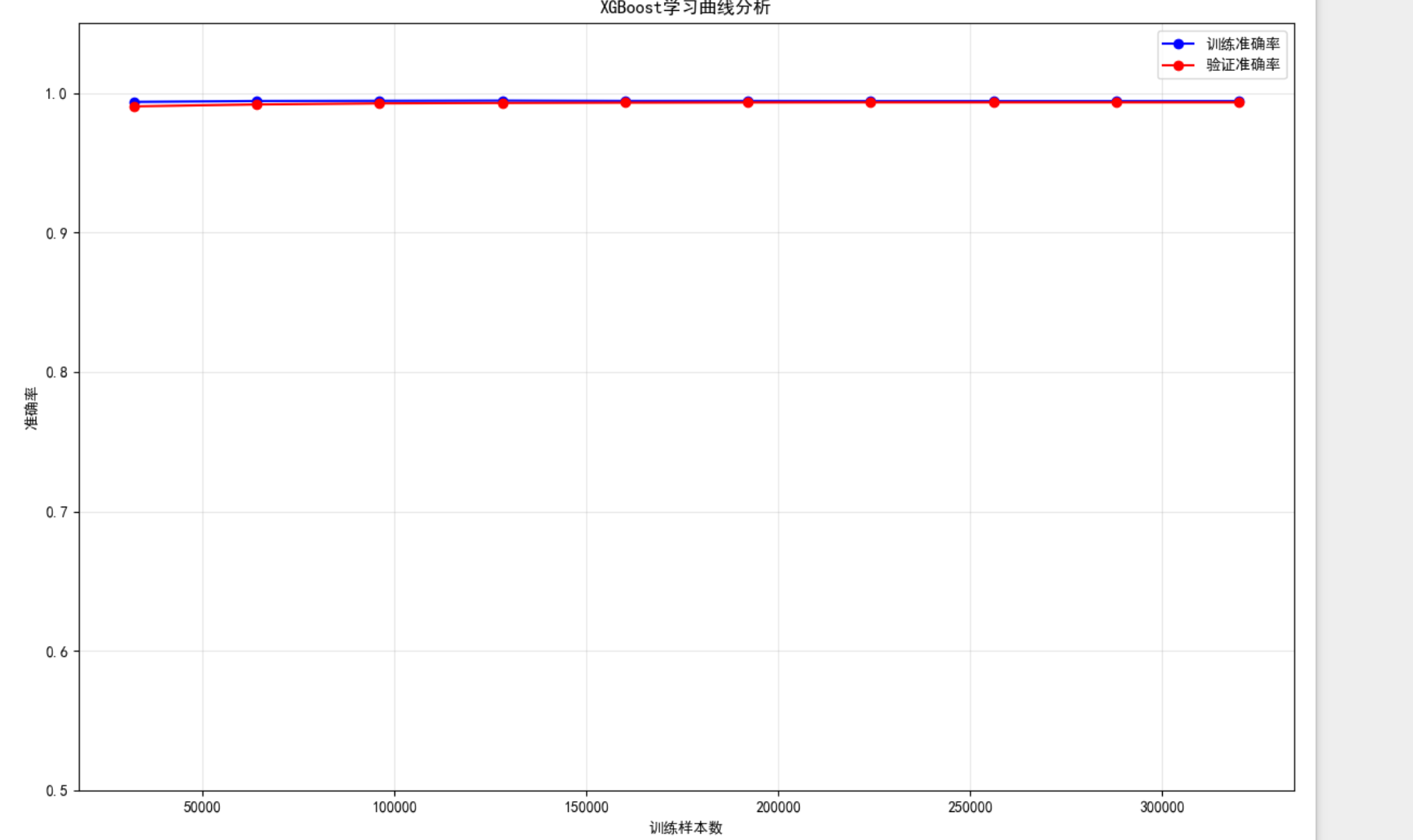


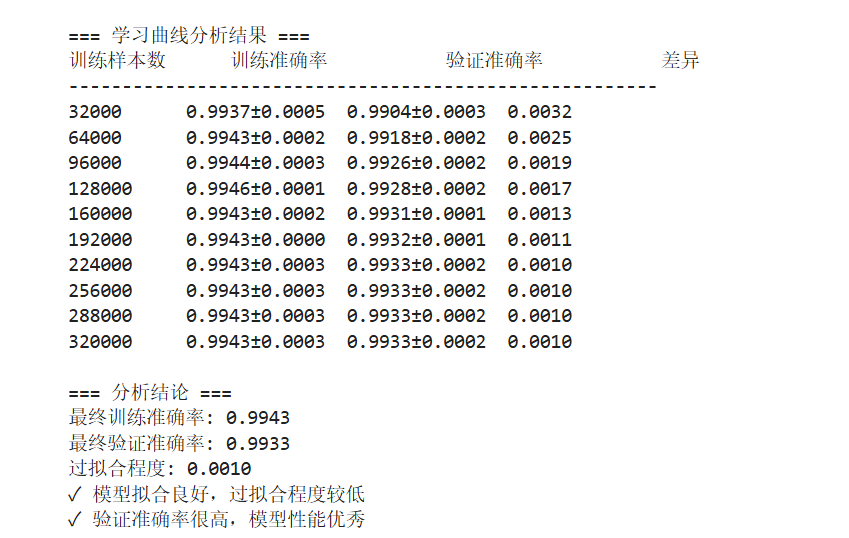
结果是这样的

准确率在**0.99**，感觉超级不靠谱



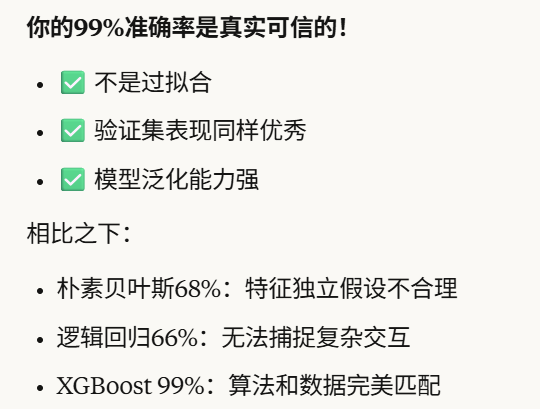
可视化看了一下，好像没发现什么问题。接下来我想验证一下模型是不是过拟合了，于是我让ai生成一个模型学习曲线分析，然后看结果





还是没什么问题，也没有过拟合。然后我百思不得其解始终不敢相信：



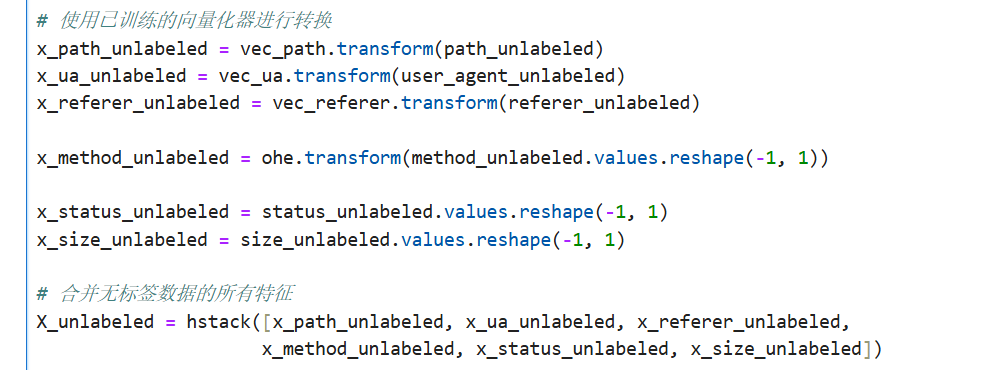


于是我就半信半疑，先保留看法吧。不过从这里也能看出XGBoost的优势。

半监督模型尝试

再次从原来的上千万数据中随机抽取40万数据，这个和训练用的我感觉应该很少概率会有重复的数据，基本可以忽略不计。这40万数据就不再打标签了，让模型来直接预测。然后找出置信度高的打上伪标签，然后再次进行新模型的训练再次预测一直重复迭代得到最终模型，这就是常用的半监督学习方式。

首先数据处理和之前的一模一样，记得不要再重新创建向量器就是用原来的就行



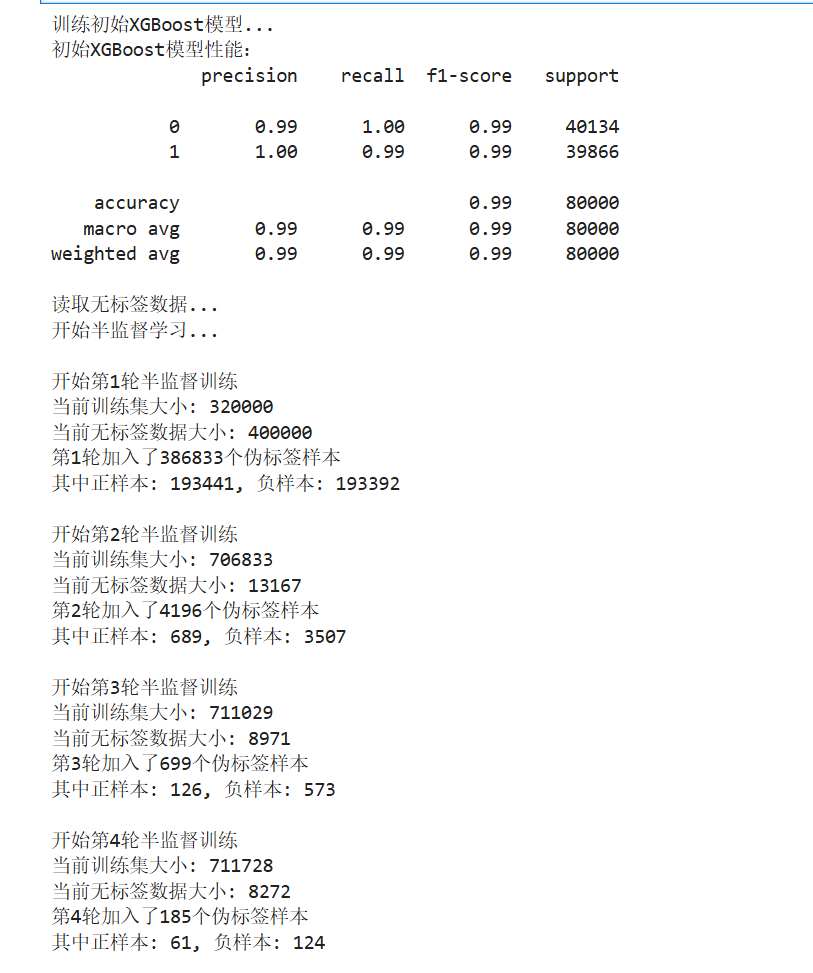
然后就开始半监督训练，不断进行迭代

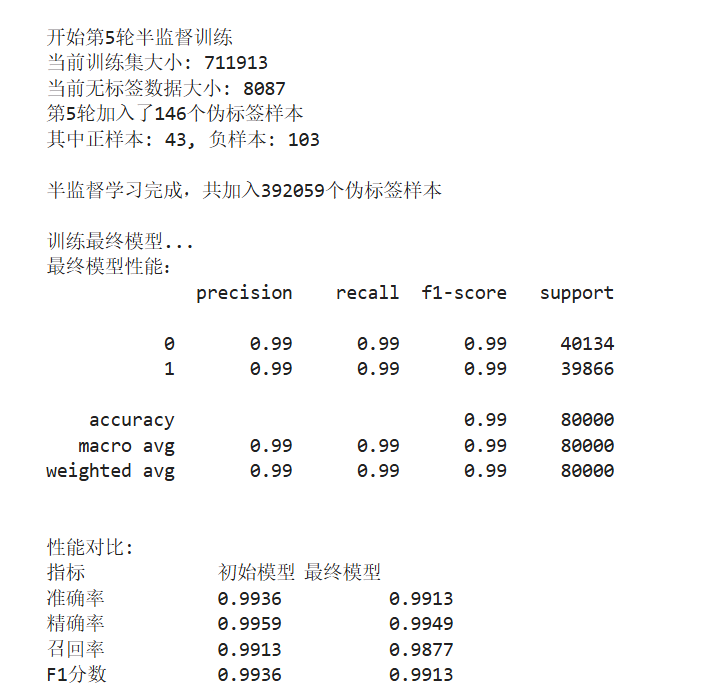




最后使用**固定测试集**的方法测试一下最终模型

结果：





可以看到模型精度还是在0.99，可能还是因为之前的那部分原因吧。总的来说来是很成功的。最后保存一下模型



结束！！！

这次算是加深了对半监督模型的理解，以及使用为标签的方法。同时也了解了安全日志的基本结构，及正则化表达。

这只是简单的学习尝试，并未涉及复杂考量，请注意