**西南财经大学**

[](http://www.swufe.edu.cn/version3/xuexiaogaikuang/xiaohui.jpg)Southwestern University of Finance and Economics

**课程论文**

**学 期：** **2020-2021第二学期**

**课程名称： 机器学习**

**论文题目：基于集成学习的公司培训人员留职预测**

**姓 名： 胡宇辰**

**学 号： 2201201Z5003**

**所在学院： 工商管理学院**

**专 业： 大数据管理**

**2021年 6 月**

**目录**

[1.理论部分： 3](#_Toc73219765)

[1.1 方差与偏差： 3](#_Toc73219766)

[1.1.1 方差与偏差的定义 3](#_Toc73219767)

[1.1.2 偏差与方差和集成学习的关系： 4](#_Toc73219768)

[1.2 集成学习介绍 4](#_Toc73219769)

[1.2.1 集成学习的定义 4](#_Toc73219770)

[1.2.2 集成学习的分类 5](#_Toc73219771)

[1.2.3模型学习方法介绍 5](#_Toc73219772)

[1.3模型简要介绍： 8](#_Toc73219773)

[2. 实验部分 11](#_Toc73219774)

[2.1背景简介 11](#_Toc73219775)

[2.2数据来源 11](#_Toc73219776)

[2.3.1将类别变量转换为数值型 12](#_Toc73219777)

[2.3.2缺失值处理 12](#_Toc73219778)

[2.3.3样本不平衡查看 13](#_Toc73219779)

[2.3.4相关性查看 14](#_Toc73219780)

[2.3.5标准化及划分数据集 14](#_Toc73219781)

[2.4实验一：留职预测实验 15](#_Toc73219782)

[2.4.1未调参结果 15](#_Toc73219783)

[2.4.2 调参说明 15](#_Toc73219784)

[2.4.3重要参数说明 16](#_Toc73219785)

[2.4.4调参后结果 16](#_Toc73219786)

[2.5实验二：模型对比实验 17](#_Toc73219787)

[2.5.1 比较策略 17](#_Toc73219788)

[2.5.2 实验结果截图 17](#_Toc73219789)

2.5.3 实验结果分析 18

[3.总结 20](#_Toc73219790)

# 基于集成学习的公司培训人员留职预测

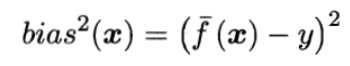
# 1.理论部分：

## 1.1 方差与偏差：

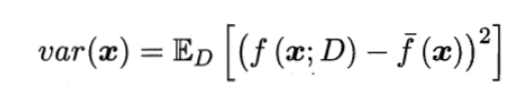
### 1.1.1 方差与偏差的定义

在机器学习中，我们用训练数据集去训练一个模型，通常的做法是定义一个误差函数，通过将这个误差的最小化过程，来提高模型的性能。然而我们学习一个模型的目的是为了解决训练数据集这个领域中的一般化问题，单纯地将训练数据集的损失最小化，并不能保证在解决更一般的问题时模型仍然是最优，甚至不能保证模型是可用的。这个训练数据集的损失与一般化的数据集的损失之间的差异就叫做泛化误差。

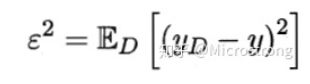
偏差是用所有可能的训练数据集训练出的所有模型的输出的平均值与真实模型的输出值之间的差异。



方差是不同的训练数据集训练出的模型输出值之间的差异。



噪声的存在是学习算法所无法解决的问题，数据的质量决定了学习的上限。假设在数据已经给定的情况下，此时上限已定，我们要做的就是尽可能的接近这个上限。



泛化误差可分解得到：

preview

### 1.1.2 偏差与方差和集成学习的关系：

Bagging算法是对训练样本进行采样，产生出若干不同的子集，再从每个数据子集中训练出一个分类器，取这些分类器的平均，所以是降低模型的方差（variance）。Bagging算法和Random Forest这种并行算法都有这个效果。

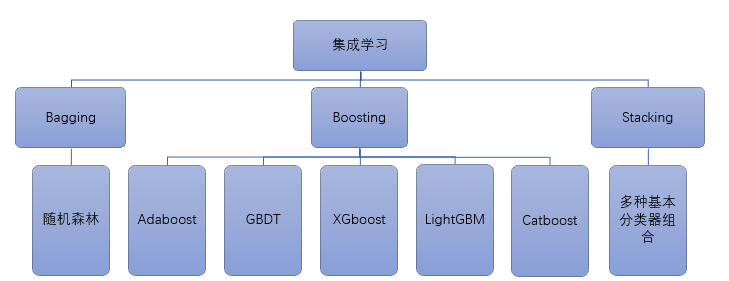
Boosting则是迭代算法，每一次迭代都根据上一次迭代的预测结果对样本进行权重调整，所以随着迭代不断进行，误差会越来越小，所以模型的偏差（bias）会不断降低。

## 1.2 集成学习介绍

### 1.2.1 集成学习的定义

集成学习 (Ensemble Learning) 算法的基本思想就是将多个分类器组合，从而实现一个预测效果更好的集成分类器。

### 1.2.2 集成学习的分类



### 1.2.3模型学习方法介绍

**（1）Bagging：**

**Bagging 的核心思路是 — — 民主。**

Bagging 的思路是所有基础模型都一致对待，每个基础模型手里都只有一票。然后使用民主投票的方式得到最终的结果。大部分情况下，经过 bagging 得到的结果方差更小。

**算法流程：**

从原始样本集中抽取训练集。每轮从原始样本集中使用Bootstraping的方法抽取n个训练样本（在训练集中，有些样本可能被多次抽取到，而有些样本可能一次都没有被抽中）。共进行k轮抽取，得到k个训练集。（k个训练集之间是相互独立的）

每次使用一个训练集得到一个模型，k个训练集共得到k个模型。（注：这里并没有具体的分类算法或回归方法，我们可以根据具体问题采用不同的分类或回归方法，如决策树、感知器等）

对分类问题：将上步得到的k个模型采用投票的方式得到分类结果；对回归问题，计算上述模型的均值作为最后的结果。（所有模型的重要性相同）



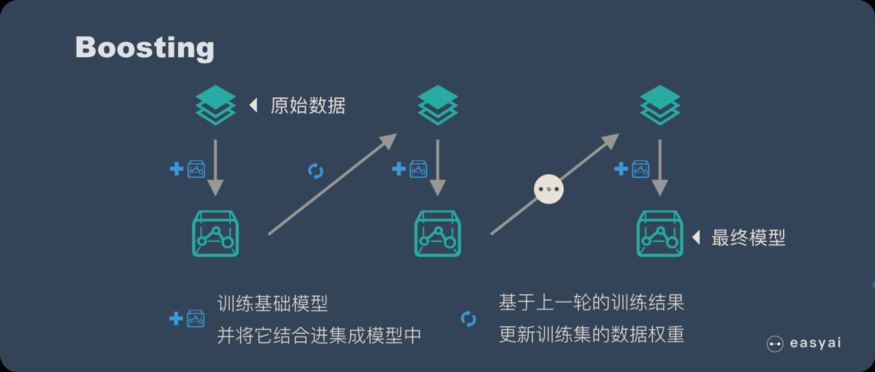
**（2）Boosting**

**Boosting 的核心思路是 — — 挑选精英。**

Boosting 和 bagging 最本质的差别在于他对基础模型不是一致对待的，而是经过不停的考验和筛选来挑选出「精英」，然后给精英更多的投票权，表现不好的基础模型则给较少的投票权，然后综合所有人的投票得到最终结果。大部分情况下，经过 boosting 得到的结果偏差更小。

**算法流程：**

1. 通过加法模型将基础模型进行线性的组合。
2. 每一轮训练都提升那些错误率小的基础模型权重，同时减小错误率高的模型权重。
3. 在每一轮改变训练数据的权值或概率分布，通过提高那些在前一轮被弱分类器分错样例的权值，减小前一轮分对样例的权值，来使得分类器对误分的数据有较好的效果。

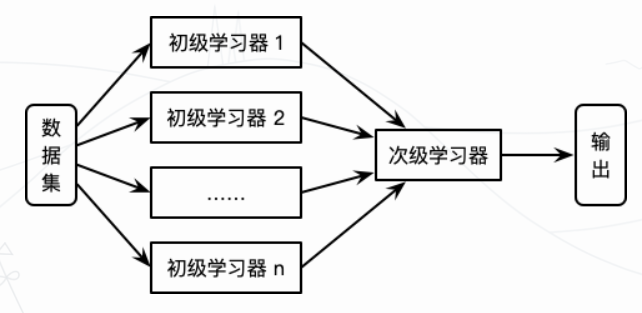


（3）Stacking

Stacking 的核心思路：

* 利用初级学习算法对原始数据集进行学习，同时生成一个新的数据集。
* 根据从初级学习算法生成的新数据集，利用次级学习算法学习并得到最终的输出。

注：对于初级学习器，可以是相同类型也可以是不同类型的。在新的数据集中，初级学习器的输出被用作次级学习器的输入特征，初始样本的标记仍被用作次级学习器学习样本的标记。Stacking 算法的流程如下图所示：



**（4）学习方法对比**

Bagging和Boosting的区别总结如下：

* 1. **样本选择上**： Bagging方法的训练集是从原始集中有放回的选取，所以从原始集中选出的各轮训练集之间是独立的；而Boosting方法需要每一轮的训**练集不变**，只是训练集中每个样本在分类器中的**权重发**生变化。而权值是根据上一轮的分类结果进行调整
  2. **样例权重上**： Bagging方法使用均匀取样，所以每个样本的权重相等；而Boosting方法根据错误率不断调整样本的权值，错误率越大则权重越大
  3. **预测函数上**： Bagging方法中所有预测函数的权重相等；而Boosting方法中每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重
  4. **并行计算上**： Bagging方法中各个预测函数可以并行生成；而Boosting方法各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果。

## 1.3模型简要介绍：

**（1）随机森林**：（<https://easyai.tech/ai-definition/random-forest/>）

Random Forest（随机森林）是 Bagging 的扩展变体，它在以决策树为基学习器构建 Bagging 集成的基础上，进一步在决策树的训练过程中引入了随机特征选择，因此可以概括 RF 包括四个部分：

1. 随机选择样本（放回抽样）；
2. 随机选择特征；
3. 构**建决策树；**
4. 随机森林投票（平均）。

**（2）AdaBoost: (提升错分数据点的权重)**

**（**<https://easyai.tech/ai-definition/adaboost/> ）

Adaboost算法的核心思想就是由分类效果较差的弱分类器逐步的强化成一个分类效果较好的强分类器。在创建第一个树之后，每个训练实例上的树的性能用于加权创建的下一个树应该关注每个训练实例的注意力。难以预测的训练数据被赋予更多权重，而易于预测的实例被赋予更少的权重。模型一个接一个地顺序创建，每个模型更新训练实例上的权重，这些权重影响序列中下一个树所执行的学习。构建完所有树之后，将对新数据进行预测，并根据训练数据的准确性对每棵树的性能进行加权。

**（3）GBDT:****(拟合梯度的残差)（**[**https://zhuanlan.zhihu.com/p/45145899**](https://zhuanlan.zhihu.com/p/45145899)**）**

GBDT是Gradient Boosting Decision Tree的简写，中文为梯度提升决策树，是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。GBDT由三个关键概念构成：Gradient Boosting (GB)，Regression Decision Tree (DT 或 RT) 和 Shrinkage。具体可参见网址

**（4）XGBoost: (二阶泰勒展开\正则项\列抽样\缺失值处理)**

**（**[**https://blog.csdn.net/wuzhongqiang/article/details/104854890**](https://blog.csdn.net/wuzhongqiang/article/details/104854890)**）**

如果boosting算法每一步的弱分类器生成都是依据损失函数的梯度方向，则称之为梯度提升(Gradient boosting)，XGBoost算法是采用分步前向加性模型，只不过在每次迭代中生成弱学习器后不再需要计算一个系数。XGBoost算法通过优化结构化损失函数（加入了正则项的损失函数，可以起到降低过拟合的风险）来实现弱学习器的生成，并且XGBoost算法没有采用搜索方法，而是直接利用了损失函数的一阶导数和二阶导数值，并通过预排序、加权分位数等技术来大大提高了算法的性能。

**（5）LightGBM: (基于histogram的决策树算法\直方图做差加速)**

**（**[**https://blog.csdn.net/wuzhongqiang/article/details/105350579**](https://blog.csdn.net/wuzhongqiang/article/details/105350579)**）**

Light对比GBDT和XGboost，采用了单边梯度抽样算法(GOSS)和互斥特征捆绑算法(EFB), 我们上面说到，GOSS可以减少样本的数量，而EFB可以减少特征的数量，这样就能降低模型分裂过程中的复杂度。

**（6）Catboost: (分类特征 \预测偏移)**

**（**[**https://mp.weixin.qq.com/s/xloTLr5NJBgBspMQtxPoFA**](https://mp.weixin.qq.com/s/xloTLr5NJBgBspMQtxPoFA)**）**

CatBoost是一种基于对称决策树（oblivious trees）为基学习器实现的参数较少、支持类别型变量和高准确性的GBDT框架，主要解决的痛点是高效合理地处理类别型特征，这一点从它的名字中可以看出来，CatBoost是由Categorical和Boosting组成。此外，CatBoost还解决了梯度偏差（Gradient Bias）以及预测偏移（Prediction shift）的问题，从而减少过拟合的发生，进而提高算法的准确性和泛化能力。

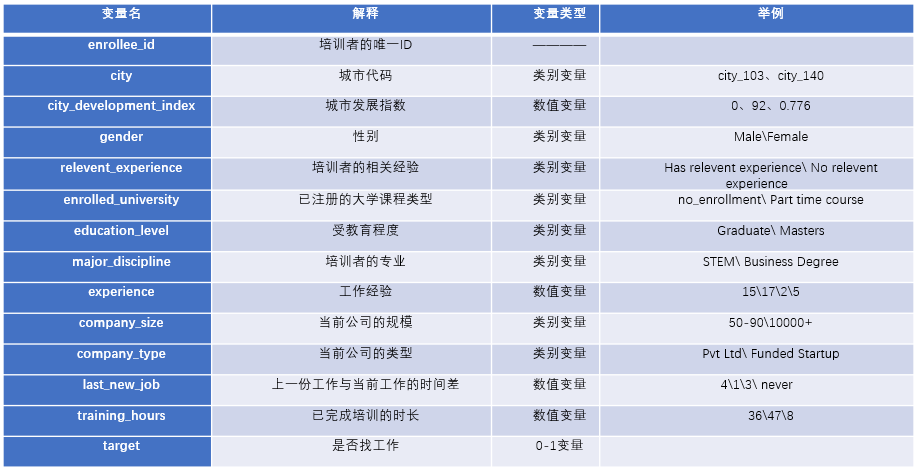
# 2. 实验部分

## 2.1背景简介

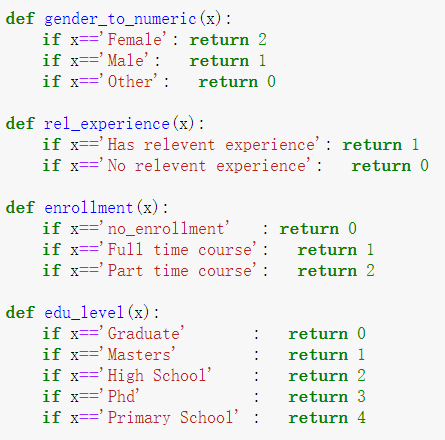
通常大部分数据公司会开展一些数据相关的培训课程，一方面可以为公司获得一定的名声和收益，另外一方面数据公司可以通过培训来寻找有意向留职的员工，本课程论文旨在利用集成学习相关方法，通过对参加培训者的相关特征，预测培训者寻找新工作或将为公司工作的可能性。

## 2.2数据来源

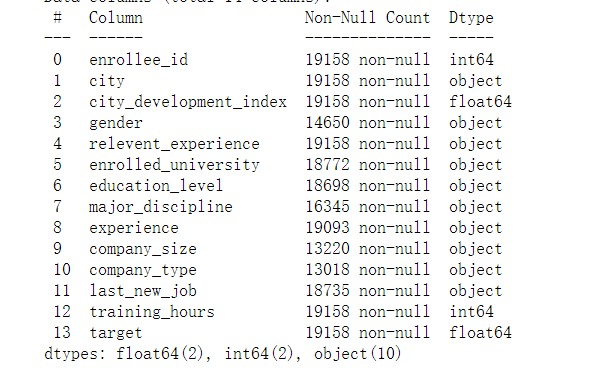
本文所用数据来自https://www.kaggle.com,包含19158条数据,共14个变量。具体变量及相关信息如下表所示：

2.3数据预处理

### 2.3.1将类别变量转换为数值型

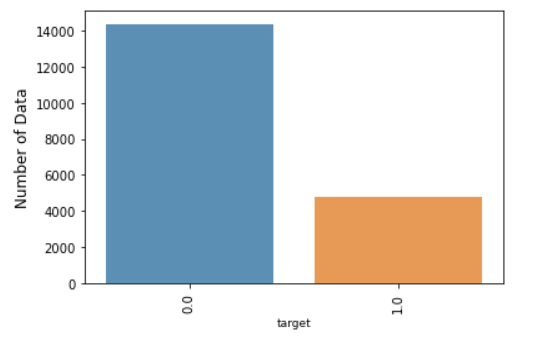


### 2.3.2缺失值处理



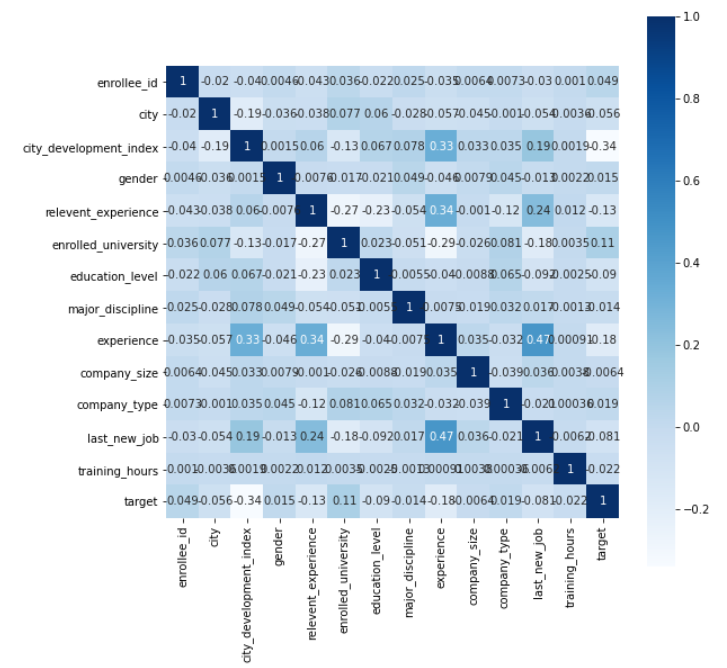
将变量‘city\_development\_index ’、‘gender’、‘enrolled\_university’、‘education\_level’、‘major\_discipline’、‘experience’、‘company\_size’、‘company\_type’、‘last\_new\_job’中的缺失值以均值填充。

### 2.3.3样本不平衡查看



正负样本比大约在1：3左右，样本不平衡不明显，故后续可不做样本不平衡处理。

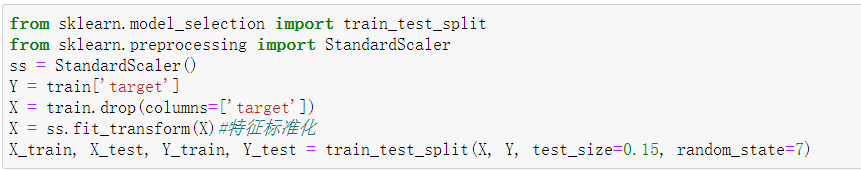
### 2.3.4相关性查看



变量之间相关性基本都在0.3以下，其中‘last\_new\_job’与‘experience’相关性达到0.47，但也是可接受范围以内，因此未对变量做相关性处理。

### 2.3.5标准化及划分数据集

将数据集以15%：85%划分为测试集与训练集。

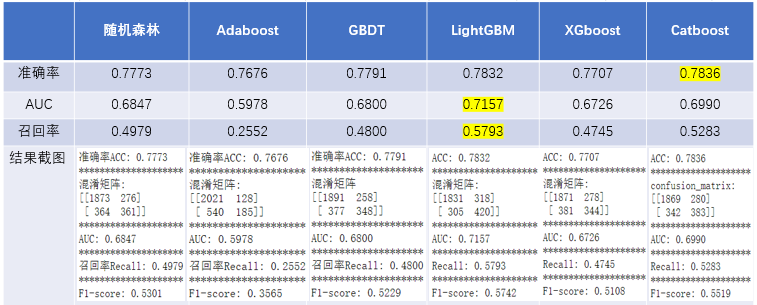


## 2.4实验一：留职预测实验

### 2.4.1未调参结果

选用了常用的六种集成算法模型对数据进行训练，预测培训者是否会在培训后留职。

未调参结果：



未调参结果展示出准确率最高的为Catboost模型，AUC与召回率最高的均为LightGBM模型，其中，LightGBM模型在各项指标上均保持良好的性能，能够达到约78.5%的预测准确率。

### 2.4.2 调参说明

**（1）调参目的：**偏差和方差的协调

**（2）**一般来说集成模型的重要参数分成两个部分——框架参数与基模型参数。

模型框架参数：

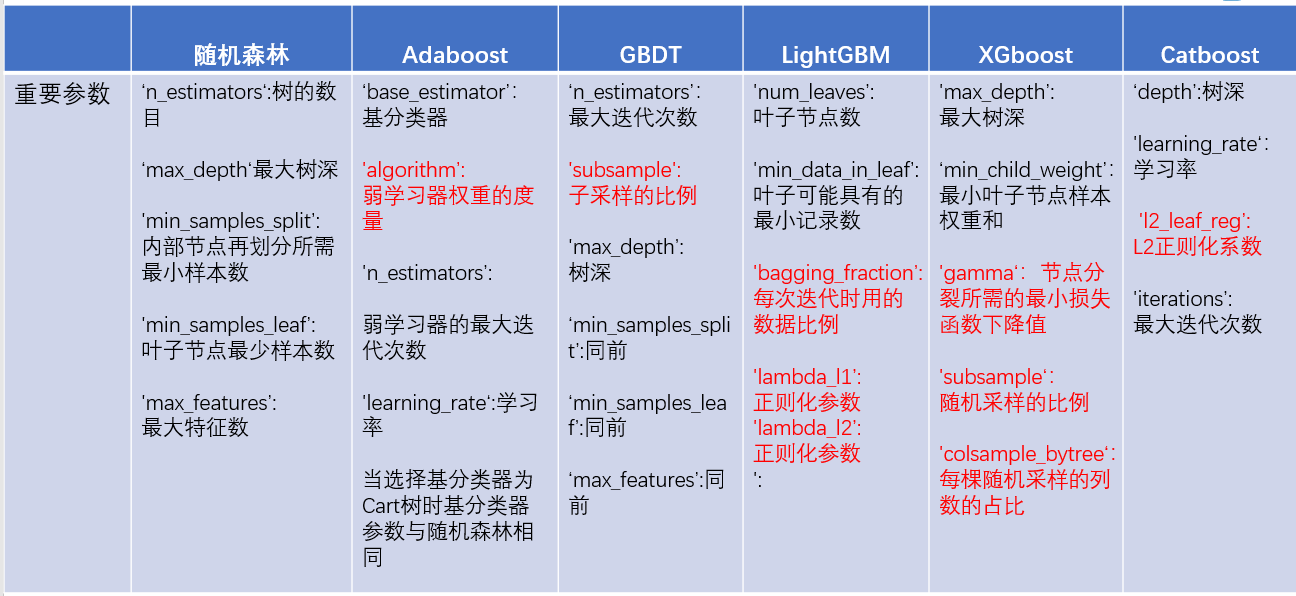
弱学习器的个数、学习率、泛化能力参数等

基模型参数：

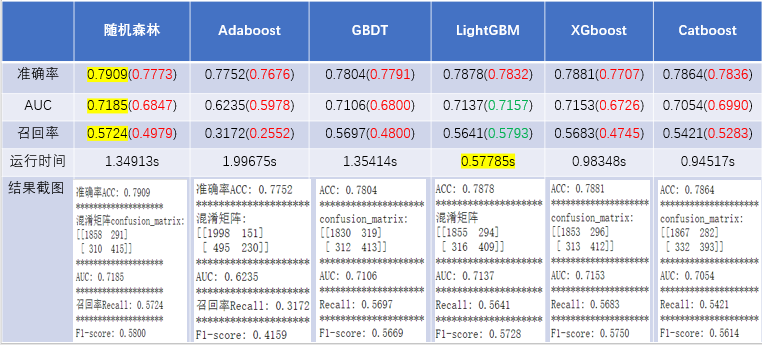
弱学习器的基本参数，例如随机森林中的决策树树的最大特征数、 最大树深、叶子节点最少样本数等

**（3）调参方法**：网格搜索

### 2.4.3重要参数说明



### 2.4.4调参后结果



调参后结果如上图所示，其中括号里的数字为为调参前的数据，红色代表调参后效果变好，绿色代表效果变差，对于LightGBM模型，AUC和召回率变差原因可能是因为LightGBM参数过多，调参未考虑到最好情况所致，其余模型调参后效果均变好。黄色背景代表效果最好的模型。从表格我们可以看到随机森林的预测效果是最好的，其预测准确率高到约80%。另外LightGBM的运行时间仍然是最短的。

## 2.5实验二：模型对比实验

通过以集成方法与基分类器的组合对比不同方法下模型的准确性、AUC、运行时间等

### 2.5.1 比较策略

**集成方法选择：**

Bagging、Adaboost、Stacking

**基分类器选择：**

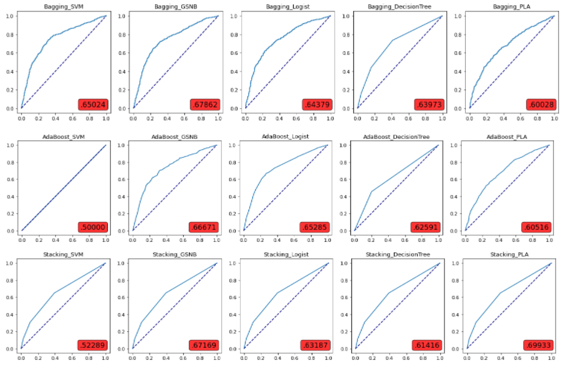
SVM、贝叶斯分类器、逻辑回归、决策树、感知机

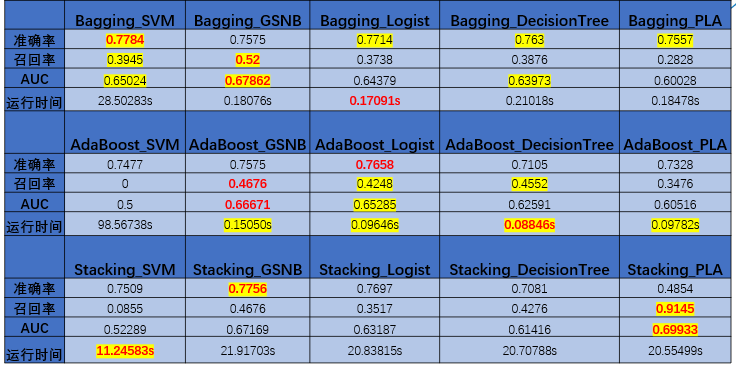
**采取两两组合的方式生成共15个模型进行比较**

（其中因为Stacking方法较为特殊，因此在选择其分类器时将其中一种基分类器作为第二层分类器即最终分类器，其他分类器作为第一层分类器）

### 2.5.2 实验结果截图



2.5.3实验结果分析



表格中红色数字代表在学习方法一定时，不同基分类器的最佳得分，黄色背景代表在基分类器一定时，不同学习方法的最佳得分。

当采用Bagging学习方法时，SVM基分类器的准确率最高，贝叶斯分类器的召回率与AUC最高，Logist的运行时间最短。当采用AdaBoost时，Logist的准确率最高，贝叶斯分类器的召回率及AUC最高，决策树运行时间最短。当采用Stacking学习方法时，贝叶斯分类器的准确率最高，感知机的召回率及AUC最高，SVM运行时间最短。

当采用SVM基分类器时，Bagging方法的准确率，召回率和AUC最优，Stacking运行时间最短。当采用贝叶斯基分类器时，Bagging方法的召回率和AUC最优，Adaboost的运行时间最短，Stacking的准确率最高。采用Logist基分类器时，，Bagging方法的准确率最高，Adaboost的运行时间，召回率和AUC最优。当采用决策树基分类器时，Bagging方法的准确率和AUC最高，Adaboost的召回率和运行时间最优。当采用感知机基分类器时，Bagging学习方法的准确率最高，Stacking学习的召回率和AUC最优，Adaboost的运行时间最短。

几点特别的地方：

1. 在15个模型中，准确率最高的为Bagging\_SVM模型，召回率最高的为Stacking\_PLA模型，AUC最高的为Stacking\_PLA模型，运行时间最短的为Bagging\_Decision Tree模型。综合表现最好的学习方法为Bagging学习方法，最好的基模型为贝叶斯分类。
2. AdaBoost\_SVM的召回率为0，AUC为0.5，考虑其原因可能是因为SVM本身就为强分类器，对其使用Boosting学习方法使其对训练数据的拟合程度过高（过拟合），因此导致在预测集上的分类错误。
3. Stacking\_SVM运行时间最短，原因在于Stacking算法的基分类器并未包含SVM算法，因此在第一层训练时时间大大减少。

# 3.总结

本报告首先介绍了集成模型的一些基本理论，包括偏差与方差，集成学习方法和集成模型，其次针对公司培训人员的相关特征进行了留职预测实验和模型对比实验。

本报告的不足之处在于：1、个人对模型的理解还不够深入。2、在对模型调参时效果较差，3、进行对比实验时基模型的参数未进行一个好的调参。4、对比实验中只进行了浅层的对比。5、数据量较小，对比效果不明显。