**华中科技大学**

**计算机科学与技术学院**

**《机器学习》结课报告**



专 业： 计算机科学与技术

班 级： 计算机202008班

学 号： U202015533

姓 名： 徐瑞达

成 绩：

指导教师： 何琨

**完成日期： 2022年 7 月 3 日**

目录

[1. 实验题目: 智能家居使用场景识别挑战赛 2](#_Toc107763341)

[2. 实验要求 2](#_Toc107763342)

[2.1 实验任务 2](#_Toc107763343)

[2.2 数据说明 2](#_Toc107763344)

[2.2.1 数据集组成说明 2](#_Toc107763345)

[2.2.2 数据量统计说明 2](#_Toc107763346)

[2.3 评测标准 3](#_Toc107763347)

[3. 算法设计 3](#_Toc107763348)

[3.1 数据处理 3](#_Toc107763349)

[3.2 模型算法 4](#_Toc107763350)

[3.3 模型融合 4](#_Toc107763351)

[4. 实验环境与平台 4](#_Toc107763352)

[5. 实现与分析 4](#_Toc107763353)

[5.1成员分工 4](#_Toc107763354)

[5.2数据预处理（撰写:刘鉴之） 4](#_Toc107763355)

[5.2.1数据读取 4](#_Toc107763356)

[5.2.2数据统计 5](#_Toc107763357)

[5.2.3合并特征 6](#_Toc107763358)

[5.3特征工程（撰写:共同完成） 7](#_Toc107763359)

[5.4模型预测一（撰写:刘鉴之） 7](#_Toc107763360)

[5.4.1 逻辑回归模型 7](#_Toc107763361)

[5.4.2 决策树模型 7](#_Toc107763362)

[5.4.3 随机森林模型 8](#_Toc107763363)

[5.5模型预测二（撰写:徐瑞达） 8](#_Toc107763364)

[5.5.1 LightGBM模型 8](#_Toc107763365)

[5.5.2 LightGBM模型调参 9](#_Toc107763366)

[5.5.3 XGBoost模型 9](#_Toc107763367)

[5.5.4 XGBoost模型调参 10](#_Toc107763368)

[5.6 模型融合（撰写:徐瑞达） 10](#_Toc107763369)

[5.7 代码整理与优化（撰写:徐瑞达） 10](#_Toc107763370)

[6. 实验结果 10](#_Toc107763371)

[7. 个人体会 10](#_Toc107763372)

# 实验题目: 智能家居使用场景识别挑战赛

某开展智能家居业务的公司在全国各地设有不同等级的代理商，为了让用户切身感受到智能家居产品的智能化和便捷性，每个代理商均有自己的智能家居体验店和展厅。

根据该公司的发展策略，需要让终端用户在智能家居体验过程中能够更加强烈、清晰的感受到智能家居系统的优点，系统会针对展厅类的场景做特殊的场景优化。为了实现这一场景下的优化，就**需要系统能够准确、快速的分析出当前智能家居产品使用环境是真实的家庭还是智能化体验的公共区域**。

# 2. 实验要求

## 2.1 实验任务

根据账号信息、设备列表、控制操作日志、设备上报日志等数据，分析账号对应的智能家居产品体系的应用场景。

## 2.2 数据说明

### 2.2.1 数据集组成说明

* 训练集和测试集均包含四类数据，测试集包含使用场景标签，用于识别算法训练，第二部分不包含场景标签，用于测试。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据类别 | 对应文件 | 变量 | 格式 | 说明 |
| 账号信息 | cus.csv | uid | string | 账号ID |
| label | int | 智能家居产品体系的使用场景 |
| 设备列表 | devList.csv | uid | string | 账号ID |
| did | string | 设备ID |
| type | string | 设备型号 |
| area | string | 设备所在区域 |
| 控制操作日志 | control.csv | uid | string | 账号ID |
| did | string | 远程控制的设备ID |
| time | bigint | 远程控制设备的时间 |
| form | string | 远程控制设备的方式 |
| data | string | 远程对设备下发的控制日志 |
| 设备上报日志 | devUpdate.csv | uid | string | 账号ID |
| did | string | 上报日志的设备ID |
| time | bigint | 设备上报日志的时间 |
| data | string | 设备上报的日志内容 |

表2.1 数据集组成

### 2.2.2 数据量统计说明

* 测试集的账号信息文件与训练集相比，不包含label列；
* 控制操作日志和设备上报日志中，data列部分数据存在json型数据，需要进行额外的读取与处理；
* 设备列表的type与area列、控制操作日志的form列，可能会对预测结果产生影响，且均为文本数据，需要进行编码或者词频统计。

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集 | 数据类别 | 规模 | 备注 |
| 训练集 | cus.csv | **915×2** |  |
| devList.csv | 8858×4 | area列数据可能含逗号 |
| control.csv | 334006×5 | 存在json型数据 |
| devUpdate.csv | 346797×4 | 存在json型数据 |
| 测试集 | cus.csv | **267×1** |  |
| devList.csv | 2501×4 | area列数据可能含逗号 |
| control.csv | 112371×5 | 存在json型数据 |
| devUpdate.csv | 113574×4 | 存在json型数据 |

表2 数据量统计

## 2.3 评测标准

本实验采用指标对预测结果进行评价：

（1）统计（正确预测环境场景），（错将家庭场景预测为公共区域），（错将公共区域预测为家庭记录）

（2）使用（1）中统计值，计算模型准确率和召回率，公式如下：

（3）使用第二步计算结果，计算,得到评测指标，公式如下：

# 3. 算法设计

本实验属于**二分类**问题，需要根据数据集预测智能家居的应用场景是**家庭场景**还是**公共区域。**

## 数据处理

### 3.1.1数据读取

由于数据集中的devList.csv、control.csv、devUpdate.csv等文件，均在某一列的数据中包含逗号，因此需要额外进行处理，因此定义了函数my\_readcsv处理这类数据。

### 3.1.2数据分析

由于数据集中包含不同数据类别的文件，因此需要对不同数据类别中的数据进行统计。同时，应该对不同数据类别中变量取值不同可能会对预测结果造成影响的变量进行编码或者词频统计，以得到新的特征便于特征工程的构建。

1. 设备列表中的area变量
2. 设备控制日志中的form变量

### 3.1.3特征工程

使用相关系数法、递归特征消除法、模型分析法、人工选择法，分别选择相关度较高的特征，取并集后得到选择的特征子集。

1. 相关系数法

通过计算特征与特征之间的**相关系数**的大小，可**判定两两特征之间的相关程度**。

两特征间的相关系数corr(x1,x2)取值区间在[-1, 1]之间，取值关系如下：

* 相关系数值小于0表示负相关，即x1与x2是**互补**特征；
* 相关系数值等于0表示无相关；
* 相关系数值大于0表示正相关，即x1与x2是**替代**特征；

由此，可以取相关系数值的绝对值，把值大于90%~95%的两两特征中的某一个特征剔除，这样，便实现了对特征的筛选与数据降维，避免不重要的特征过多，使得模型的性能下降。

1. 递归特征消除法

sklearn.feature\_selection提供的RFE利用以下原理实现对特征的筛选：

* 将待筛选的k个特征作为初始特征子集输入到随机森林分类器中，计算每个特征的重要性，并利用交叉验证方法得到初始特征子集的分类精度；
* 将特征重要性最低的一个特征移除后得到的特征子集再次输入到随机森林分类器中，重新计算每个特征的重要性和该特征子集的分类精度；
* 递归地重复第二步，直至特征子集为空，最后得到k个不同特征数量的特征子集，选择**分类精度最高的特征子集**作为最优特征组合。

1. 模型分析法
2. 人工选择

缺失值处理，异常值处理，分析数据分布情况等。对于模型的效果非常重要

时间允许的话可以分析各个特征对预测值的影响（相关性）决定采用什么特征，可以和决策树部分“信息增益”知识点联系起来

## 3.2 模型算法

通过对逻辑回归、决策树、随机森林等不同模型进行准确率评估，最终选取随机森林模型、LightGBM模型、XGBoost模型参与模型融合并进行最后的预测。

1. 逻辑回归模型（Logistic regression）

虽然逻辑回归模型中带有“回归”二字，但逻辑回归模型其实是一个分类模型。逻辑回归模型的模型简单，而且可解释性强。该模型的具体参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 类型 | 默认值 | 说明 |
| tol | float | 0.0001 | 设置停止迭代求解的阈值 |
| max\_iter | int | 100 | 设置最大的迭代次数 |
| multi\_class | string | ‘auto’ | 设置分类方式 |
| random\_state | int | None | 设置随机数种子 |

1. 决策树模型

由课内知识可知，构造决策树的基本算法有多种，如使用信息增益进行特征选择的ID3算法，分类回归树CART算法。对于分类问题，决策树的最后一层叶子结点才是分类标签。该模型的具体参数如下：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 默认值 | 说明 |
| splitter | ‘best’ | 设置找到最佳切分点的特征范围 |
| max\_depth | None | 防止过拟合 |
| max\_leaf\_nodes | None | 通过限制最大叶子节点数防止过拟合 |
| min\_impurity\_split | - | 设置阈值以限制决策树的增长 |

1. 随机森林模型

随机森林模型通过建立多个决策树并将它们融合起来得到一个更加准确和稳定的模型，其随机性体现在随机选取特征和随机选取样本。该模型的具体参数如下：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 默认值 | 说明 |
| n\_estimators | 100 | 设置树的棵树 |
| max\_depth | - | 设置树的最大深度，防止过拟合 |
| min\_samples\_leaf | - | 设置叶子的最小样本数量 |
| min\_samples\_split | - | 设置分支节点的最小样本数量 |

1. LightGBM模型

LightGBM是由微软推出的可扩展机器学习系统，是基于GBDT（梯度提升决策树）算法的分布式梯度提升框架。LightGBM模型简单易用，鲁棒性强，直接支持缺失值与类别特征，无需对数据额外进行特殊处理。该模型的具体参数如下：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 分类 | 说明 |
| num\_leaves | 基本参数 | 设置树的模型复杂度，较大时准确率较高，但可能导致过拟合 |
| max\_depth | 基本参数 | 设置树的深度，防止过拟合 |
| feature\_fraction | 针对训练速度 | 使用特征的子抽样 |
| learning\_rate | 针对准确率 | 选择较小的学习率能获得稳定较好的模型性能 |
| num\_iterations | 针对准确率 | boosting的迭代次数，过大会导致过拟合 |
| min\_child\_sample | 针对稳定性 | 一个叶子上的最小数据量，用于提高模型泛化能力 |

1. XGBoost模型

XGBoost实现了梯度提升树(GBDT)模型，可以自动处理缺失值，也常用于分类问题。该模型的具体参数如下：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 默认 | 说明 |
| gamma | 0 | 指定节点分裂所需最小损失函数下降值，值越大，算法越保守 |
| max\_depth | 6 | 设置树的深度，防止过拟合 |
| learning\_rate | 0.3 | 选择较小的学习率能获得稳定较好的模型性能 |
| subsample | 1 | 用于避免过拟合 |
| colsample\_bytree | 1 | 控制每棵随机采样的列数的占比，即控制特征采样 |

## 3.3 模型融合

模型融合往往可以无痛涨点，融合有很多种方式，相同模型不同参数预测值取平均，不同模型预测值取平均，或是boosting等等

# 4. 实验环境与平台

|  |  |
| --- | --- |
| **环境配置** | **具体版本** |
| 操作系统 | Windows11专业版22H2 |
| 处理器 | Intel(R) Core(TM) i5 1.00GHz/8GB RAM |
| 编程语言 | Python 3.10.2 |
| 集成开发环境IDE | PyCharm 2022.1.3(Professional Edition) |

# 5. 实现与分析

## 5.1成员分工

|  |  |
| --- | --- |
| **具体模块** | **分工划分** |
| 数据读取、数据统计 | 刘鉴之 |
| 特征工程 | 共同完成 |
| 逻辑回归、决策树、随机森林模型 | 刘鉴之 |
| LightGBM、XGBoost模型及调参 | 徐瑞达 |
| 模型融合 | 徐瑞达 |
| 代码整理与优化 | 徐瑞达 |

## 5.2数据预处理（撰写:刘鉴之）

### 5.2.1数据读取

鉴于数据中存在json或包含逗号的特殊数据字段，因此定义了函数my\_readcsv实现对特殊数据的额外处理，该函数接受文件路径和分隔符作为参数，读取完成后返回DataFrame对象。具体实现的部分代码如下：

def my\_readcsv(path, sep=','):  
 # 使用codecs读取整个文件到lines  
 try:  
 lines = codecs.open(path, encoding='utf-8').readlines()  
 except:  
 lines = codecs.open(path, encoding='latin-1').readlines()  
 # 获取文件表头header  
 header = lines[0].strip().split(sep)  
 # 获取文件内容列表content  
 content = []  
 for line in lines[1:]:  
 line = line.strip()  
 try:  
 # 如果不包含json型数据,则直接使用sep分割line  
 index = [i for i, x in enumerate(line) if x == ',']  
 if len(index) == len(header) - 1:  
 content.append(line.split(sep))  
 else:  
 json\_list = []  
 # 将非json型数据读取至json\_list中  
 index = [0] + index  
 for idx in range(len(header) - 1):  
 json\_list.append(line[index[idx]:index[idx + 1]].strip(sep))  
 json\_list.append(line[index[len(header) - 1]:].strip(sep).replace(',', ';'))  
 content.append(json\_list)  
 except:  
 pass  
 # 返回DataFrame  
 return pd.DataFrame(content, columns=header)

### 5.2.2数据统计

在该部分中，使用模块pandas的groupby方法对各个数据类别对应DataFrame的uid列进行分组统计，并对不同变量应用不同的统计方法（使用agg函数）。具体各个数据类别的变量统计方法如下表。

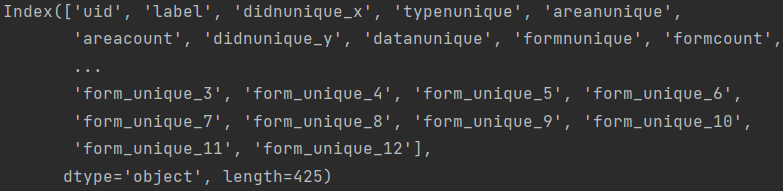
|  |  |  |
| --- | --- | --- |
| 数据类别 | 变量统计方法 | 说明 |
| devList | 'did': 'nunique', 'type': 'nunique', 'area':['nunique','unique','count'] | 统计设备列表中各账号对应的设备种类数、型号种类数、区域种类数、区域类型集合、区域数目 |
| control | 'did': 'nunique', 'data': 'nunique', 'form':['nunique','unique','count'] | 统计控制日志中各账号对应的设备种类数、日志种类数、命令类型种类数、命令类型集合、命令数目 |
| devupdate | 'did': 'nunique', 'data': 'nunique', | 统计上报日志中各账号对应的设备数目、日志数目 |

### 5.2.3合并特征

使用模块pandas的merge方法将以上数据统计后得到的各个DataFrame，然后使用TF-IDF技术对设备列表中的areaunique变量、操作控制日志中的formunique进行文本数据信息统计，这里举例对formunique的统计。

# 统计formunique列词频  
tfidf = TfidfVectorizer(max\_features=13)  
# 使用'default'替代formunique为0的字段  
train\_control\_tfidf=tfidf.fit\_transform(train\_feat['formunique'].apply(lambda x: 'default' if isinstance(x, int) is True else ' '.join(x)))  
test\_control\_tfidf =tfidf.fit\_transform(test\_feat['formunique'].apply(lambda x: 'default' if isinstance(x, int) is True else ' '.join(x)))  
# 设置列名  
features = list(tfidf.get\_feature\_names\_out())  
train\_control\_tfidf = pd.DataFrame(train\_control\_tfidf.toarray(),  
columns=['form\_unique\_' + str(features.index(x)) for x in features])  
test\_control\_tfidf = pd.DataFrame(test\_control\_tfidf.toarray(),  
columns=['form\_unique\_' + str(features.index(x)) for x in features])  
# 合并统计信息并删除formunique列  
train\_feat = pd.concat([train\_feat, train\_control\_tfidf], axis=1)  
train\_feat = train\_feat.drop(['formunique'], axis=1)  
test\_feat = pd.concat([test\_feat, test\_control\_tfidf], axis=1)  
test\_feat = test\_feat.drop(['formunique'], axis=1)

以上工作完成后，即可得到处理后的DataFrame，输出train\_feat的列如下：



同时，在程序中使用DataFrame的to\_csv方法将程序每次运行时统计后的数据输出到文件中，便于人工分析查看。因此具体统计结果可见附件train\_feat.csv和test\_feat.csv。

可见，在经过数据处理过后，被统计的变量变为了数值型特征，而且对于area和form变量，还利用TF-IDF技术得到了各area值和form值对应的统计信息，这样就根据原始数据构造出了便于进行算法训练的特征。

## 5.3特征工程（撰写:共同完成）

在特征工程部分，我们利用不同的选择特征的方法对特征进行了筛选，具体介绍如下。

### 5.3.1 相关系数法

### 5.3.1 递归特征消除法

### 5.3.1 模型分析法

### 5.3.1 人工选择

## 5.4模型预测一（撰写:刘鉴之）

在本项目中，分别采用了逻辑回归模型、决策树模型、随机森林模型、LightGBM模型、XGBoost模型等模型进行预测。其中前三个模型的预测由**刘鉴之**完成，后两个模型的预测和调参由**徐瑞达**完成。最后，重点选取了LightGBM模型和XGBoost模型进行调参、优化与预测。

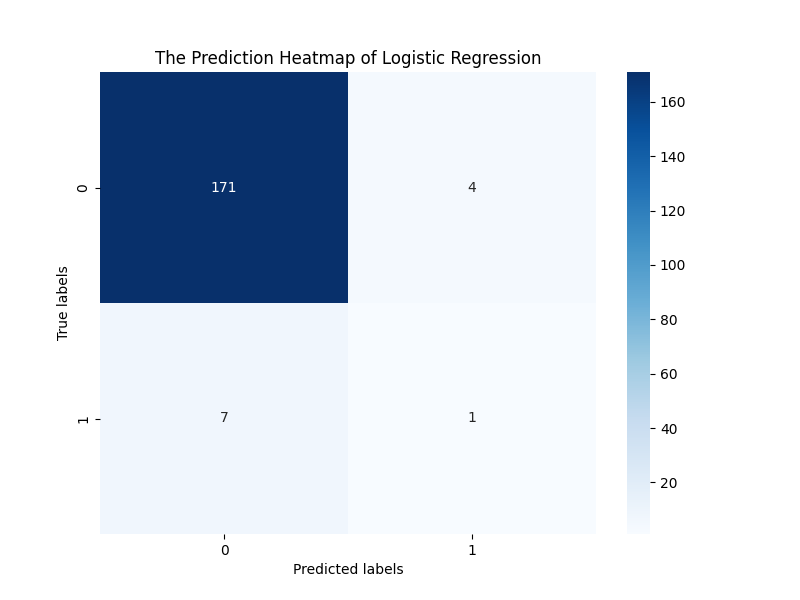
首先需要使用训练集对模型进行训练，划分训练集的代码如下：

# 划分训练集

X = train\_feat.drop(['uid', 'label'], axis=1)  
y = train\_feat['label']  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

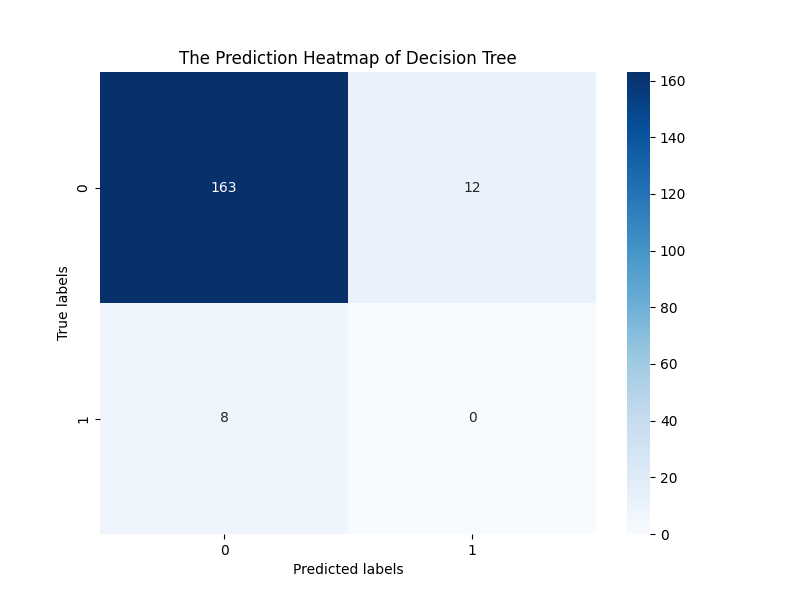
### 5.4.1 逻辑回归模型

直接调用sklearn模块中的LogisticRegression方法进行训练即可，简单的逻辑回归模型对于训练集的预测准确率较高，但是在线上测试时效果不佳。使用逻辑回归模型对训练集进行预测时的预测结果热力图如下图所示：



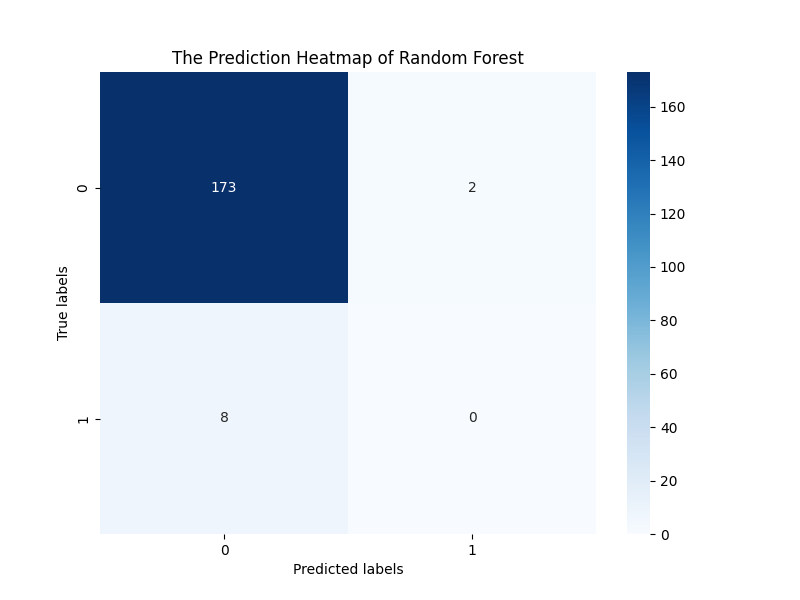
### 5.4.2 决策树模型

直接调用sklearn模块中的DecisionTreeClassifier方法进行训练即可，该模型对训练集的预测准确率往往是所选取的几种模型中最低的，线上结果也不尽人意。使用决策树模型对训练集进行预测时的预测结果热力图如下图所示：



### 5.4.3 随机森林模型

直接调用sklearn模块中的RandomForestClassifier方法进行训练即可，在未采用LightGBM模型和XGBoost模型之前，随机森林模型的线上测试结果最佳，是值得选取的模型之一。使用随机森林模型对训练集进行预测时的预测结果热力图如下图所示：

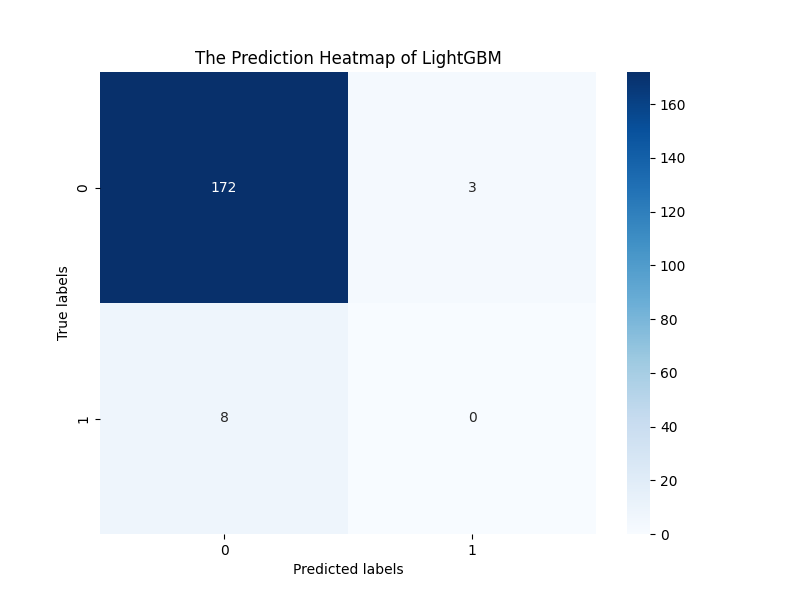


## 5.5模型预测二（撰写:徐瑞达）

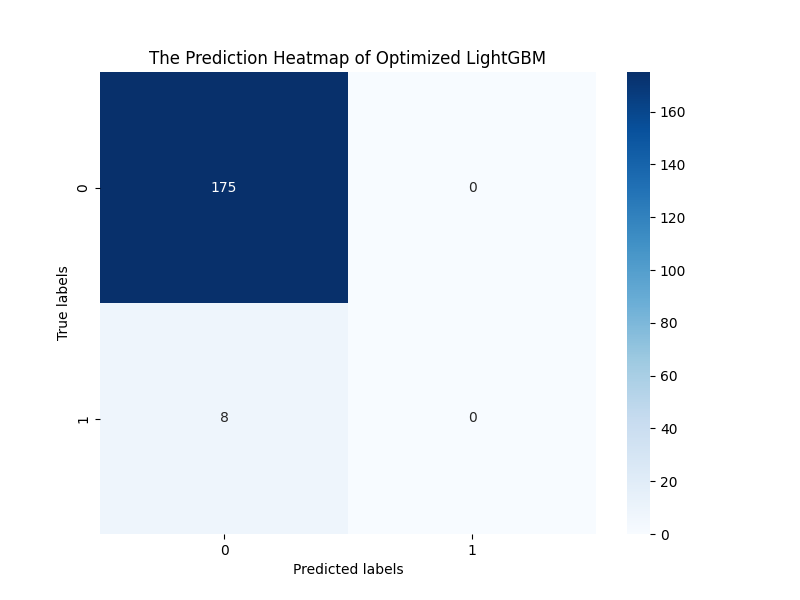
### 5.5.1 LightGBM模型

直接调用lightgbm模块中的LGBMClassifier方法进行训练即可。未经过调参的LightGBM模型对训练集的预测结果已经超过了逻辑回归模型和决策树模型，接近随机森林模型的结果，经过调参后，其准确率提升了0.02左右，高于随机森林模型。

使用调参前的LightGBM模型对训练集进行预测时的预测结果热力图如下图所示：



使用调参后的LightGBM模型对训练集进行预测时的预测结果热力图如下图所示：

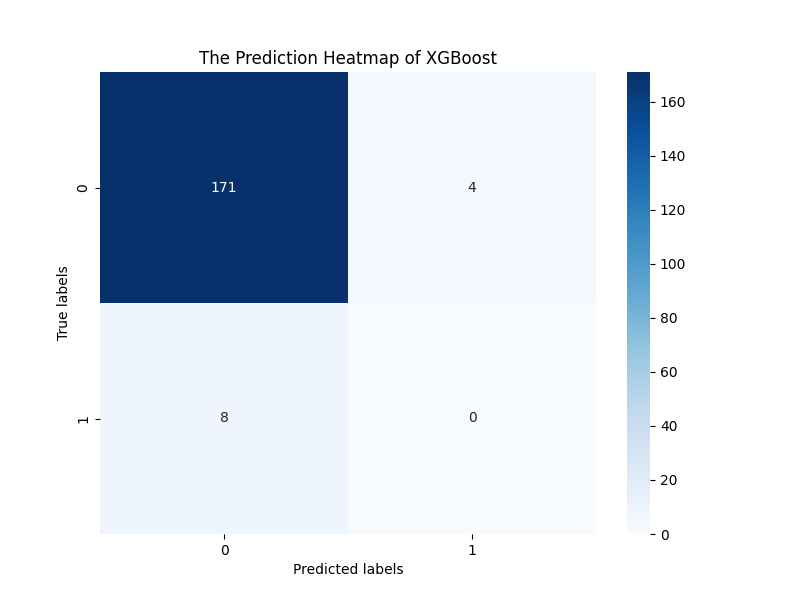


### 5.5.2 LightGBM模型调参

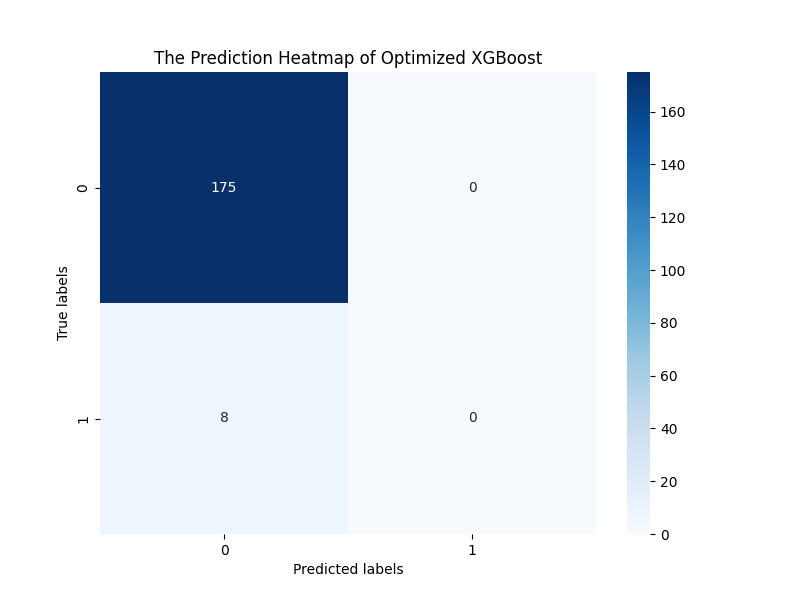
### 5.5.3 XGBoost模型

直接调用xgboost模块中的XGBClassifier方法进行训练即可。未经过调参的XGB模型对训练集的预测结果接近LightGBM模型的结果，经过调参后，其准确率与调参后的LightGBM模型相当，因此后期尝试使用这两种模型进行融合。

使用调参前的XGBoost模型对训练集进行预测时的预测结果热力图如下图所示：



使用调参前的XGBoost模型对训练集进行预测时的预测结果热力图如下图所示：



### 5.5.4 XGBoost模型调参

## 5.6 模型融合（撰写:徐瑞达）

## 5.7 代码整理与优化（撰写:徐瑞达）

关键模块的实现，以及这个实现对于得分的长进，不要贴大段代码

每段代码尽量别超过半页

鼓励同学们对不同设计的结果进行分析，思考为什么A不如B

这里也可以做一些好看的可视化，比如超参选择和得分的关系什么的

# 6. 实验结果

在这里贴出最终得分就行，顺便带个一两句

# 7. 个人体会

通过这次实验，