重庆邮电大学

学生实验实习报告册

学年学期：2022-2023 学年(秋)学期

课程名称：数据工程综合实践

学生学院：计算机学院/人工智能学院

专业班级：数据科学与大数据技术

学生学号：XXXXXX

学生姓名：陈浩如

联系电话：19823324153

重庆邮电大学教务处印制

目　录

教师评阅记录表

【重要说明】

- 学生提交报告册最终版时，**必须包含此页**，否则不予成绩评定。
- 本报告册模板内容格式除确实因为填写内容改变了布局外，**不得变更其余部分的格式**，否则不予成绩评定。

报告是否符合考核规范	☑ 符合　☐ 不符合
报告格式是否符合标准	☑ 符合　☐ 不符合
报告是否完成要求内容	☑ 是　☐ 否
报告评语：	
报告成绩：	
评阅人签名（签章）　XXX 2023 年 1 月 10 日	

实验或实习报告

课程名称	数据工程综合实践	课程编号	A2041050
开课学院	计算机学院/人工智能学院		
指导教师	XXX		
实验实习地点	综合实验大楼 B517		

学号	姓名
*****	*****

一、 赛题解读与方案概述

(1)赛题解读

1.赛题背景与目的

赛题提供用户在 2016 年 1 月 1 日至 2016 年 6 月 30 日之间真实线上线下消费行为，预测用户在 2016 年 7 月领取优惠券后 15 天以内的使用情况。

2.数据字段表分析

表一、用户线下消费和优惠卷领取行为

Field	Description
User_id	用户ID
Merchant_id	商户ID
Coupon_id	优惠券ID：null表示无优惠券消费，此时Discount_rate和Date_received字段无意义
Discount_rate	优惠率：x \in [0,1]代表折扣率；x:y表示满x减y。单位是元
Distance	user经常活动的地点离该merchant的最近门店距离是x*500米（如果是连锁店，则取最近的一家门店），x\in[0,10]；null表示无此信息，0表示低于500米，10表示大于5公里；
Date_received	领取优惠券日期
Date	消费日期：如果Date=null & Coupon_id != null，该记录表示领取优惠券但没有使用，即负样本；如果Date!=null & Coupon_id = null，则表示普通消费日期；如果Date!=null & Coupon_id != null，则表示用优惠券消费日期，即正样本；

由此表可以看出用户线下消费分为两种情况，一种是没有使用优惠卷的普通消费行为，另一种是使用优惠卷的消费行为，即正样本。

故要对两种消费行为分别处理，提取指定的特征情况。

表二、用户线上点击\消费和领取优惠卷行为

Field	Description
User_id	用户ID
Merchant_id	商户ID
Action	0 点击， 1购买，2领取优惠券
Coupon_id	优惠券ID：null表示无优惠券消费，此时Discount_rate和Date_received字段无意义。“fixed”表示该交易是限时低价活动。
Discount_rate	优惠率：x \in [0,1]代表折扣率；x:y表示满x减y；“fixed”表示低价限时优惠；
Date_received	领取优惠券日期
Date	消费日期：如果Date=null & Coupon_id != null，该记录表示领取优惠券但没有使用；如果Date!=null & Coupon_id = null，则表示普通消费日期；如果Date!=null & Coupon_id != null，则表示用优惠券消费日期；

与线下用户行为表一致，也分为两种消费情况，同时存在限时折扣活动，要分别预处理，将’fixed’字段修正，以及带有’fixed’字段的数据处理清楚。

表三、用户 020 线下优惠卷使用预测样本

Field	Description
User_id	用户ID
Merchant_id	商户ID
Coupon_id	优惠券ID
Discount_rate	优惠率：x \in [0,1]代表折扣率；x:y表示满x减y.
Distance	user经常活动的地点离该merchant的最近门店距离是x*500米（如果是连锁店，则取最近的一家门店），x\in[0,10]；null表示无此信息，0表示低于500米，10表示大于5公里；
Date_received	领取优惠券日期

由于此字段提供了一整个月信息,相当于信息泄露,要好好利用label_field里的数据构造用户当月情况。

(二)方案综述

首先从用户线下消费行为数据表中提取用户(User_field)、商家(Merchant_field)、优惠卷特征(Coupon_field)，再结合用户-商家(User_Merchant_field)、用户-优惠卷(User_Coupon_field)等交叉特征提取，同时提取用户线上消费行为(Online_field)等系列特征,以及预测区间段的泄露数据系统特征(Leak_field)。

做完特征工程后利用 XGBoost 模型预测优惠卷使用概率，不断调参得到最优解。

二、 数据预处理

1. 用户线下数据预处理

(1) 缺失数据填充：

```
data['Distance'].fillna(-1, inplace=True)
data['Discount_rate'].fillna(-1, inplace=True)
data['Coupon_id'].fillna(-1, inplace=True)
```

(2) 折扣率与满减等情况探索，以及最低满减额度

```
data['discount_rate'] = data['Discount_rate'].map(lambda x: (float(str(x).split(':')[0]) - float(str(x).split(':')[1]))
                                                            / (float(str(x).split(':')[0])) if ':' in str(x) else float(x))
data['flag_of_manjian'] = data['Discount_rate'].map(lambda x: 1 if ':' in str(x) else 0)
data['manjian_at_least_cost'] = data['Discount_rate'].map(lambda x: int(str(x).split(':')[0]) if ':' in str(x) else -1)
```

(3)时间类型数据转换及时间数据的探索

```
data['date_received'] = pd.to_datetime(data['Date_received'], format="%Y%m%d")
columns = data.columns.tolist()
if 'Date' in columns:
    data['date'] = pd.to_datetime(data['Date'], format="%Y%m%d")
data['weekday_receive'] = data['date_received'].map(lambda x: x.weekday())  # 星期几
data['is_weekend'] = data['weekday_receive'].map(lambda x: 1 if x == 6 or x == 7 else 0)  # 判断领券日是否为休息日
data = pd.concat([data, pd.get_dummies(data['weekday_receive'], prefix='week')], axis=1)  # one-hot离散星期几
data['is_yuechu'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 10 and x.day > 0 else 0)  #判断月初
data['is_yuezhong'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 20 and x.day > 10 else 0)  #判断月中
data['is_yuemo'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 30 and x.day > 20 else 0)    #判断月末
```

2. 用户线上数据预处理

(1) 缺失数据填充：

```
data['Discount_rate'].fillna(-1, inplace=True)
```

```
#将fixed转化成0号优惠卷
data['Coupon_id'].fillna(-1, inplace=True)
```

```
#操作处理
data['Action'].fillna(-1, inplace=True)
```

(2) 折扣率转换以及限时低价活动的转换

```
data['flag_of_manjian'] = data['Discount_rate'].map(lambda x: 1 if ':' in str(x) else 0)
data['is_fix'] = data['Coupon_id'].map(lambda x: 1 if str(x) == 'fixed' else 0)
data['manjian_at_least_cost'] = data['Discount_rate'].map(lambda x: int(str(x).split(':')[0]) if ':' in str(x) else -1)
```

(3) 缺失优惠卷的转换以及限时低价活动的转换

```
#将fixed转化成0号优惠卷
data['Coupon_id'].fillna(-1, inplace=True)
data['Coupon_id'] = data['Coupon_id'].map(lambda x: x if str(x) != 'fixed' else 0)
```

(4) 时间类型数据转换及时间数据的探索

```
#时间处理
data['date_received'] = pd.to_datetime(data['Date_received'], format='%Y%m%d')
columns = data.columns.tolist()
if 'Date' in columns:
    data['date'] = pd.to_datetime(data['Date'], format='%Y%m%d')
data['weekday_receive'] = data['date_received'].map(lambda x: x.weekday())  # 星期几
data['is_weekend'] = data['weekday_receive'].map(lambda x: 1 if x == 6 or x == 7 else 0)  # 判断领券日是否为休息日
data = pd.concat([data, pd.get_dummies(data['weekday_receive'], prefix='week')], axis=1)  # one-hot离散星期几
data['is_yuechu'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 10 and x.day > 0 else 0)  # 判断月初
data['is_yuezhong'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 20 and x.day > 10 else 0)  # 判断月中
data['is_yuemo'] = data['date_received'].map(lambda x: -1 if pd.isnull(x) else 1 if x.day <= 30 and x.day > 20 else 0)  # 判断月末
```

三、 数据划分与打标

1. 数据划分

划分方案：划分了两个训练集，这样可以扩充一倍的训练样本。

	预测区间（提取label）	特征区间（提取feature）
测试集	20160701~20160731	20160315~20160630
训练集1	20160515~20160615	20160201~20160514
训练集2	20160414~20160514	20160101~20160413

具体划分代码：

```
print('划分数据集 1 中')
train_history_field1 = train_data[train_data['date'].isin(pd.date_range('2016/1/1',
periods=104)) |(train_data['date_received'].isin(pd.date_range('2016/1/1',
periods=104)) & train_data['date'].map(lambda x : pd.isnull(x)))]
# [20160101, 20160413]
train_history_online_field1 =
online_data[online_data['date'].isin(pd.date_range('2016/1/1', periods=104))
|(online_data['date_received'].isin(pd.date_range('2016/1/1', periods=104)) &
```

```
online_data['date'].map(lambda x : pd.isnull(x)))]
train_label_field1 =
train_data[train_data['date_received'].isin(pd.date_range('2016/4/14', periods=30))]
# [20160414, 20160514]

print('划分数据集 2 中')

train_history_field2 = train_data[train_data['date'].isin(pd.date_range('2016/2/1',
periods=104)) |(train_data['date_received'].isin(pd.date_range('2016/2/1',
periods=104)) & train_data['date'].map(lambda x : pd.isnull(x)))]
# [20160201, 20160514]
train_history_online_field2 =
online_data[online_data['date'].isin(pd.date_range('2016/2/1', periods=104))
|(online_data['date_received'].isin(pd.date_range('2016/2/1', periods=104)) &
online_data['date'].map(lambda x : pd.isnull(x)))]
train_label_field2 =
train_data[train_data['date_received'].isin(pd.date_range('2016/5/15', periods=30))]
# [20160515, 20160615]


print('划分测试集中')

test_history_field = train_data[train_data['date'].isin(pd.date_range('2016/3/15',
periods=108)) |(train_data['date_received'].isin(pd.date_range('2016/3/15',
periods=108)) & train_data['date'].map(lambda x : pd.isnull(x)))]
# [20160101, 20160413]
test_history_online_field =
online_data[online_data['date'].isin(pd.date_range('2016/3/15', periods=108))
|(online_data['date_received'].isin(pd.date_range('2016/3/15', periods=108)) &
online_data['date'].map(lambda x : pd.isnull(x)))]
test_label_field = test_data.copy()
```

2. 数据打标

将 15 天内核销的优惠卷的’label’列打为 1，其他为 0

```
def get_label(data1):
    data = data1.copy()
    data['label'] = list(map(lambda x, y: 1 if (x - y).total_seconds() / (60 * 60 * 24) <= 15 else 0, data['date'], data['date_received']))
    return data
```

四、 特征工程

1.用户线下消费特征集(User_field)

共提取了 70 条特征，包含用户领卷数、用户核销数、用户未核销数等等，以及与时间情况和距离情况和折扣率情况的交叉特征，具体在此不再列举。

采用 groupby 函数加自定义函数 apply 或者自带的统计函数提取特征，以用户领卷数为例，展示该行代码。

```
tmp = pd.DataFrame(data[(data['Coupon_id'].map(lambda x: x != -1)) &
                  (data['Date_received'].map(lambda x: x !=
                  -1))].groupby(keys[0])['cnt'].count()).reset_index()
tmp.columns = [keys[0], prefixs + 'receive_cnt']
```

然后采用自定义的merge及填充空值函数mer,将提取的特征与数据集合并。

```
u_feat = mer(u_feat, tmp, keys[0], 0)
```

mer 函数具体如下

```
def mer(data1, data2, key, fill):
    data1 = pd.merge(data1, data2, on = key, how = 'left')
    data1.fillna(fill, downcast='infer', inplace = True)
    return data1
```

2.商家线下消费特征集(Merchant_field)

共提取了 39 条特征，包含商家领卷数，商家核销数，商家未核销数等等，以及与时间情况和距离情况和折扣率情况的交叉特征，具体在此不再列举。

同时采用 groupby 函数做同样的特征提取和合并处理。

3. 优惠卷线下消费特征集(Coupon_field)

共提取了 5 条特征，包含优惠卷领卷数，优惠卷核销数，优惠卷未核销数等等，具体在此不再列举。

同时采用 groupby 函数做同样的特征提取和合并处理。

4.用户商家交叉特征群(User_Merhcant_field)

共提取了 21 条特征，包含用户领取特定商家优惠卷数，用户核销特定商家优惠卷数，用户未核销特定商家优惠卷数等等，以及与时间情况和距离情况和折扣率情况的交叉特征，具体在此不再列举。

同时采用 groupby 函数做同样的特征提取和合并处理。

5.用户优惠卷交叉特征群(User_Coupon_field)

共提取了 3 条特征，包含用户领取特定优惠卷数，用户核销特定优惠卷数，用户未核销特定优惠卷数。

同时采用 groupby 函数做同样的特征提取和合并处理。

6.预测域泄露数据集(Leak_field)

共提取了 56 条特征，包含用户领取优惠卷数，用户领取特定优惠卷数，用户领取优惠卷平均距离等等， 以及与时间情况和距离情况和折扣率情况的交叉特征，具体在此不再列举。

同时采用 groupby 函数做同样的特征提取和合并处理。

7.用户线上操作数据集(Online_field)

共提取了 7 条特征，包含用户线上操作次数、用户线上点击次数、用户线上购买次数等等，具体在此不再列举。

同时采用 groupby 函数做同样的特征提取和合并处理。

8. 融合各个特征集

将提取好的特征集删去重复列，然后拼接到一起。

```
history_feat = get_user_offline_featrue(history_field, label_field)
merchant_feat = get_Merchant_featrue(history_field, label_field)
coupon_feat = get_Coupon_featrue(history_field, label_field)
um_feat = get_user_Merchant_featrue(history_field, label_field)
uc_feat = get_user_coupon_featrue(history_field, label_field)
leak_feat = get_leak_featrue(label_field)
online_feat = get_online_featrue(online_field, label_field)

share_characters = list(set(label_field.columns.tolist()) &
                        set(history_feat.columns.tolist()) &
                        set(merchant_feat.columns.tolist()) &
                        set(leak_feat.columns.tolist()) &
                        set(coupon_feat.columns.tolist()) &
                        set(um_feat.columns.tolist()) &
                        set(online_feat.columns.tolist()) &
                        set(uc_feat.columns.tolist())
                        )
label_field.index = range(len(label_field))
dataset = pd.concat([label_field, history_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, leak_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, merchant_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, coupon_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, um_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, online_feat.drop(share_characters, axis=1)], axis=1)
dataset = pd.concat([dataset, uc_feat.drop(share_characters, axis=1)], axis=1)
```

然后最后对输出的 dataset 将 label 列放到最后，删去无用的数据列，调整数据格式。

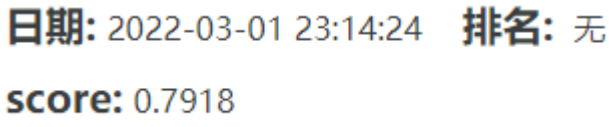
```
if 'Date' in dataset.columns.tolist():
    dataset.drop(['Merchant_id', 'Discount_rate', 'Date','date_received', 'date'], axis=1, inplace=True)
    label = dataset['label'].tolist()
    dataset.drop(['label'], axis=1, inplace=True)
    dataset['label'] = label
else:
    dataset.drop(['Merchant_id', 'Discount_rate','date_received'], axis=1, inplace=True)

dataset['User_id'] = dataset['User_id'].map(int)
dataset['Coupon_id'] = dataset['Coupon_id'].map(int)
dataset['Date_received'] = dataset['Date_received'].map(int)
dataset['Distance'] = dataset['Distance'].map(int)
if 'label' in dataset.columns.tolist():
    dataset['label'] = dataset['label'].map(int)

dataset.drop_duplicates(keep='first', inplace=True)
dataset.index = range(len(dataset))
```

9. 第二次提交任务二相较第一次的优化情况

(1)添加热启动特征：构建用户 15 天内消费用卷情况、商家 15 天内消费用卷情况、优惠卷 15 天内消费用卷情况、用户商家交叉 15 天内消费用卷情况、用户优惠卷交叉 15 天内消费用卷情况、商家优惠卷交叉 15 天内消费用卷情况。共 82 条特征。

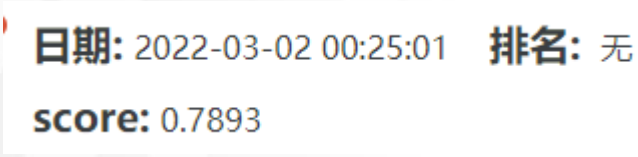
五、 模型设计与融合

1. 采用单 XGBoost 模型预测，进行单折 5000 轮训练，输出 AUC 情况如下。

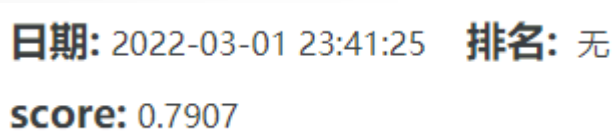


2. 采用单 Lightgbm 模型预测，进行单折 5000 轮训练，输出 AUC 情况如

下。

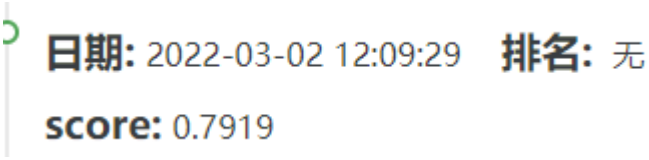


3. 采用单 GBDT 模型预测，输出 AUC 情况如下。

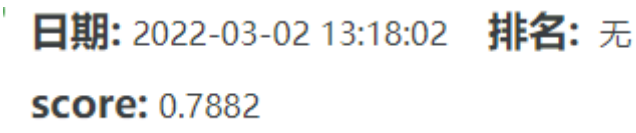


综上所述，单模型 XGBoost 进行单折 5000 轮训练的 AUC 情况较为理想。

4. 删除重要性较低的特征，重新进行 XGBoost 单折 5000 轮训练，有了一定提升。



5. 随后尝试了五折交叉验证训练 Xgboost，但 AUC 情况反而降下来了。

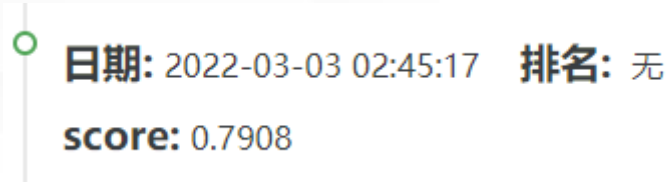


目前只将项目 AUC 情况训练到了 0.7919，特征筛选还没有做好，模型调参也没有调到最优解，模型融合和多模型选择更是没有做好。感到了目前的能力上限，更进一步的操作还需要学习。

6. 第二次提交任务二相较第一次的优化情况

在新构建了热启动特征后，重新跑一次单折 5000 轮 XGBoost 训练。AUC 达到了 0.8016。

尝试筛特征，将特征数从 **283** 个筛到了 **150** 个，效果反而变差。



不断尝试筛特征，通过 Kbest 函数分析特征相关性选出前 K 个关联特征，最后线上提交 AUC 情况反而没有直接采用 283 个特征好。

最后仍旧采用 283 个特征，开始调参。

第一套参数 0.8016

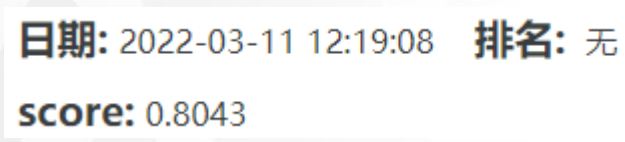
```
params = {'booster': 'gbtree',
          'objective': 'binary:logistic',
          'eval_metric': 'auc',
          'silent': 1,
          'eta': 0.01,
          'max_depth': 5,

          'min_child_weight': 1,
          'gamma': 0,
          'lambda': 1,
          'colsample_bylevel': 0.7,
          'colsample_bytree': 0.7,
          'subsample': 0.9,
          'scale_pos_weight': 1}
```

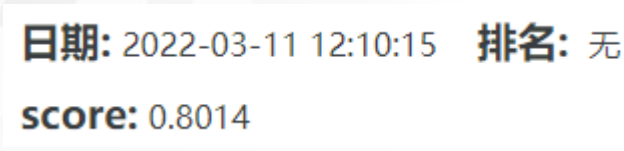
第二套参数 **0.8021**

```
params = {'booster': 'gbtree',
          'objective': 'binary:logistic',
          'eval_metric': 'auc',
          'gamma': 0.1,
          'min_child_weight': 1.1,
          'max_depth': 5,
          'lambda': 10,
          'subsample': 0.7,
          'colsample_bytree': 0.7,
          'colsample_bylevel': 0.7,
          'eta': 0.01,
           # 'tree_method': 'exact',
          'tree_method': 'gpu_hist',
          'seed': 0,
          'nthread': 12,
          'predictor': 'gpu_predictor',
          }
```

然后发现了 XGBoost 的训练效果受到 XGBoost 版本很大的影响，将 XGBoost 版本从 1.2.2 更新到了 1.5.2，继续采用这两套参数，出现以下结果：

第一套参数：0.8014，第二套参数：0.8043.





阅读到关于 **Bagging** 的思想，考虑是否能采用 **Bagging** 的思想，由于两个版本两套参数运行出的四个结果是相对独立且效果较优的，类比于 **Bagging** 中要求的有差异的独立的且效果较好的基训练器，考虑采用 **Bagging** 对于回归问题的加权融合方法。

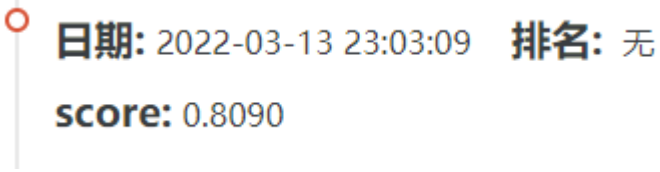
经过多次探索得到最佳加权融合比例，将 **AUC** 训练到了 **0.8090**

加权比例：

(1) 0.8014 * 0.3 + 0.8016 * 0.7 = 0.8079

(2) 0.8021 * 0.2 + 0.8043 * 0.7 = 0.8054

(3) 0.8079 * 1 + 0.8054 * −0.05 = 0.8090



六、 项目总结

这整个 020 优惠卷预测项目让我感到了数据挖掘的魅力，但苦于自己的知识水平和见识太少，以及经验不充分，没有将分数刷到很高。

这半个月左右的时间不断地尝试模型训练，以及查阅了各种资料，拜读了范磊学长的论文，以及仔细推理和复现了第一赛季第一名 Wepe 队伍的特征工

程代码，才达到了目前的AUC分数。有了如下几点收获：

1.特征工程

特征工程是数据挖掘的重中之重，一切的基础，如果没有在数据中提取足够的特征，那无论怎么调整模型都是无效的。特征提取需要对数据的敏感程度以及足够的思维能力，从Wepe的特征工程中发现了Leak_field的存在让我的AUC暴涨了20%。但我本身对于数据的敏感程度还不够，还需要经验的积累。

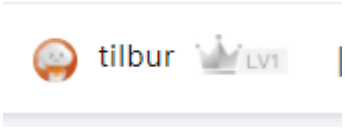
同时从论坛中学到了如何判定当前特征工程是否完善，就是用100%的数据集训练，再把这份数据集当作预测集，直接预测当前数据情况，判断AUC情况。由于这种方式会造成过拟合，但很容易就能看出特征提取的是否充足，因为在过拟合的情况下，如果特征提取的充足，AUC情况就会逐渐接近1.000，如果连使用同一份数据集来做训练和预测，在过拟合的情况下都不能达到较高的AUC分数，那就证明特征工程做的还不够。

2.模型选择和调参

集成学习是当前竞赛中最大的热门，采用了XGBoost和Lightbgm，但XGBoost的效果更好，在调整参数和运行环境后，得到了四套相对独立且不同的效果好的预测结果，借助Bagging的思想，采用加权融合，得到了比原来最好的预测结果更好的结果。

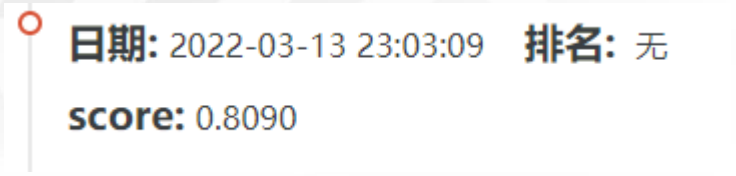
如果没有办法再提升，就去学习吧。

七、 最优线上成绩与历史提交成绩记录（截图，另请务必提供线上测评 ID 以便查验;历史提交成绩记录指全部的成绩提交记录，而不是某一天的）



线上测评 ID：

最优线上成绩：



历史提交成绩记录：

日期: 2022-03-13 23:17:26 **排名:** 无
score: 0.8065

日期: 2022-03-13 23:15:50 **排名:** 无
score: 0.8065

日期: 2022-03-13 23:14:58 **排名:** 无
score: 0.8030

日期: 2022-03-13 23:03:09 **排名:** 无
score: 0.8090

日期: 2022-03-13 23:02:35 **排名:** 无
score: 0.8082

日期: 2022-03-13 23:01:24 **排名:** 无
score: 0.8089

日期: 2022-03-13 23:00:52 **排名:** 无
score: 0.8089

日期: 2022-03-13 23:00:18 **排名:** 无
score: 0.8048

日期: 2022-03-13 22:59:42 **排名:** 无
score: 0.8090

日期: 2022-03-13 22:59:09 **排名:** 无
score: 0.8082

日期: 2022-03-12 13:13:15 **排名:** 无
score: 0.8083

日期: 2022-03-12 13:09:51 **排名:** 无
score: 0.8054

日期: 2022-03-12 10:15:19 **排名:** 无
score: 0.8079

日期: 2022-03-12 10:12:45 **排名:** 无
score: 0.8054

日期: 2022-03-11 14:37:51 **排名:** 无
score: 0.8049

日期: 2022-03-11 14:37:15 **排名:** 无
score: 0.8043

日期: 2022-03-11 13:40:00 **排名:** 无
score: 0.8057

日期: 2022-03-11 13:39:22 **排名:** 无
score: 0.8048

日期: 2022-03-11 13:37:42 **排名:** 无
score: 0.8053

日期: 2022-03-11 13:32:14 **排名:** 无
score: 0.8014

日期: 2022-03-11 12:57:31 **排名:** 无
score: 0.7958

日期: 2022-03-11 12:19:08 **排名:** 无
score: 0.8043

日期: 2022-03-11 13:40:00 **排名:** 无
score: 0.8057

日期: 2022-03-11 13:39:22 **排名:** 无
score: 0.8048

日期: 2022-03-11 13:37:42 **排名:** 无
score: 0.8053

日期: 2022-03-11 13:32:14 **排名:** 无
score: 0.8014

日期: 2022-03-11 12:57:31 **排名:** 无
score: 0.7958

日期: 2022-03-11 12:19:08 **排名:** 无
score: 0.8043

日期: 2022-03-11 12:15:47 **排名:** 无
score: 0.7993

日期: 2022-03-11 12:10:15 **排名:** 无
score: 0.8014

日期: 2022-03-10 20:40:11 **排名:** 无
score: 0.7987

日期: 2022-03-10 19:58:23 **排名:** 无
score: 0.7985

日期: 2022-03-10 19:20:30 **排名:** 无
score: 0.7914

日期: 2022-03-10 18:41:07 **排名:** 无
score: 0.7958

日期: 2022-03-10 17:40:54 **排名:** 无
score: 0.7934

日期: 2022-03-10 17:15:28 **排名:** 无
score: 0.7956

日期: 2022-03-10 12:31:02 **排名:** 无
score: 0.7866

日期: 2022-03-10 12:00:06 **排名:** 无
score: 0.7600

日期: 2022-03-10 11:49:57 **排名:** 无
score: 0.7800

日期: 2022-03-10 00:58:54 **排名:** 无
score: 0.7922

日期: 2022-03-09 18:29:52 **排名:** 无
score: 0.8016

日期: 2022-03-03 02:45:17 **排名:** 无
score: 0.7908

日期: 2022-03-03 02:01:59 **排名:** 无
score: 0.7908

日期: 2022-03-02 18:00:11 **排名:** 无
score: 0.7895

日期: 2022-03-02 13:24:51 **排名:** 无

score: 0.7829

日期: 2022-03-02 13:22:51 **排名:** 无

score: 0.7778

日期: 2022-03-02 13:18:02 **排名:** 无

score: 0.7882

日期: 2022-03-02 12:52:08 **排名:** 无

score: 0.7855

日期: 2022-03-02 12:33:13 **排名:** 无

score: 0.7918

日期: 2022-03-02 12:09:29 **排名:** 无

score: 0.7919

日期: 2022-03-02 00:25:01 **排名:** 无

score: 0.7893

日期: 2022-03-02 00:01:12 **排名:** 无

score: 0.5252

日期: 2022-03-01 23:41:25 **排名:** 无
score: 0.7907

日期: 2022-03-01 23:14:24 **排名:** 无
score: 0.7918

日期: 2022-03-01 21:56:48 **排名:** 无
score: 0.7514

日期: 2022-03-01 21:09:02 **排名:** 无
score: 0.7449

日期: 2022-03-01 21:02:00 **排名:** 无
score: 0.7346

日期: 2022-02-28 17:11:57 **排名:** 无
score: 0.7441

日期: 2022-02-28 17:03:00 **排名:** 无
score: 0.7384

日期: 2022-02-28 16:58:46 **排名:** 无
score: 0.7411

日期: 2022-02-28 16:53:47 **排名:** 无
score: 0.7408

日期: 2022-02-28 16:49:45 **排名:** 无
score: 0.7391

日期: 2022-02-28 16:44:28 **排名:** 无
score: 0.7355

日期: 2022-02-28 16:41:32 **排名:** 无
score: 0.7351

日期: 2022-02-28 16:31:39 **排名:** 无
score: 0.7391

日期: 2022-02-28 16:27:14 **排名:** 无
score: 0.7405

日期: 2022-02-28 16:19:53 **排名:** 无
score: 0.7413

日期: 2022-02-27 19:38:50 **排名:** 无
score: 0.7316

日期: 2022-02-27 19:03:33 **排名:** 无
score: 0.7343

日期: 2022-02-27 18:33:48 **排名:** 无
score: 0.7412

日期: 2022-02-27 18:15:49 **排名:** 无
score: 0.7433

日期: 2022-02-27 17:43:31 **排名:** 无
score: 0.7432

日期: 2022-02-27 17:00:46 **排名:** 无
score: 0.7392

日期: 2022-02-27 16:26:22 **排名:** 无
score: 0.7373

日期: 2022-02-27 15:43:27 **排名:** 无
score: 0.7393

日期: 2022-02-27 13:49:58 **排名:** 无
score: 0.7378

日期: 2022-02-27 01:28:30 **排名:** 无
score: 0.7243

日期: 2022-02-26 23:25:52 **排名:** 无
score: 0.6816

日期: 2022-02-26 23:08:30 **排名:** 无
score: 0.5924

日期: 2022-02-26 22:55:08 **排名:** 无
score: 0.7198

日期: 2022-02-26 22:39:07 **排名:** 无
score: 0.7175

日期: 2022-02-25 17:12:13 **排名:** 无
score: 0.7139

日期: 2022-01-15 17:35:59 **排名:** 无
score: 0.7187

日期: 2022-01-15 16:50:25 **排名:** 无
score: 0.5808

日期: 2022-01-13 22:13:04 **排名:** 无
score: 0.5808

日期: 2022-01-13 21:37:24 **排名:** 无
score: 0.7187

日期: 2022-01-13 21:26:24 **排名:** 无

score: 0.5808

日期: 2022-01-13 21:11:05 **排名:** 无

score: 0.5835

日期: 2022-01-13 20:51:53 **排名:** 无

score: 0.5882

日期: 2022-01-13 18:56:04 **排名:** 无

score: 0.4836

日期: 2022-01-13 18:47:02 **排名:** 无

score: 0.5905

日期: 2022-01-13 17:05:40 **排名:** 无

score: 0.5893

日期: 2022-01-13 15:43:37 **排名:** 无

score: 0.5904

日期: 2022-01-12 18:55:29 **排名:** 无

score: 0.5904

日期: 2022-01-12 18:02:10 **排名:** 无

score: 0.5786

日期: 2022-01-12 17:53:30 **排名:** 无

score: 0.5336