

企业非法集资风险预测

一、赛题理解及结果

1) 赛题理解与认识

a) 赛题背景

非法集资是一种犯罪活动，是指单位或者个人未依照法定程序经有关部门批准，以发行股票、债券、彩票、投资基金证券或者其他债权凭证的方式向社会公众筹集资金，并承诺在一定期限内以货币、实物以及其他方式向出资人还本付息或给予回报的行为。非法集资包括非法吸收公众存款和集资诈骗。

非法集资对社会和群众都带来了极大的危害，近年来通过互联网金融平台进行的非法集资活动危害愈加扩大。非法集资的形式也多变，涉及投资理财、P2P 网贷、房地产、教育、私募股权等多种形态。

以往对非法集资行为的监控和识别主要以人工收集相关信息，并通过人工的经验和对财务报表等大量数据进行统计分析比较，人工识别的准确率和效率都比较低。本赛题旨在通过机器学习、深度学习等方法训练出一个预测模型，通过学习企业的相关信息来预测企业是否具有非法集资风险。

b) 赛题相关业务理解

非法集资行为的特征同时具有以下四个性质：

- 非法性：未经有关部门批准，违规向社会筹集资金，如未经批准公开、非公开发行股票、债券等。
- 利诱性：承诺在一定期限内以货币、实物、股权等方式还本息或者给付汇报。
- 公开性和社会性：向社会公众即社会不特定对象吸收资金，通过媒体、推介会、传单、手机短信、熟人推介等途径向社会公开宣传。

非法集资企业一般有以下特征：

- 工商异常：未按期公示、股东变更频繁等。
- 网站异常：网站未备案、网站打不开等。
- 司法异常：涉诉次数多、行政处罚多等。
- 经营异常：短期大量对外投资、短期招聘大量人员等。
- 负面舆情：失联跑路、提现困难、负面新闻多等。
- 关联风险：与问题企业股权关联、与问题企业法人关联等。

非法集资的风险来源主要有：

- 基本风险：企业的基本信息如企业类型、企业门类等；经营资质如经营项目、经营范围等；变更信息如变更事项、变更频率等。
- 行为风险：企业的招聘信息如招聘人数、是否异地经营等；产品信息如产品个数、产品描述、收益率等；企业知识产权。
- 舆情风险：企业的舆情信息如负面新闻次数。
- 遵从风险：企业涉及的案件信息、被投诉举报次数等。
- 族群风险：企业的族谱信息，与其他非法集资企业的关联关系。

c) 赛题数据理解

赛题数据集包括约 25000 家企业数据，其中约 15000 家企业带标注数据作为训练集，剩余数据作为测试集。使用的数据均为脱敏后的真實数据，来源于互联网大数据，部分字段内容在部分企业中有缺失。赛题共有 8 个数据集：

- 数据集1：base_info.csv，包含所有企业的基本信息，包括企业经营方式、行业、地址、注册资本等。
- 数据集2：annual_report_info.csv，包含企业的年报信息，包括从业人数、经营状况、人员分布等。
- 数据集3：tax_info.csv，包含企业的税收信息，包括税收时间、税种、税率、税额等。
- 数据集4：change_info.csv，包含企业的变更信息，包括变更时间、变更内容等。
- 数据集5：news_info.csv，包含企业的新闻舆情信息，包括新闻时间、新闻正负面性等。
- 数据集6：other_info.csv，包含企业其他信息，包括企业知识产权、裁判文书数量等。
- 数据集7：enterprise_info.csv，带标注的企业数据，是否非法集资标注。
- 数据集8：enterprise_evaluate.csv，验证集，score 列为空。

数据集较多，但企业基本数据包含了企业的大部分信息，即表明企业非法集资的基本风险。其他数据集由于缺失值较多、有效企业较少，故企业的行为风险等方面特征较难挖掘。所以大部分的时间可以用来挖掘处理企业的基本风险。

2) A榜结果，排名（截图）

企业非法集资
风险预测

中国计算机学会 & 中国科大智慧城市研究院

队伍 / 人数: 3403 / 4210 奖金: ¥ 50,000

A 榜 (初赛 10.13 ~ 12.06) B 榜

赛制规则 数据与评测 排榜 参赛队伍 作品提交 交流讨论 常见问题 已报名

初赛排行榜 A 榜 B 榜

周榜单 我的成绩
到目前为止，您的最好成绩为 **0.84720037** 分，第 **45** 名，在本阶段中，您已超越 **813** 支队伍。

my dear dalao please daidai wo

45 ↓ 2 my dear dalao please... 0.84720037 125 2020-12-04 21:02 查看建模过程

3) B榜结果，排名（截图）

企业非法集资
风险预测

中国计算机学会 & 中国科大智慧城市研究院

队伍 / 人数: 3403 / 4210 奖金: ¥ 50,000

A 榜 (初赛 10.13 ~ 12.06) B 榜

赛制规则 数据与评测 排榜 参赛队伍 作品提交 交流讨论 常见问题 已报名

初赛排行榜 A 榜 B 榜

周榜单 我的成绩
到目前为止，您的最好成绩为 **0.84342005** 分，第 **27** 名，在本阶段中，您已超越 **845** 支队伍。

my dear dalao please daidai wo

27 ↑ 14 my dear dalao please... 0.84342005 1 2020-12-06 09:01 查看建模过程

4) 全部提交历史记录 (截图)

10.19 ~ 12.06 总计 126 次提交，全部提交记录长图：

[last_day_01.csv](#)

所在赛程： 初赛 - B榜 状态 / 得分 **0.84342005246** [查看日志](#) 提交时间 **2020/12/06 09:01**

备注： final

[last_day_03.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84335478372** [查看日志](#) 提交时间 **2020/12/04 21:12**

备注： 无备注信息

[last_day_02.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84720037045** [查看日志](#) 提交时间 **2020/12/04 21:02**

备注： 无备注信息

[last_day_01.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84576914479** [查看日志](#) 提交时间 **2020/12/04 20:48**

备注： 无备注信息

[submit2020_12_03_03.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84261559501** [查看日志](#) 提交时间 **2020/12/03 23:59**

备注： 无备注信息

[submit2020_12_03_02.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84261559501** [查看日志](#) 提交时间 **2020/12/03 23:57**

备注： pseudo

[submit2020_12_03_01.csv](#)

所在赛程： 初赛 - A榜 状态 / 得分 **0.84187867506** [查看日志](#) 提交时间 **2020/12/03 23:44**

备注： 45,55 retrain

[submit2020_12_02_03.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84105631507 查看日志	2020/12/02 23:57

备注: 让我们一起摇摆~

[submit2020_12_02_02.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84710900710 查看日志	2020/12/02 16:15

备注: 不搞了不搞了

[submit2020_12_02_02.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	程序错误 查看日志	2020/12/02 16:10

备注: 不搞了不搞了

[submit2020_12_02_01.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83962992371 查看日志	2020/12/02 10:59

备注: 无备注信息

[submit2020_12_01_03.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84348265085 查看日志	2020/12/01 23:59

备注: 无备注信息

[submit2020_12_01_02.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84187867506 查看日志	2020/12/01 23:55

备注: 无备注信息

[submit2020_12_01_01.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84261559501 查看日志	2020/12/01 23:34

备注: 无备注信息

[submit2020_11_30_03.csv](#) ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.84183629189 查看日志	2020/11/30 23:23

备注：无备注信息

submit2020_11_30_02.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.84027566162** 查看日志 **2020/11/30 23:14**

备注：无备注信息

submit2020_11_30_01.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83949432859** 查看日志 **2020/11/30 23:06**

备注：无备注信息

submit2020_11_29_03.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83935881240** 查看日志 **2020/11/29 23:56**

备注：911

submit2020_11_29_02.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83847647912** 查看日志 **2020/11/29 23:55**

备注：0.55 891

submit2020_11_29_01.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.84257628356** 查看日志 **2020/11/29 23:47**

备注：depth = 5, num_leaves = (10, 12), learning_rate = 0.01, n_estimators = 500

submit2020_11_27_03.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.84257628356** 查看日志 **2020/11/27 23:35**

备注：depth = 6, num_leaves = (8, 12), 填补enttypeitem, 过滤opscope, 5in, 3null S_score: 0.8441535583658499 S_std: 0.0013356606987076495 S_5-flo_std: 0.009582108348816053 f1_score: 0.8464149454330416 f1_std: 0.0012117196055160167 f1_5-flo_std: 0.00867676329374378

submit2020_11_27_02.csv ↴

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.84183629189** 查看日志 **2020/11/27 23:21**

备注：depth = 6, num_leaves = (8, 12), 填补enttypeitem, 过滤opscope S_score: 0.8446477568409806 S_std: 0.0007738557783017095 S_5-flo_std: 0.011024039063245691 f1_score: 0.8470699354140072 f1_std: 0.0006464458235557146 f1_5-flo_std: 0.009983609901513006 label1 = 9

[submit2020_11_27_01.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84119259714

[查看日志](#)

提交时间

2020/11/27 23:16

备注: depth = 6, num_leaves = (8, 12), 填补enttypeitem, S_score: 0.8411275851694283 S_std: 0.000893370391662944 S_5-fold_std: 0.012688120398782319 f1_score: 0.8435884761980276 f1_std: 0.0008189783007452315 f1_5-fold_std: 0.011446618968497056

[submit2020_11_26_03.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84265810240

[查看日志](#)

提交时间

2020/11/26 21:26

备注: +report_null, +bg120,131 num_leaves = (6, 8) 耗子尾汁

[submit2020_11_26_02.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84031780006

[查看日志](#)

提交时间

2020/11/26 20:35

备注: depth = 6, num_leaves = (8, 12)

[submit2020_11_26_01.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84261559501

[查看日志](#)

提交时间

2020/11/26 19:28

备注: 清理opscope

[submit2020_11_25_02.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84261559501

[查看日志](#)

提交时间

2020/11/25 23:58

备注: 去5个in

[submit2020_11_25_01.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.84339422733

[查看日志](#)

提交时间

2020/11/25 23:53

备注: baseline 补enttype, 5个in

[submission_2020_11_20_4.csv](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.83393314090

[查看日志](#)

提交时间

2020/11/20 23:53

备注: 5-fold

[submission_2020_11_20_2.csv](#)

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83785579230 查看日志	2020/11/20 23:31

备注: +cb & report

[submission_2020_11_20_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83413903414 查看日志	2020/11/20 23:17

备注: stacking(LR)

[submission_2020_11_19_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82925583099 查看日志	2020/11/19 23:54

备注: nnb调lgbm

[submission_2020_11_19_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83357356631 查看日志	2020/11/19 23:32

备注: nnb+4个null

[submission_2020_11_19_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83357356631 查看日志	2020/11/19 22:07

备注: +report

[submission_2020_11_18_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82695630339 查看日志	2020/11/18 23:55

备注: +bg120131

[submission_2020_11_18_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82721556949 查看日志	2020/11/18 23:40

备注: +2个freq

[submission_2020_11_18_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83234163567 查看日志	2020/11/18 10:42

备注: opfrom_ym -> opfrom_year_freq & near & this & year_cat(label_enc)

submission_2020_11_17_3.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83076123721 查看日志	2020/11/17 14:09

备注: +report & bg

submission_2020_11_17_2.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83147683020 查看日志	2020/11/17 10:14

备注: +len

submission_2020_11_17_1.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83363671181 查看日志	2020/11/17 09:46

备注: opscopecluster -> opscope

submission_2020_11_16_3.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82910318062 查看日志	2020/11/16 22:44

备注: 特征选择, 去重要度较低的列以及位置列

submission_2020_11_16_2.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83199645665 查看日志	2020/11/16 22:24

备注: +opfrom_year_cat, year^townsign^enttypeitem

submission_2020_11_16_1.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83234163567 查看日志	2020/11/16 22:02

备注: 用-1填空值, 5个in, 3个null_count, news表3个year_max

submission_2020_11_15_3.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83436107448 查看日志	2020/11/15 23:56

备注: +2个lgbm

submission_2020_11_15_2.csv ↓

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.83120684909 查看日志	2020/11/15 23:51
备注: 0.6rf + 0.2xgb + 0.2lgbm		

submission_2020_11_15_1.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.83581646594** [查看日志](#) 提交时间 **2020/11/15 23:47**

备注: new new baseline插入参数, 插入5个in

submission_2020_11_14_4.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.82800970680** [查看日志](#) 提交时间 **2020/11/14 09:57**

备注: label1 = 940 - year^town^enttype + five in tags cluster30 -> cluster3000 cv = (0.842, 0.839) 1.0lgbm no matter.

submission_2020_11_14_3.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.82758970358** [查看日志](#) 提交时间 **2020/11/14 00:40**

备注: 回溯 label1 = 884 only + year^town^enttypeminu(lb_enc) cv = (0.843, 0.840) early stopping cv = (0.844, 0.844) 0.830 -> ? 白给的王

submission_2020_11_14_2.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.81479782511** [查看日志](#) 提交时间 **2020/11/14 00:24**

备注: label1 = 934, - oploc, orgid, jobid, + year^town^enttypeminu(lb_enc) industryco -> industryco_cb 1.0lgbm cv = (0.846, 0.842) 0.830 -> ?

submission_2020_11_13_3.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.82766975184** [查看日志](#) 提交时间 **2020/11/13 12:14**

备注: stacking, 0.826 or 0.836

submission_2020_11_13_2.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.83047943689** [查看日志](#) 提交时间 **2020/11/13 10:47**

备注: 0.3xgb+0.3lgb+0.4rf opfrom_year_cat_townsign only cv: (0.841, 0.840)

submission_2020_11_13_1.csv ↴
所在赛程:
初赛 - A榜

状态 / 得分 **0.82933654102** [查看日志](#) 提交时间 **2020/11/13 00:02**

备注: + opfrom_year_cat_townsign - opfrom_year_cv becomes -> 0.842, 0.839

[submission_2020_11_12_3.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82910318062[查看日志](#)

提交时间

2020/11/12 23:44

备注: + townsign ^ opfrom_year_cat cv improves 0.001

[submission_2020_11_12_2.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82910318062[查看日志](#)

提交时间

2020/11/12 23:38

备注: + townsign ^ enttypeminu + townsign ^ opfrom_year_cat cv not change, worst folds improved

[submission_2020_11_12_1.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.83133581258[查看日志](#)

提交时间

2020/11/12 21:01

备注: 耗子尾汁

[submission_2020_11_11_3.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82192160679[查看日志](#)

提交时间

2020/11/11 15:08

备注: 多表, 预言个屁, dpx说815啊, 很快啊

[submission_2020_11_11_2.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82281669317[查看日志](#)

提交时间

2020/11/11 12:47

备注: 开裂

[submission_2020_11_11_1.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82346805007[查看日志](#)

提交时间

2020/11/11 12:05

备注: 无备注信息

[submission_2020_11_10_3.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82192160679[查看日志](#)

提交时间

2020/11/10 23:47

备注: num_leaves 6->8, 本地还是跟上次一样差, 估计还是825左右

[submission_2020_11_10_2.csv ↓](#)

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82209589811 查看日志	2020/11/10 23:45

备注: opscope_cluster -> 7

[submission_2020_11_10_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82889656714 查看日志	2020/11/10 23:31

备注: one_hot industryphy

[submission_2020_11_9_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82903129827 查看日志	2020/11/09 21:18

备注: 我裂开了, 0.836+cross features(town,entitem,opfrom)

[submission_2020_11_9_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.80659779101 查看日志	2020/11/09 20:37

备注: 无备注信息

[submission_2020_11_9_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.81326393163 查看日志	2020/11/09 20:04

备注: 无备注信息

[submission_2020_11_8_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.81661842206 查看日志	2020/11/08 22:14

备注: 无备注信息

[submission_2020_11_8_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.81942209204 查看日志	2020/11/08 22:08

备注: 无备注信息

[submission_2020_11_8_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.81862005536 查看日志	2020/11/08 22:02

备注: last one

[submission_2020_11_7_3.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82380338319** [查看日志](#) **2020/11/07 00:10**

备注: 单走一个随机森林, 傻逼

[submission_2020_11_7_2.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82554387552** [查看日志](#) **2020/11/07 00:01**

备注: 896, 0.6

[submission_2020_11_7_1.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82477080017** [查看日志](#) **2020/11/07 00:01**

备注: 969, 0.5

[submission_2020_11_6_3.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83622877811** [查看日志](#) **2020/11/06 12:28**

备注: 0.55 预言家: 0.828

[submission_2020_11_6_2.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83571131229** [查看日志](#) **2020/11/06 11:58**

备注: 0.815

[submission_2020_11_6_1.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83035671437** [查看日志](#) **2020/11/06 10:45**

备注: 3000clusters, +n, p, np, for, pub 预言家: 0.828

[submission_2020_11_5_3.csv](#) ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82671148519** [查看日志](#) **2020/11/05 23:59**

备注: 0.545, opscope->opscope_cluster 预言家说0.825

[submission_2020_11_5_2.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.82377265783** 提交时间 **2020/11/05 22:27**

备注: 叠加cluster 0.823

[submission_2020_11_5_1.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.82665137360** 提交时间 **2020/11/05 21:08**

备注: opscope -> opscope_cluster 盲猜0.828

[submission_2020_11_4_3.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.82592837759** 提交时间 **2020/11/04 11:12**

备注: 回滚0.8328, 提高n, 盲猜0.828

[submission_2020_11_4_2.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.83041653563** 提交时间 **2020/11/04 10:48**

备注: - q, qn, 我觉得0.830吧?

[submission_2020_11_4_1.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.82871750977** 提交时间 **2020/11/04 10:09**

备注: +'FORINVESTSIGN_min', 'PUBSTATE_max'

[submission_2020_11_3_3.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.83284844353** 提交时间 **2020/11/03 23:59**

备注: +p, +pn

[submission_2020_11_3_2.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.83205947697** 提交时间 **2020/11/03 23:56**

备注: 回滚0.830, +negtive

[submission_2020_11_3_1.csv](#)

所在赛程: 初赛 - A榜 状态 / 得分 **0.82810406341** 提交时间 **2020/11/03 00:11**

备注: stacking 0.3scf + 0.3rf + 0.2xgb + 0.2lgbm

submission_2020_11_2_3.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.83054538762** 查看日志 2020/11/02 23:59

备注: 调参

submission_2020_11_2_2.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82346805007** 查看日志 2020/11/02 23:48

备注: +opform, +compform

submission_2020_11_2_1.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82020449636** 查看日志 2020/11/02 12:08

备注: 回滚0.828, oploc -> oploc_cat

submission_2020_11_1_3.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82731084297** 查看日志 2020/11/01 11:38

备注: 回滚0.828, 删forregcap, forreccap, congro

submission_2020_11_1_2.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82426497156** 查看日志 2020/11/01 11:32

备注: 删exenum

submission_2020_11_1_1.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82871750977** 查看日志 2020/11/01 11:15

备注: opfrom_year -> opfrom_ym

submission_2020_10_31_3.csv ↓

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82651690266** 查看日志 2020/10/31 23:56

备注: 原0.825baseline, 删俩id列

submission_2020_10_31_2.csv ↓

[submission_2020_10_31_2.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82114447461[查看日志](#)

提交时间

2020/10/31 23:37

备注: 原0.825baseline, 删2020, label1: 905

[submission_2020_10_31_1.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.81914539087[查看日志](#)

提交时间

2020/10/31 23:32

备注: 这要白给啊, 删掉opscope, 删掉2020

[submission_2020_10_30_4.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82473961233[查看日志](#)

提交时间

2020/10/30 15:49

备注: label1 : 927 去无用列, opfrom_year -> opfrom_year_cat

[submission_2020_10_30_3.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82173796546[查看日志](#)

提交时间

2020/10/30 12:15

备注: 无备注信息

[submission_2020_10_30_2.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

程序错误[查看日志](#)

提交时间

2020/10/30 12:13

备注: 比赛要笑着打:)

[submission_2020_10_30_1.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82102396340[查看日志](#)

提交时间

2020/10/30 10:48

备注: op_from_year -> opfrom_year_cat +opscope_invest

[submission_2020_10_29_3.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82377265783[查看日志](#)

提交时间

2020/10/29 00:14

备注: 阿巴阿巴

[submission_2020_10_29_2.csv ↓](#)

所在赛程:

初赛 - A榜

状态 / 得分

0.82816986543[查看日志](#)

提交时间

2020/10/29 00:08

备注: ababab

[submission_2020_10_29_1.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82281669317** [查看日志](#) **2020/10/29 00:02**

备注: +industryphy_replace & enttypeitem_replace

[submission_2020_10_28_3.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82578790726** [查看日志](#) **2020/10/28 23:09**

备注: +bg_count_120 & bg_count_131, 微调参数

[submission_2020_10_28_2.csv ↓](#)

所在赛程: 状态 / 得分 提交时间

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82578790726** [查看日志](#) **2020/10/28 12:42**

备注: 这个是真的真的真的baseline啊

[submission_2020_10_28_1.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.81691858349** [查看日志](#) **2020/10/28 00:52**

备注: 去opfrom_year 加'parnum', 'townsign', 'enttypeminu', 'industryco', 'enttype', 'opfrom_year', 'FORINVESTSIGN_min', 'bg_count_131', 'bg_count_120', 'bg_count',

[submission_2020_10_27_3.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.79653162939** [查看日志](#) **2020/10/27 23:57**

备注: 'industryphy_replace', 'enttypeitem_replace', 'regcap_cat', 'reccap_cat', 'op_year_cat', 'opfrom_year' 6 features 0.5xgb+0.5lgbm

[submission_2020_10_27_2.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.81825678769** [查看日志](#) **2020/10/27 00:11**

备注: 新版 baseline 调参

[submission_2020_10_27_1.csv ↓](#)

所在赛程: 状态 / 得分 提交时间
初赛 - A榜 **0.82562516866** [查看日志](#) **2020/10/27 00:06**

备注: 第一版模型 (之前最优) , 阈值设为 0.45

[submission_2020_10_26_3.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82707886851

[查看日志](#)

提交时间

2020/10/26 12:49

备注: 帅啊 (+bg_per_year)

[submission_2020_10_26_2.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82765367951

[查看日志](#)

提交时间

2020/10/26 12:45

备注: 掉分吧 (+public_year)

[submission_2020_10_26_1.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82535074389

[查看日志](#)

提交时间

2020/10/26 00:11

备注: 别奶了 (+op_year & 调参, 0.3xgb+0.3lgbm+0.4rf)

[submission_2020_10_25_3.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82578790726

[查看日志](#)

提交时间

2020/10/25 23:29

备注: dpx毒奶875~887 (全新baseline)

[submission_2020_10_25_2.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.81582860786

[查看日志](#)

提交时间

2020/10/25 00:40

备注: bagging? (0xgb+0lgbm+1rf)

[submission_2020_10_25_1.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82562516866

[查看日志](#)

提交时间

2020/10/25 00:36

备注: 仪式感 (0.1xgb+0.4lgbm+0.5rf)

[submission_2020_10_24_3.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82066094315

[查看日志](#)

提交时间

2020/10/24 00:43

备注: 真的睡了 (+rp_year)

[submission_2020_10_24_2.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82465318635

[查看日志](#)

提交时间

2020/10/24 00:15

备注: 洗澡 (count & count_120 & count_131)

[submission_2020_10_24_1.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82342131268

[查看日志](#)

提交时间

2020/10/24 00:09

备注: 睡觉 (count & count_120)

[submission_2020_10_23_3.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82651690266

[查看日志](#)

提交时间

2020/10/23 23:59

备注: 无备注信息

[submission_2020_10_23_2.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82520759999

[查看日志](#)

提交时间

2020/10/23 23:12

备注: bg_count -> bg_count_120

[submission_2020_10_23_1.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82032125708

[查看日志](#)

提交时间

2020/10/23 00:35

备注: add feature bg

[submission_2020_10_21_4.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.82289433806

[查看日志](#)

提交时间

2020/10/22 12:14

备注: 我不想努力了 (去stacking)

[submission_2020_10_22_2.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.77995161638

[查看日志](#)

提交时间

2020/10/22 12:11

备注: stacking

[submission_2020_10_22_1.csv](#) ↴

所在赛程:

初赛 - A榜

状态 / 得分

0.78731659937

[查看日志](#)

提交时间

2020/10/22 09:34

备注: baseline (/ features)

[submission_2020_10_21_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.80283341570 查看日志	2020/10/21 11:54

备注: 无备注信息

[submission_2020_10_21_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.79971437516 查看日志	2020/10/21 10:22

备注: 无备注信息

[submission_2020_10_21_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.79049477692 查看日志	2020/10/21 09:32

备注: 无备注信息

[submission_2020_10_20_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.79837506462 查看日志	2020/10/20 12:35

备注: 无备注信息

[submission_2020_10_20_2.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.79661005764 查看日志	2020/10/20 12:13

备注: 无备注信息

[submission_2020_10_20_1.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.79273654451 查看日志	2020/10/20 10:46

备注: 无备注信息

[submission_2020_10_19_3.csv](#) ↴

所在赛程:	状态 / 得分	提交时间
初赛 - A榜	0.82492684981 查看日志	2020/10/19 22:45

备注: 无备注信息

[submission_2020_10_19_2.csv](#) ↴

所在赛程：
初赛 - A榜

状态 / 得分
0.8137309448

提交时间
2020/10/19 22:10

备注：无备注信息

[submission_2020_10_19_1.csv](#) ↗

所在赛程：
初赛 - A榜

状态 / 得分
0.79463148790

提交时间
2020/10/19 20:17

备注：无备注信息

热门

商务合作

关注DF公众号



二、组员及分工

组员	学号	分工
章子寅	201830671211	前期各自自学熟悉数据挖掘流程，完成各自的baseline。后期交流，EDA、提供构建特征思路。（特殊情况：主导完成另一赛题（大数据时代的Serverless工作负载预测）的代码编写，A榜rk67，B榜rk87。）
郑冰升	201830664169	前期各自自学熟悉数据挖掘流程，完成各自的baseline。后期交流，组织代码框架，特征工程，模型训练，参数调优。

三、算法说明

1) 方案概述

base_info 数据集包含了所有企业的基本信息，是企业非法集资基本风险来源。比赛前期仅针对该表进行分析预测，也能取得不错的结果，故重点对该表的特征进行提取，构建新特征、交叉特征。而其他数据集中包含的企业个数较少、数据质量较差，企业的行为风险很难体现，直接增加这类特征反而降低模型预测准确率，故简单构建一些统计特征来反应企业非法集资行为风险。最后，使用 LightGBM 模型进行预测，由于单模型预测方差较大，预测时采取多个相同模型进行参数扰动再对预测结果进行投票/取平均值，本地得分采用多次重复的随机 5 折交叉验证，建立较为靠谱的本地 cv。最后的提交基于 21 个 LightGBM 的预测结果，对难分类样本进行与 XGBoost 和 CatBoost 模型的投票，预测结果有较可观的提升。

2) 算法流程

a) 数据预处理

首先是缺失值的处理，数据集中有多列几乎完全缺失 (forreccap, congro, forregcap)，直接丢弃。分析发现，企业类型特征高度相关 (enttype, enttypeitem, enttypeminu, enttypegb)，而 enttypegb 列不存在空值，直接根据该列进行填充。其余特征发现填充后预测效果反而较差，直接用 -1 标识或不填充交由模型处理。

对于数据集中的日期特征 (opfrom, opto 等)，将其转化为时间戳，再进行分箱泛化，防止过拟合。对于文本描述的特征 (opscope)，由于存在符号、格式的问题，简单进行正则过滤。对于类别特征，直接进行计数编码，但将出现次数较少的类别统一替换为一个新类，进行泛化，部分进行 LabelEncode。

b) 构建新特征

构建数据完整度特征，企业信息完整度与非法集资风险关联，构建是否存在存在于其他数据集的01 特征 (in_news, in_report, in_tax, in_change, in_other)、构建基本信息表的信息完整度特征 (base_null_count)、在其他信息表中的信息完整度特征 (other_null_count)、企业年报表中的信息完整度特征 (report_null_count)。

由于企业信息脱敏，一些地址信息无法直接提取，但可以通过前缀比较得知是否在同一地区，故提取经营地址、企业机构代码等地址相关列的前缀，构造是否异地的特征 (district_FLAG1/2/3)。企业人数规模也是重要特征，构建人员特征 (person_SUM, person_null_SUM)。企业经营范围，进行聚类后发现效果一般，故直接提取经营范围的个数 (opscope_COUNT)。

EDA 发现企业年份和企业城镇标志与企业非法集资有很强相关性，主要表现为企业非法集资的集群在时间和空间上的迁移，在空间上非法集资现象有从乡镇到城市再到乡镇多发的趋势，在时间上非法集资案件逐年增加并在 2016 年达到顶峰后逐年减少。故先对年份进行分箱 (opfrom_cat)，构建交叉特征 (opfprom_cat_townsing)。

对于新闻表、税收表等其他数据集，简单构建统计特征 (min, max, mean, sum, count, nunique, std)。

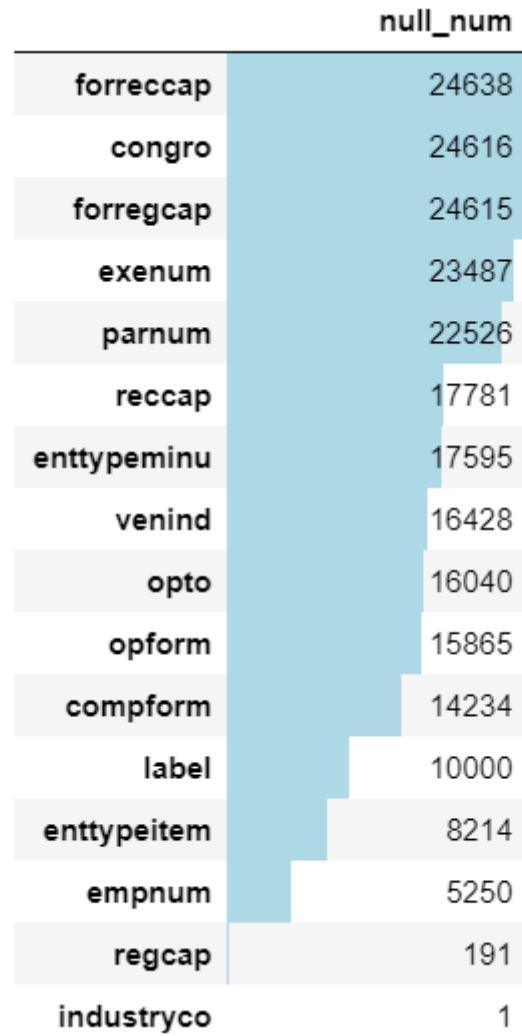
c) 交叉验证 & 模型训练

通过对抗验证，发现训练集与测试机分布极端不均匀，数据集的切分对模型预测结果有极大的影响，故在五折交叉验证的基础上，不固定随机种子，进行多次重复的随机划分的五折交叉验证，旨在得到一个中肯的本地 cv 得分。模型采取轻量的 LightGBM，由于单个模型预测方差较大，采用多个 LightGBM 进行参数扰动，对预测结果进行投票/取平均，得到最后的预测结果。由于发现难分类样本比例较大，最终提交结果是在 21 个 LightGBM 的参数扰动预测基础上，对难分类样本进行与 XGBoost 和 CatBoost 模型的投票，预测结果在本地和线上均有可观提升。

四、核心代码

1) EDA

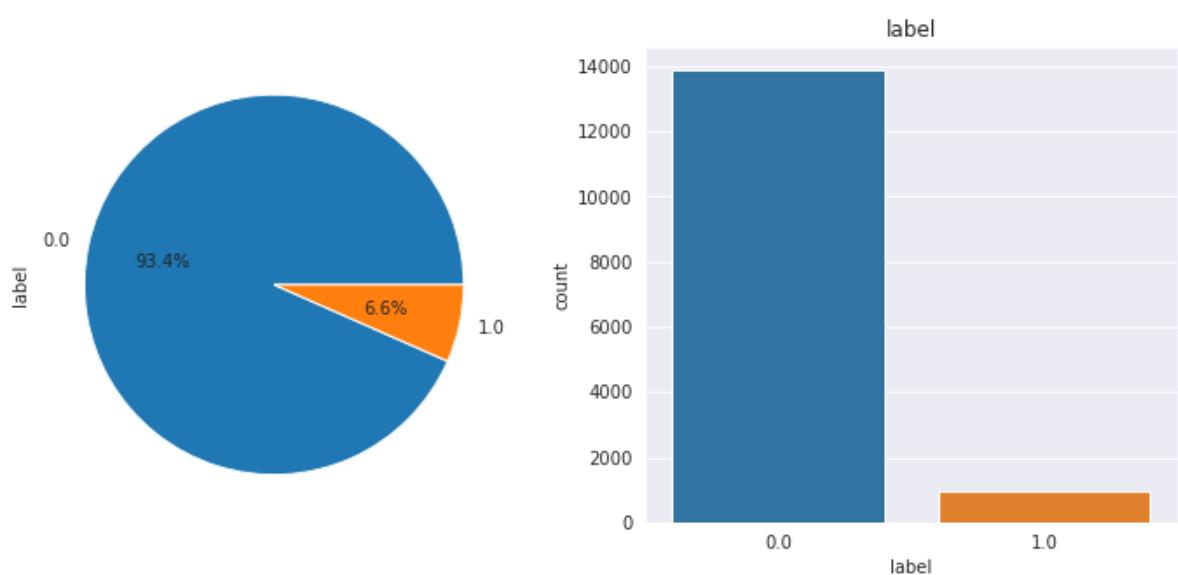
```
all_data.isnull().sum().sort_values(ascending =  
False).to_frame('null_num').style.bar(color = 'lightblue')
```



```

fig, ax = plt.subplots(1, 2, figsize = (12, 5))
data.label.value_counts().plot.pie(autopct = '%1.1f%%', ax = ax[0])
sns.countplot(x = 'label', data = data, ax = ax[1])
ax[1].set_title('label')
plt.show()

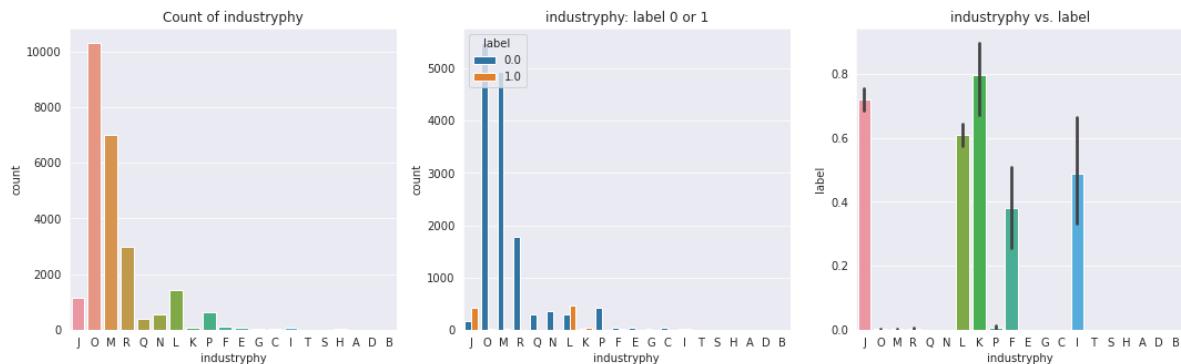
```



```

fig, ax = plt.subplots(1, 3, figsize = (18, 5))
sns.countplot(x = 'industryphy', data = data, ax = ax[0])
ax[0].set_title('Count of industryphy')
sns.countplot(x = 'industryphy', hue = 'label', data = data, ax = ax[1])
ax[1].set_title('industryphy: label 0 or 1')
sns.barplot(x = 'industryphy', y = 'label', data = data, ax = ax[2])
ax[2].set_title('industryphy vs. label')
plt.show()

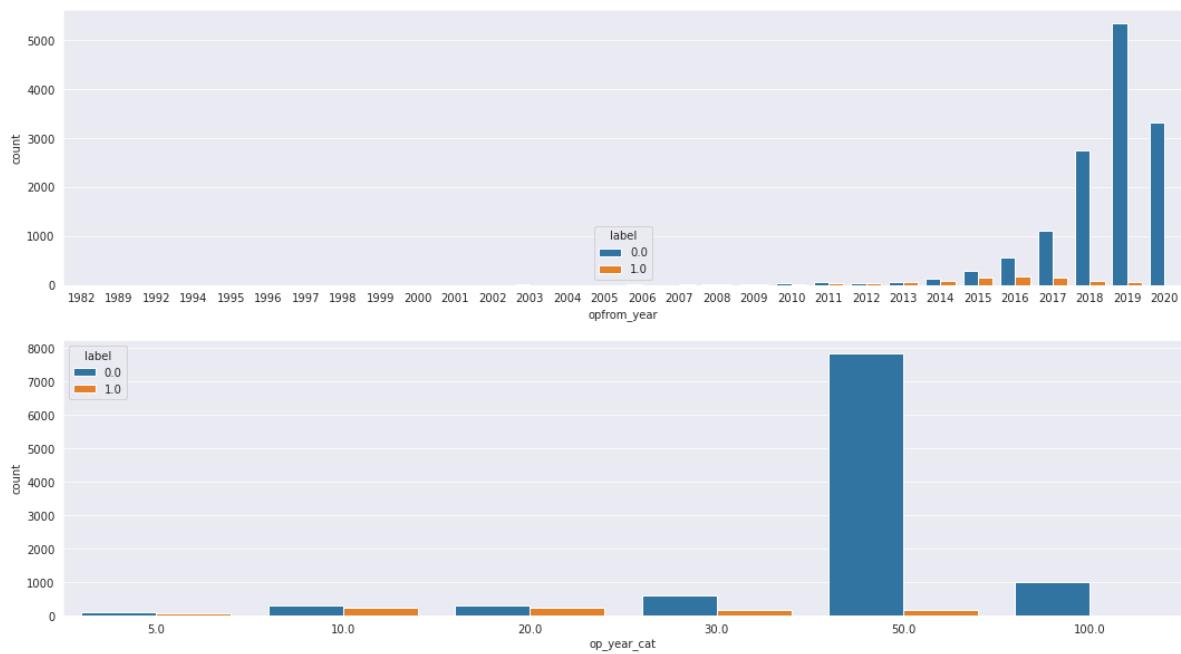
```



```

fig, ax = plt.subplots(2, 1, figsize = (18, 10))
sns.countplot(data['opfrom_year'], hue = data['label'], ax = ax[0])
sns.countplot(data['op_year_cat'], hue = data['label'], ax = ax[1])
plt.show()

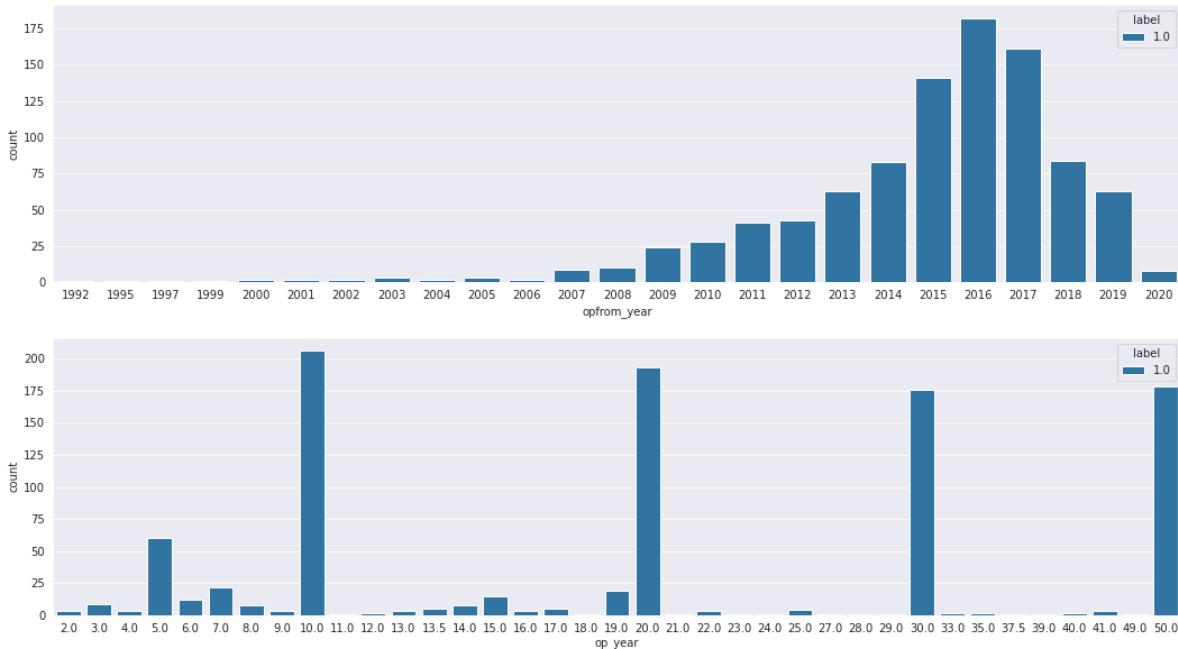
```



```

data = train.copy()
data = data[(data['op_year'].notnull()) & (data['opfrom_year'] <= 2020) &
            (data['op_year'] <= 50) & (data.label == 1)]
fig, ax = plt.subplots(2, 1, figsize = (18, 10))
sns.countplot(data['opfrom_year'], hue = data['label'], ax = ax[0])
sns.countplot(data['op_year'], hue = data['label'], ax = ax[1])
plt.show()

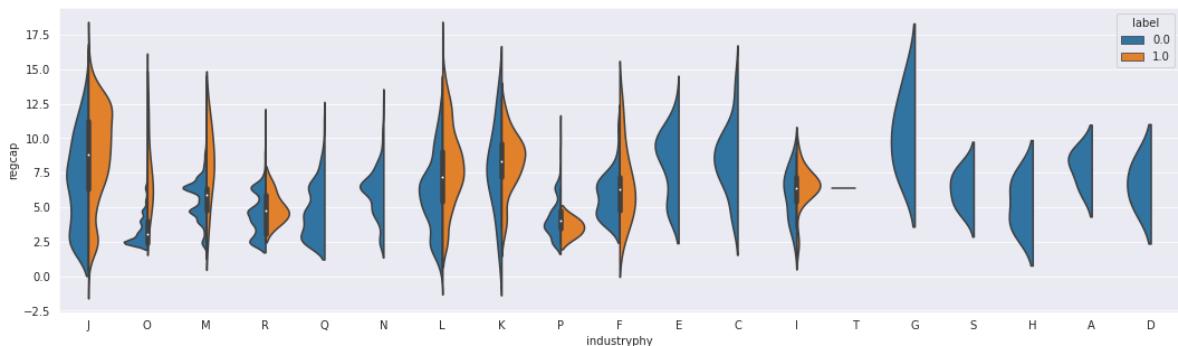
```



```

data = train.copy()
data = data[data['regcap'] != 0]
plt.figure(figsize = (18, 5))
fig, ax = plt.subplots(1, 1, figsize = (18, 5))
sns.violinplot(x = 'industryphy', y = 'regcap', hue = 'label', data = data,
split = True, ax = ax)
plt.show()

```



2) 特征工程

```

# 过滤缺失值过多的列
def filter_col_by_nan(df, ratio = 0.01):
    cols = []
    for col in df.columns:
        if df[col].isna().mean() >= (1 - ratio):
            cols.append(col)
    return cols

base_info = base_info.drop(filter_col_by_nan(base_info, 0.01), axis = 1)
report_info = report_info.drop(filter_col_by_nan(report_info, 0.01), axis = 1)

# 填充缺失值
base_info['enttypeitem'] = base_info['enttypeggb'].map(lambda x : int(x / 10) *
10)
base_info['enttypeminu'] = base_info['enttypeggb']
base_info.drop(['enttypeggb'], axis = 1, inplace = True)

```

```

# 是否在其他表出现
base_info['in_news'] = np.nan
base_info.loc[base_info['id'].isin(news_info['id']), 'in_news'] = 1
base_info['in_report'] = np.nan
base_info.loc[base_info['id'].isin(report_info['id']), 'in_report'] = 1
base_info['in_tax'] = np.nan
base_info.loc[base_info['id'].isin(tax_info['id']), 'in_tax'] = 1
base_info['in_change'] = np.nan
base_info.loc[base_info['id'].isin(change_info['id']), 'in_change'] = 1
base_info['in_other'] = np.nan
base_info.loc[base_info['id'].isin(other_info['id']), 'in_other'] = 1

# 计数编码特征
cat_col = ['oplocdistrict', 'industryphy', 'industryco', 'enttype',
           'enttypeitem', 'enttypeminu', 'opfrom_cat',
           'dom', 'oploc', 'opform']
for col in cat_col:
    base_info[col + '_COUNT'] =
base_info[col].map(base_info[col].value_counts())
col_idx = base_info[col].value_counts()
for idx in col_idx[col_idx < 8].index:
    base_info[col] = base_info[col].replace(idx, -1)

# 信息完整度特征
base_null_cols = ['enttypeitem', 'enttypeminu', 'exenum', 'empnum', 'parnum',
                  'reccap', 'opto', 'venind', 'opform', 'compform']
base_info['base_null_count'] = base_info[base_null_cols].apply(lambda x :
x.isnull().sum(), axis = 1)
other_null_cols = ['in_news', 'in_report', 'in_change', 'in_tax', 'in_other']
base_info['other_null_count'] = base_info[other_null_cols].apply(lambda x :
x.isnull().sum(), axis = 1)
base_info['total_null_count'] = base_info['base_null_count'] +
base_info['other_null_count']
base_info[other_null_cols] = base_info[other_null_cols].fillna(0)
report_null_cols = ['FUNDAM', 'EMPNUMSIGN', 'BUSSTNAME', 'COLGRANUM',
                    'FORINVESTSIGN', 'STOCKTRANSIGN']
report_info['report_null_count'] = report_info[report_null_cols].apply(lambda x :
x.isnull().sum(), axis = 1)

# 异地经营特征
base_info['district_FLAG1'] = (base_info['orgid'].fillna('')).apply(lambda x:
str(x)[:6]) == \
    base_info['oplocdistrict'].fillna('').apply(lambda x: str(x)
[:6])).astype(int)
base_info['district_FLAG2'] = (base_info['orgid'].fillna('')).apply(lambda x:
str(x)[:6]) == \
    base_info['jobid'].fillna('').apply(lambda x: str(x)[:6])).astype(int)
base_info['district_FLAG3'] =
(base_info['oplocdistrict'].fillna('').apply(lambda x: str(x)[:6]) == \
    base_info['jobid'].fillna('').apply(lambda x: str(x)[:6])).astype(int)

# 人数特征
base_info['person_SUM'] = base_info[['empnum', 'parnum', 'exenum']].sum(1)
base_info['person_NULL_SUM'] = base_info[['empnum', 'parnum',
                                         'exenum']].isnull().astype(int).sum(1)

# 经营范围
def conv_opscope(x):

```

```

x = x.replace('(', '(').replace(')', ')').replace(';', ';', ';').replace('.',
'.').replace(',', ',', ',')
x = x.replace('`', '`').replace(':', ':').replace(';', ';', ';').replace('.', '.')
x.replace('\"', '\"').replace('\"', '\"')
x = re.sub('\(未([^\(\)])*金融([^\(\)])*服务\)', '[未经批准不得融资]', x)
x = re.sub('\((([^(\)])*)\)', '', x)
x = x.strip('*\\')
x = ',' + x
x = re.sub('\\,(.*):', ',', , x)
x = x.lstrip(',')
return x
base_info['opscope'] = base_info['opscope'].map(conv_opscope)
base_info['opscope_COUNT'] = base_info['opscope'].apply(lambda x :
len(x.split(',')))

# 时间分箱，时间、空间的交叉特征
def conv_opfrom(x):
    if x <= 2006: return '1979_2006'
    elif x <= 2009: return '2007_2009'
    elif x <= 2016: return '2010_2016'
    elif x <= 2018: return '2017_2018'
    elif x <= 2019: return '2019_2019'
    elif x <= 2020: return '2020_2020'
    else: return -1
base_info['opfrom_cat'] = base_info['opfrom'].dt.year.map(conv_opfrom)
base_info['opfrom_cat'] = base_info['opfrom_cat'].astype(str) + '_' +
base_info['townsign'].astype(str)

# 其他表的统计特征
news_info_df = news_info.groupby('id').agg({'public_date': ['count', 'max',
'min', 'mean']}).reset_index()
news_info_df.columns = ['id', 'public_date_COUNT', 'public_MAX', 'public_MIN',
'public_MEAN']
news_info_df2 = pd.pivot_table(news_info, index = 'id', columns =
'positive_negrive', aggfunc = 'count').reset_index()
news_info_df2.columns = ['id', 'news_COUNT1', 'news_COUNT2', 'news_COUNT3']
news_info_df = pd.merge(news_info_df, news_info_df2)

tax_info_df = tax_info.groupby('id').agg({
    'TAX_CATEGORIES': ['count', 'nunique'],
    'TAX_ITEMS': ['nunique'],
    'TAXATION_BASIS': ['count'],
    'TAX_AMOUNT': ['max', 'mean', 'std'],
})
change_info_df = change_info.groupby('id').agg({
    'bgxmdm': ['count', 'nunique'],
    'bqa': ['nunique'],
    'bgh': ['nunique'],
    'bgrq': ['nunique'],
})
report_info['BUSSTNAME'] = report_info['BUSSTNAME'].map({'清算': 1, '停业': 2,
'歇业': 3, '开业': 4})
report_info_df = report_info.groupby('id').agg({
    'ANCHEYEAR': ['max'],
    'BUSSTNAME': ['min'],
    'FUNDAM': ['max', 'std'],
})

```

```

    'EMPNUM': ['max', 'std'],
    'UNEEMPLNUM': ['max', 'sum', 'std'],
    'PUBSTATE' : ['max', 'nunique'],
    'FORINVESTSIGN' : ['max'],
    'WEBSITSIGN' : ['max'],
    'STOCKTRANSIGN' : ['min']
})

```

3) 交叉验证

```

# 建立本地对准确率、召回率、F1的加权评分
def cal_score(confus, parse = True):
    p0 = 0 if confus[0][0] == 0 else confus[0][0] / (confus[0][0] + confus[1][0])
    r0 = 0 if confus[0][0] == 0 else confus[0][0] / (confus[0][0] + confus[0][1])
    p1 = 0 if confus[1][1] == 0 else confus[1][1] / (confus[1][1] + confus[0][1])
    r1 = 0 if confus[1][1] == 0 else confus[1][1] / (confus[1][1] + confus[1][0])
    f1_0 = 0 if p0 == 0 or r0 == 0 else 2 * p0 * r0 / (p0 + r0)
    f1_1 = 0 if p1 == 0 or r1 == 0 else 2 * p1 * r1 / (p1 + r1)
    f = f1_1
    s = 0.5 * p1 + 0.3 * r1 + 0.2 * f
    return 'P = {:.5f}, R = {:.5f}, F1 = {:.5f}, S = {:.5f}'.format(p1, r1, f, s)
    if parse else (p1, r1, f, s)

# 重训练，对难分类样本进行加权融合预测
def retrain(train_X, train_y, test_X, test_y, lower = 0.45, upper = 0.55):
    train = train_X.copy()
    train['label'] = train_y.copy()
    test = test_X.copy()
    test['label'] = test_y.copy()
    test_index_res = test[(test.label > lower) & (test.label < upper)].index
    test.loc[test_index_res, 'label'] *= 0.5

    xgb_clf = xgb.XGBClassifier(**{
        'learning_rate' : 0.03,
        'n_estimators' : 150,
        'max_depth' : 5,
        'min_child_weight' : 5,
        'objective' : 'binary:logistic',
        'random_state' : 2021,
    })

    cab_clf = cab.CatBoostClassifier(**{
        'learning_rate' : 0.03,
        'iterations' : 150,
        'depth' : 5,
        'loss_function' : 'Logloss',
        'random_state' : 2021,
        'silent' : True
    })

    xgb_clf.fit(train.drop(['label'], axis = 1), train['label'])
    test.loc[test_index_res, 'label'] += 0.25 *
    xgb_clf.predict_proba(test.loc[test_index_res].drop(['label'], axis = 1))[:, 1]

```

```

cab_clf.fit(train.drop(['label'], axis = 1), train['label'])
test.loc[test_index_res, 'label'] += 0.25 *
cab_clf.predict_proba(test.loc[test_index_res].drop(['label'], axis = 1))[:,1]

return test['label']

# 五折交叉验证，随机划分，不设置随机种子
def k_fold_serachParmeters(model, train_val_data, train_val_kind, test_kind):
    mean_S = 0
    mean_f1 = 0
    mean_f1Train = 0
    n_splits = 5
    test_S_scores = []
    test_f1_scores = []

    cat_features = ['oplocdistrict', 'industryphy', 'industryco', 'enttype',
                    'enttypeitem', 'enttypeminu', 'opfrom_cat',
                    'dom', 'oploc', 'opform']

    sk = StratifiedKFold(n_splits = n_splits, shuffle = True)
    pred_Test = np.zeros(len(test_kind))
    for train, test in sk.split(train_val_data, train_val_kind):
        x_train = train_val_data.iloc[train]
        y_train = train_val_kind.iloc[train]
        x_test = train_val_data.iloc[test]
        y_test = train_val_kind.iloc[test]

        pred = y_test.copy()
        pred = pseudo_label(model, x_train, y_train, x_test, pred, -0.05, 1.05)
        pred = [1 if x > 0.5 else 0 for x in pred]
        fper_class = eval_score(y_test, pred)
        pred_Train = model.predict(x_train)
        pred_test = model.predict_proba(test_kind)[:, 1]

        #
        # print('before:', np.sum(pred_test))
        # pred_test = retrain(x_train, y_train, test_kind, pred_test, 0.45,
        # 0.55)
        # print('after:', np.sum(pred_test), '\n')

        pred_Test += pred_test / n_splits
        fper_class_train = eval_score(y_train, pred_Train)
        mean_f1 += fper_class['f1'] / n_splits
        mean_f1Train += fper_class_train['f1'] / n_splits
        test_f1_scores.append(fper_class['f1'])
        S_score = my_score_func(y_test, pred)
        mean_S += S_score / n_splits
        test_S_scores.append(S_score)

    test_S_std = np.std(test_S_scores)
    test_f1_std = np.std(test_f1_scores)
    return mean_S, mean_f1, test_S_std, test_f1_std, pred_Test

```

4) 模型训练

```
np.random.seed(20212233)

score_tta = None
s_list = []
f1_list = []
s_std_list = []
f1_std_list = []

tta_fold = 21 # 21个LGB
pred_list = []
for _ in range(tta_fold):
    num_leaves = np.random.randint(6, 10)
    min_child_samples = np.random.randint(2, 6)
    max_depth = 5
    learning_rate = np.random.randint(1, 4)
    n_estimators = 150 if learning_rate == 3 else (250 if learning_rate == 2
else 450)
    learning_rate /= 100

    clf = lgb.LGBMClassifier(
        num_leaves = num_leaves, min_child_samples = min_child_samples,
        max_depth = max_depth, learning_rate = learning_rate,
        n_estimators = n_estimators, n_jobs = -1)

    S_sco, f1_sco, S_std, f1_std, test_pred = k_fold_serachParmaters(clf,
                           train_data.drop(['id', 'opscope', 'label'], axis =
1),
                           train_data['label'],
                           test_data.drop(['id', 'opscope'], axis = 1),
                           )

    if score_tta is None:
        score_tta = test_pred / tta_fold
    else:
        score_tta += test_pred / tta_fold
    pred_list.append(test_pred)
    s_list.append(S_sco)
    S_std_list.append(S_std)
    f1_list.append(f1_sco)
    f1_std_list.append(f1_std)

print('S_score:', np.array(s_list).mean(), 'S_std:', np.array(s_list).std(),
'S-5-flod_std:', np.array(S_std_list).mean())
print('')
score_list = ['{:.9f}'.format(x) for x in sorted(s_list)]
print(', '.join(score_list[:10]))
print(', '.join(score_list[-10:]))
print('\n')
print('f1_score:', np.array(f1_list).mean(), 'f1_std:', np.array(f1_list).std(),
'f1-5-flod_std:', np.array(f1_std_list).mean())
print('')
score_list = ['{:.9f}'.format(x) for x in sorted(f1_list)]
print(', '.join(score_list[:10]))
print(', '.join(score_list[-10:]))
```

5) 参数调优

```
# 绘制学习曲线
def plot_learning_curve(estimator, train_X, train_y, cv = 10, train_sizes =
np.linspace(0.1, 1.0, 5), title = ''):

    plt.figure(figsize = (12, 6))
    plt.title(title)
    plt.xlabel('Training examples')
    plt.ylabel('Score')
    train_sizes, train_scores, test_scores = learning_curve(estimator, train_X,
train_y,
                                                    cv = cv, train_sizes
= train_sizes,
                                                    scoring = 'f1',
random_state = 2021)
    train_scores_mean = np.mean(train_scores, axis = 1)
    train_scores_std = np.std(train_scores, axis = 1)
    test_scores_mean = np.mean(test_scores, axis = 1)
    test_scores_std = np.std(test_scores, axis = 1)
    plt.grid()

    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                    train_scores_mean + train_scores_std,
                    alpha = 0.1, color = 'g')
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                    test_scores_mean + test_scores_std,
                    alpha = 0.1, color = 'r')
    plt.plot(train_sizes, train_scores_mean, 'o-', color='g',
label = 'traning score')
    plt.plot(train_sizes, test_scores_mean, 'o-', color='r',
label = 'testing score')
    plt.legend(loc = 'best')
    return plt

model = lgb.LGBMClassifier(
    num_leaves = 8, min_child_samples = 5,
    max_depth = 5, learning_rate = 0.03,
    n_estimators = 150, n_jobs = -1)

plot_learning_curve(
    model,
    train_data.drop(['id', 'opscope', 'label'], axis = 1),
    train_data['label'], cv = 5)

# 随机网格搜索参数
cv_params = {
#     'learning_rate' : np.arange(0.01, 0.3, 30),
#     'n_estimators' : range(50, 1001, 50),
#     'max_depth' : range(10, 60, 1),
#     'num_leaves' : range(10, 14, 1),
#     'min_child_weight' : range(3, 30, 1),
}
other_params = {
    'learning_rate' : 0.03,
    'n_estimators' : 150,
    'max_depth' : 5,
```

```
'num_leaves' : 8,
'subsample' : 0.8,
'feature_fraction' : 0.8,
'boosting_type' : 'gbdt',
'objective' : 'binary',
'random_state' : 20201109,
}
model = LGBMClassifier(**other_params)
grid_search = RandomizedSearchCV(
    estimator = model,
    param_distributions = cv_params,
    scoring = my_score,
    cv = StratifiedKFold(n_splits = 5, shuffle = True, random_state = 2020),
    verbose = True,
    n_jobs = -1,
    random_state = 20202,
)
grid_search.fit(x, y)
evaluate_result = grid_search.cv_results_
best_model = grid_search.best_estimator_
print(pd.DataFrame(evaluate_result))
print(grid_search.best_params_)
```