

**大数据挖掘与机器学习**

**课程设计**

**题 目** 银行精准营销客户存款预测

**学 院** 商学院

**专 业**

**班 级**

**学 生**

**指导教师** 蔡永明

**二〇二二年七月二日**

注：

去年课程设计题目：为中国数据挖掘顶级大赛，中国计算机学CCF大数据与计算智能大赛（CCF Big Data & Computing Intelligence Contest，简称CCF BDCI），第九届大数据与人工智能领域的算法、应用和系统大型挑战赛中的“中原银行个贷违约预测”。大赛链接<https://www.datafountain.cn/competitions/530>

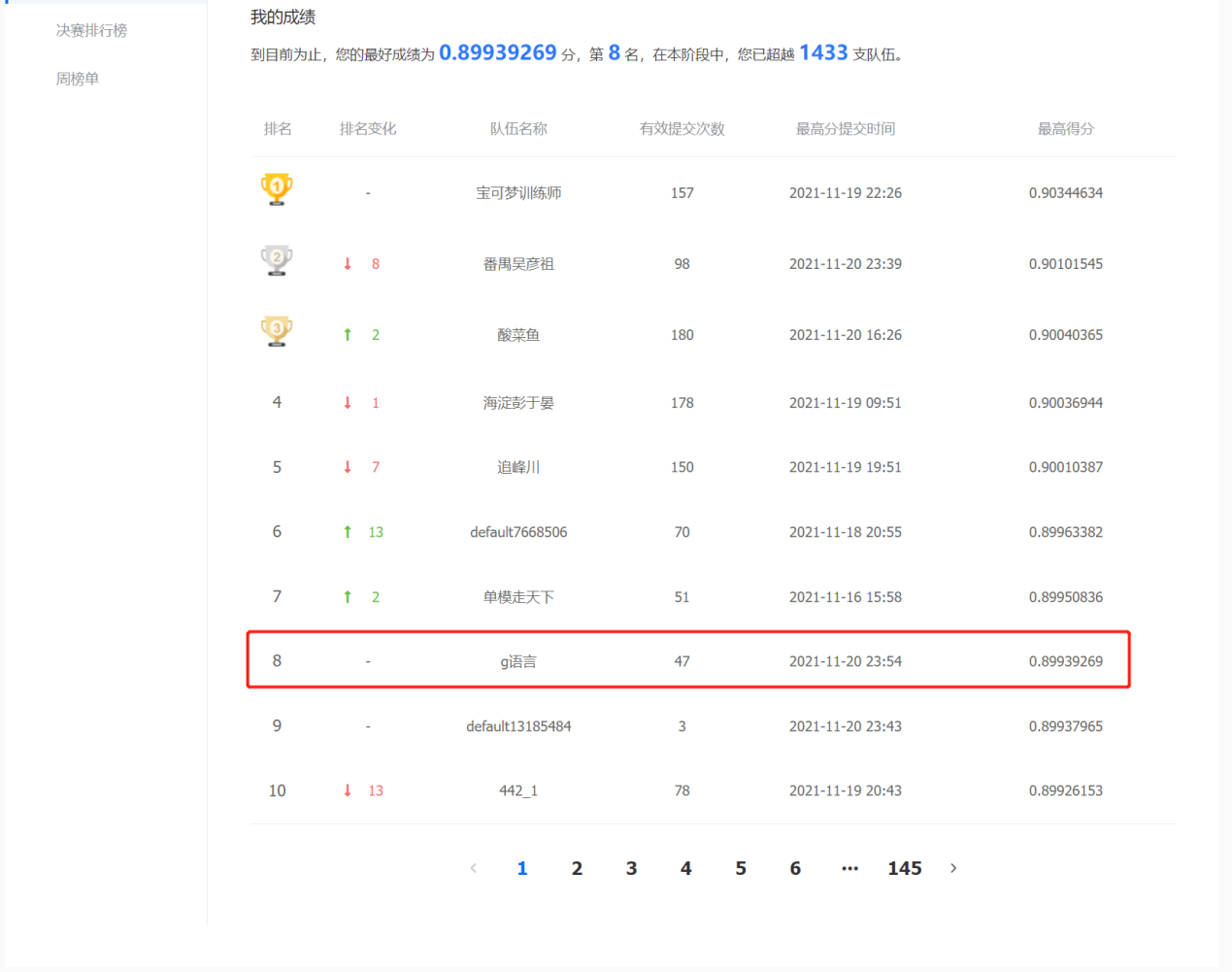
学生直接报名参加比赛，官网比赛A磅成绩为课程设计成绩。

今年没有组织大家参赛，感兴趣的同学，登录网站自己看看找适合的比赛参加！

本范例是信管1903段智龙同学团队（信管1904万杰、信管1903冉令坤、信管1901鹿晓琳、信管1903于欣煜）的参赛方案（也是我们的课程设计）。在1500多支队伍中，段智龙带领团队，获得A榜第8名的成绩（B榜成绩略有失误，无缘大奖），在前20的榜单里，几乎全是985名校队伍。

本范例仅作为课程设计学习，**请尊重段同学及其团队的知识版权，请勿外传**。

大赛排名页截图（A榜）



**目 录**

[1 原始数据的理解 4](#_Toc28143)

[1.1 所需数据集 4](#_Toc14415)

[1.2 数据集之间的关联 4](#_Toc17077)

[1.3 数据字段及含义 6](#_Toc17764)

[1.4 数据的基本描述 9](#_Toc31844)

[1.5 数据与现实意义 11](#_Toc27696)

[2 数据可视化 18](#_Toc10390)

[2.1 data\_all数据集的大体分布 18](#_Toc12662)

[2.2 train\_public和test\_public分布情况的比较 23](#_Toc30795)

[2.3 train\_internet和test\_public分布情况的比较 26](#_Toc27738)

[2.4 特征之间的相关程度 30](#_Toc12217)

[2.5 特征重要性 31](#_Toc23403)

[3 数据探索 33](#_Toc14738)

[3.1 删除字段 33](#_Toc29527)

[3.2 缺失值处理 34](#_Toc8589)

[3.3 定类和定序字段处理 35](#_Toc15301)

[3.4 数据标准化 36](#_Toc6206)

[3.5 数据归一化 37](#_Toc29797)

[4 数据预处理 38](#_Toc29294)

[4.1 构建新特征 38](#_Toc19267)

[5 数据建模 42](#_Toc26788)

[5.1 模型选择 42](#_Toc30421)

[5.2 模型构建思想 42](#_Toc16247)

[5.3 模型的分析 43](#_Toc22882)

[5.4 模型的调参 46](#_Toc2012)

[5.5 Stack的选择 47](#_Toc16561)

[6 数据后处理： 48](#_Toc24216)

[6.1 目的 48](#_Toc11298)

[6.2 过程 48](#_Toc3187)

[7 结论 51](#_Toc5281)

# 原始数据的理解

## 所需数据集

**训练数据**

  train\_public.csv 个人贷款违约记录数据

  train\_internet.csv 某网络信用贷产品违约记录数据

**测试数据**

  test\_public.csv 用于测试的数据，获取榜单排名

## 数据集之间的关联

train\_public.csv 个人贷款违约记录数据集包含10000条记录，train\_internet.csv 某网络信用贷产品违约记录数据集包含700000条记录，test\_public.csv 用于测试的数据集包含5000条数据。总共715000条记录，数据集大小适中。

（1）train\_internet.csv 数据集

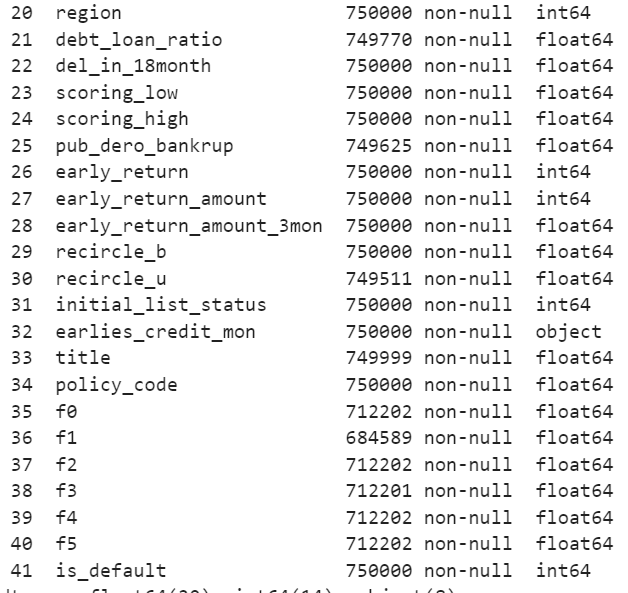
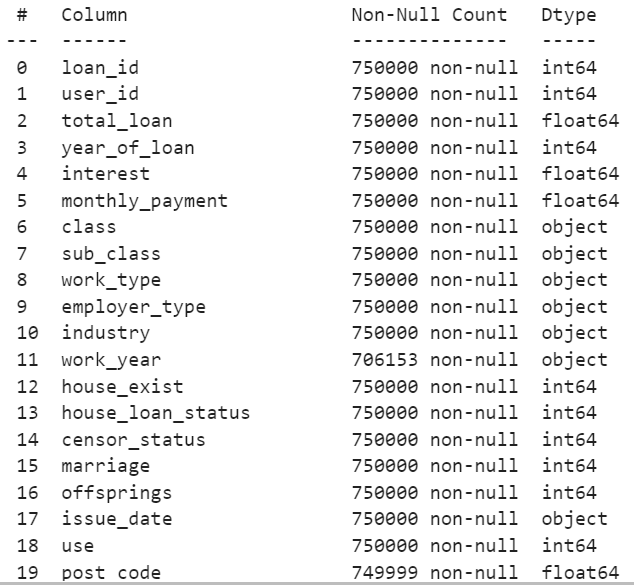


图1

解释：train\_internet.csv中的数据类型有float64类型的数据20个、int64类型的数据有14个、object类型的数据有8个。

（2）train\_public.csv 数据集

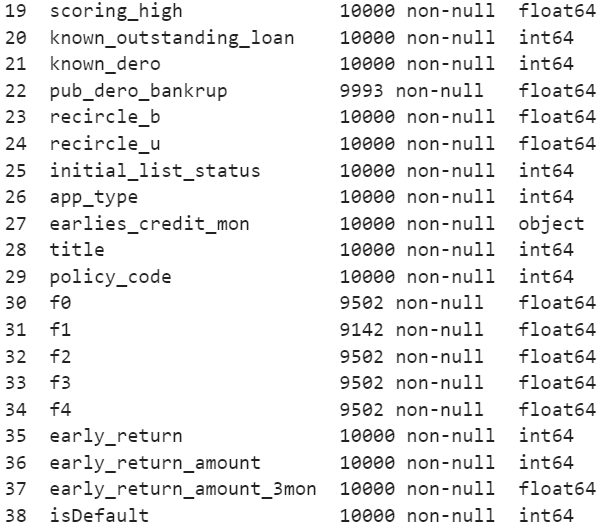
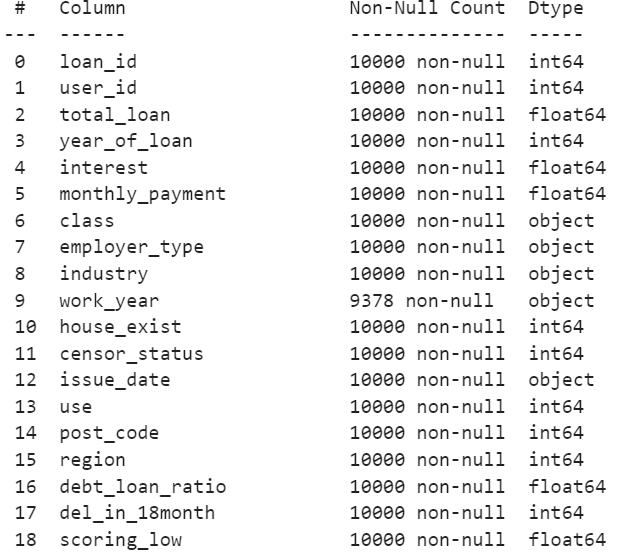


图2

解释：train\_public.csv中的数据类型有float64类型的数15个、int64类型的数据有18个、object类型的数据有6个。

（1）test\_public.csv 数据集

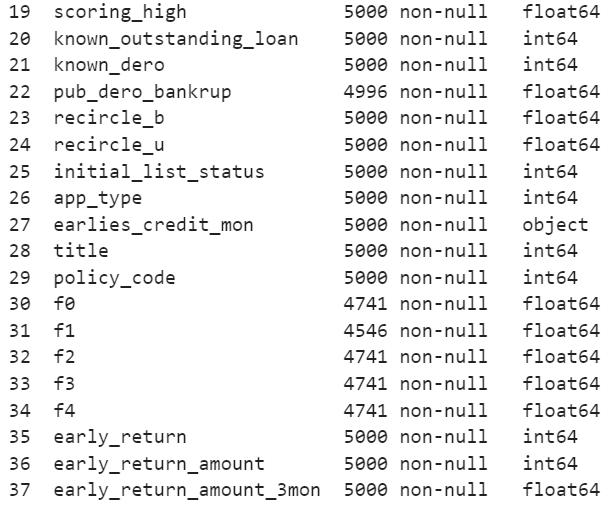


图3

解释：test\_public.csv中的数据类型有float64类型的数15个、int64类型的数据有17个、object类型的数据有6个。

for i in inter.columns:

if i not in train\_public.columns:

print(i)

总结：train\_internet.csv数据与train\_public.csv数据的区别：

Internet数据集字段个数比public数据集字段个数多3个，其中internet中含有的属性而public不含有的属性有6个，如下：

sub\_work\_type，house\_loan\_status，marriage，offsprings，f5

is\_default

Public中含有的属性而Internet中不含有的属性有4个，如下：

known\_outstanding\_loan，known\_dero，app\_type，isDefault

train\_public.csv与test\_public.csv属性字段相同。

## 数据字段及含义

（1）train\_public.csv数据字段含义描述

|  |  |
| --- | --- |
| 字段 | 字段描述 |
| loan\_id | 贷款记录唯一标识 |
| user\_id | 借款人唯一标识 |
| total\_loan | 贷款数额 |
| year\_of\_loan | 贷款年份 |
| interest | 当前贷款利率 |
| monthly\_payment | 分期付款金额 |
| grade | 贷款级别 |
| employment\_type | 所在公司类型（世界五百强、国有企业、普通企业…） |
| industry | 工作领域（传统工业、商业、互联网、金融…） |
| work\_year | 工作年限 |
| home\_exist | 是否有房 |
| censor\_status | 审核情况 |
| issue\_month | 贷款发放的月份 |
| use | 贷款用途类别 |
| post\_code | 贷款人申请时邮政编码 |
| region | 地区编码 |
| debt\_loan\_ratio | 债务收入比 |
| del\_in\_18month | 借款人过去18个月逾期30天以上的违约事件数 |
| scoring\_low | 借款人在贷款评分中所属的下限范围 |
| scoring\_high | 借款人在贷款评分中所属的上限范围 |
| known\_outstanding\_loan | 借款人档案中未结信用额度的数量 |
| known\_dero | 贬损公共记录的数量 |
| pub\_dero\_bankrup | 公开记录清除的数量 |
| recircle\_bal | 信贷周转余额合计 |
| recircle\_util | 循环额度利用率 |
| initial\_list\_status | 贷款的初始列表状态 |
| app\_type | 是否个人申请 |
| earlies\_credit\_mon | 借款人最早报告的信用额度开立的月份 |
| title | 借款人提供的贷款名称 |
| policy\_code | 公开可用的策略\_代码=1新产品不公开可用的策略\_代码=2 |
| f系列匿名特征 | 匿名特征f0-f4，为一些贷款人行为计数特征的处理 |
| early\_return | 借款人提前还款次数 |
| early\_return\_amount | 贷款人提前还款累积金额 |
| early\_return\_amount\_3mon | 近3个月内提前还款金额 |

1. train\_internet.csv数据字段含义描述

|  |  |
| --- | --- |
| loan\_id | 网络贷款记录唯一标识 |
| user\_id | 用户唯一标识 |
| total\_loan | 网络贷款金额 |
| year\_of\_loan | 网络贷款期限（year） |
| interest | 网络贷款利率 |
| monthly\_payment | 分期付款金额 |
| class | 网络贷款等级 |
| sub\_class | 网络贷款等级之子级 |
| work\_type | 工作类型（公务员、企业白领、创业…） |
| employment\_type | 所在公司类型（世界五百强、国有企业、普通企业…） |
| industry | 工作领域（传统工业、商业、互联网、金融…） |
| work\_year | 就业年限（年） |
| house\_ownership | 是否有房 |
| house\_loan\_status | 房屋贷款状况（无房贷、正在还房贷、已经还完房贷） |
| censor\_status | 验证状态 |
| marriage | 婚姻状态（未婚、已婚、离异、丧偶） |
| offsprings | 子女状态(无子女、学前、小学、中学、大学、工作) |
| issue\_date | 网络贷款发放的月份 |
| use | 贷款用途 |
| post\_code | 借款人邮政编码的前3位 |
| region | 地区编码 |
| debt\_loan\_ratio | 债务收入比 |
| del\_in\_18month | 借款人过去18个月信用档案中逾期60天内的违约事件数 |
| scoring\_low | 借款人在信用评分系统所属的下限范围 |
| scoring\_high | 借款人在信用评分系统所属的上限范围 |
| pub\_dero\_bankrup | 公开记录清除的数量 |
| early\_return | 提前还款次数 |
| early\_return\_amount | 提前还款累积金额 |
| early\_return\_amount\_3mon | 近3个月内提前还款金额 |
| recircle\_bal | 信贷周转余额合计 |
| recircle\_util | 循环额度利用率，或借款人使用的相对于所有可用循环信贷的信贷金额 |
| initial\_list\_status | 网络贷款的初始列表状态 |
| earlies\_credit\_line | 网络贷款信用额度开立的月份 |
| title | 借款人提供的网络贷款名称 |
| policy\_code | 公开策略=1不公开策略=2 |
| f系列匿名特征 | 匿名特征f0-f5，为一些网络贷款人行为计数特征的处理 |

## 数据的基本描述

**数据空缺值方面**：

在train\_internet.csv数据集中，共有12个特征含有空缺值。其中post\_code和title只有一行数据有空缺值，其次debt\_loan\_ratio，pub\_dero\_bankrup，recircle\_u有几百行数据有空缺值，work\_year以及f系列特征中含有空缺值的行数达到了37800~65411行之多。

在train\_public.csv和test\_public.csv数据集中的数据比较相似，含有空缺值的数据及特征也比较一致，都含有7个特征有空缺值。其中pub\_dero\_bankrup都只有几行数据含有空缺值，而work\_year以及f0~f4系列的特征就含有几百条数据中含有空缺值，在train\_public.csv的10000条数据中占比不大，而在test\_public.csv的5000条数据中占比还比较重要。

**使用descibe()方法对数值型数据进行基本情况的分析**：

（1）train\_internet



图4

1. train\_public



图5

1. test\_public



图6

## 数据与现实意义

(1)loan\_id是贷款记录唯一标识，代表特定人在特定时间贷款特定金额，相当于主键，能够唯一的标识一条记录。一个loan\_id只能检索出一条记录，不可重复。

(2)user\_id是贷款人的唯一标识，用id号来代表每个贷款人，不使用姓名，可以避免重复，同时简化存储量。

(3)total\_loan是贷款数额，在现实贷款中有重要的意义，贷款数额是指借款人与放款人签订的合同数额，是一个不变的数额，通俗一些讲，就是借钱人向给钱人所借去的金钱的数目。在贷款中首先需要明确贷款数额，要从多方面考虑，比如贷款人还款能力，贷款人可用额度等。

(4)year\_of\_loan指贷款年份，贷款年份指借去金额之后还款的日期，贷款年份的判定是贷款日的年份加期限，贷款涉及到利率，及需要支付一定的利息，所以贷款年份对于计算利息来说至关重要。一般情况来说，不同的贷款年份，利率不一样，贷款年份越多，利率越小。但同时随着贷款年份的增多，所需要偿还的利息也会增多，所以需要根据借款人的需求和能力，设置合理的贷款年份，尽量延长贷款时间的同时，降低贷款利息。对于个人贷款来说，银行的贷款年份最长是30年，一般不超过20年。还款方式一般有：等额本息还款，等额本金还款，一次性还本付息，阶段性等额本息还款，先息后本等。

1. interest指当年贷款的利率，对于贷款来说，最重要的就是利率，利率是利息计算的重要依据。贷款利率，是银行等金融机构发放贷款时向借款人收取利息的利率。主要分为三类：中央银行对商业银行的贷款利率;商业银行对客户的贷款利率;同业拆借利率。利率分为按照时间分为年利率、日利率、月利率，换算公式如下：日利率(0/000)=年利率(%)÷360=月利率(‰)÷30，月利率(‰)=年利率(%)÷12。
2. monthly\_payment是分期付款金额，对于贷款来说，一般采用的还款方式是分期付款，分期还款是**按照最初的分期总金额乘以一定的费率系数来计算**，并不是按照每月剩余的未还金额来计算的。这是最为普遍的还款方式，一般银行也会推荐借钱方采纳这类方式还款，由于这类还款方式的还款额每月都一样，方便记忆，借钱方也便于安排收支，适合收入稳定的借钱方。不过，由于利息不会随本金的偿还而减少，正因如此还款总利息相对较高，且银行资本占用时间长。一般对于按照年为单位的贷款来说，还款的分期一般以月为单位，每月需要还款多少有一定的规则和计算标准。
3. grade是贷款级别，早期贷款级别划分为正常、逾期、呆滞、呆账四种类型，后三种合称为不良贷款。后来根据借款人的实际还款能力进行了五级分类，按照风险程度排列分别为：正常、关注、次级、可疑、损失。后三种可划分为不良贷款。正常的贷款是指无逾期。借款人能够履行合同，一直能正常还本付息，不存在任何影响贷款本息及时全额偿还的消极因素，银行对借款人按时足额偿还贷款本息有充分把握。贷款损失的概率为0。这个比赛最终的isdefault的值就是预测贷款人是否是属于正常贷款，是否存在着违约行为。
4. employment\_type指所在公司的类型，一个人的还款能力何其收入水平是直接挂钩的，而一个人的收入水平又和其工作是直接挂钩的，一般来说，公司的层次越高，相对收入就越高，相对而言还款能力就越强，存在违约的可能性就越小。
5. industry指工作的领域，有传统的工业，商业，互联网金融等，随着网络化的趋势逐步加强，一些与计算机相关的领域收入相对变得较高，比如互联网相关的职业的排名比较靠前。在收入比较高的工作领域，还款能力相对较强，违约概率相对较低。
6. work\_year是贷款人的工作年限，一方面，工作年限越长，相对而言工资越高，另一方面，工作年限越长，手中的积蓄会相对较多，还款能力较强，违约概率较小。
7. home\_exist是贷款人是否有房，相对于其他的数据来说，这个字段，对于违约预测的影响比较小，但从大体趋势上来看，有房的贷款人的还款能力还是相较于没房的贷款人的还款能力还是高一些的。
8. censor\_status是审核情况，当个人提出贷款申请时，银行会对个人的贷款申请进行审核，贷款审核是银行贷款风险把控的关键环节，需要核实借款人的常住地和身份证明；是否有稳定的收入来源，能否按时偿还贷款本息；有没有银行认可的担保物，名下有没有固定资产；是否有个人不良信用记录。根据审核情况来进行贷款的发放，对于审核所不通过的贷款人的申请给予驳回，降低贷款风险，减少贷款违约率。
9. issue\_month是指贷款的发放月份，从贷款发放开始，就开始正式计入贷款时间，并按照利率进行计算，来规定所要归还的本息，发放月份主要是影响本息的计算，从而间接的影响还款能力。
10. use是指贷款的用途，贷款用途是指贷款的具体去向。它反映贷款用于解决生产经营活动哪一方面的资金需要，反映贷款同哪些生产要素相结合，反映贷款用在再生产过程哪个环节上。银行贷款按贷款用途或对象不同，可分为工商业贷款、农业贷款、消费者贷款、有价证券经纪人贷款等，在进行银行贷款时，必须要明确贷款用途，根据贷款用途判断违约概率 。
11. post\_code贷款人申请时的邮政编码，邮政编码是实现邮政现代化的必需工具，最终目的是使贷款过程中的信件在传递过程中提高速度和准确性，因此在贷款时务必写明邮政编码。在以前邮政编码可以确当一个人所在的大体位置，但是随着数据库的发展，邮政编码的作用逐渐减小。继续保留邮编，是因为这相当于一种社会性的规范，但其对于违约预测影响不大。
12. region是地区编码，发挥的作用和邮政编码大体类似，都是为了消息信件等准确传达，猜测其对于违约预测的影响，应该体现在地区的经济发展水平对于个人收入和思想教育的影响。
13. debt\_loan\_ratio是债务收入比，又可称为DIT，负债收入比率=年负债/年税后收入，该指标反映支出能力的强弱，临界值为40%，达到这一数值则说明短期偿债能力可以得到保证。该比例能反映客户在一定时期财务状况的良好程度。债务收入比是银行挑选客户的标准之一。银行使用债务收入比以及信用记录来评估借款人是否具备偿还贷款的能力。每个贷方（银行或网贷）都会设置自己的DTI要求。个人信用贷款提供者通常比抵押贷款（房贷、车贷）提供者允许更高的DTI。银行和其他网贷提供者希望在向潜在借款人发放贷款之前都希望看到较低的DTI。较高的DTI可能表示一个人的债务过多，甚至每月赚取的收入不足于还债，这是一个非常危险的个人财务信号。DIT是进行个贷违约预测的重要字段，从多方面反应了贷款人的还款能力。DIT越低，还款能力越高，违约概率就越低。
14. del\_in\_18month是指借贷人过去18个月违约的事件数，如果过去十八个月违约次数过多，则证明还款人这次贷款也有极大可能违约，所以应当谨慎审核其贷款申请。这个字段对于最终违约预测的估计的影响也是比较大的。
15. scoring\_low和scoring\_high是贷款人在贷款评分中所属的上限和下限，银行会对个人进行综合评分，综合评分简单来说就是银行根据借款人自身的情况评估出来的一些项目的总分。如果这个总分达到了银行的贷款标准，那么就可以按照流程申请贷款。“综合评分”只是对贷款申请人申请资格的一个综合的量化评判，影响它的因素有很多，你的婚姻状况、工作、职位、职业、收入水平、负债、房屋性质、贷款方式、学历、个人信用记录等等都决定了你评分的多少，只有这些分数相加起来的总和达到了评分标准，你才有可能拿到贷款。但是光看总分还不够，有的银行会对于单个的评分项设置上下限，当由单个评分低于其下限时，则并不能申请贷款。显然，贷款评分中的上下限越高，则意味着个人的贷款评分越高，违约的概率就较低。
16. known\_outstanding\_loan是指借款人档案中未结信用额度的数量，这种未结的信用额度分为两种，一种是因为违约而导致未结的信用额度，另一种则是因为额度不到期而未结的信用额度。对于第一种，未结信用额度越多，代表违约越多，这次贷款的违约概率就越大，对于第二种，未结信用额度越高，则代表需要偿还的资金越多，则违约的概率也就越高。所以总结来说，未结信用额度数量越多，违约的概率也就越高。
17. known\_dero是指损害公共记录的数量，**公共记录**是征信报告中最重要的组成部分，包括欠税记录、民事判决记录、强制执行记录、行政处罚记录、社会保险参保缴费记录、住房公积金缴费记录、获得许可记录、获得认证记录、获得资质记录、获得奖励记录、出入境检验检疫绿色通道信息、进出口商品免检信息、进出口免检分类监管信息、上市公司或有事项、拥有专利情况、公共事业缴费记录。损害公共记录的数量越多，意味着贷款人的素质越低，同时也以为着违约的概率越高。
18. pub\_dero\_bankrup指公开清楚记录的数目，是指对于以前已经违约过的记录，并且长时间不在损害，可以予以清除，这个指标有两个方面的反应，一是之前损害公共记录的数量，二是个人改正的态度。对于这个指标，应该采用辩证的角度来进行分析。
19. recircle\_bal指信贷周转余额合计，周转信贷就是银行承诺借款一定额度给企业，如果企业没有贷够足够的额度，则对剩余部分付一定的承诺费。 如果企业信誉恶化，即使签订了信贷协定 ，企业也可能得不到借款。 所以不是任何时候都必须满足企业的借款要求。周转信贷协定是银行从法律上承诺向企业提供不超过某一最高限额的贷款协定。在协定有效期内，只要企业累计借款总额未超过最高限额，银行必须满足企业任何时候提出的借款要求。企业享有周转协定，通常要对贷款限额的未使用部分付给银行一笔承诺费。
20. recircle\_util是指循环额度利用率，**循环额度**一般是指银行为客户提供的可以在约定期限内分次提款、**循环**使用的贷款**额度**。循环额度的利用率越高，代表个人资金周转越快，从而还款能力就越强，违约概率就越小。
21. initial\_list\_status是指贷款的初始列表状态，有0和1之分，代表在贷款初始状态下有无贷款，1代表有，0代表无。
22. app\_type是指是否个人申请，银行个人贷款分主要分为个人信用贷款和消费类抵押贷款。个人申请银行贷款最基本的条件有如下几点:在贷款银行所在地有固定住所、有常住户口或有效居住证明、年龄在65周岁(含)以下、具有完全民事行为能力的中国公民;有正当职业和稳定的收入，具有按期偿还贷款本息的能力;具有良好的信用记录和还款意愿，无不良信用记录;明确的贷款用途，且贷款用途符合相关规定。因为本次赛题为个贷违约预测，所以只考虑个人贷款。
23. earlies\_credit\_mon是指借款人最早报告的信用额度开立的月份，一般情况下来说，最早报告的信用额度开立的月份越早，代表越清楚信用规则，违约的概率相对较低。
24. title是指借款人提供的贷款名称，这里来说，只作为一个标识处理，对于个人还款能力的预测和违约概率影响并不大，因此可以忽略不计。
25. f系列匿名特征是匿名特征f0-f4，为一些贷款人行为计数特征的处理，这些数据在预处理之前并不知道其实际意义和价值，需要经过数据的预处理判断数据是否可用，和对于违约预测的影响。
26. early\_return是指借款人提前还款次数，提前还贷指借款人将自己的贷款部分向银行提出提前还款的申请，并保证以前月份没有逾期且归还当月贷款；按照银行规定日期，将贷款部分全部一次还清或部分还清。提前还款的次数越多，代表借款人在规定日期之前或者规定日期还款的可能性越大，因此违约的概率就越小。
27. early\_return\_amount是指贷款人提前还款累积金额。借款人提前还款分为提前偿还部分贷款和提前全额偿还贷款，对于第一种，有两种还款方式：一种是在每月还款额固定的情况下缩短还款期。另一种是减少每月还款额，但还款期保持不变。第一种方法的优点是可以减少利息，而第二种方法的优点是可以减少每月还款并减轻每月还款的压力。对于第二种，全额提前还清意味着借款人立即还清所有欠款。提前还款累积的金额越多，表明借款人还款的资金流动较好，债务偿还能力较好，违约的概率较低。
28. early\_return\_amount\_3mon是指贷款人三个月内提前还款累积金额。这个给字段和上一个字段的区别就是三个月的界限，上一个字段，没有对时间进行约束，相对这个字段来说，较为粗略，而这个字段是三个月内的期限，能够更接近这次贷款时间，同时能够更好，更准确的预测还款能力和违约的概率。

# 数据可视化

因为官方给了我们三组数据集（train\_internet，train\_public，test\_public），在前面我们分析过的对所给数据集之间的关联，这三个数据集在特征、数据量上都有所不同，我们通过观察三个表的分布情况，来分析哪些特征是三个数据集共有的，可做相同的操作；哪些特征的数据之间分布情况差距过大，做一些删减工作；从而使清洗出的数据更加有效，提高数据质量，因为数据质量的好坏决定了数据分析与数据预测能否获得理想的结果。

## data\_all数据集的大体分布

因为从之前的特征分析中可知，train\_public数据集和test\_public数据集的特征字段及数据概况差不多，所以我们首先将train\_public和test\_public拼接而成形成具有15000行数据的data\_all数据集，对它进行了简单的分布情况展示，以易于对后面和另外两个数据集的比较做参考，因为data\_all数据集的分布情况基本可以代表我们所需要预测数据的test\_public的分布情况。

data\_all数据集是将train\_public数据集和test\_public数据集纵向合并相同特征过后得到的总public数据集，我们通过观察两个public数据集的数据在每个特征中的分布情况，粗略地了解数据的集中分布情况、最大值最小值等大体信息，为数据的大致走向有个粗略的印象，为后面比较train\_internet和test\_public的分布情况以及比较train\_public和test\_public的分布情况作参考。

在一些机器学习相关的竞赛中，给定的训练集和测试集中的部分特征本身很有可能就存在分布不一致的问题。实际应用中，随着业务的发展，训练样本分布也会发生变化，最终导致模型泛化能力不足。所以我们必要的进行的一步就是检查训练和测试集特征分布一致性。

最常用的检查训练集和测试集特征分布一致性的方法之一就是使用KDE(核密度估计)分布图。核密度估计(kernel density estimation)是在概率论中用来估计未知的密度函数，属于非参数检验方法之一，通过核密度估计图可以比较直观的看出数据样本本身的分布特征。所以我们这里选用KDE(核密度估计)分布图来观察数据样本本身的分布特征，即使用seaborn的kdeplot方法对合并后的data\_all数据集进行核密度估计并可视化。

代码展示：

plt.figure(figsize=(8,40\*4))

for i,col in enumerate(data\_all.columns[2:-2]):

cond = data\_all['origin']=='train'

train\_col=data\_all[col][cond]#训练集

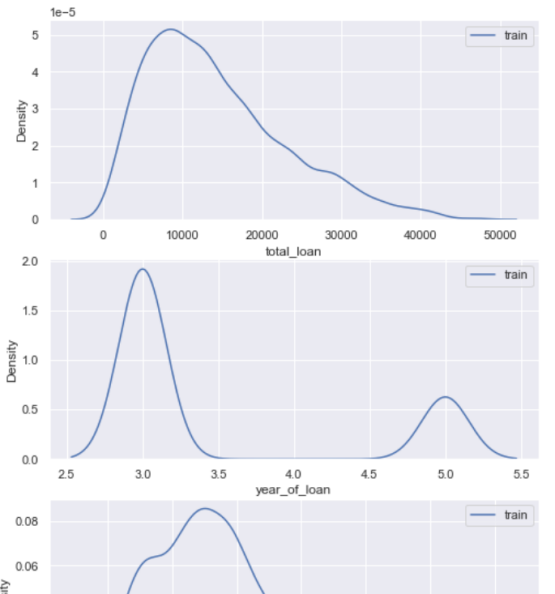
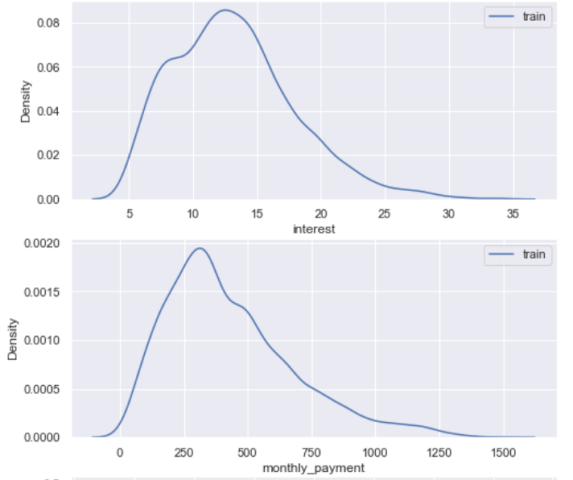
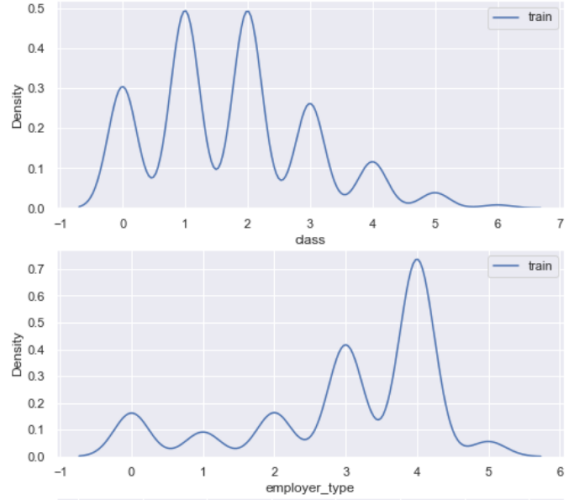
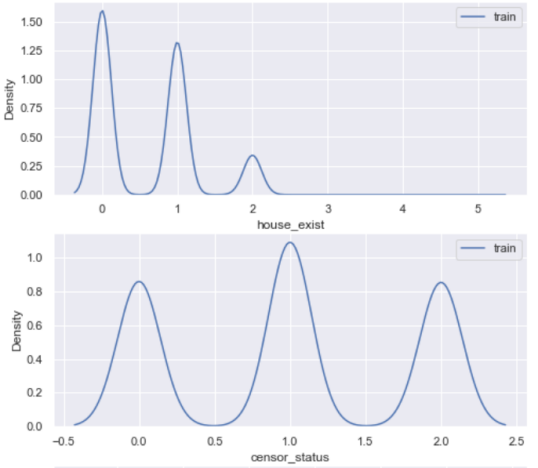
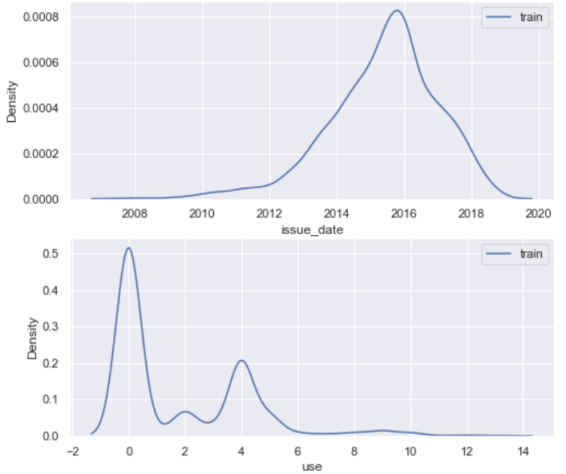
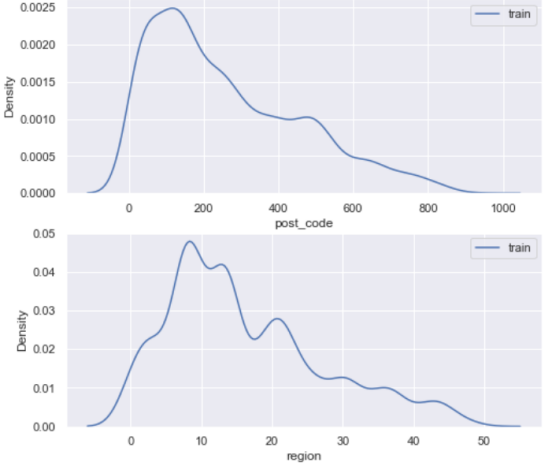
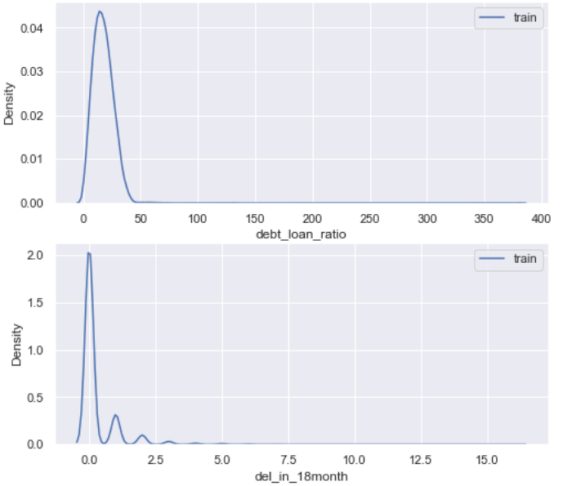
axes=plt.subplot(40,1,i+1)

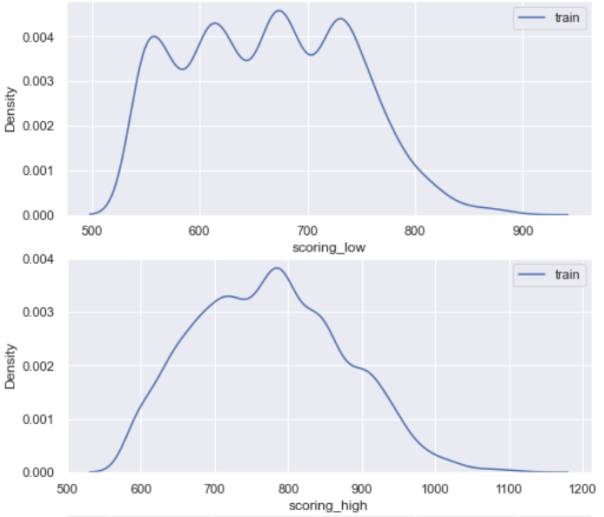
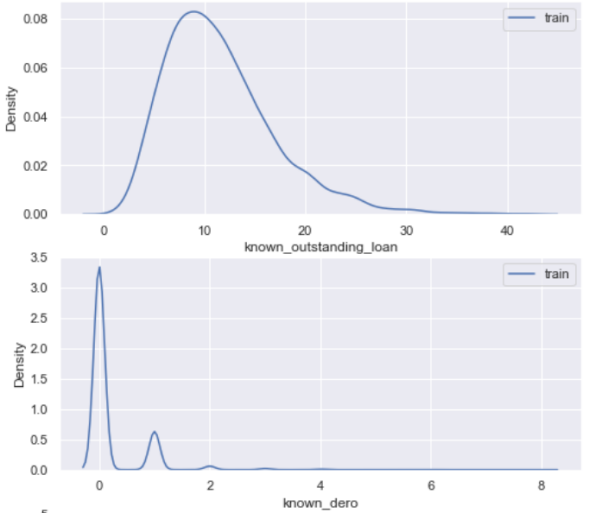
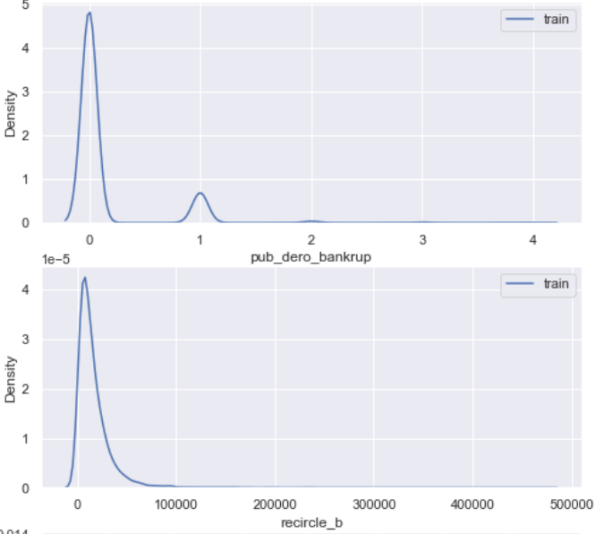
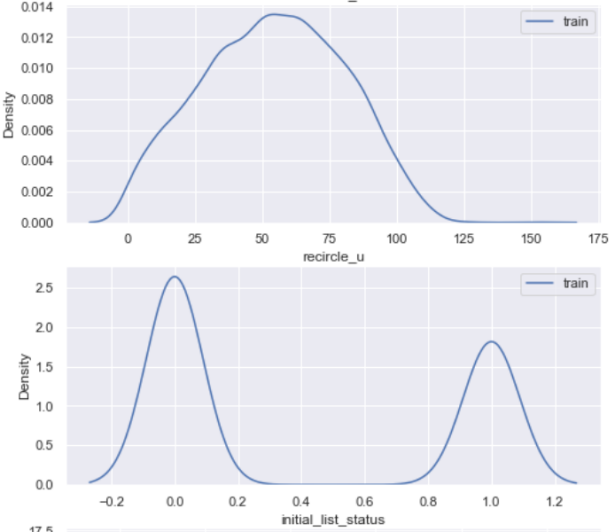
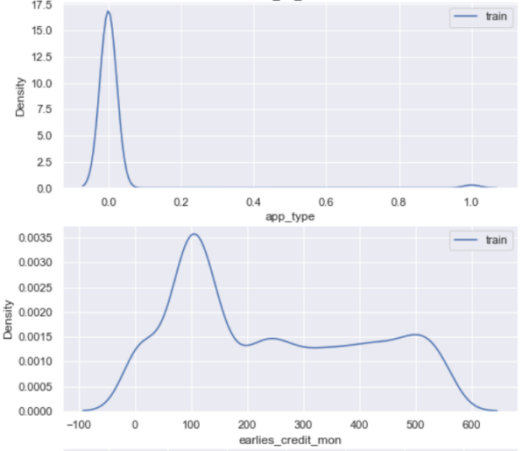
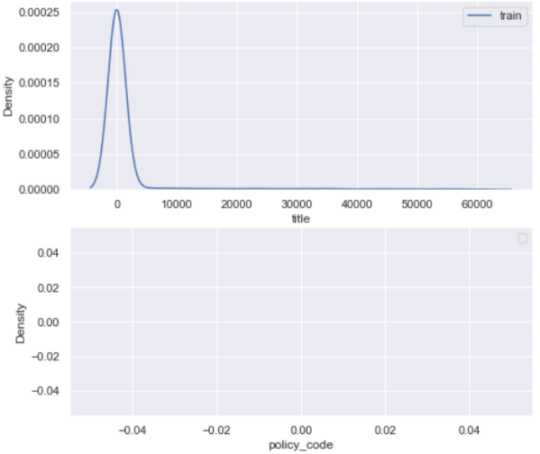
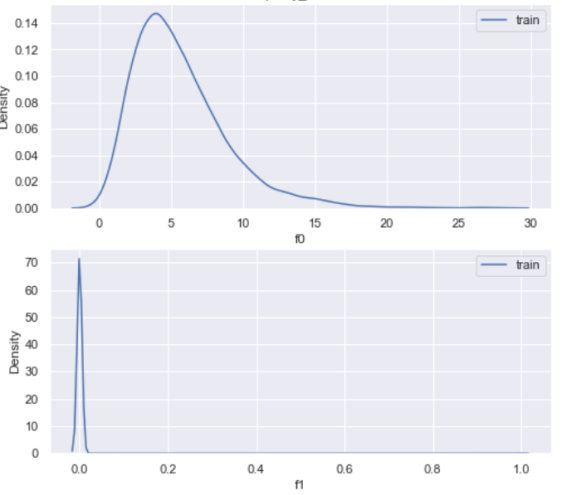
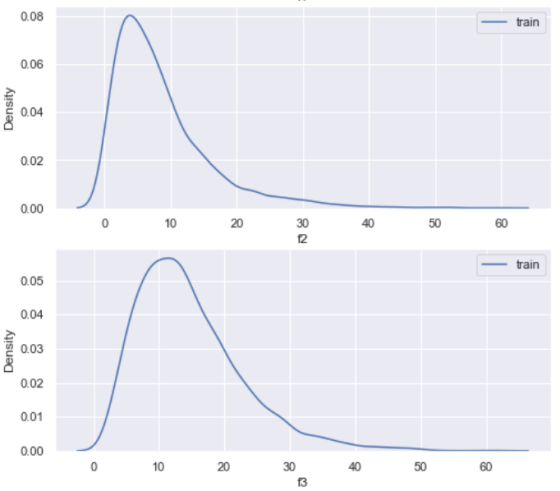
ax=sns.kdeplot(test\_col,shade=False,ax=axes)

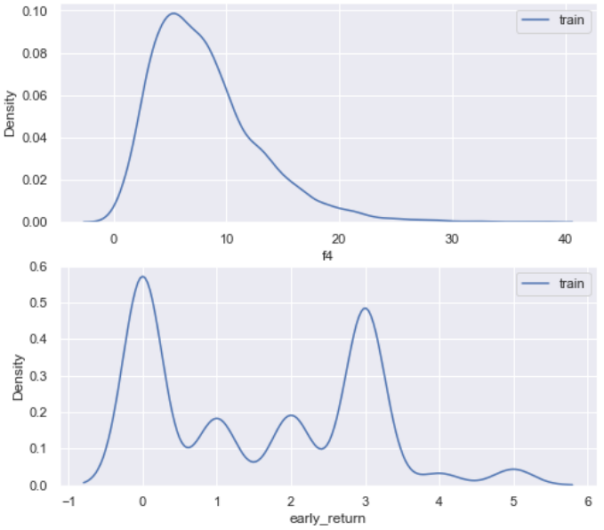
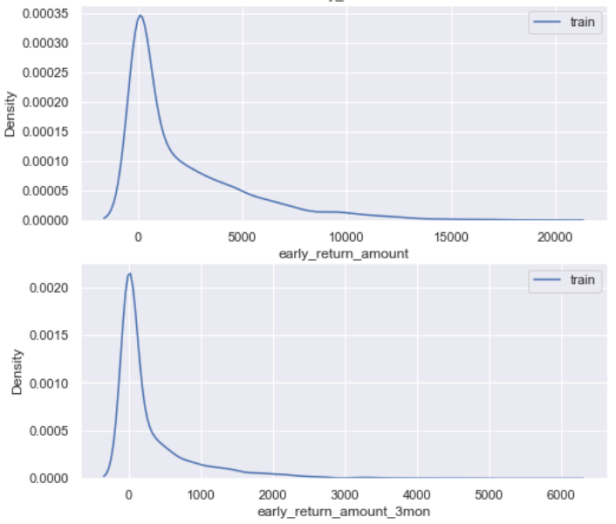
sns.kdeplot(train\_col,shade=False,ax=ax)

plt.legend(['train','test'])

运行出的data\_all的分布情况如图所示：

由上图可知，total\_loan的数据集中在数值10000左右，interest，monthly\_payment，issue\_date，post\_code，debt\_loan\_ratio，known\_outstanding\_loan，scoring\_high，recircle\_u，f0，f2，f3等特征的分布情况都比较相似，都是集中在某一数值附近，呈正态分布形态，说明这些特征值都较稳定。

## train\_public和test\_public分布情况的比较

在对train\_public和test\_public特征分布情况的对比中，可以发现不仅只是官方给的原数据集中他们的特征字段一样，而且在每一个特征的数据分布情况也都基本一致，也证明了test\_public的数据拟合情况应该与train\_public的数据相似，如果将train\_public放进算法中训练，一定能更好更加准确的预测test\_public的贷款违约情况。

代码展示：

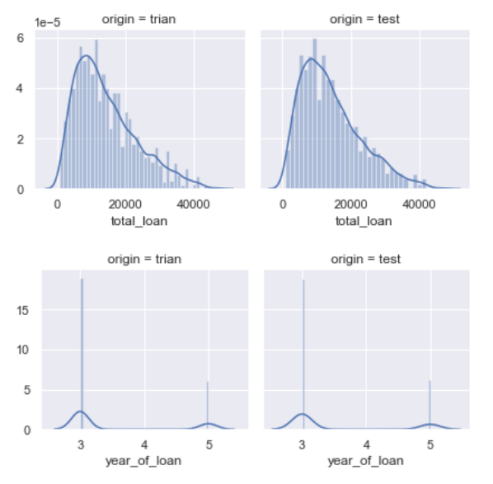
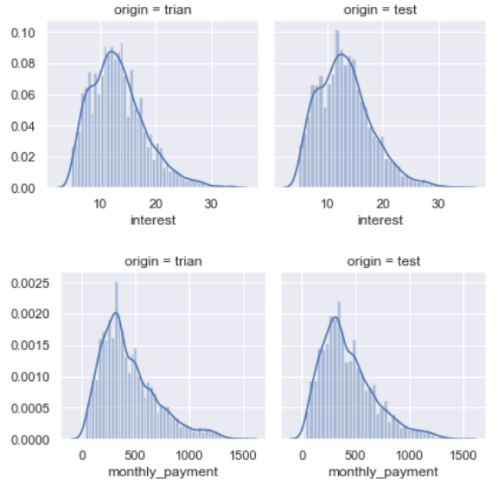
plt.figure(figsize=(9,6))

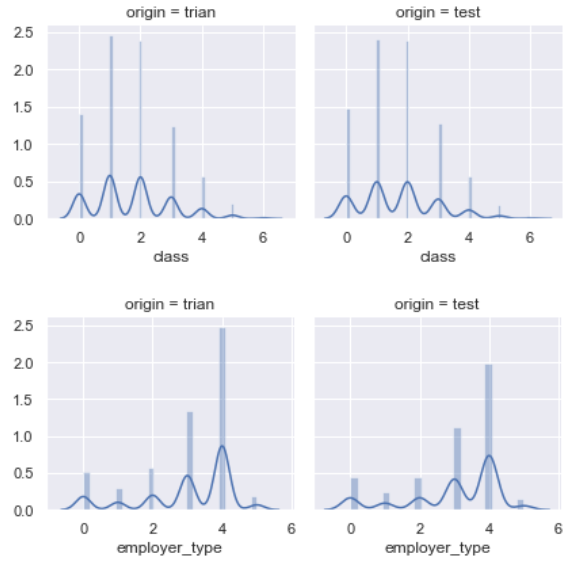
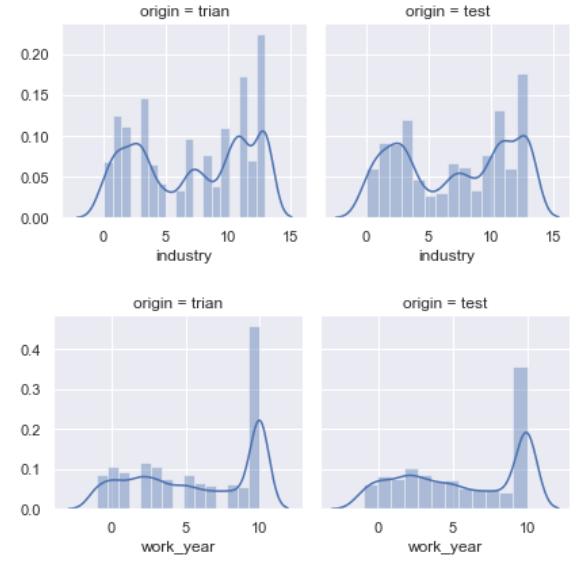
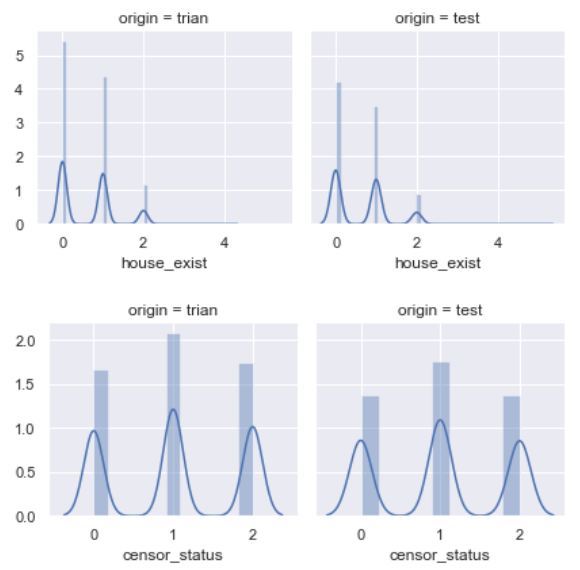
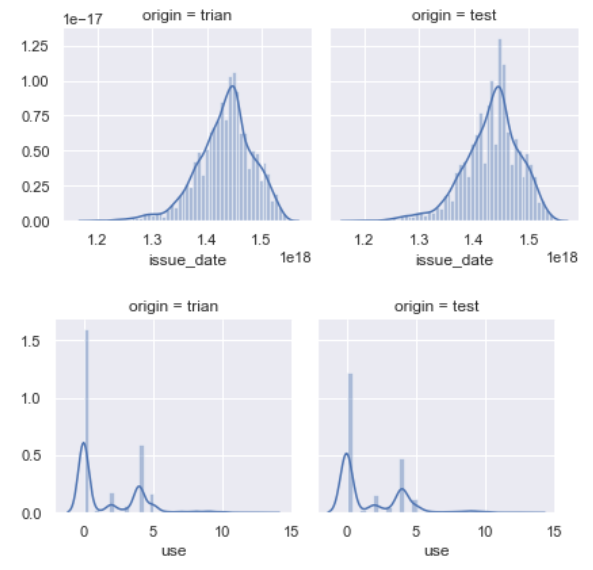
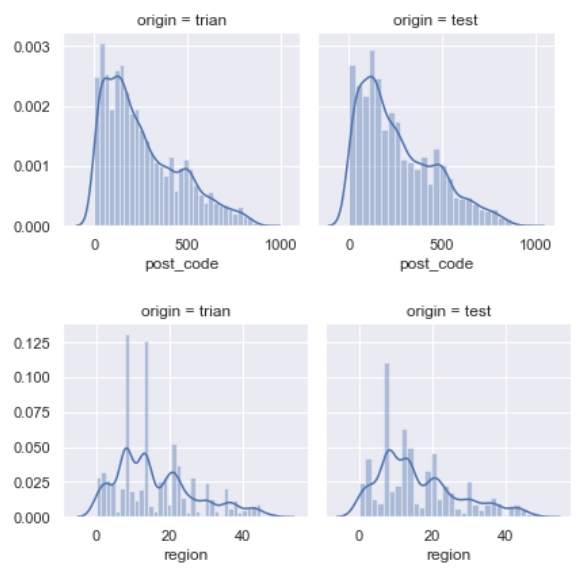
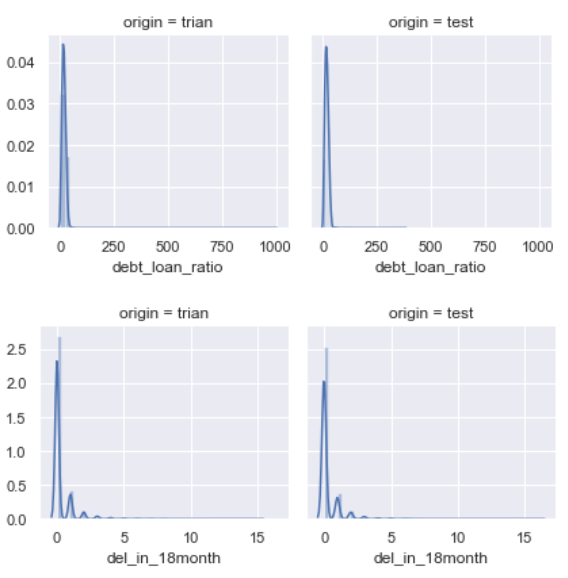
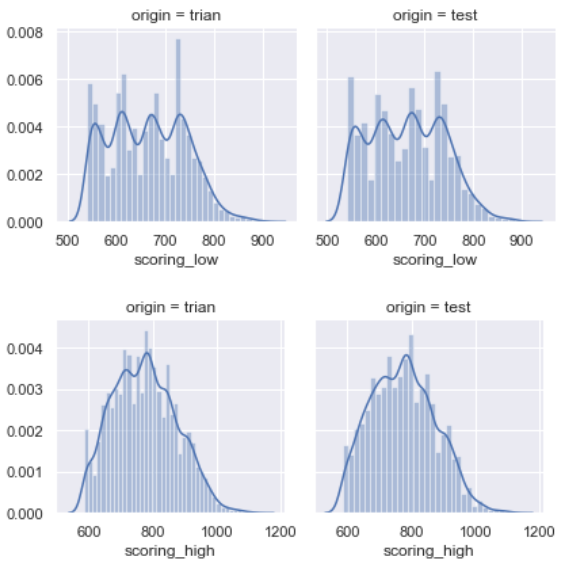
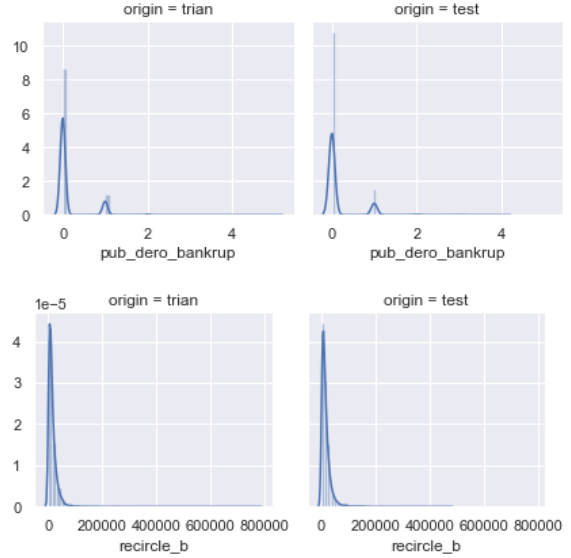
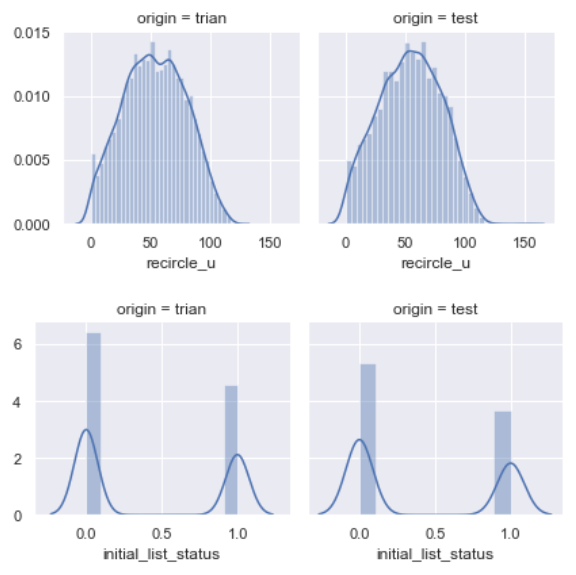
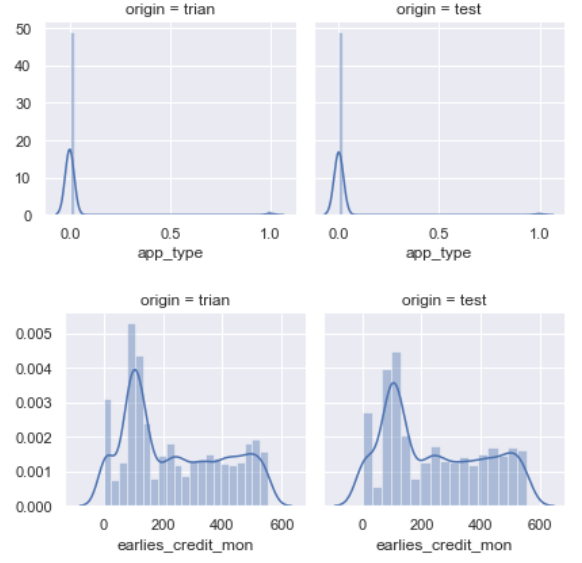
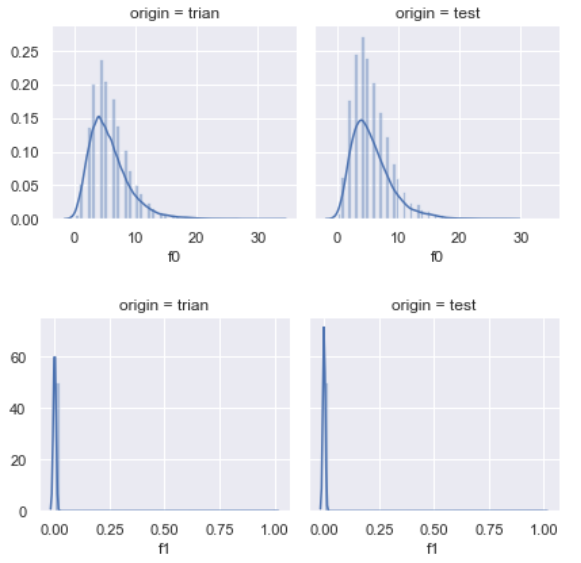
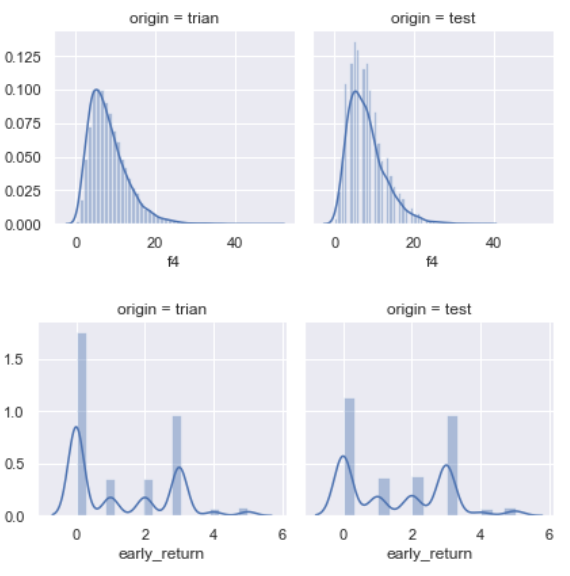
for col in data\_all.columns[2:-2]:

g=sns.FacetGrid(data\_all,col='origin')

g.map(sns.distplot,col)

以下是对两个数据集中的每个特征分布情况的对比图，可以很清晰很直观的观察到两个数据集的分布情况基本一致：

## train\_internet和test\_public分布情况的比较

众所周知，大部分机器学习算法都有一个假设：训练数据样本和未知的测试样本来自同一分布。如果测试数据的分布跟训练数据不一致，那么就会影响模型的效果。在一些机器学习相关的竞赛中，给定的训练集和测试集中的部分特征本身很有可能就存在分布不一致的问题。实际应用中，随着业务的发展，训练样本分布也会发生变化，最终导致模型泛化能力不足。

在检查train\_internet和test\_public数据集的分布情况过程中，我们确实发现了很大的问题，先展示分析train\_internet和test\_public数据集的代码实现过程以及运行结果。

代码展示：

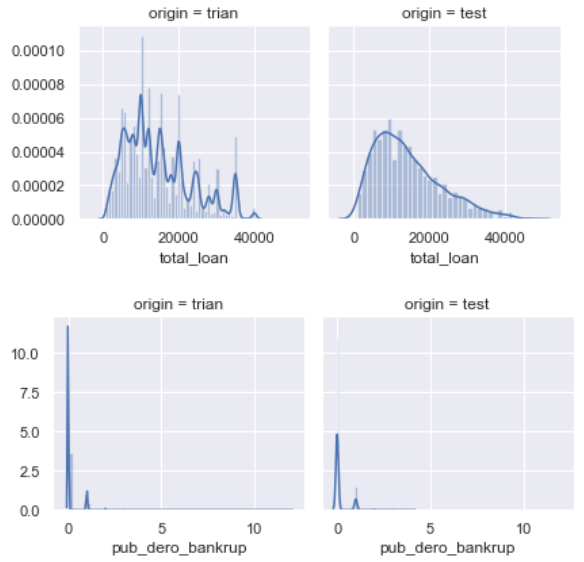
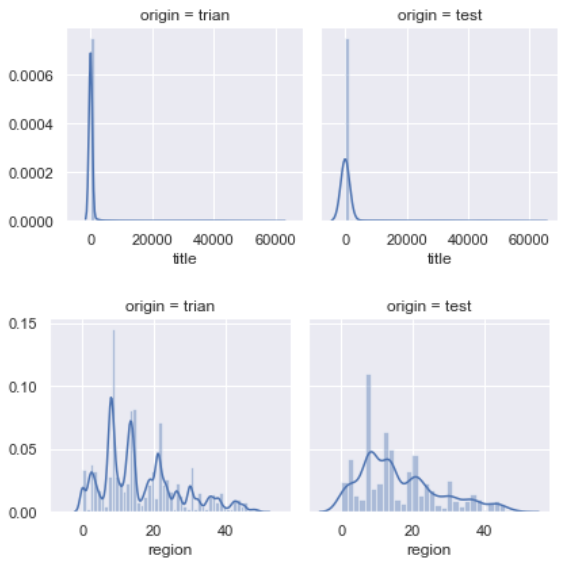
plt.figure(figsize=(9,6))

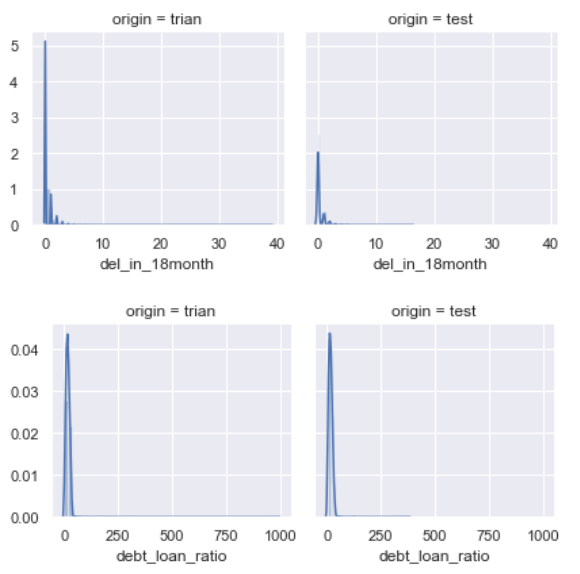
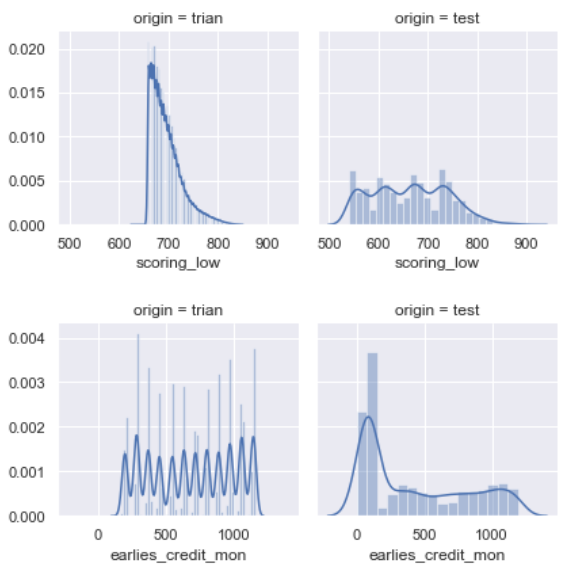
for col in data\_all2.columns[2:-5]:

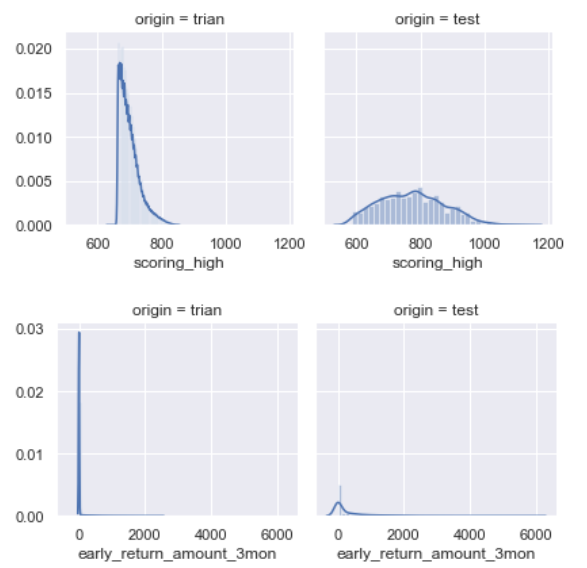
g=sns.FacetGrid(data\_all2,col='origin')

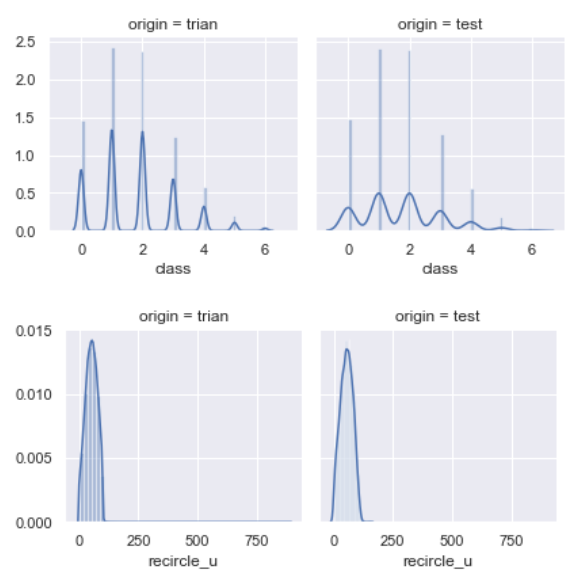
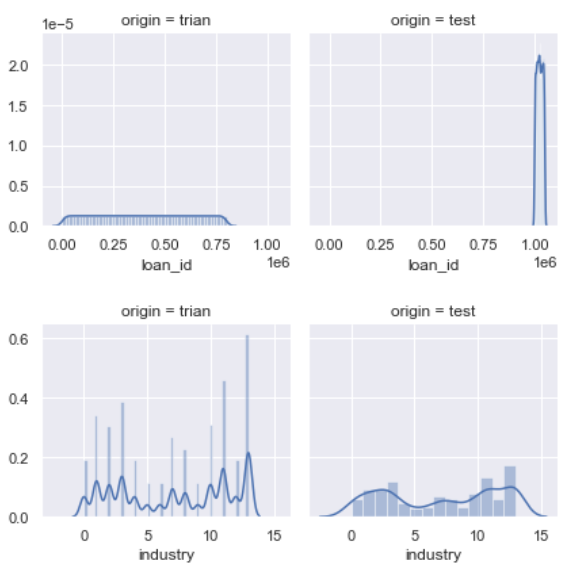
g.map(sns.distplot,col)

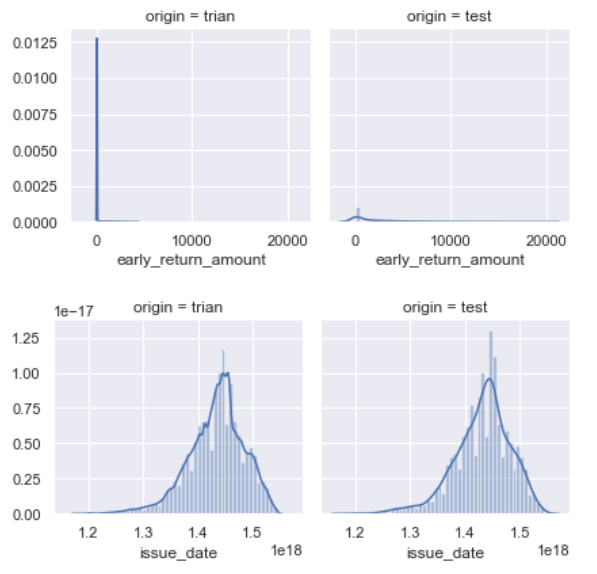
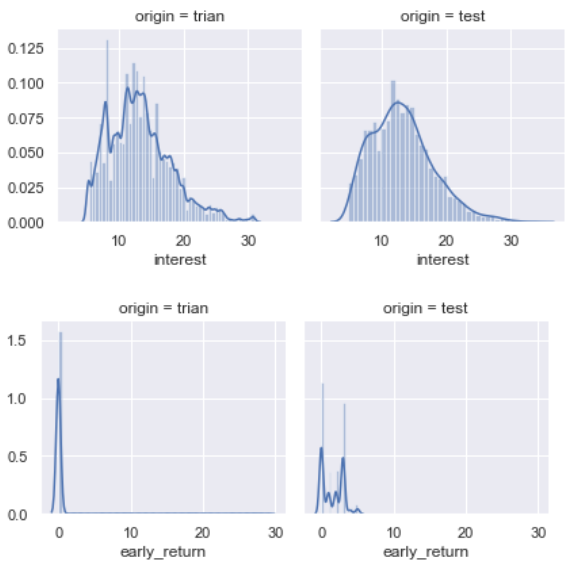
运行结果：

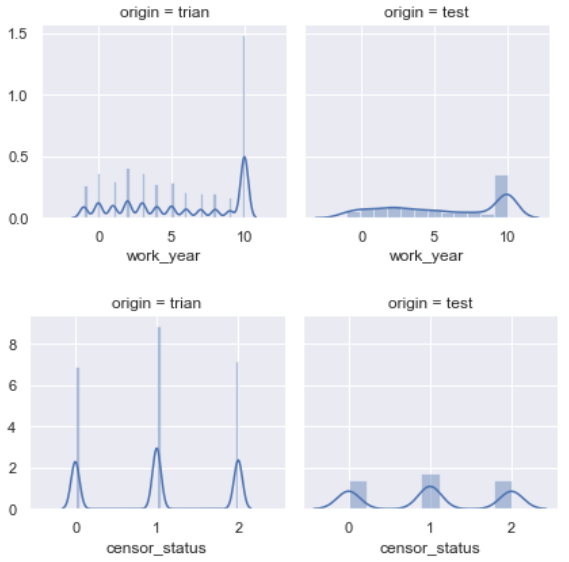
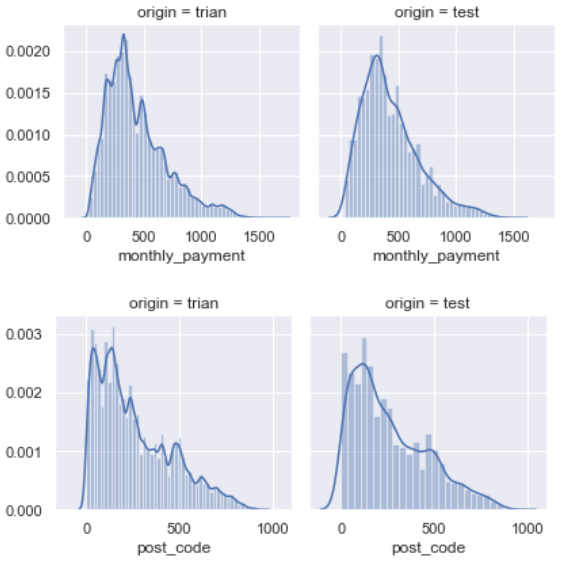


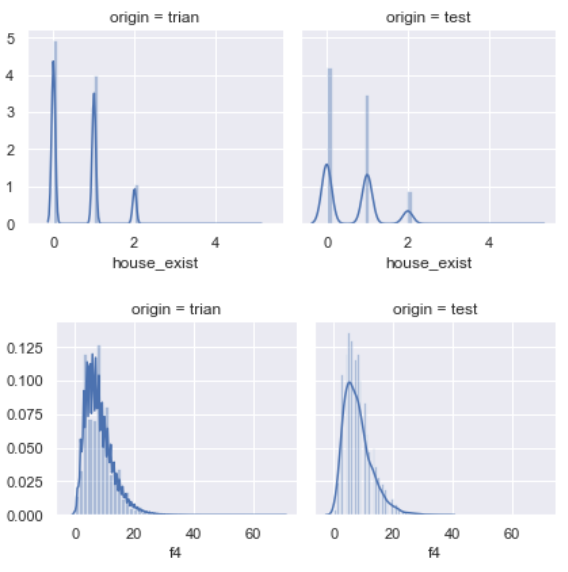
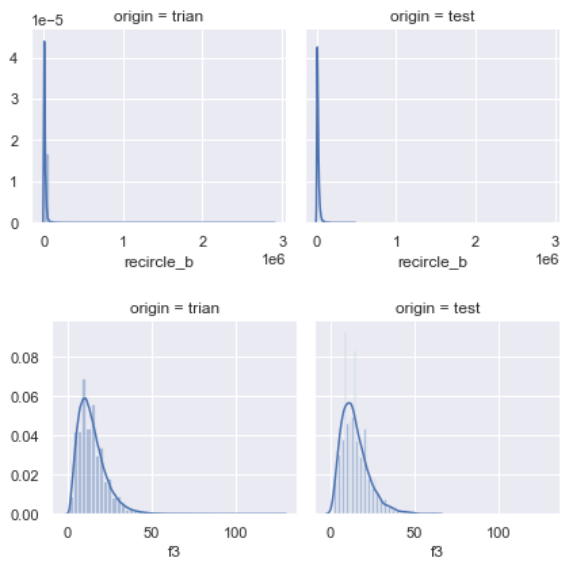


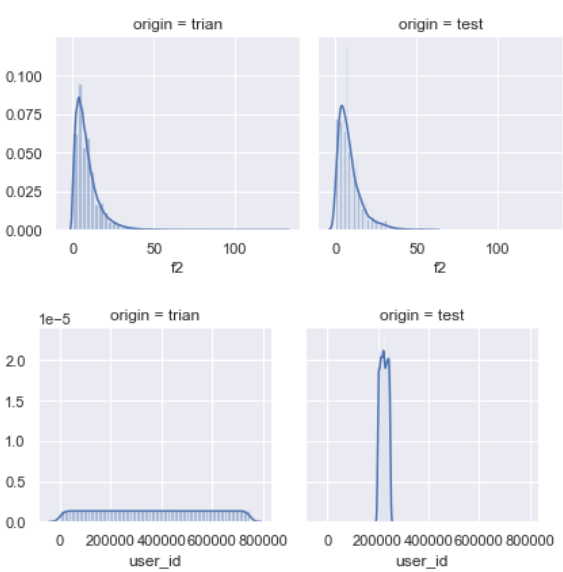
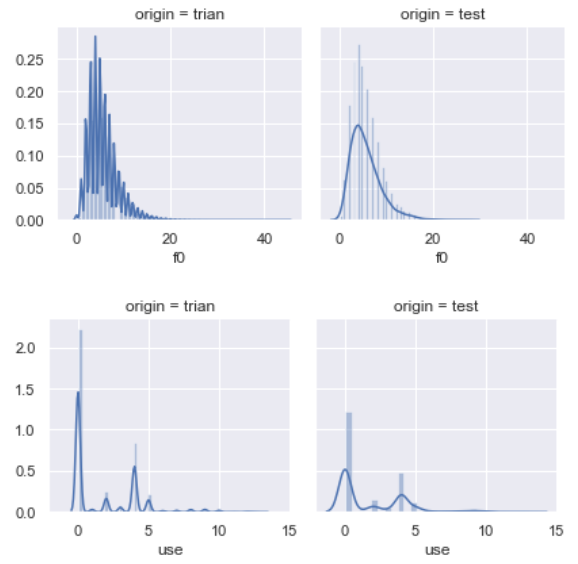


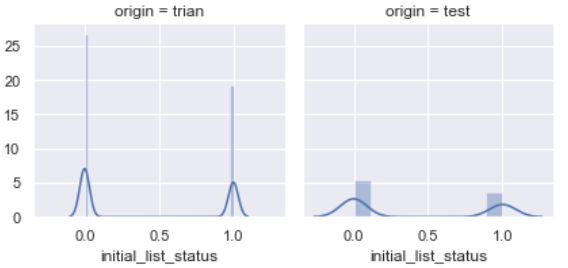












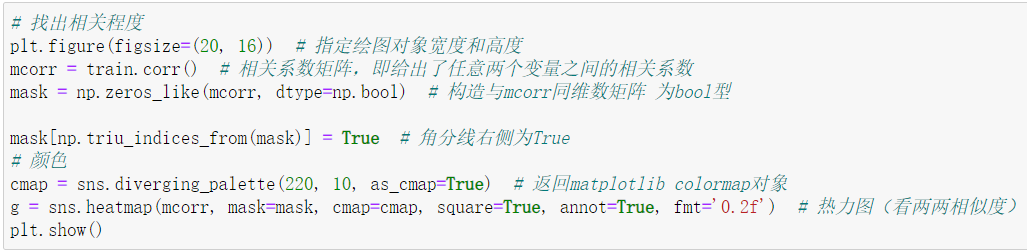
从上图可知，除了少数几个特征比如del\_in\_18month，debt\_loan\_ratio，issue\_date，monthly\_payment，recircle\_b，f2等在train\_internet和test\_public上的分布情况有一部分相同，但是其他大多数特征在两个数据集上的分布情况几乎无相似之处，说明train\_internet和test\_public之间存在着巨大的差异，在这些特征下，训练集的数据和测试集的数据分布不一致，会影响模型的泛化能力，这将会造成通过train\_internet数据集训练出来的模型可能并不适于test\_public，所以当发现这一问题时，我们小组也都陷入了瓶颈，因为train\_internet和test\_public甚至train\_public都差距太大，显得train\_internet数据很脏以至于让我们无法修改。

但最终，**我们小组得出的解决办法是：由于train\_internet分布情况与test\_public差距太大，但train\_public和test\_public分布相似，所以以train\_public为主，在train\_internet里挑选数据。**

## 特征之间的相关程度

根据热力图中不同方块颜色对应的相关系数的大小，可以判断出变量之间相关性的大小。从以上热力图可以看出，total\_loan和monthly\_payment，scoring\_high和scoring\_low,f3和f4的相关性较大。

代码展示：



热力图展示：

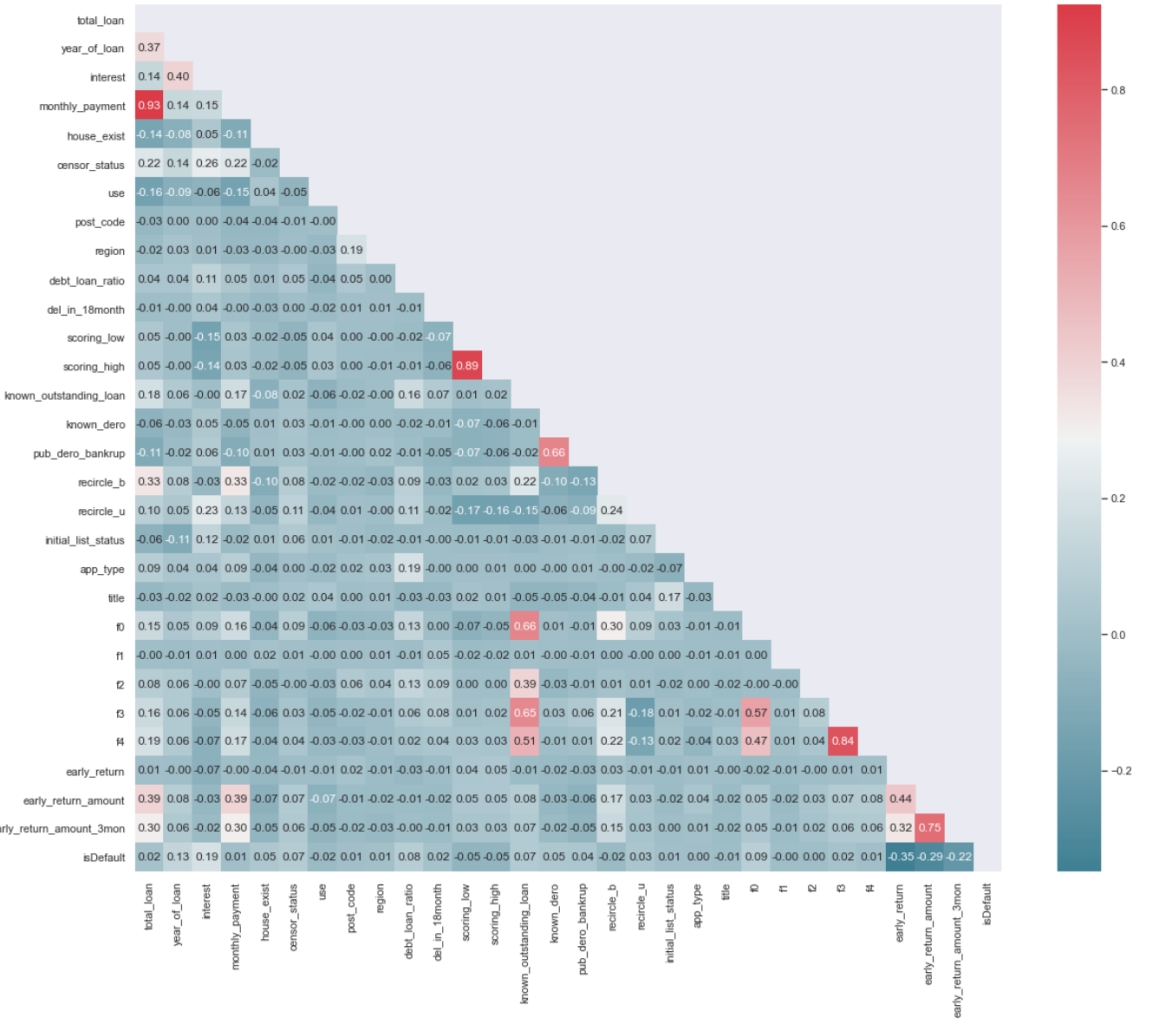


图7 热力图

## 特征重要性

特征重要性是一种为预测模型的输入特征评分的方法，该方法揭示了进行预测时每个特征的相对重要性。可以为涉及预测数值的问题（称为回归）和涉及预测类别标签的问题（称为分类）计算特征重要性得分。这些得分非常有用，可用于预测建模问题中的多种情况，可以帮助我们更好地了解数据集，相对得分突出显示了哪些特征可能与目标最相关，反之则突出哪些特征最不相关。

以下是在lightgbm算法下得出的特征重要性显示，前三个特征：issue\_date，recircle\_b，debt\_loan\_ratio与预测是否会违约最相关，而相反排序靠后的几个特征最不相关。

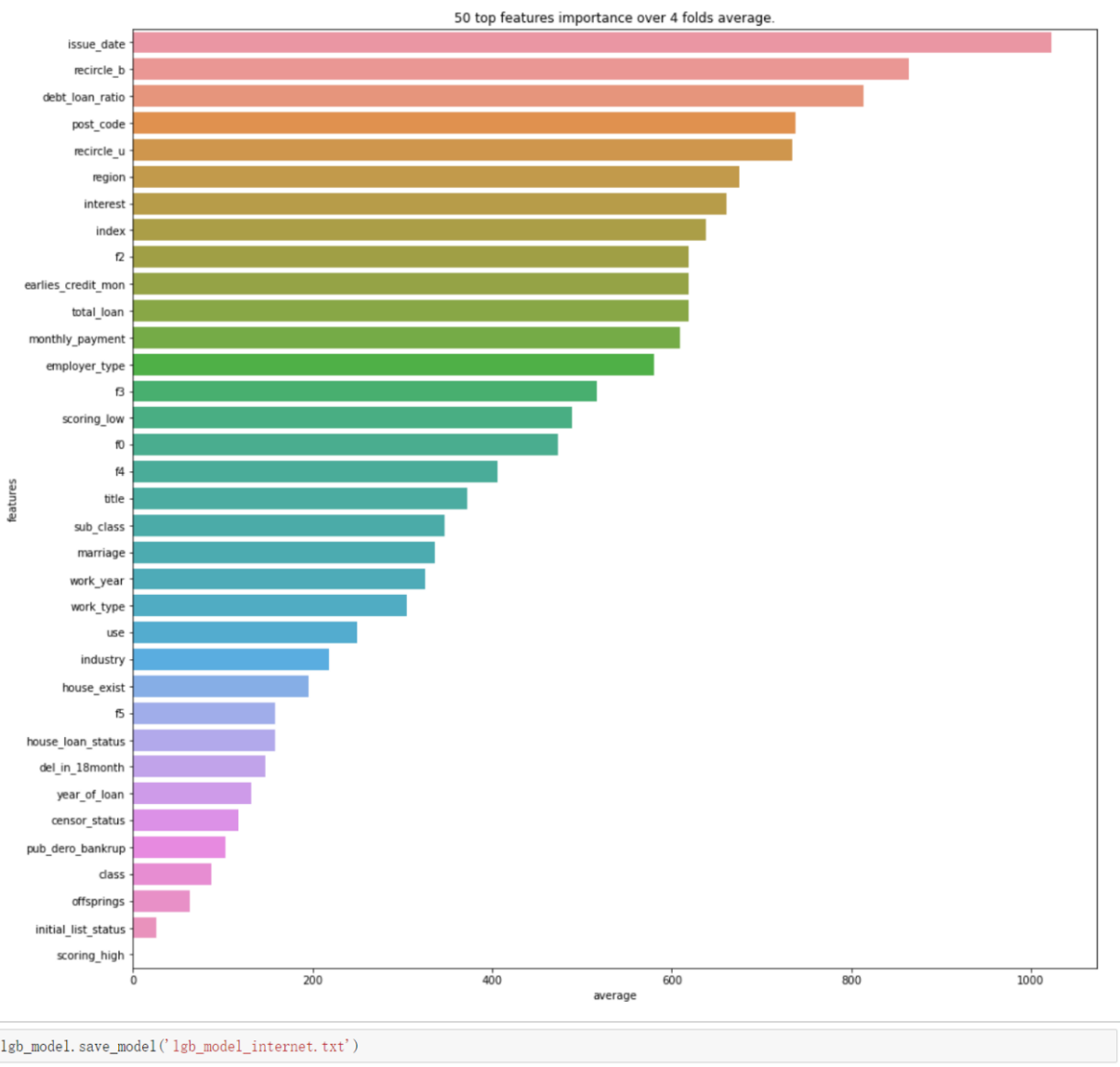


图8 特征重要性

# 数据探索

## 删除字段

1. 先删除Internet数据集中多出来的字段：sub\_work\_type，house\_loan\_status，marriage，offsprings，f5，sub\_class，work\_type。

然后删除train\_public数据集中有的字段但是Internet中不含有的字段：known\_outstanding\_loan，known\_dero，app\_type。

test数据集与train\_public数据集做相同的删除操作。代码如下：

internet.drop(['sub\_work\_type','house\_loan\_status','marriage','offsprings','f5','sub\_class',

'work\_type'],axis=1,inplace=True)

public.drop(['known\_outstanding\_loan','known\_dero','app\_type'],axis=1,inplace=True)

1. 删除无用字段：

三个数据集都删除"loan\_id", "user\_id", "title",”earlies\_credit\_mon”这四个无用字段。

数据集.drop(["loan\_id", "user\_id", "title",”earlies\_credit\_mon”],axis=1, inplace=True)

## 缺失值处理

1. work\_year字段缺失值处理

from sklearn.impute import SimpleImputer //先导入处理缺失值的类

对internet数据集处理：work\_year填充众数

train\_internet.loc[:,"work\_year"]=SimpleImputer(strategy="most\_frequent").fit\_transform(train\_internet.loc[:,"work\_year"].values.reshape(-1,1))

对train\_public数据集处理：work\_year填充众数

train\_internet.loc[:,"work\_year"]=SimpleImputer(strategy="most\_frequent").fit\_transform(train\_internet.loc[:,"work\_year"].values.reshape(-1,1))

对test\_public数据集处理：work\_year填充众数

train\_internet.loc[:,"work\_year"]=SimpleImputer(strategy="most\_frequent").fit\_transform(train\_internet.loc[:,"work\_year"].values.reshape(-1,1))

1. 对f0—f4缺失值填充众数

test\_public.loc[:,"f0"]=SimpleImputer(strategy="most\_frequent").fit\_transform(test\_public.loc[:,"f0"].values.reshape(-1,1))

test\_public.loc[:,"f1"]=SimpleImputer(strategy="most\_frequent").fit\_transform(test\_public.loc[:,"f1"].values.reshape(-1,1))

test\_public.loc[:,"f2"]=SimpleImputer(strategy="most\_frequent").fit\_transform(test\_public.loc[:,"f2"].values.reshape(-1,1))

test\_public.loc[:,"f3"]=SimpleImputer(strategy="most\_frequent").fit\_transform(test\_public.loc[:,"f3"].values.reshape(-1,1))

test\_public.loc[:,"f4"]=SimpleImputer(strategy="most\_frequent").fit\_transform(test\_public.loc[:,"f4"].values.reshape(-1,1))

其他代码相同。

1. internet数据集共有70万条数据，在填充完空缺值比较大的字段后，还有debt\_loan\_ratio，pub\_dero\_bankrup，recircle\_u这三个字段还有不超过

500的空缺值，对数据影响很小，所以删除空缺值。

Train\_public数据集中pub\_dero\_bankrup字段好友7个空缺值，所以直接删除。

Test\_public数据集中已没有空缺值。

## 定类和定序字段处理

1.train\_internet.csv数据集处理

class，employer\_type，industry这三个字段，转换成1，2，3...数值型。

代码如下：

#导入包

from sklearn.preprocessing import LabelEncoder

#数据转换

cols=['employer\_type','industry', 'class']#'work\_year'

encoder=LabelEncoder()

for col in cols:

data\_all2[col]=encoder.fit\_transform(data\_all2[col])

#结束

2.train\_public.csv和test\_public.csv数据集处理

由于一开始就将三个数据集的字段进行统一化，所以字段名和字段类型都是相同的，所以对这两个数据集进行的定类和定序字段的处理与internet数据集相同。

class，employer\_type，industry这三个字段，转换成1，2，3...数值型。代码省略。

## 数据标准化

1.对total\_loan贷款总数和monthly\_payment每月还款金额经行标准化处理

#导入标准化使用的类

from sklearn.preprocessing import StandardScaler

#

total\_loan：train\_internet.iloc[:,0:1]=StandardScaler().fit\_transform(train\_internet.iloc[:,0:1])

monthly\_payment：

train\_internet.iloc[:,3:4]=StandardScaler().fit\_transform(train\_internet.iloc[:,3:4])

## 数据归一化

1. 对其字段做归一化处理：

#导入包

from sklearn.preprocessing import MinMaxScaler

#设置参数

scaler = MinMaxScaler(feature\_range=[0,1])

#调用

cope3 = pd.DataFrame(scaler.fit\_transform(train.loc[:,5:]),columns=train.columns)

对除了已经做过标准化的数据，都需要进行归一化操作。

**PS：以上操作都是同时对三个数据集进行的操作，代码只显示了，对train\_public\_internet.csv数据集进行的操作。经过探索后这些操作都不能使成绩提高，甚至会使成绩下降，所以不采用以上预处理方法。**

# 数据预处理

## 构建新特征

对三个数据集做相同的操作。

1.从work\_year字段中提取工作年长，构建性特征。代码如下：

#导入包re使用正则表达式

import re

#定义一个函数，其中将工作年份中的空值赋值为-1，其他的只提取字符串中的数字。

def workYearDIc(x):

if str(x)=='nan':

return -1

x = x.replace('< 1','0')

return int(re.search('(\d+)', x).group())

#调用函数，提取特征值：

train['work\_year'] = train['work\_year'].map(workYearDIc)

2.从issue\_data字段中提取贷款发放的年，月，天，构成新的特征。

#先使用datatime()方法将时间变成标准日期形式

train['issue\_date'] = pd.to\_datetime(train['issue\_date'])

#提取日期中的年

train['issue\_date\_year'] = train['issue\_date'].apply(lambda x: x.year)

train['issue\_date\_month'] = train['issue\_date'].apply(lambda x: x.month)

train['issue\_date\_dayofweek'] = train['issue\_date'].dt.apply(lambda x: x.dayofweek)

3.对earlies\_credit\_mon转换形式构建新特征

#定义一个函数

import re

def findDig(val):

fd = re.search('(\d+-)', val)

if fd is None:

return '1-'+val

return val + '-01'

#调用函数

test\_public['earlies\_credit\_mon']= pd.to\_datetime(test\_public['earlies\_credit\_mon'].map(findDig))

#设置一个阈值：timeMax = pd.to\_datetime('1-Dec-21')，当时间大于阈值是，就把阈值赋值给它。

train.loc[train\_data['earlies\_credit\_mon']>timeMax,'earlies\_credit\_mon']= train.loc[train\_data['earlies\_credit\_mon']>timeMax,'earlies\_credit\_mon']+ pd.offsets.DateOffset(years=-100)

4.对上面的earlies\_credit\_mon字段进行格式转换后，分别提取年和月，作为新的特征

train['earliesCreditMon'] = train\_data['earlies\_credit\_mon'].dt.month

train['earliesCreditYear'] = train\_data['earlies\_credit\_mon'].dt.year

1. 将class字段中定序数据改为数值型1，2，3..之后，将值大于4的都改为4，构建新特征

#自定义函数f()

def f(x):

if x>=4:

return 4

return x

#调用函数

train\_data['issue\_date'] = pd.to\_datetime(train\_data['issue\_date'])

test\_public['class'] = test\_public['class'].map(f)

1. 构建信贷周转余额率新特征recircle\_ratio

train\_data['recircle\_ratio'] = train\_data['recircle\_b']/train\_data['total\_loan']

test\_public['recircle\_ratio'] = test\_public['recircle\_b']/test\_public['total\_loan']

1. internet与public进行选取交集特征，通过模型来选择阈值isdeafault<0.05的数据，这个方法需要迭代3次

代码：InteId = IntePre.loc[IntePre.isDefault<0.05, 'loan\_id'].tolist()

**PS:对三个数据集都做了以上构建新特征操作。**

# 数据建模

## 模型选择

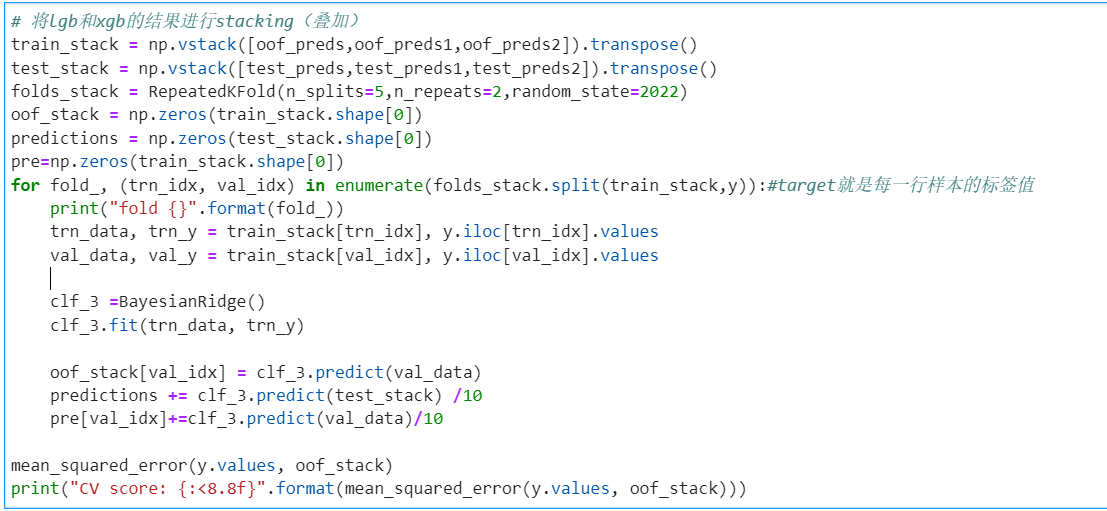
本次比赛是机器学习比赛，我们选择树模型集成学习模型，lightgbm, xgboost还有catboost算法作为我们比赛的模型。

## 模型构建思想

使用lightgbm，xgboost和catboost模型分别在train训练集上进行10折交叉验证，在将训练好的模型在测试集test上预测出概率值，之后将train再次输入三个模型之中，预测出train的概率值，将train的预测值拼接起来形成一个5000\*3的train\_stack，将test的预测值也做相同的操作，拼接成一个5000\*3的test\_stack。



进行模型的stacking，将train\_stack输入BayesianRidge之中，进行5折重复次的交叉验证，以防止过拟合，训练出BayesianRidge，之后将test\_stack输入模型之中，将模型融合起来，这样就能得到stack堆叠之后的结果。

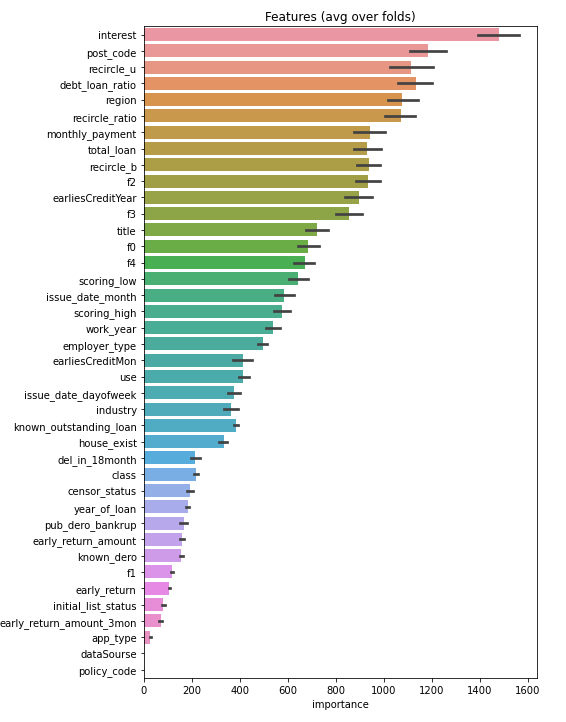


## 模型的分析

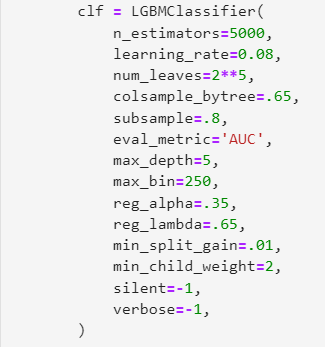
（1）lightgbm

lightgbm算法是我们选择的第一个算法，是一种基于树模型的boost算法，该算法有很多优势，可以自动处理空值，内存占用小，并且lightgbm采用HistoGram算法，使得运算速度快。

我们采用lightgbm模型进行第一次数据分析，通过生成的特征重要度进行排序，筛选过滤掉两个id特征。

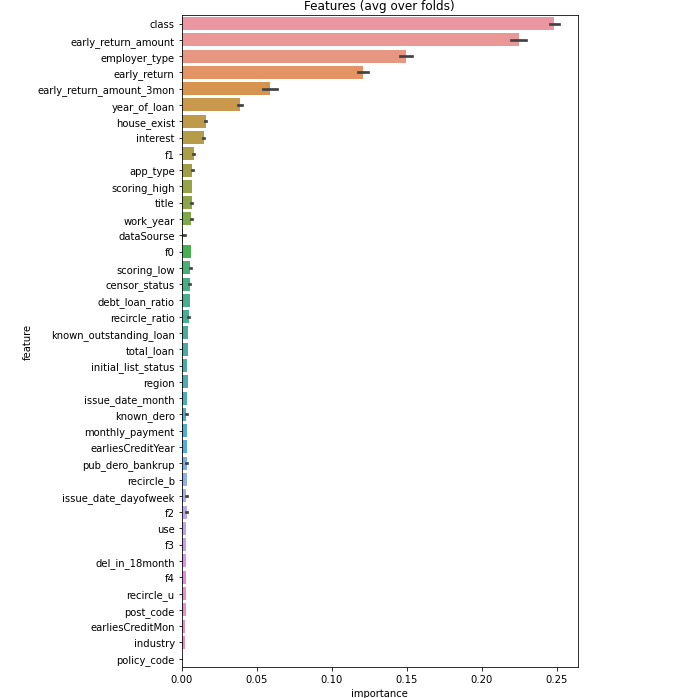


在调参方面，我们是借鉴baseline跑出来的最好成绩参数为基础，在通过optuna进行自动化调参，生成最优参数。



（2）xgboost

xgboost是我们选用的第二个算法，该算法和lightgbm算法相比，并没有太强的优势，选用xgboost一个目的是为了防止lightgbm的过拟合，而是为了在stack的时候对lightgbm进行辅助。Xgboost生成的特征重要性分布与lightgbm有很大差距，发现xgboost与lightgbm的关注点不同，希望通过该模型可以补全lightgbm的短板，在stack时可以提高分数。



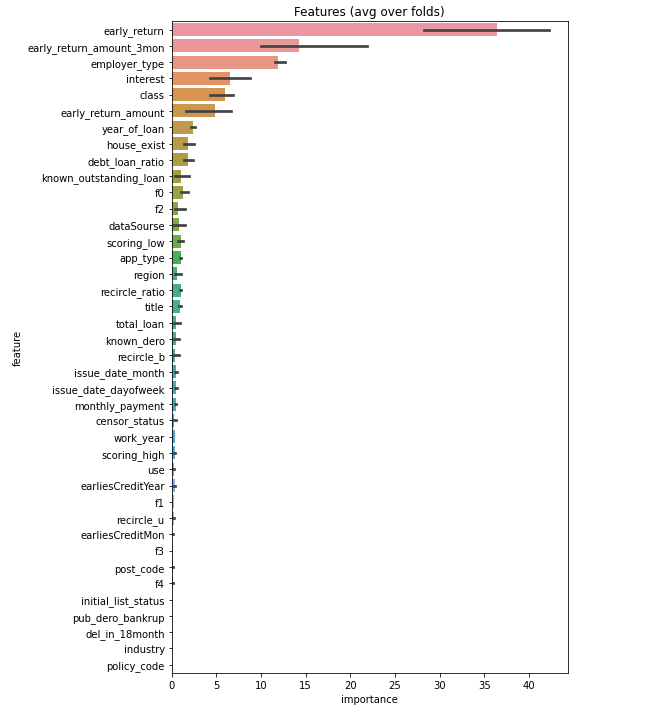
大多数的树形baseline都是用的lightgbm，所以我们xgboost算法的参数是在lightgbm参数的基础上进行的微调。



（3）catboost

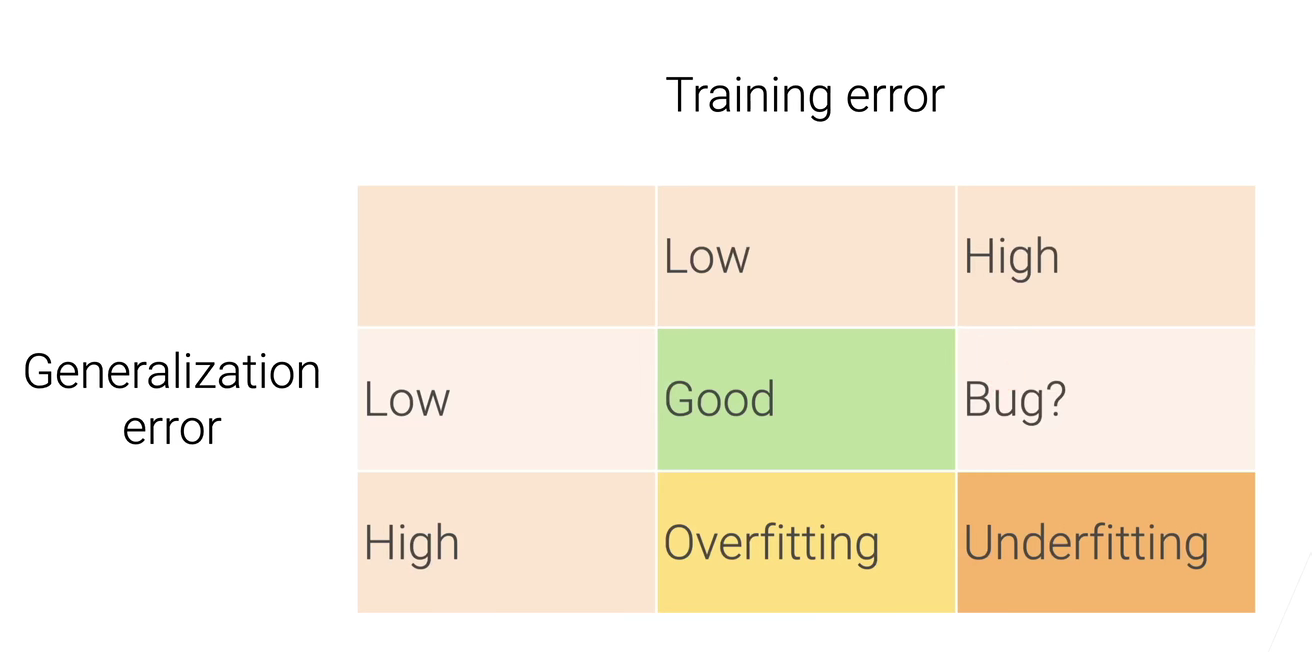
这是我们选择的第三个算法，这个算法有很多优点，这个算法可以直接用GPU训练，加快了训练速度，同时这个算法可以自动处理非数字型数据，但catboost算法在填充空值方面并不比lightgbm强，catboost只能在参数里选择填充空值的数值，无法像lightgbm一样去自行填充空值，并比较两个空值哪一个能使成绩最优。

Catboost算法关注点与前两个算法不一样，但看起来更接近人对这些特征的关注度，比较符合人对这些特征的排序。



因为Catboost算法能力很强，所以我选择加强模型的参数，以让模型更好的去拟合数据集，并在stack中融合得出更好的结果。

## 模型的调参



在模型的训练过程中，我们遇到了training error 高，但generalization error低的情况，通过大体分析，排除是模型bug原因，推测应该是训练数据对模型来说要复杂很多，没法找出数据中隐藏的规律，所以我们在baseline的参数选择上增加模型的复杂度，将树的个数增加，并添加catboost来辅助判断。

## Stack的选择

Stack的模型我们选择比较简单的模型进行stack，以防止数据的过拟合，最初选择的mlp，svm，还有BayesianRidge三种算法，通过多次测试，发现mlp和svm的拟合能力在5000\*3的数据集上完全过拟合，根本不适用，最终选择BayesianRidge作为stack的算法，进行融合。（ps：sklearn对神经网络的封装真差！）

# 数据后处理：

## 目的

在将前面工作中的XGBoost和LightGBM模型预测的结果中，预测值介于0.5附近的数据进行二次预测。我们选择运用简单的算法进行拟合，利用KNN临近算法将待测数据与train\_public数据集中的数据进行分析比对，得到进一步优化的结果。

## 过程

1. 模型的搭建

模型的特点：KNN算法的核心思想是，如果一个样本在特征空间中的K个最相邻的样本中的大多数属于某一个类别，则该样本也属于这个类别，并具有这个类别上样本的特性。该方法在确定分类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。KNN方法在类别决策时，只与极少量的相邻样本有关。由于KNN方法主要靠周围有限的邻近的样本，而不是靠判别类域的方法来确定所属类别的，因此对于类域的交叉或重叠较多的待分样本集来说，KNN方法较其他方法更为适合

模型的构建思想：

* 准备数据，对数据进行预处理。
* 计算测试样本点（也就是待分类点）到其他每个样本点的距离
* 对每个距离进行排序，然后选择出距离最小的K个点。
* 对K个点所属的类别进行比较，根据[少数服从多数](https://baike.baidu.com/item/%E5%B0%91%E6%95%B0%E6%9C%8D%E4%BB%8E%E5%A4%9A%E6%95%B0/10350157)的原则，将测试样本点归入在K个点中占比最高的那一类。

1. 模型的效果效率：

对train\_public 中的5000条数据进行简单的数据清洗，利用train\_test\_split包将train\_public进行4：1比例划分测试机与验证机,并提取标签。随机种子设为22。

代码展示：

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=22)

导入sklearn.neighbors下的KNeighborsClassifier,RadiusNeighborsClassifier算法包

搭建模型，进行预测

model1 = KNeighborsClassifier(n\_neighbors=2)

model1.fit(X\_train, Y\_train)

score1 = model1.score(X\_test, Y\_test)

首次搭建的模型K值设为2，并输出sore1查看预测结结果，模型对train\_public的预测正确率稳定在0.83左右。

1. 模型的改进

增大K值，修改模型标识每个样本的近邻样本权重的模式为uniform自定义权重，半径最近邻法使用auto。经过反复试验，当K为35时，模型对验证集的预测正确率达到最高。

导入待预测数据，对其进行以上相通的数据清洗，并对其进行预测，将结果导出，融合到最终结果中。

# 结论

从今年9月开始了解竞赛，到现在AB榜结束，历时三个月，我们小组从第一次提交得出0.81063839320的成绩，到最终得出得分0.89939268534，A榜排名第八的成绩。在这一过程中，我们经历了许多挫折，都在一次次失败经验中寻找解决方法，在队长的指导下，我们购买书籍，学习网上视频，不断丰富我们的知识，增长我们的能力，在比赛的过程中每一个人都努力查找资料, 仔细检查, 认真核对，逐步将我们的成绩提高，我们的成绩离不开每一位小组成员的努力。

我们小组主要划分两个部分，一小组3人负责数据预处理，另一小组2人负责算法。算法组负责调试算法，不断改进算法，数据预处理小组分析处理数据后交给算法组进行预测，如果成绩降低就改进预处理方法，如果成绩有所提高就进行记录，不断寻求最好的预处理手段。

对于数据预处理方面，主要是对数据进行了删除无用数据字段，填充空缺值，转换类型—将定类和定序数据转换成数字型方便运算，然后就是进行数据标准化，归一化处理。其中比较重要的就是挑选有用的数据字段进行构建新的特征值。算法主要是使用了catboost，LightGBM，xgboost三种主要的算法模型，并在算法的基础上进行调参，以使达到理想的结果。

最后，通过这次比赛（也是最后的课程设计），我们更加扎实的掌握了数据挖掘有关方面的知识，在比赛过程中虽然遇到了一些问题，但经过一次又一次的思考，一遍又一遍的检查找出错误的原因，最终也获得了不错的成绩。实践出真知，通过团队协作，使我们掌握的知识不再是纸上谈兵，培养了我们理论联系实际的能力。在课程设计过程中，我们不断发现错误，改正错误，领悟知识，提高能力。这次大赛很考验小组协作能力的，我们小组在队长的带领下，分工明确，责任详细，多次开会总结经验和不断改进，最终取得了不错的成果。