**金融风控——贷款违约预测**

**学生姓名： 杨晴**

2023年8月

**目录**

[一、项目介绍 2](#_Toc142396493)

[二、探索性数据分析（EDA） 2](#_Toc142396494)

[2.1 数据集概述 2](#_Toc142396495)

[2.2 缺失值 4](#_Toc142396496)

[2.3 唯一值 5](#_Toc142396497)

[2.4 特征分析 5](#_Toc142396498)

[2.4.1 数值型特征 5](#_Toc142396499)

[2.4.2 非数值型特征 8](#_Toc142396500)

[2.5 相关性分析 9](#_Toc142396501)

[三、数据预处理 10](#_Toc142396502)

[3.1 特征转换 10](#_Toc142396503)

[3.1.1 非数值型特征 10](#_Toc142396504)

[3.1.2 离散型特征 11](#_Toc142396505)

[3.2 缺失值处理 11](#_Toc142396506)

[3.2.1 众数填充 11](#_Toc142396507)

[3.2.2 KNNImputer填充 11](#_Toc142396508)

[3.3 特征选择 12](#_Toc142396509)

[3.3.1 相关系数法 12](#_Toc142396510)

[3.3.2 互信息法 13](#_Toc142396511)

[3.3.3 RFE特征消除法 14](#_Toc142396512)

[3.3.4 scorecardpy评分卡工具 15](#_Toc142396513)

[3.4 归一化 16](#_Toc142396514)

[四、模型评估 16](#_Toc142396515)

[4.1 模型 16](#_Toc142396516)

[4.1.1 LR 16](#_Toc142396517)

[4.1.2 RandomForest 18](#_Toc142396518)

[4.1.3 XGBoost 18](#_Toc142396519)

[4.1.4 LightGBM 19](#_Toc142396520)

[4.2 调参 20](#_Toc142396521)

[4.2.1 XGBoost 20](#_Toc142396522)

[4.2.2 LightGBM 21](#_Toc142396523)

[4.3 总结 21](#_Toc142396524)

# 一、项目介绍

该项目以金融风控中的个人信贷为背景，要求选手根据贷款申请人的数据信息预测其是否有违约的可能，以此判断是否通过此项贷款。

# 二、探索性数据分析（EDA）

## 2.1 数据集概述

该数据来自某信贷平台的贷款记录，总数据量超过120w，包含47列变量信息，其中15列为匿名变量。为了保证比赛的公平性，将会从中抽取80万条作为训练集，20万条作为测试集A，20万条作为测试集B，同时会对employmentTitle、purpose、postCode和title等信息进行脱敏。

对训练集数据的简单查看：

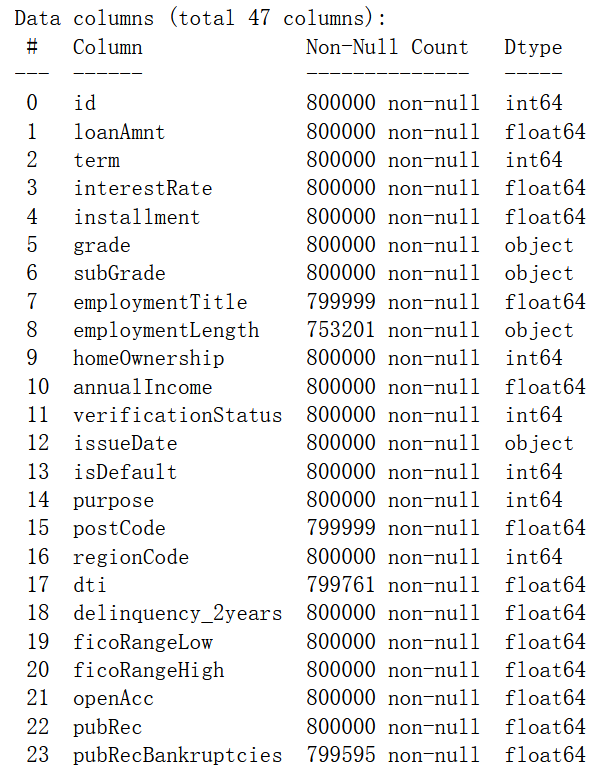
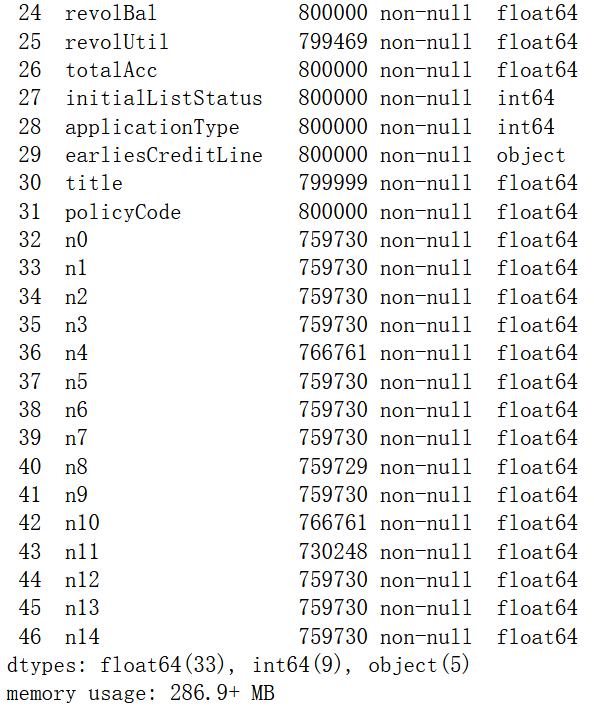
 

图2-1. 训练集的info() 结果

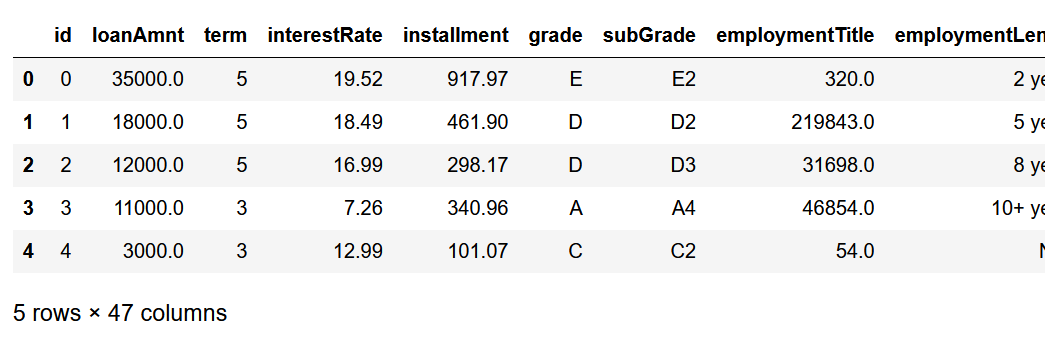


图2-2. 训练集的head() 结果

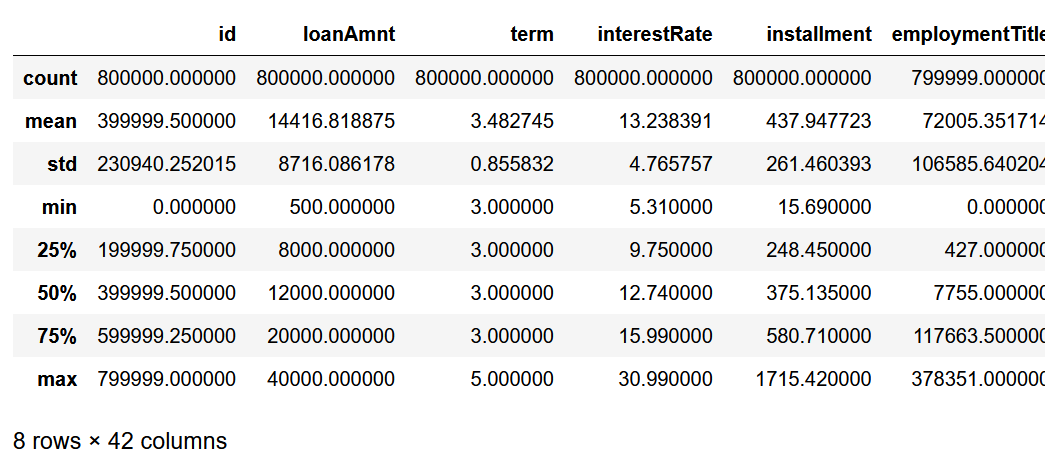


图2-3. 训练集的describe() 结果

对测试集数据的简单查看：

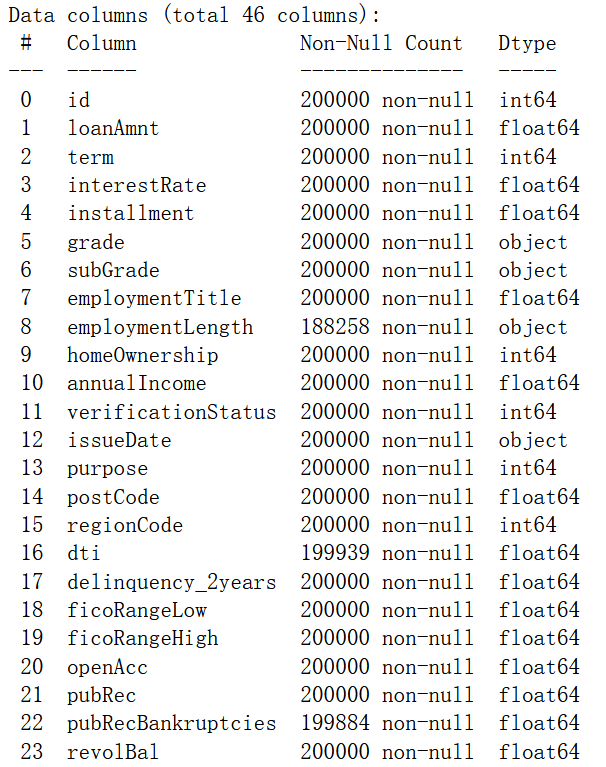
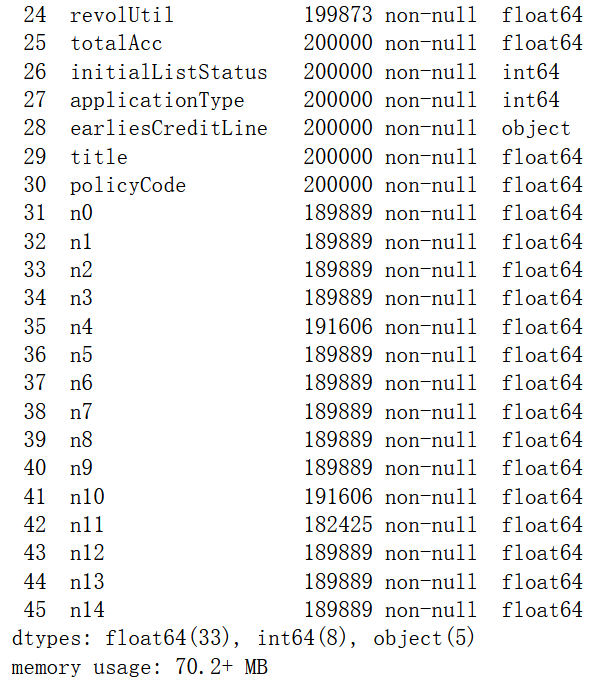
 

图2-4. 测试集的info() 结果

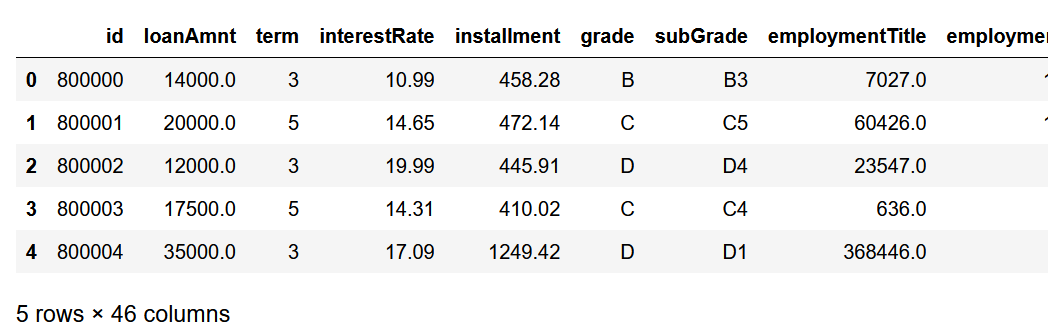


图2-5. 测试集的head() 结果

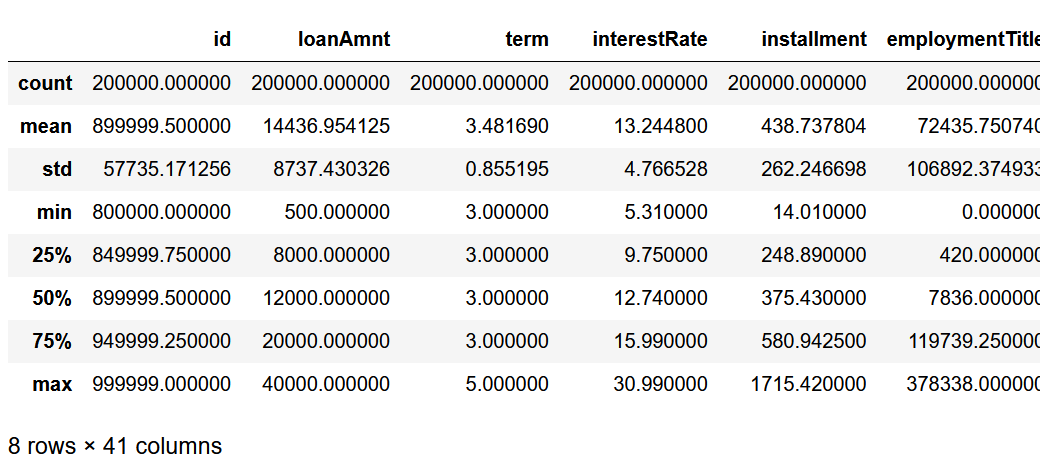


图2-6. 测试集的describe() 结果

## 2.2 缺失值

通过missingno库来查看训练集和测试集的缺失值：

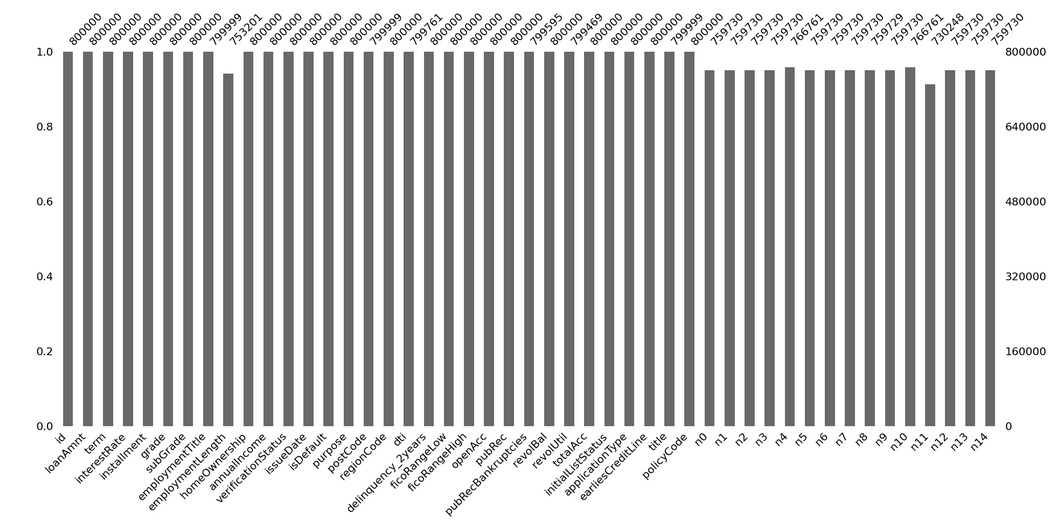


图2-7. 训练集的缺失值

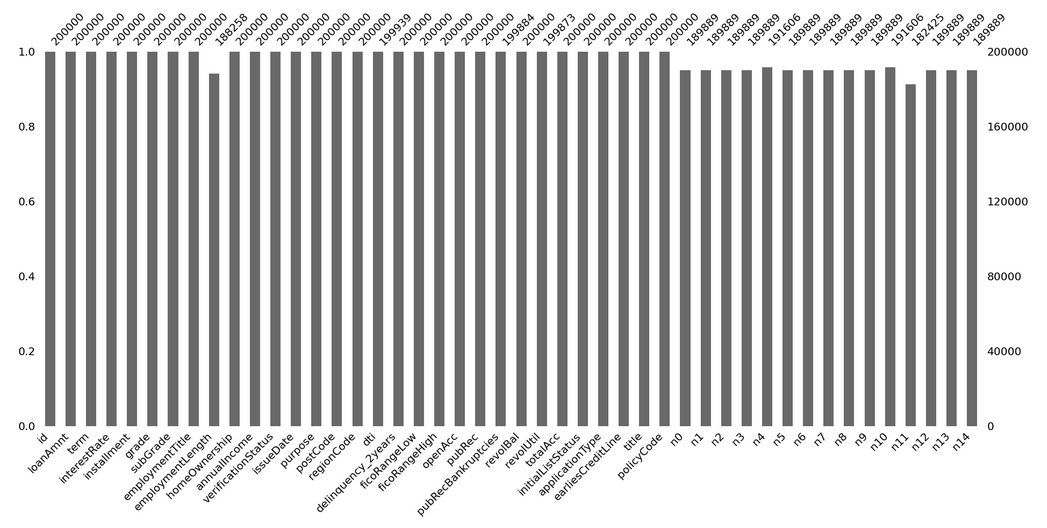


图2-8. 测试集的缺失值

由图2-8可知，训练集中特征 employmentTitle, employmentLength, postCode, dti, pubRecBankruptcies, revolUtil, title, n0-n14 均存在缺失值，其中所有的缺失值数量都不超过50%；测试集中特征 employmentLength, dti, pubRecBankruptcies, title, n0-n14 均存在缺失值，所有的缺失值数量都不超过50%。

## 2.3 唯一值

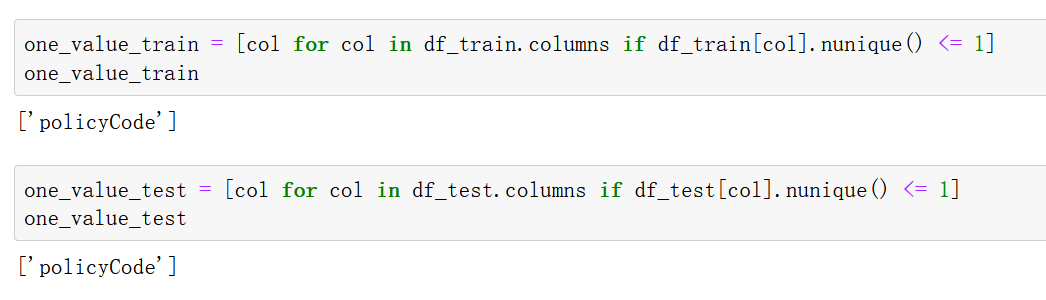


图2-9. 训练集和测试集获取唯一值

训练集和测试集当中都有唯一值的特征 policyCode，后续可以考虑删除该特征。

## 2.4 特征分析

从训练集和测试集的info()结果来看，存在type为 int、float、object 的特征，下面对它们进行更详细的分类查看。

### 2.4.1 数值型特征

数值型特征有：[id, loanAmnt, term, interestRate, installment, employmentTitle, homeOwnership, annualIncome, verificationStatus, isDefault, purpose, postCode, regionCode, dti, delinquency\_2years, ficoRangeLow, ficoRangeHigh, openAcc, pubRec, pubRecBankruptcies, revolBal, revolUtil, totalAcc, initialListStatus, applicationType, title, policyCode, n0, n1, n2, n3, n4, n5, n6, n7, n8, n9, n10, n11, n12, n13, n14]，共42个。

其中连续型数值特征：[id, loanAmnt, interestRate, installment, employmentTitle, annualIncome, purpose, postCode, regionCode, dti, delinquency\_2years, ficoRangeLow, ficoRangeHigh, openAcc, pubRec, pubRecBankruptcies, revolBal, revolUtil, totalAcc, title, n0, n1, n2, n3, n4, n5, n6, n7, n8, n9, n10, n13, n14]，共33个。

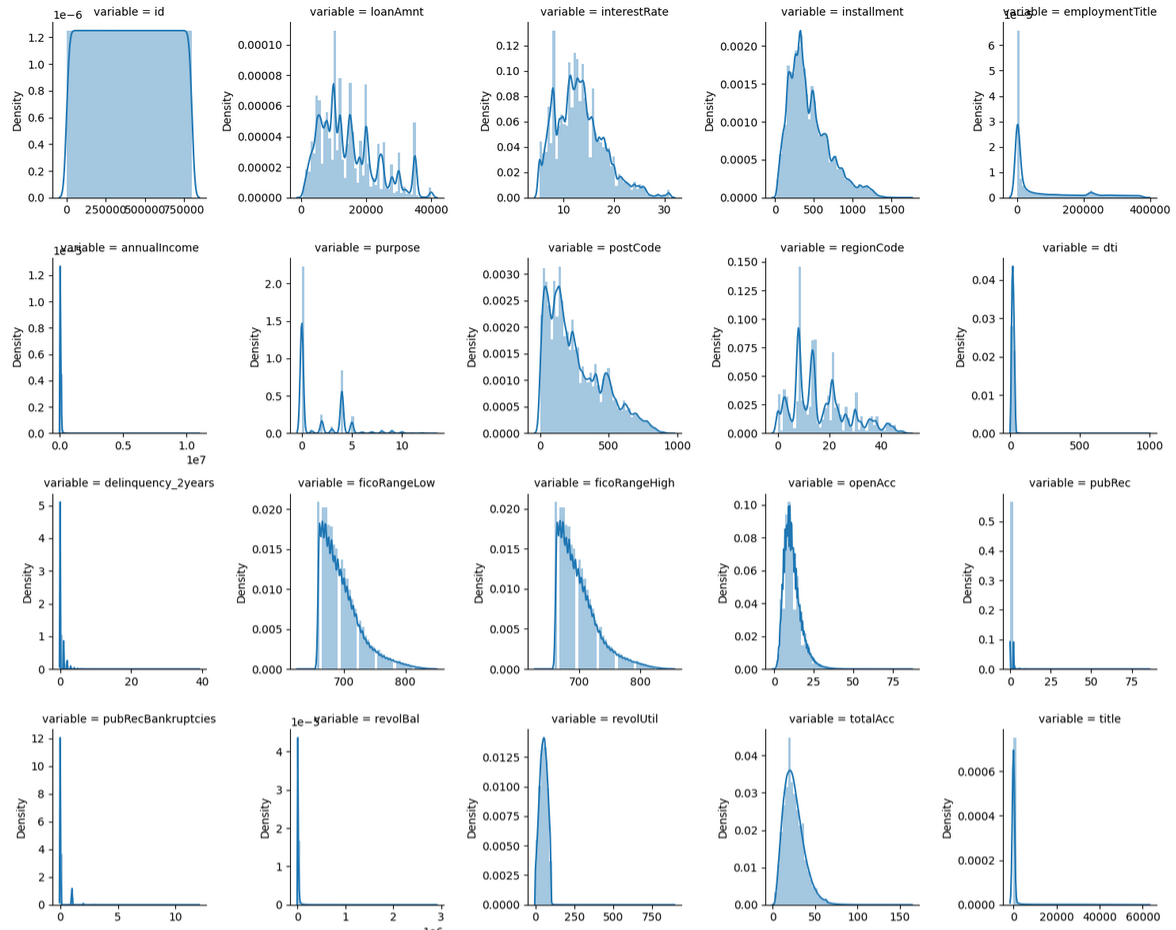


图2-10. 连续型特征分布（部分）

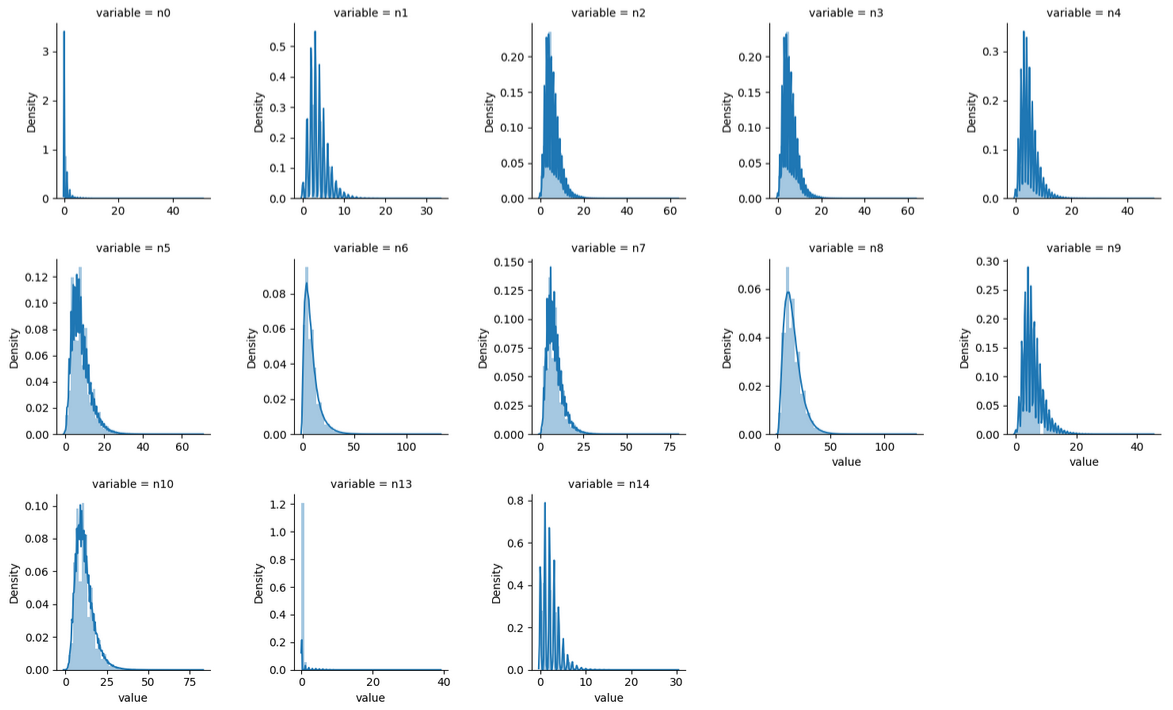


图2-11. 连续型特征分布（部分）

离散型数值特征：[term, homeOwnership, verificationStatus, isDefault, initialListStatus, applicationType, policyCode, n11, n12]，共9个，其中 isDefault 是标签数据。

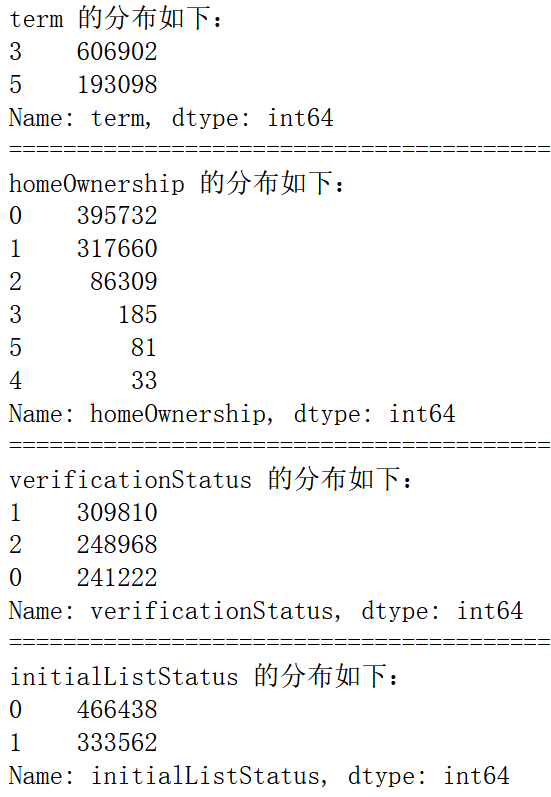
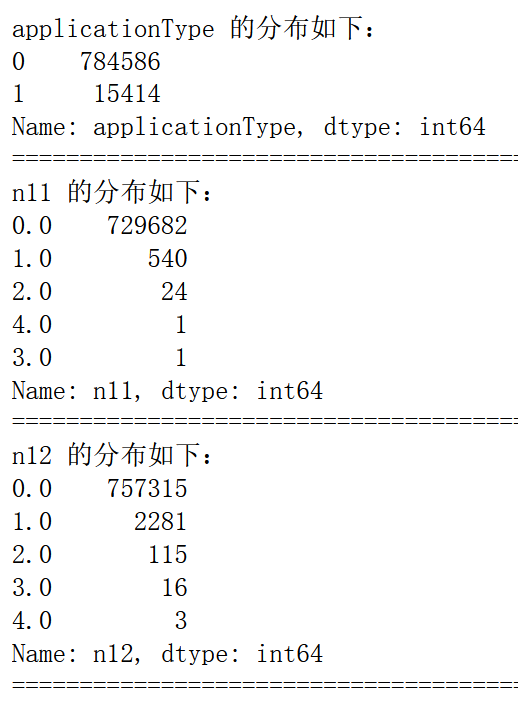
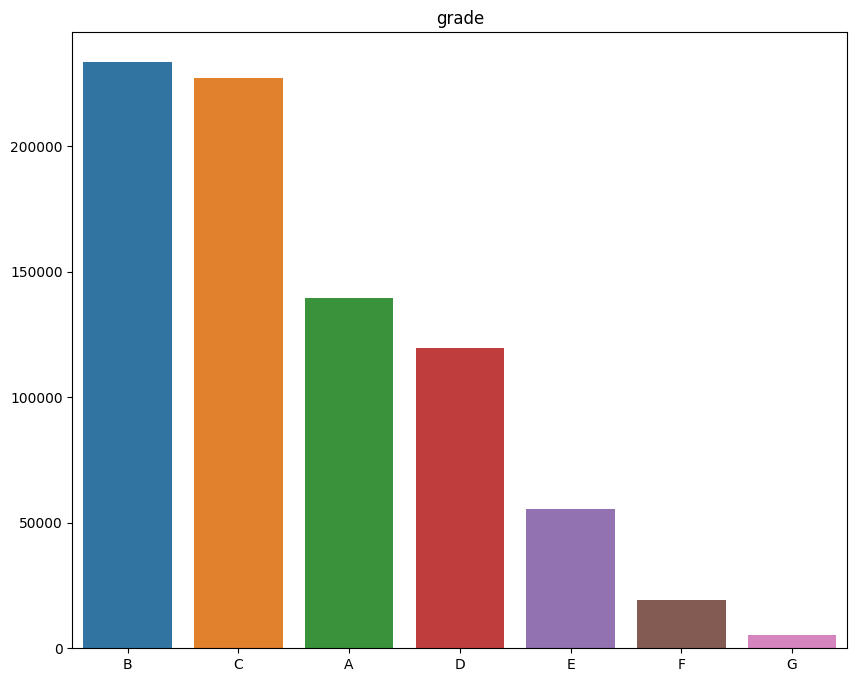
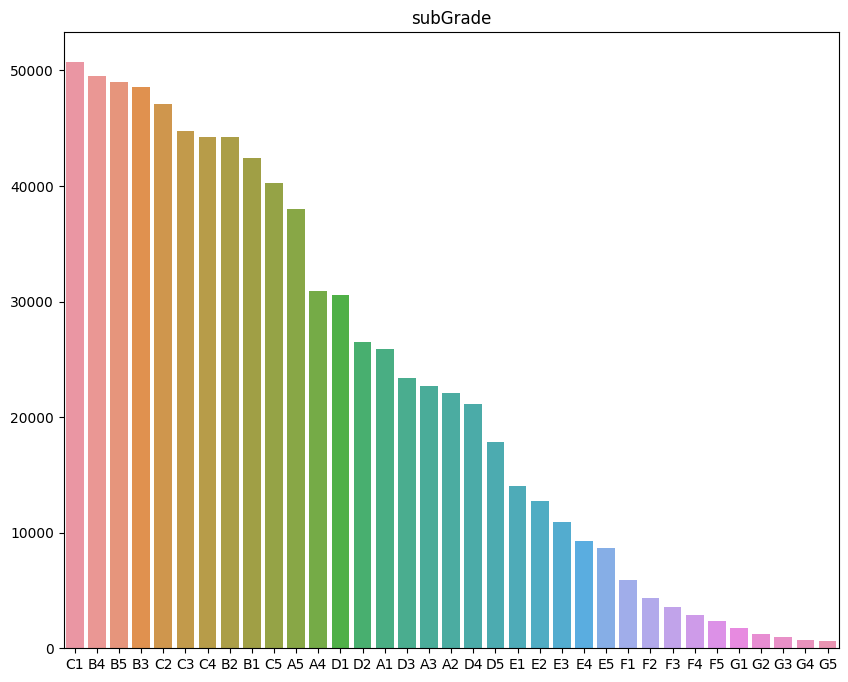
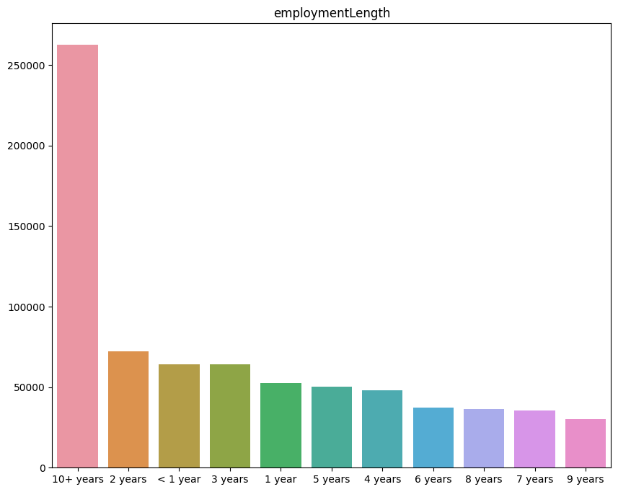
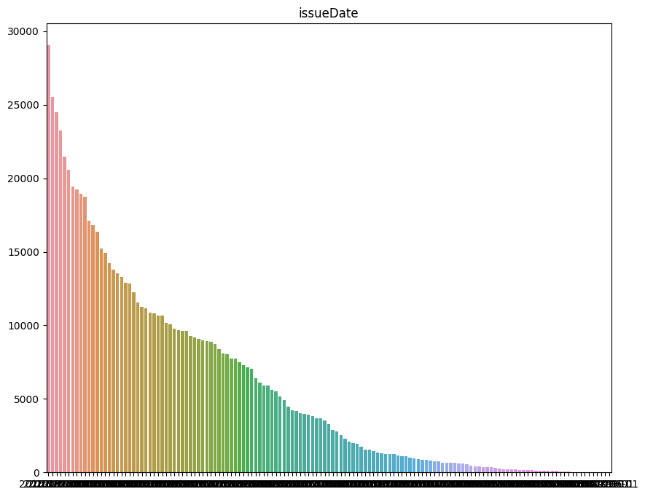
 

图2-12. 离散型特征分布

### 2.4.2 非数值型特征

非数值型特征有：[grade, subGrade, employmentLength, issueDate, earliesCreditLine]，共5个。

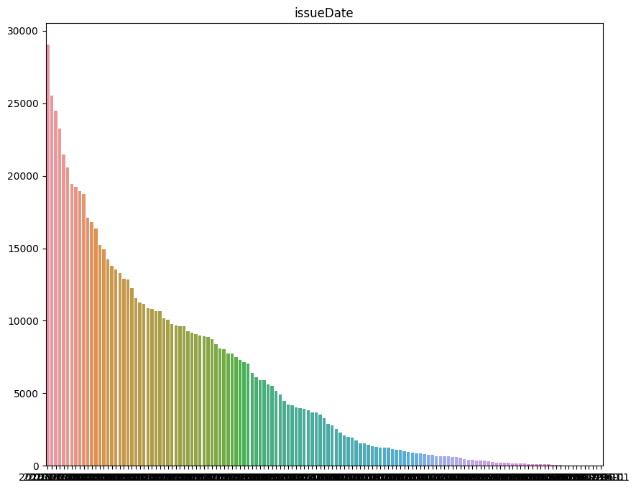


图2-13. 非数值型特征分布

可视化一些特征与标签的关系：

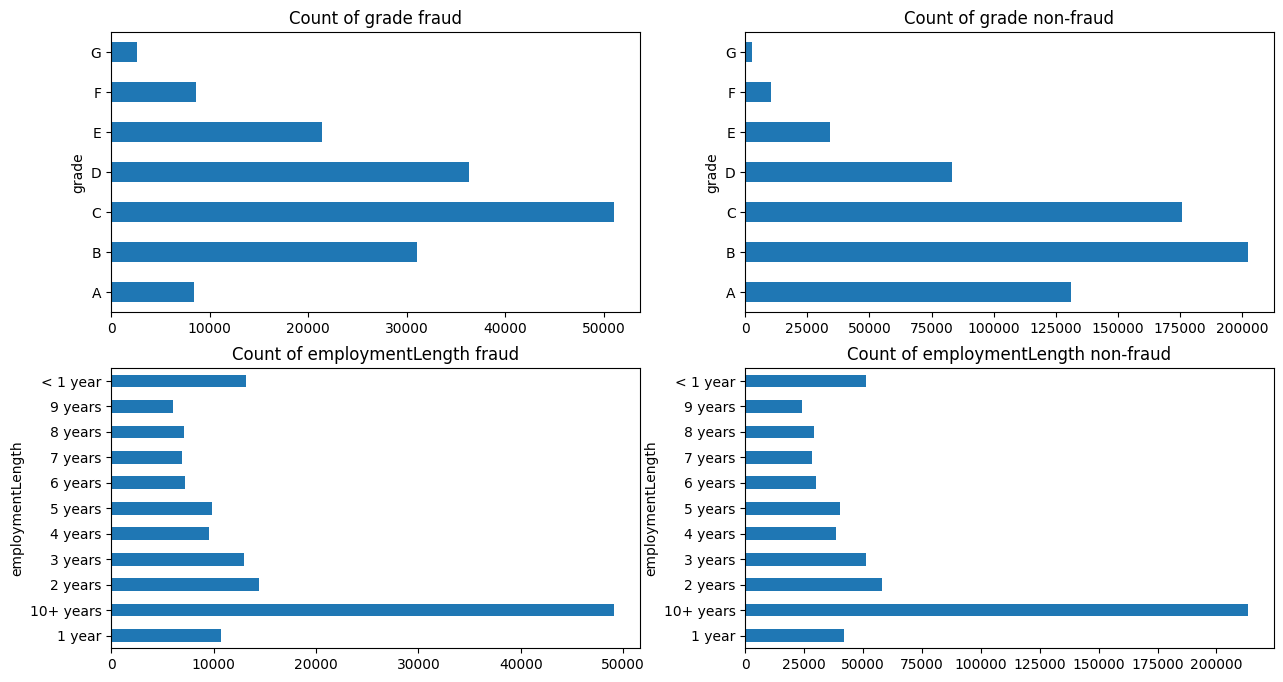


图2-13. 非数值型特征分布

由图2-13可知，特征 grade 在违约数据中主要是C等级最多，在未违约数据中则是B等级最多；特征 employmentLength 在违约和未违约的数据中均是就业年限超过10年的数据最多。

## 2.5 相关性分析

从图2-14的标签分布可以看出，该数据集是一个不平衡的数据集，违约：未违约 ≈ 1：4。

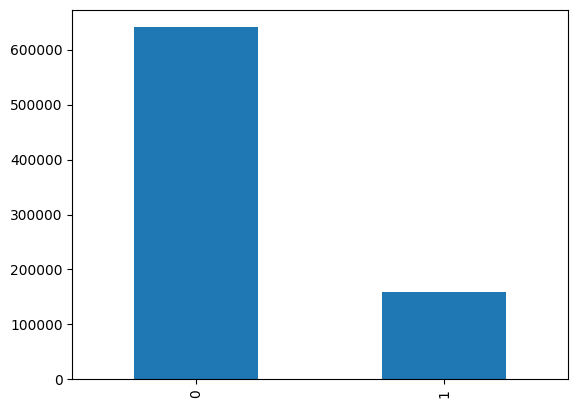


图2-14. 非数值型特征分布

可视化标签与其他46个特征之间的相关性，通过热力图进行展示。从热力图中可知，特征 n2、n3 与 n9 极为相似，特征 openAcc 与 n10 极为相似，可以考虑只保留 n2 与 openAcc。

筛选20个与标签最为相近的特征，通过热力图进行展示，见图2-14。

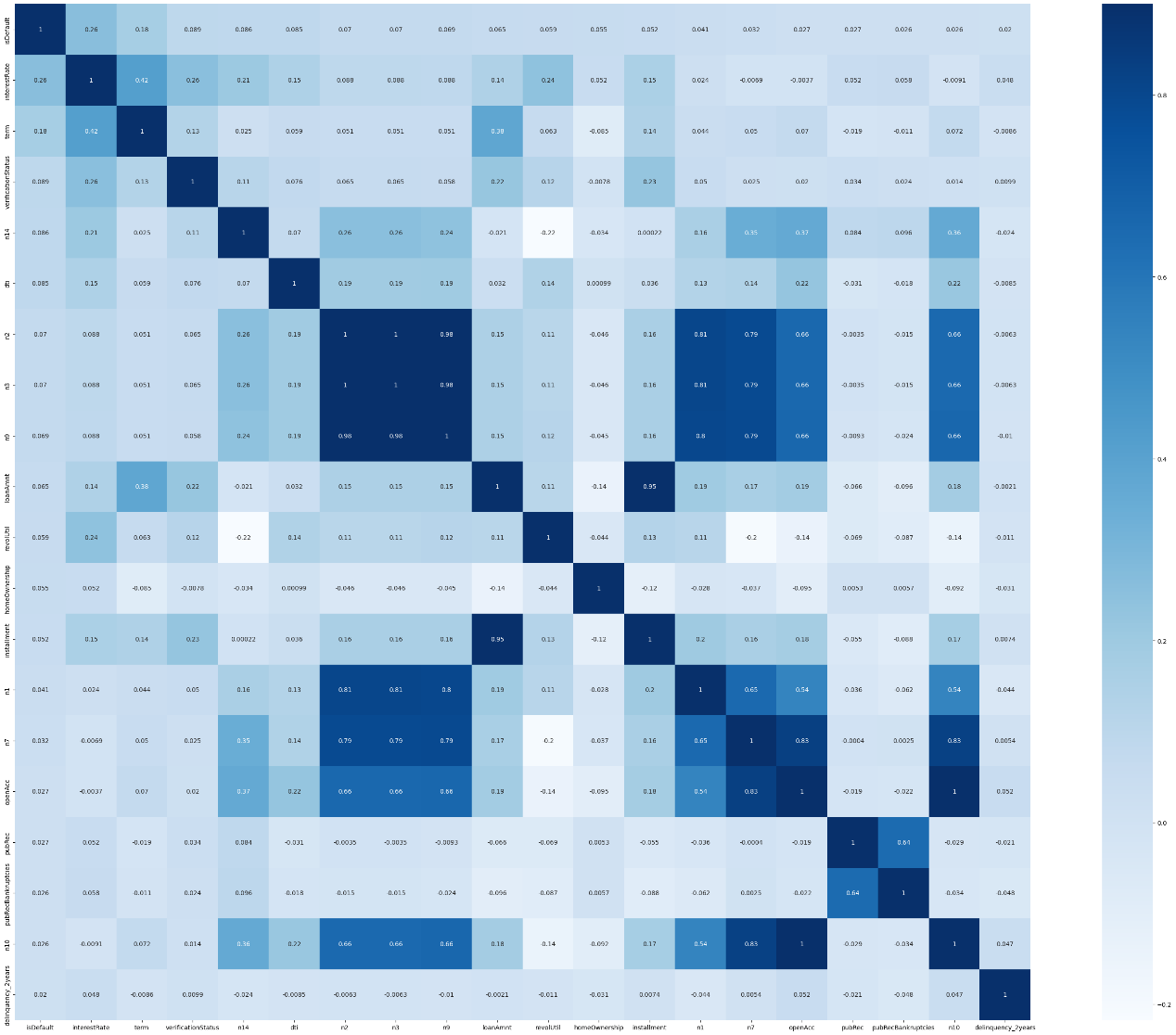


图2-15. 20个最相关的特征

由图2-15可知，标签与其他属性的相关性很弱。

# 三、数据预处理

数据预处理是指在进行机器学习和数据分析之前，对原始数据进行一系列的处理和转换操作，以使数据更加适合用于建模和分析。数据预处理是数据科学中的关键步骤之一，它可以影响模型的性能、泛化能力和可解释性。

## 3.1 特征转换

选择删去无意义的特征 id，具有唯一值的特征 policyCode，以及与 n2 极为相似的特征 n3、n9 、与 openAcc 极为相似的特征 n10。

### 3.1.1 非数值型特征

对于 grade 特征，通过excel表可以看出，grade总共有A、B、C、D、E、F、G共7个等级，采用0、1、2、3、4、5、6、7来对其进行映射。

对于subGrade特征，它是grade特征的子级，总共有35种类别，采用one-hot编码的方式来对其进行处理。

对于employmentLength特征，它代表的是就业年限，通过查看它的数据分布，发现就业年限大于10年的数据有26万多条，远超过就业年限处于10年以内的，故将就业年限大于10年的取值定为10年，就业年限小于1年的取值定为1年，在对其进行处理，即将 ’2 years’ 变为2，’3 years’ 变为3，以此类推。

对于issueDate特征，原始格式是字符串，先将其转化为datetime日期格式，然后再求出该列数据的最小日期，根据最小日期构造出时间间隔issueDateDT这个特征，再将issueDate特征从原始数据中删除。

对于earliesCreditLine特征，它代表的是借款人最早报告的信用额度开立的月份，此处用对应的年份数据进行代替，即将 ’ 2001/8/1’ 变为 2001，’ 2002/5/1’ 变为 2002，以此类推。

### 3.1.2 离散型特征

对于离散型特征，直接对它们进行one-hot编码处理。

## 3.2 缺失值处理

通过查看训练集和测试集，发现：

训练集存在缺失值的特征：employmentTitle, employmentLength, postCode, dti, pubRecBankruptcies, revolUtil, title, n0, n1, n2, n4, n5, n6, n7, n8, n13, n14；测试集存在缺失值的特征：employmentLength, dti, pubRecBankruptcies, revolUtil, n0, n1, n2, n4, n5, n6, n7, n8, n13, n14，采用两种填充方式对它们进行填充：众数填充和KNNImputer填充。因为数据集具有较多的异常值，故未采用均值来填充。

### 3.2.1 众数填充

众数填充是一种常用的缺失值填充方法。在众数填充中，对于某个特征的缺失值，使用该特征的众数（即出现次数最多的值）来填充缺失位置。

### 3.2.2 KNNImputer填充

KNNImputer是一种常用的缺失值填充方法，核心思想是基于K最近邻算法，通过利用已有数据中与缺失值样本最接近的K个邻居的特征信息，来估计缺失值。优点在于它不会引入新的偏差，能够提供更准确的缺失值估计，特别是在数据中存在复杂的关联关系时表现良好。

## 3.3 特征选择

特征选择是一项重要的数据预处理技术，它在机器学习和数据挖掘中起着至关重要的作用。在处理原始数据时，往往会涉及大量的特征，但并非所有特征都对模型的性能有积极的影响。相反，过多的特征可能会引发问题，如维度灾难和过拟合，导致模型的效率降低并且泛化能力受损。为了避免这些问题，特征选择的目标是从原始特征集合中选择最具有代表性和相关性的特征子集，优化模型的训练和预测性能。以下方法的特征选择结果文件位于 ’data\_progress\_results/’ 文件夹下，其中 ’data\_progress\_results/mode’、’data\_progress\_results/knn’ 文件夹里分别存放众数填充和KNNImputer填充后特征选择的结果。

### 3.3.1 相关系数法

相关系数法是一种简单而有效的特征选择技术，在机器学习和数据分析领域得到广泛应用。它主要适用于处理连续型特征和数值型目标变量之间的关系。该方法通过计算特征与目标变量之间的相关系数，衡量它们之间的线性相关性，从而确定哪些特征对目标变量的预测具有较强的解释能力。

在本项目中，通过threshold阈值来筛选相关特征，最终筛选结果如下：

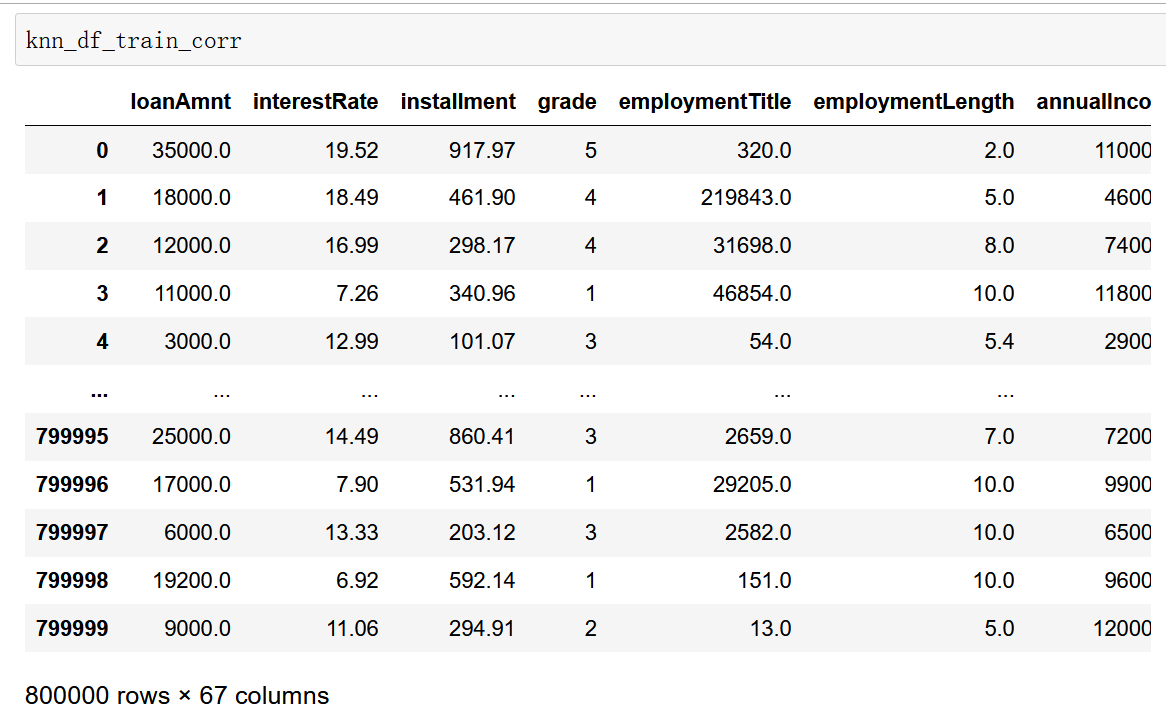


图3-1. 通过KNNImputer填充后再通过相关系数法筛选的结果

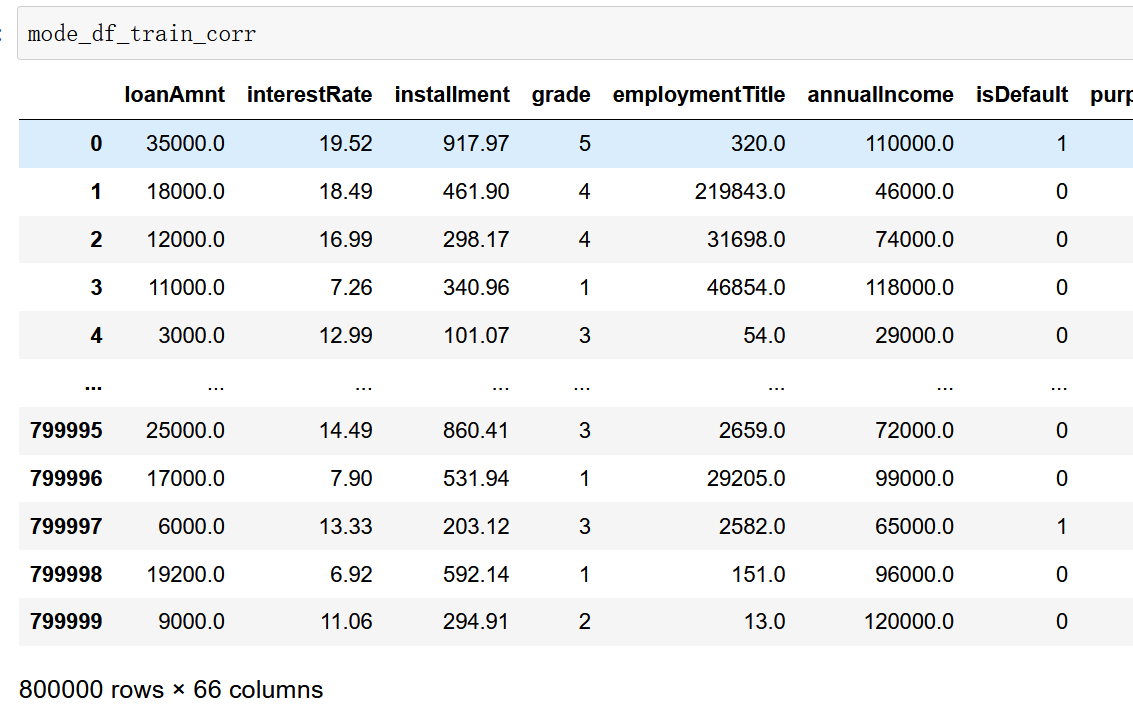


图3-2. 通过众数填充后再通过相关系数法筛选的结果

### 3.3.2 互信息法

互信息法被用来衡量每个特征与目标变量之间的相关性和重要性，其步骤是通过计算每个特征与目标变量之间的互信息值，来衡量特征对目标变量的重要程度。互信息值越大，表示特征与目标变量之间的关联性越强。

在本项目中，通过人为设定保留的特征数来筛选特征，最终筛选结果如下：

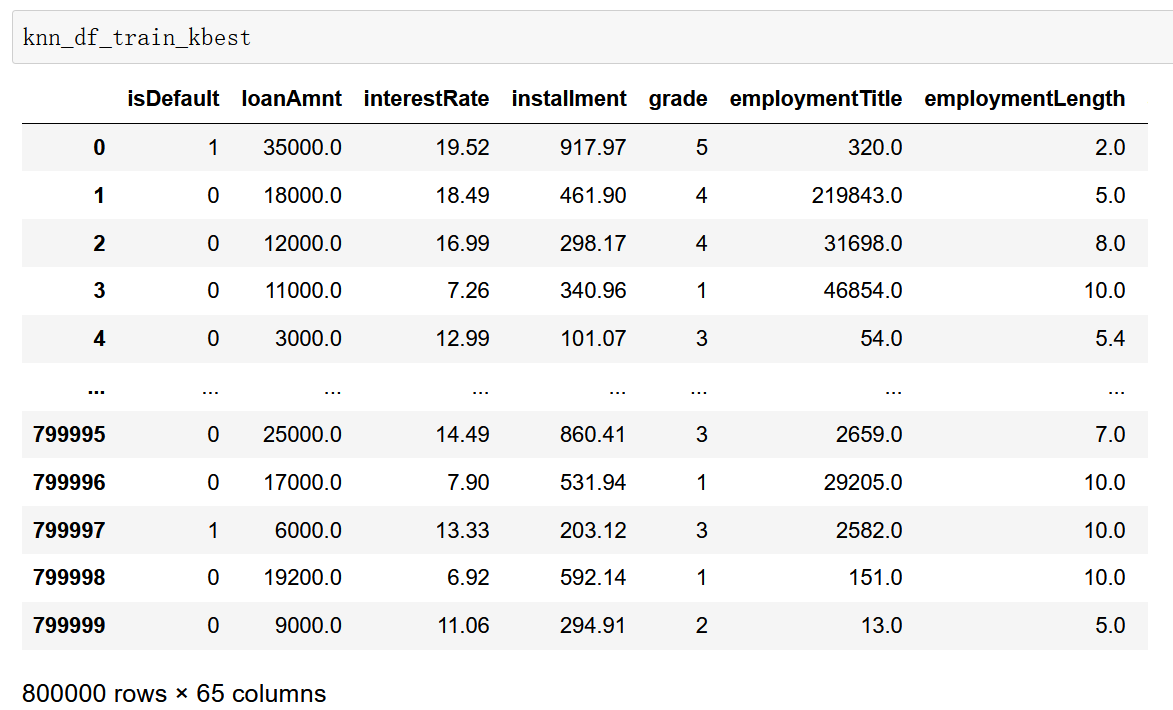


图3-3. 通过KNNImputer填充后再通过互信息法筛选的结果

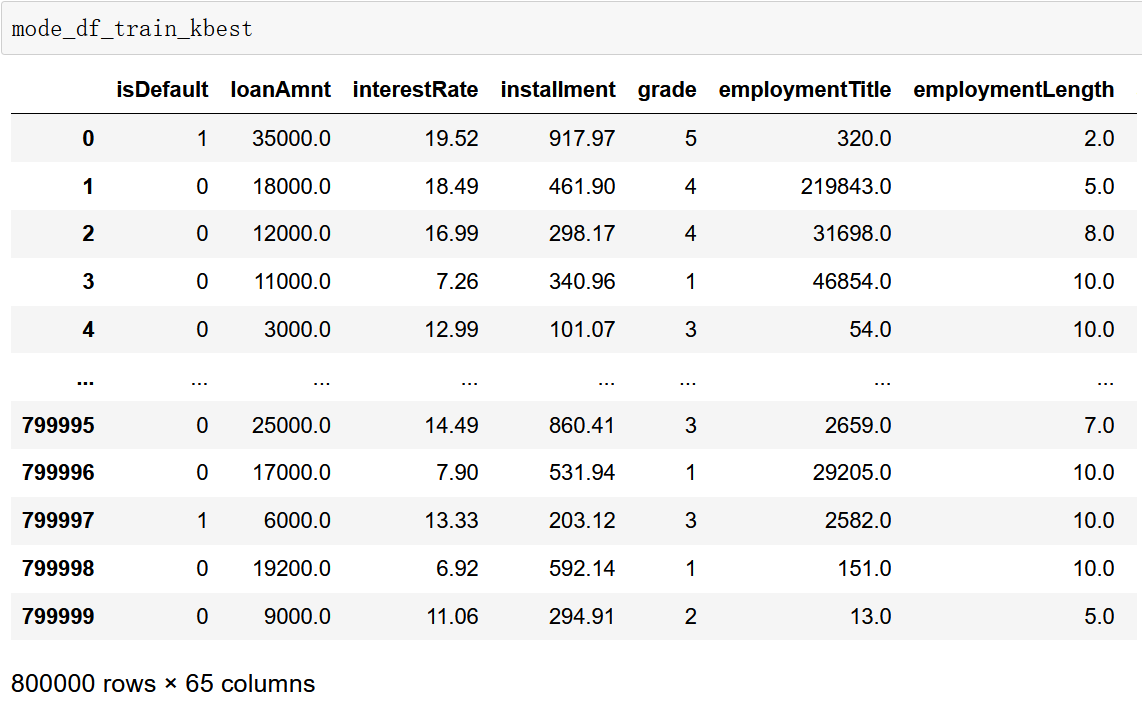


图3-4. 通过众数填充后再通过互信息法筛选的结果

### 3.3.3 RFE特征消除法

RFE（Recursive Feature Elimination）特征消除法是一种常用的特征选择方法，用于在机器学习和数据挖掘任务中选择对模型性能最有贡献的特征子集。它是一种递归的过滤方法，通过不断剔除最不重要的特征，直到达到预定的特征数量或满足某个停止条件。

在本项目中，通过人为设定保留的特征数来筛选特征，最终筛选结果如下：

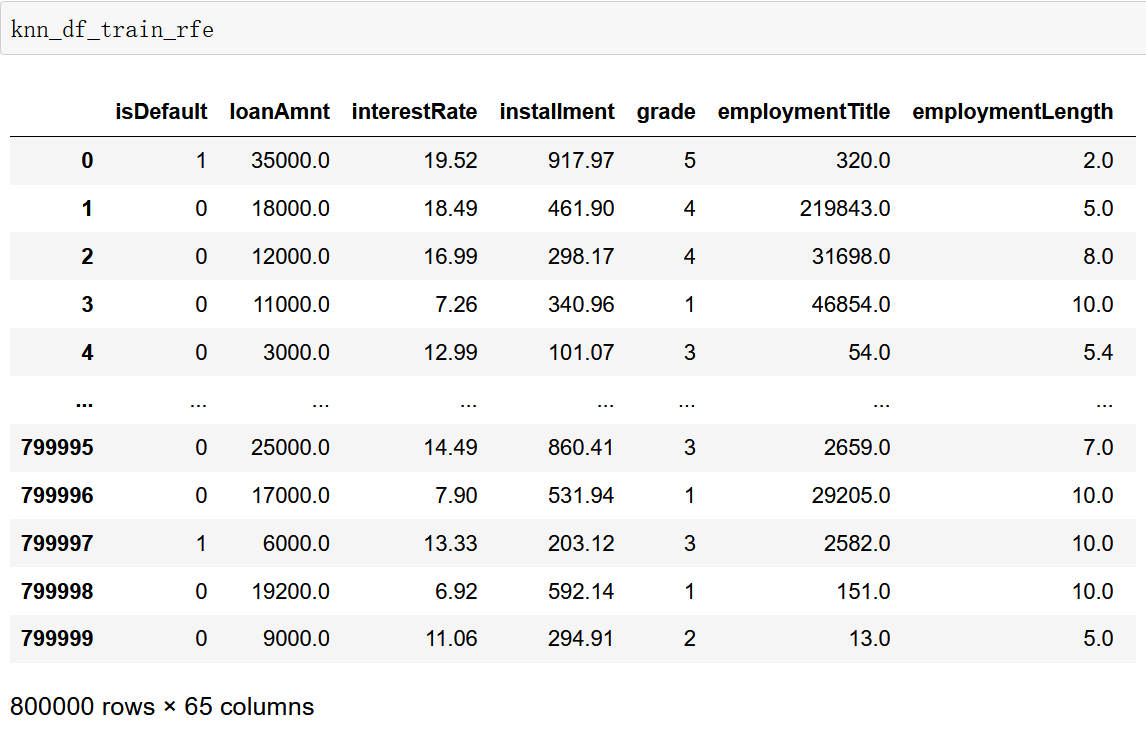


图3-5. 通过KNNImputer填充后再通过RFE特征消除法筛选的结果

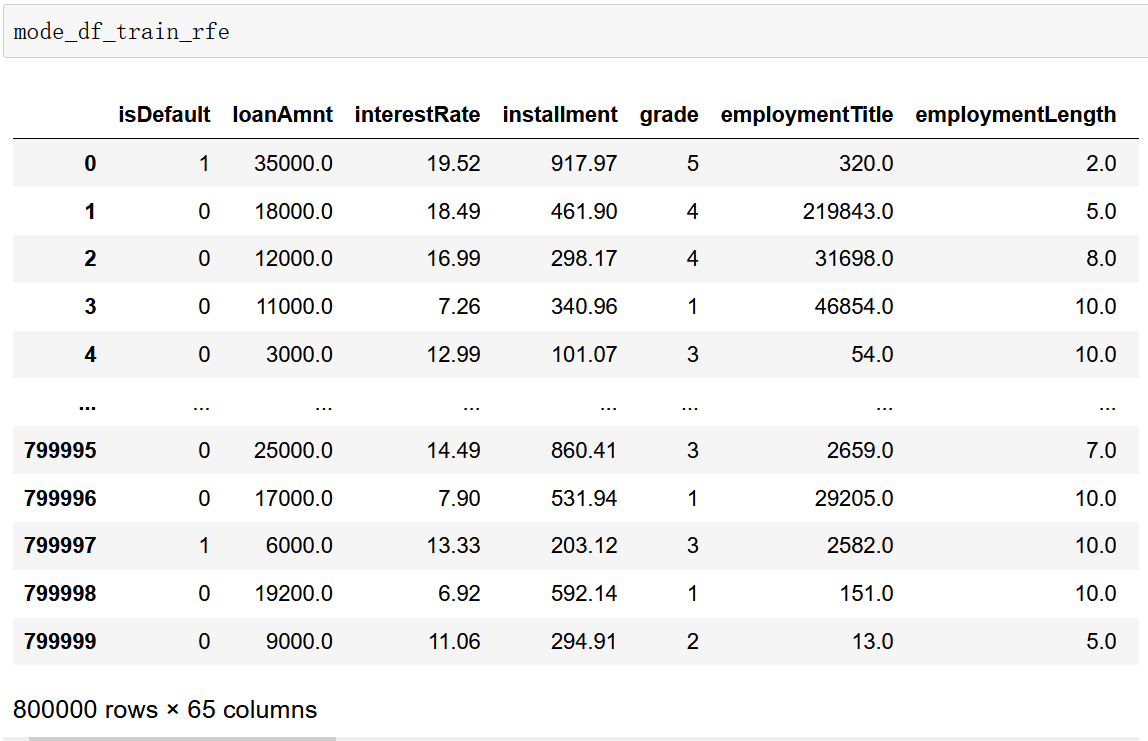


图3-6. 通过众数填充后再通过RFE特征消除法筛选的结果

### 3.3.4 scorecardpy评分卡工具

scorecardpy是一个用于构建评分卡的Python库，它提供了一套简单且强大的工具，用于在信用风险评估、金融风控等领域中构建评分卡模型。评分卡是一种常用的风险评估工具，用于将各种特征转化为分数，从而对个体的信用风险进行评估和预测。其主要功能包括数据划分、变量选择、数据分箱等。

在本项目中，通过scorecardpy库里筛选变量的函数var\_filter来对特征进行筛选，最终筛选结果如下：

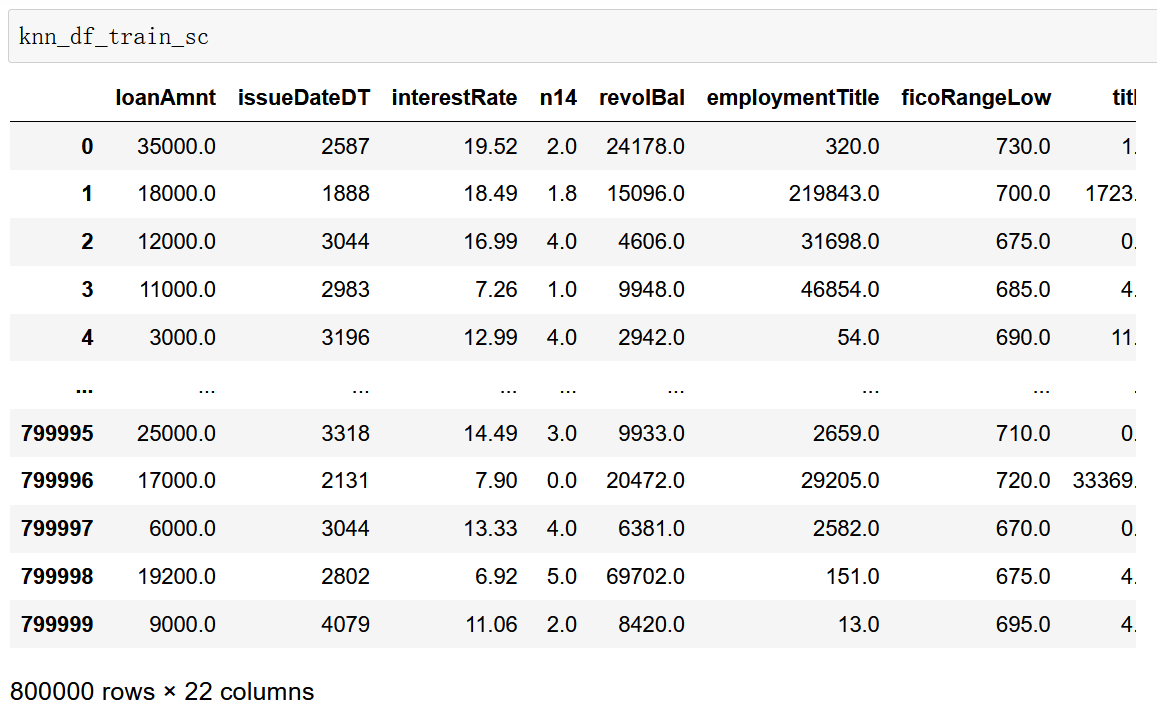


图3-7. 通过KNNImputer填充后再通过评分卡工具筛选的结果

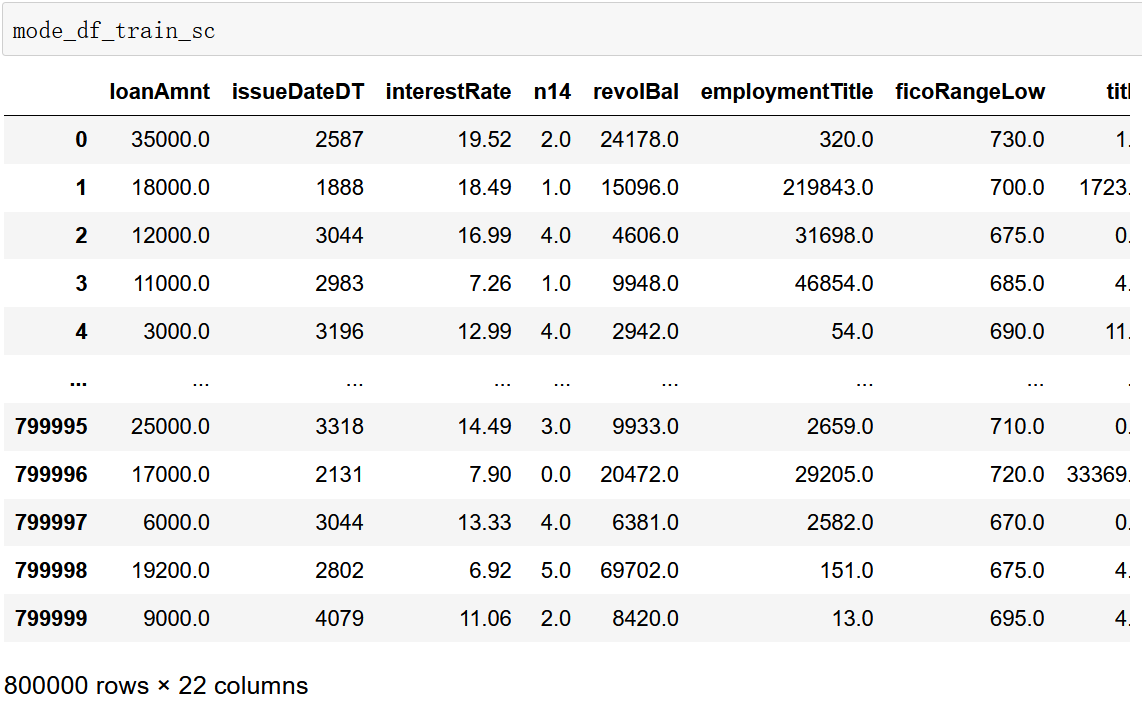


图3-8. 通过众数填充后再通过评分卡工具筛选的结果

## 3.4 归一化

RobustScaler是一种数据归一化方法，他可以通过中位数和四分位数范围来缩放数据。具体来说，它将每个特征的值减去该特征的中位数，然后再除以该特征的四分位数范围。与其他归一化方法（如MinMaxScaler或StandardScaler）不同，RobustScaler是一种鲁棒性较强的标准化方法，对于含有异常值的数据集表现较好。因为该数据集通过箱线图查看可知有较多特征具有异常值，但从现实意义来说，不好定义这些特征的异常值，没有对这些异常值进行处理，故采用RobustScaler方法对数据进行归一化。

# 四、模型评估

## 4.1 模型

因为是不平衡数据集，故在进行训练、测试集划分时采用 ’stratify’ 参数，保持测试集与整个数据集里result的数据分类比例一致。为了保证结果的可复现性，对于每个模型，设置其随机种子均为42。

### 4.1.1 LR

逻辑回归模型是一种用于解决分类问题的统计学习方法。它在数据挖掘、机器学习和统计分析等领域被广泛应用。逻辑回归的主要目标是根据已有的特征数据，预测某个样本属于某个类别的概率，并将其映射到一个[0, 1]的范围内，从而进行分类。

以下是不同处理过后的数据集在LR模型上的具体表现，此处只是简单测试了一下，并没有使用交叉验证方法：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| dataset | model | acc | precision | recall | f1 | auc |
| knn\_no\_other | lr | 0.8002 | 0.3200 | 0.0013 | 0.0025 | 0.6261 |
| knn\_corr | lr | 0.8014 | 0.5173 | 0.0692 | 0.1221 | **0.7018** |
| knn\_kbest | lr | 0.7990 | 0.4802 | 0.0933 | 0.1562 | 0.6915 |
| knn\_rfe | lr | 0.8007 | 0.5033 | 0.0842 | 0.1442 | 0.6999 |
| knn\_sc | lr | 0.8014 | 0.5173 | 0.0692 | 0.1221 | **0.7018** |
| mode\_no\_other | lr | 0.8002 | 0.3333 | 0.0014 | 0.0029 | 0.6266 |
| mode\_corr | lr | 0.7992 | 0.4853 | 0.1035 | 0.1707 | 0.6949 |
| mode\_kbest | lr | 0.8007 | 0.5036 | 0.0750 | 0.1305 | 0.6971 |
| mode\_rfe | lr | 0.8018 | 0.5293 | 0.0611 | 0.1095 | 0.6992 |
| mode\_sc | lr | 0.8020 | 0.5327 | 0.0615 | 0.1103 | **0.7066** |

根据上表的结果显示，对于LR模型来说，在使用KNNImputer填充的数据集中，通过相关系数法和评分卡工具筛选特征得到的AUC取值最高，为0.7018。而在使用众数填充的数据集中，通过评分卡工具筛选特征得到的AUC取值最高，为0.7066。相比直接填充而不筛选特征，填充完毕后再进行特征筛选的结果要更好，无论采用哪种筛选特征的方法，基本上要高0.07左右。需要特别注意的是，由于这是一个不平衡的数据集，从表中可以看到，所有结果的recall和f1-score都很低。这意味着模型在识别正例（少数类）方面存在较大的困难，导致这些指标较低。

在处理这个不平衡数据集时，我也尝试采用了过采样处理（SMOTE）的方法，通过增加少数类样本来平衡数据分布。然而，令人意外的是，最终的结果相对于不进行过采样处理并没有显著的提升。为了更好地了解实验结果，我将相关数据保存在了两个文件中：'code/results/results\_smote\_knn.csv' 和 'code/results/results\_smote\_mode.csv'，感兴趣的请前往查看。

### 4.1.2 RandomForest

随机森林是一种强大且广泛应用于机器学习的集成学习方法。它通过构建多个决策树模型，并将它们的预测结果综合起来，从而得到更为准确和稳健的预测结果。随机森林的核心思想是结合多个相对简单的决策树（称为弱学习器），通过集成的方式形成一个强大的集成模型（称为强学习器）。随机森林的随机性使得其在处理大规模数据集和高维特征时表现优秀，并且具有较好的泛化能力，能够有效地避免过拟合问题。

以下是不同处理过后的数据集在RandomForest模型上的具体表现，此处只是简单测试了一下，并没有使用交叉验证方法：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| dataset | model | acc | precision | recall | f1 | auc |
| knn\_no\_other | rf | 0.8046 | 0.5775 | 0.0763 | 0.1349 | **0.7119** |
| knn\_corr | rf | 0.8046 | 0.5709 | 0.0826 | 0.1443 | **0.7119** |
| knn\_kbest | rf | 0.8037 | 0.5574 | 0.0783 | 0.1374 | 0.7106 |
| knn\_rfe | rf | 0.8041 | 0.5645 | 0.0793 | 0.1390 | 0.7117 |
| knn\_sc | rf | 0.8046 | 0.5709 | 0.0826 | 0.1443 | **0.7119** |
| mode\_no\_other | rf | 0.8044 | 0.5739 | 0.0758 | 0.1339 | **0.7119** |
| mode\_corr | rf | 0.8041 | 0.5615 | 0.0835 | 0.1454 | 0.7107 |
| mode\_kbest | rf | 0.8044 | 0.5686 | 0.0803 | 0.1408 | 0.7118 |
| mode\_rfe | rf | 0.8044 | 0.5709 | 0.0793 | 0.1393 | 0.7109 |
| mode\_sc | rf | 0.8038 | 0.5490 | 0.0921 | 0.1578 | 0.7098 |

根据上表的结果显示，对于RandomForest模型来说，在使用KNNImputer填充的数据集中，通过相关系数法和评分卡工具筛选特征的结果与未进行特征筛选的结果相同，均为0.7119。在使用众数填充的数据集中，未进行特征筛选的结果最好，为0.7119。

### 4.1.3 XGBoost

XGBoost是一种boosting算法的实现方式，旨在降低模型的偏差，即降低模型预测与实际标签的误差。通过集成多个相对简单的基学习器（称为"弱学习器"）来构建一个强大的集成模型。XGBoost的训练过程是迭代的，每次迭代学习一个新的基学习器，目标是学习前面学习器的预测残差。通过多个学习器的学习，XGBoost逐步降低了模型的残差，使得模型的预测结果更接近实际标签，从而降低模型的误差。

以下是不同处理过后的数据集在XGBoost模型上的具体表现，此处只是简单测试了一下，并没有使用交叉验证方法：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| dataset | model | acc | precision | recall | f1 | auc |
| knn\_no\_other | xgb | 0.8065 | 0.5706 | 0.1226 | 0.2019 | 0.7310 |
| knn\_corr | xgb | 0.8059 | 0.565 | 0.1187 | 0.1961 | 0.7297 |
| knn\_kbest | xgb | 0.8060 | 0.5666 | 0.1177 | 0.1949 | 0.7298 |
| knn\_rfe | xgb | 0.8065 | 0.5699 | 0.1224 | 0.2015 | **0.7311** |
| knn\_sc | xgb | 0.8059 | 0.5650 | 0.1187 | 0.1961 | 0.7297 |
| mode\_no\_other | xgb | 0.8063 | 0.5674 | 0.1228 | 0.2019 | 0.7317 |
| mode\_corr | xgb | 0.8056 | 0.5623 | 0.1166 | 0.1931 | 0.7288 |
| mode\_kbest | xgb | 0.8058 | 0.5618 | 0.1210 | 0.1990 | 0.7312 |
| mode\_rfe | xgb | 0.8063 | 0.5674 | 0.1217 | 0.2004 | **0.7320** |
| mode\_sc | xgb | 0.8051 | 0.5577 | 0.1118 | 0.1862 | 0.7276 |

根据上表的结果显示，对于XGBoost模型来说，在使用KNNImputer填充的数据集中，通过RFE特征消除法筛选特征得到的AUC取值最高，为0.7311。在使用众数填充的数据集中，也是通过RFE特征消除法筛选特征得到的AUC取值最高，为0.7320。

### 4.1.4 LightGBM

LightGBM是一种高效、快速、高性能的梯度提升树算法，主要适用于大规模数据集和高维特征。其特点包括基于直方图的决策树算法、Leaf-wise生长策略、特征并行训练和直接支持类别特征等。通过这些优化策略，LightGBM在训练速度和内存使用上有明显优势，同时提供了灵活的参数调节，平衡了训练速度和模型准确性。

以下是不同处理过后的数据集在LightGBM模型上的具体表现，此处只是简单测试了一下，并没有使用交叉验证方法：

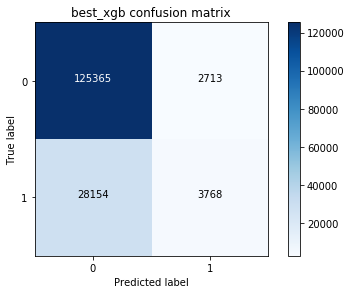
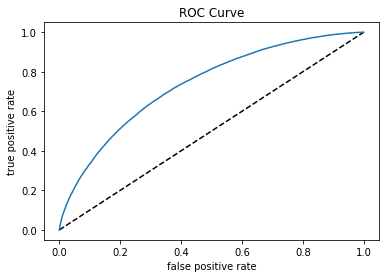
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| dataset | model | acc | precision | recall | f1 | auc |
| knn\_no\_other | lgb | 0.8065 | 0.5858 | 0.1022 | 0.1741 | 0.7315 |
| knn\_corr | lgb | 0.8060 | 0.5792 | 0.1009 | 0.1719 | 0.7303 |
| knn\_kbest | lgb | 0.8062 | 0.5837 | 0.0994 | 0.1699 | 0.7302 |
| knn\_rfe | lgb | 0.8066 | 0.5878 | 0.1018 | 0.1736 | **0.7317** |
| knn\_sc | lgb | 0.8060 | 0.5792 | 0.1009 | 0.1719 | 0.7303 |
| mode\_no\_other | lgb | 0.8067 | 0.5901 | 0.1024 | 0.1745 | **0.7320** |
| mode\_corr | lgb | 0.8062 | 0.5819 | 0.1015 | 0.1729 | 0.7301 |
| mode\_kbest | lgb | 0.8067 | 0.5879 | 0.1038 | 0.1765 | 0.7319 |
| mode\_rfe | lgb | 0.8064 | 0.5836 | 0.1036 | 0.1760 | 0.7318 |
| mode\_sc | lgb | 0.8059 | 0.5784 | 0.1004 | 0.1711 | 0.7288 |

根据上表的结果显示，对于LightGBM模型来说，在使用KNNImputer填充的数据集中，通过RFE消除法筛选特征得到的AUC取值最高，为0.7317。在使用众数填充的数据集中，未进行特征筛选的结果最好，为0.7320，经过筛选特征的结果与未进行筛选结果之间没有显著的区别。

## 4.2 调参

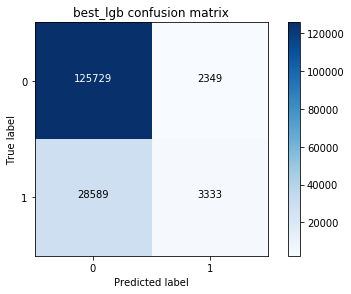
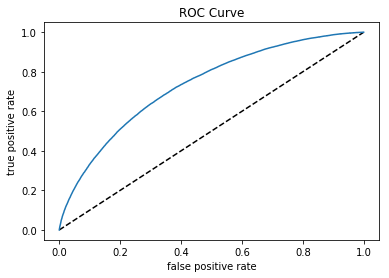
根据第一节的结果分析，这一节选用XGBoost与LightGBM来对进行填充之后使用RFE消除法获得的数据集来进行调参实验，采用GridSearchCV方法。

### 4.2.1 XGBoost

在参数n\_estimators=200, max\_depth=7, learning\_rate=0.1的条件下，XGBoost模型的结果可以达到accuracy=0.8071，precision=0.5814，recall=0.1180，f1-score=0.1962，auc=0.7341。通过调参优化模型后，实验结果显示模型性能有所提升，相比未进行调参的情况，模型的性能变得更好。

### 4.2.2 LightGBM

在参数num\_leaves=40, max\_depth=8, min\_data\_in\_leaf=40, feature\_fraction=1, min\_data\_per\_group=100, max\_cat\_threshold=10, max\_bin=300的条件下，LightGBM模型的结果可以达到accuracy=0.8066，precision=0.5866，recall=0.1044，f1-score=0.1773，auc=0.7319。在进行调参优化模型后，实验结果却显示模型性能有所下降，这可能是因为在调参过程中设置的超参数并没有得到全局最优解，而是陷入了局部最优解的情况。

## 4.3 总结

在本次实验中，对LR模型和RandomForest模型进行了性能评估，并与XGBoost模型和LightGBM模型进行了对比。从实验结果可以看出，LR模型和RF模型的总体效果相对较差，而XGBoost模型和LightGBM模型表现较为出色。XGBoost模型和LightGBM模型作为梯度提升树算法的改进版本，具有高效、快速和高性能的特点，在大规模数据集和高维特征下表现优异。它们在模型训练和预测中都表现出色，能够有效地降低偏差，提高模型的准确性和泛化能力。而LR模型和RF模型虽然也能完成任务，但在本次实验中的表现相对较一般。

调参是一个复杂的过程，涉及到多个超参数的选择，以及不同超参数组合对模型性能的影响。有时候，通过改变某个超参数的值，可能会导致模型在训练集上的性能有所提高，但这并不意味着该超参数组合在测试集或实际应用中表现也会更好。这种情况通常发生在存在多个局部最优解的情况下。在实验中观察到，对于XGBoost模型，经过调参后性能有所提升，这表明在超参数搜索空间中找到了更适合模型的设置，从而使其在测试数据上表现更好。然而，对于LightGBM模型，调参后的结果却出现性能下降的情况。这可能是由于在调参过程中选择的超参数并未达到全局最优解，导致模型在训练数据上过拟合或者陷入了局部最优解。针对这一现象，可以进一步优化调参策略，尝试更广泛的超参数搜索空间，或者考虑使用更优化的调参算法，以期望找到更合适的超参数组合，从而提升LightGBM模型的性能。

在比较了两种特征填充方式（KNNImputer和众数填充方法）后，发现它们之间的差别并不明显。这可能是由于原始训练集中的缺失值本来就不多，因此无论使用哪种方法，都能较好地填补缺失值，使得模型在训练过程中能够更好地学习到数据的规律和特征。尽管两种填充方式在某些指标上有细微差别，但整体上模型的性能并没有显著变化。

总体来看，实验结果显示了XGBoost模型和LightGBM模型相对其他模型的优越性，并且在这个特定的数据集情况下，填充方式对模型性能的影响相对较小。这些发现强调了根据具体数据集特点和任务需求来选择适当的模型和填充技术的重要性。因此，在实际应用中，应该综合考虑数据集的特征、大小以及任务目标，选择最合适的模型和填充策略，以取得更好的性能。