西南财经大学“新网银行杯”数据科学竞赛

论文题目

基于统计学习的违约概率预测模型研究

队名：_____菜_____

成员：____徐昊成_____

____周建凯_____

____王珞情_____

____刘思逸_____

名次：_____10_____

2018 年 10 月 28 日

摘 要

商业信贷体制随经济形势的不断发展，信用风险是商业银行所面临的主要风险。信用卡作为商业银行业务中的一种简单而普及的业务，如何评估从而有效地降低这类风险成为了越来越受到商业银行所关注的问题，随着高维数据的出现，我们所得到的往往只是一些其他方面的信息，因此如何从这些看似杂乱的数据中挑选出有意义的数据，并发现其中的规律成为了我们这篇文章所研究的内容。

本文在对原始数据做了分析的基础上，进行了部分的数据预处理与特征工程，概括性的介绍了 XGBoast、LightGBM、二值型变量进行回归所常用的 Logistic 回归、以及半监督的背景和理论，并结合客户信用风险预测的具体特点，在含 y 标签的训练集进行有监督学习模型的训练和不含 y 标签上的半监督模型的训练。在有监督学习模型的训练过程中采用了 GBDT+LR 模型、XGBoast 模型、LightGBM 模型进行训练，通过对三种模型 AUC 验证的比较，得到了 LightGBM 作为最好的模型及其相关参数。用该模型进行了违约概率预测，在 B 榜的 AUC 评分为 0.75467。最后本文对模型最后得到的结果进行了评价和总结，并提出了相应建议。

关键词：XGBoast 算法；LightGBM 算法；半监督模型；概率预测

目 录

第一章 引言

1.1 研究背景与意义

《巴塞尔新资本协议》将商业银行面临的主要风险定义为信用风险、市场风险以及操作风险。其中，信用风险是最主要和最复杂的风险。《巴塞尔新资本协议》对信用风险的计量提出了标准法和内部评级法，其指出有条件的银行要实施内部评级法，通过对历史数据构建模型测算客户的违约概率[1]。信用风险（CreditRisk）又称违约风险，是指债务人或交易对手无力履行合同所规定的义务或信用质量发生变化，影响金融产品价值，从而给债权人或金融产品持有人造成经济损失的风险[2]。

违约概率是影响信用风险的关键因素之一，准确地预测客户的违约概率是信用风险计量的基础。如何运用统计和机器学习模型对客户信用风险进行预测是金融机构风险管理关注的重要问题，如在客户贷款申请审批场景中，金融机构主要依据客户的信用评分做出是否准入、额度大小、利率高低等决策，在风险量化实践中还将面临如何处理和运用高维稀疏数据、如何充分利用无标签数据、如何将样本量充足的产品上的风控模型学习经验迁移到小样本或坏样本少的产品上等问题。

1.2 研究现状

商业银行客户违约概率的研究一直是国际、国内研究的热门领域。20 世纪 60 年代后，随着信用卡业务的推出，基于信用评分技术的客户违约概率研究取得了较大的发展。信用评分模型，利用可观察到的借款人的特征变量计算出一个数值（得分）来代表借款人的信用风险。特征变量的选择、根据历史数据的回归分析结果确定各自的权重是模型构建的关键。

国际上常用的建模方法主要包括判别分析、逻辑回归、神经网络等。判别分析模型是通过对一系列变量的分析，建立判别函数，使其能最好地区分违约客户和不违约客户。1966 年，Beaver 指出可以通过财务指标来预测公司的破产概率，并用单变量分析法对 79 家破产公司做了实证研究[3]。1968 年，Altman 提出了以 5 个财务比率为判别变量的线性判别分析模型——Z-score 模型[4]。1977 年，Altman 等对原始的 Z 计分模型进行扩展，建立的第二代的 ZETA 信用风险模型[5]。1995 年，对于非上市公司，Altman 对 Z 模型进行了修改，得到 Z’计分模型

[6]。

2002 年，Shi 等提出了多目标规划模型，并将其应用到信用卡组合管理中，取得了良好的效果[7]。

逻辑回归模型假设违约概率服从逻辑分布（累积违约概率采用逻辑函数形式，其值落在 0 和 1 之间），结合企业的财务状况计算出企业的违约概率。1977 年，Martin 利用 logit 和判别分析对 1975 至 1976 年间的银行进行财务风险预测[8]。1980 年，Ohlson 选取了 1970-1976 年间的 105 个工业企业进行风险预测[9]。1991 年，Platt 等使用 logit 模型检验得出，使用与行业相关的财务比率要比使用企业特有的比率得到更好的企业破产预测[10]。1992 年，Lawrence 等利用 logit 模型对活动住房贷款的违约概率进行了预测，结果发现支付历史是最重要的预测指标[11]。

神经网络模型与判别分析相似，但是没有了判别函数的变量是线性且相互独立的假设，是一种非线性的违约预测函数。Coats 等和 Zhang 等分别用神经网络模型对美国公司的财务危机和破产风险进行了预测[12][13]。神经网络模型的映射能力和泛化能力较强，但其解释能力比较差，存在过度拟合的问题。

国内对商业银行信用风险的研究主要集中在宏观层面和制度层面。《巴塞尔新资本协议》出台后对违约概率等风险要素的关注逐步加强。研究主要是在一些商业银行的部分贷款数据上运用信用评分模型，得到预测、分类的准确率，缺乏对影响客户违约概率因素的系统性分析。王春峰等先后将判别分析法、统计方法与神经网络技术相结合的组合预测方法[15]、改进蚁群算法[16]结合商业银行的实际案例进行了实证研究[14]。张维等将递归分类树方法用于商业银行信用的风险分析中[17]。方洪全等以 1333 笔实际贷款记录建立数据方，运用联机分析挖掘（on-lineanalyticalmining，OLAM），建立起两阶段信用风险评估体系[18]。徐佳娜等将人工神经网络信用风险评估技术与层次分析法相结合，建立了商业银行信用风险评估 AHP-ANN 模型。马晓君、沙靖岚、牛雪琪基于美国 P2P 平台 Lending Club 的海量真实交易数据，采用 “多观测”与 “多维度”两种数据清洗方式，运用 2016 年微软亚洲研究院提出的机器学习算法 LightGBM，兼顾权威性和创新性地对平台内贷款项目的违约风险进行预测[19]。沙靖岚将 LightGBM 算法与 XGBoost 算法运用到 P2P 网络借贷违约预测模型中，并对影响违约结果的因素进行排序与分析[20]。

1.3 本文研究目的及思路

本文主要对客户信用风险预测的方法进行了研究，并且使用 AUC 来评估模型对用户违约概率预测的准确程度。AUC 即以 False Positive Rate 为横轴，True

Positive Rate 为纵轴的 ROC（Receiver Operating Characteristic）curve 下方的面积的大小

$$AUC = \frac{\sum_i S_i}{M \times N}$$

M 为正样本个数，N 为负样本个数，M * N 为正负样本对的个数。S_i 为第 i 个正负样本对的得分，定义如下：

$$S_i = \begin{cases} 1 & score_{i-p} > score_{i-n} \\ 0.5 & score_{i-p} = score_{i-n} \\ 0 & score_{i-p} < score_{i-n} \end{cases}$$

$score_{i-p}$ 为正负样本对中模型给正样本的评分，$score_{i-n}$ 为正负样本对中模型对负样本的评分。AUC 值在[0,1]区间，越高越好。

本文的思路如图 1.1，首先，我们对原始数据做了数据可视化分析，得到了数据的分布信息。在分析的基础上，进行了部分的数据预处理与特征工程的构建。接下来对机器学习的相关理论进行了完整的阐述，并结合客户信用风险预测的具体特点，利用了包含 y 标签的训练集进行有监督学习模型的训练和半监督模型进行训练。在有监督学习模型的训练过程中采用了 GBDT+LR 模型、XGBoast 模型、LightGBM 模型进行训练。在训练的结果上对于原始数据进行预处理和特征工程的构建进一步的优化。

图 1.1 本文的思路

在有监督模型训练得到一定的成果后，尝试引入无标签的训练集进行半监督学习的训练，并进行了迁移验证。最后我们得到了本问题的解决方案以及可以改

进的方向。

1.4 本文的组织结构

本文的结构安排如下：

第一章为引言，主要介绍用户信用风险预测的研究背景和研究现状，说明了在风险量化实践中目前存在的问题，最后简单介绍了本文的研究内容。

第二章是数据可视化的分析。通过对原始数据的可视化分析，提出后续的

第三章是数据预处理与特征工程的处理。数据预处理阶段主要涉及到数值型与类别型变量的处理和缺失值的处理，特征工程则包括缺失信息特征与特征组合的构建与筛选。

第四章是模型原理的介绍。介绍了本文涉及到的 XGBoast、GBDT+LR、LightGBM 和半监督学习模型的原理。

第五章是模型求解的结果。介绍了 XGBoast、GBDT+LR、LightGBM 和半监督学习模型的求解思路，超参数设置与相应的求解结果。

第六章总结了全文的工作，分析了本文研究工作的不足之处以及未来可能的改进方向。

第二章 数据的可视化分析

探索性数据分析（Exploratory data analysis，简写为 EDA）是指对已有的数据（特别是调查或观察得来的原始数据）在尽量少的先验假定下进行探索，通过作图、制表、方程拟合、计算特征量等手段探索数据的结构和规律，从而形成值得假设的检验的一种数据分析方法。是对传统统计学假设检验手段的补充。

2.1 数据标签分布

根据赛题，0 代表低风险客户，1 代表高风险客户。我们可以做图观察标签的分布情况。如图 2.1 所示

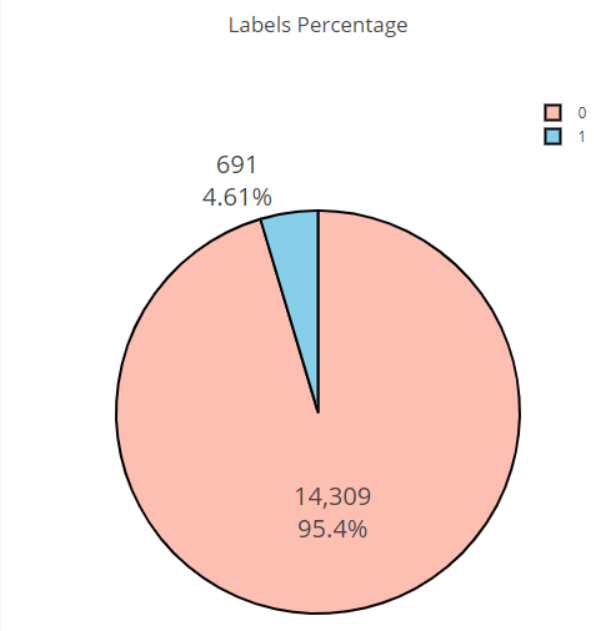

图 2.1 低/高风险客户标签分布情况

由上图可以看出，这是一个样本不平衡问题，低风险客户远远多于高风险客户。

t-SNE（t－分布随机近邻嵌入）与 PCA（主成分分析）是常用的数据可视化工具，我们可以通过这两种方法来观察数据的二维分布，来对接下来的模型求解带来启示。

（1）有标签数据分布：首先作图观察有标签数据在二维空间的分布情况，初步观察两种类别数据分布情况的差异

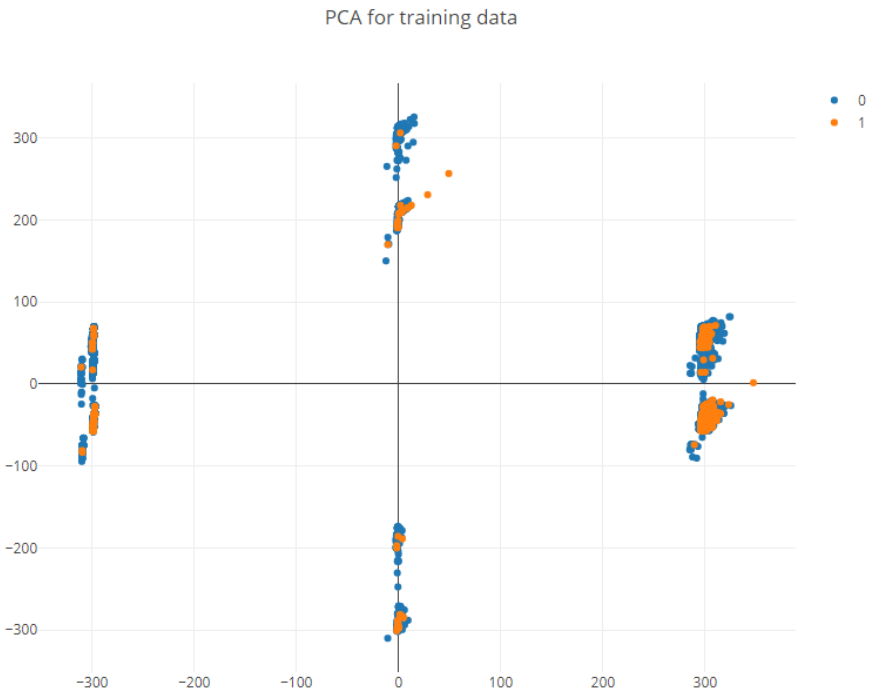

图 2.2 通过 PCA 得到的数据分布图

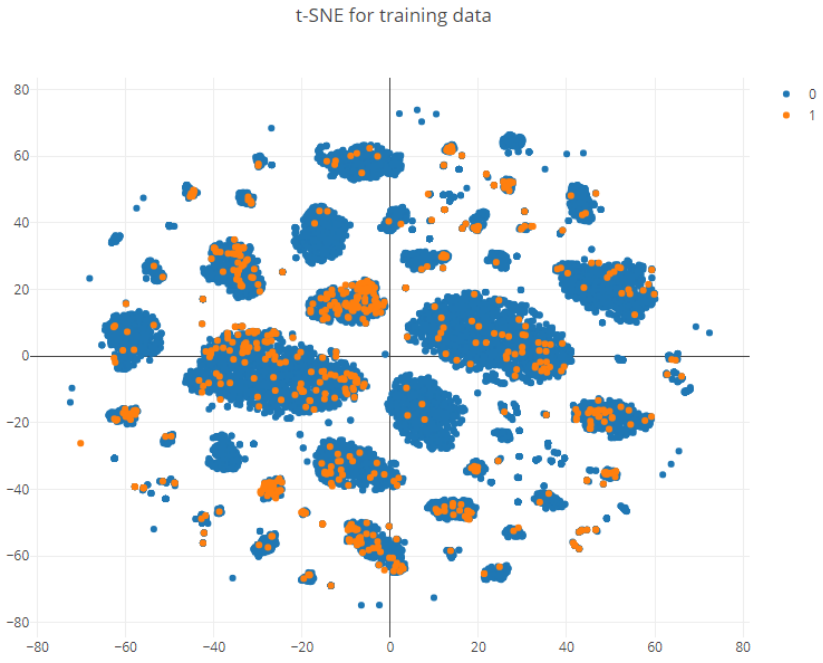

图 2.3 通过 t-SNE 得到的数据分布图

由图看出，有部分集群中高风险客户占比远远高于其他集群中高风险客户占比。

（2）无标签数据与有标签数据在二维空间中的分布：可以通过做出无标签数据与有标签数据在二维空间中的分布，来观察两种不同的数据之间的差异。

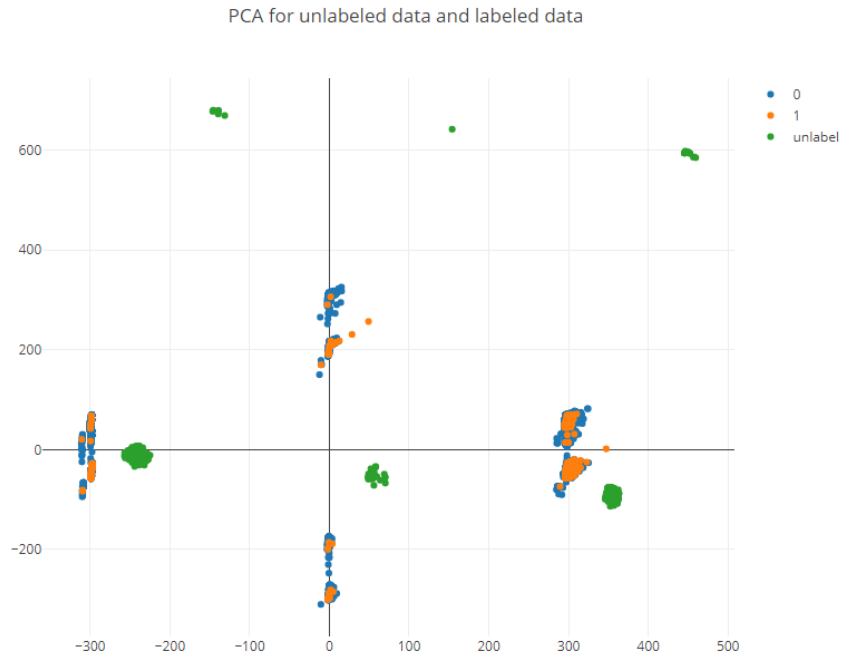

图 2.4 两种数据由 PCA 所得分布图

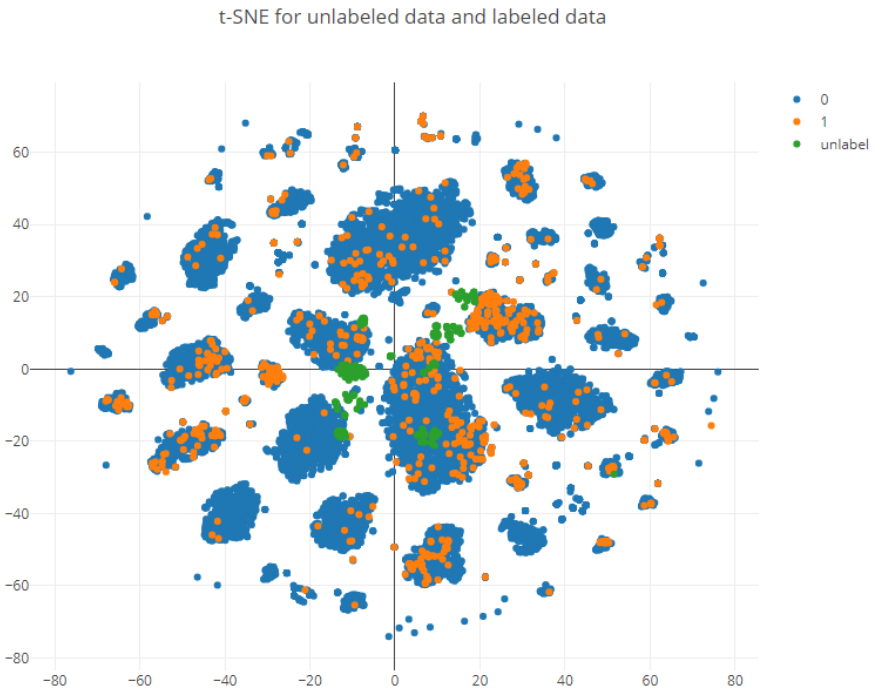

图 2.5 两种数据由 t-SNE 所得分布图

由图可以看出，无标签数据与有标签数据之间分布存在较大差异，故将无标签数据纳入模型可能会对模型产生负面影响，这在后面的半监督模型的运用中得到了验证。

（3）不同组的数据在二维空间中的分布：可以通过作图来观察不同的组之间的空间分布来得出他们之间的区别与联系，来决定是否采用分组建模的方法

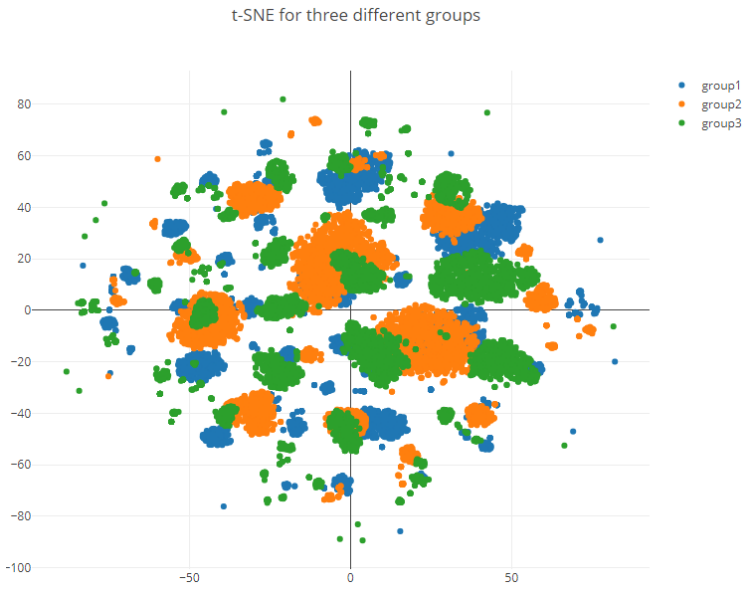

图 2..6 不同组数据分布图

由此图我们可以看出，三个组之间分布并没有太大的差异，所以我们初步判断分组求解效果可能不是很好，后面的实验也验证了我们的猜想。

2.2 特征信息的分布统计

2.2.1 特征的简单描述统计

我们首先对每种特征统计了其平均数，标准差，极值，25%、50%、75%分位数以及每个特征缺失总数及比例。接下来我们对特征缺失比例、特征种类分布以及类别特征类别数量做了简单的分析，结果如下图所示：

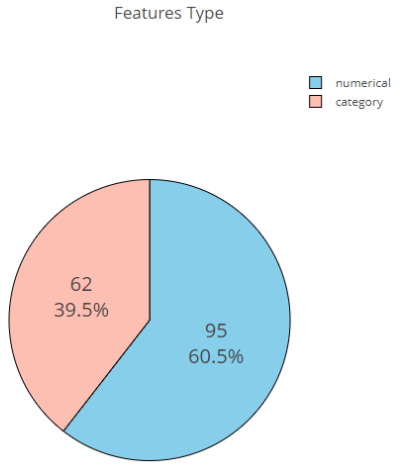

图 2.7 特征种类分布

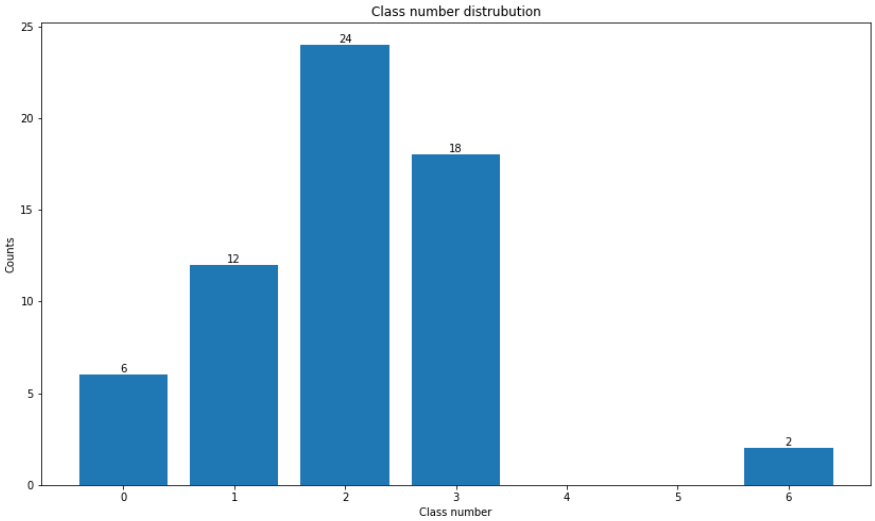

图 2.8 类别特征类别数量

从图 2.7 可以得出，数值型变量占大多数。

根据图 2.8，我们可以在后面的处理中去除类别数量为 0 和 1 的特征，或者将类别数量为 1 且含有缺失值的特征处理成两类别特征。

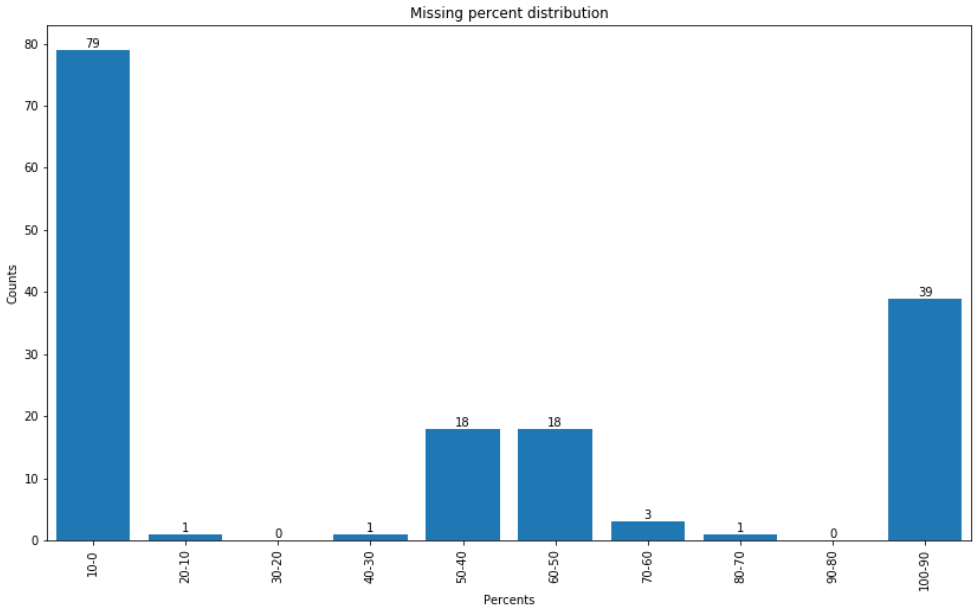

图 2.9 特征缺失比例分布

由图 2.9 可以得出有 39 个特征有极高的缺失比率，在接下来的处理中我们可以将其除去。

2.2.2 特征间的相关性统计

对于某些数据特征，如果其数值无缺失值且其值一致，也就是方差非常低。

我们通常认为低方差变量携带的信息量也很少，所以可以把它直接删除。

如果两个变量之间是高度相关的，这意味着它们具有相似的趋势并且可能携带类似的信息。同理，这类变量的存在会降低某些模型的性能（例如线性回归和逻辑斯蒂回归）。为了解决这个问题，我们可以计算独立数值变量之间的相关性。如果相关系数超过某个阈值，就删除其中一个变量。这里通过研究特征之间的皮尔森相关系数来评估特征质量，如图 2.10 所示

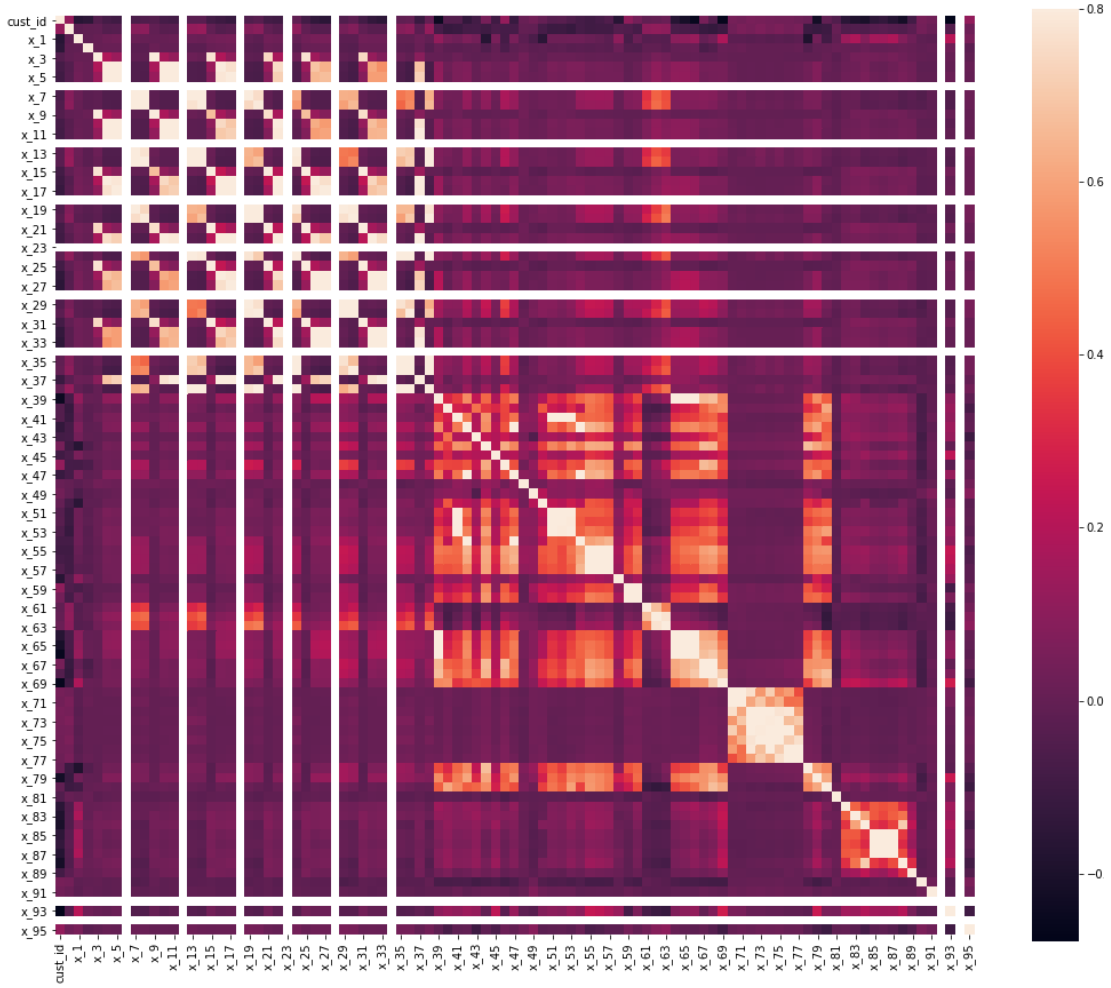

图 2.10 特征的相关性统计

由上图可以得出，大量特征之间存在极高的相关性（ρ>0.9），我们可以除去冗余特征，以便接下来的分析，并且提高模型的稳定性以及运行速度。

同时根据分析，特征中有 6 个特征值为常数，这些单一值特征对模型求解没有帮助，故可以去除 x_110，x_112， x_116， x_129， x_132， x_134。

在除去高缺失率，高相关性以及单一值特征以后，我们将特征减少到 91 个，在不影响模型性能的同时，简化特征工程的工作量，提高模型运行效率。

2.3 特征与标签的相关性统计

由于处理后的数据维度依然很高，对每个特征做详细的分析工作量大且收益有限，所以我们可以通过分析特征与目标的相关系数，选出几个相关性高的特征来做更详细的分析，我们选出来了相关系数绝对值排名前四的特征来做下一步的分析，如图 2.11

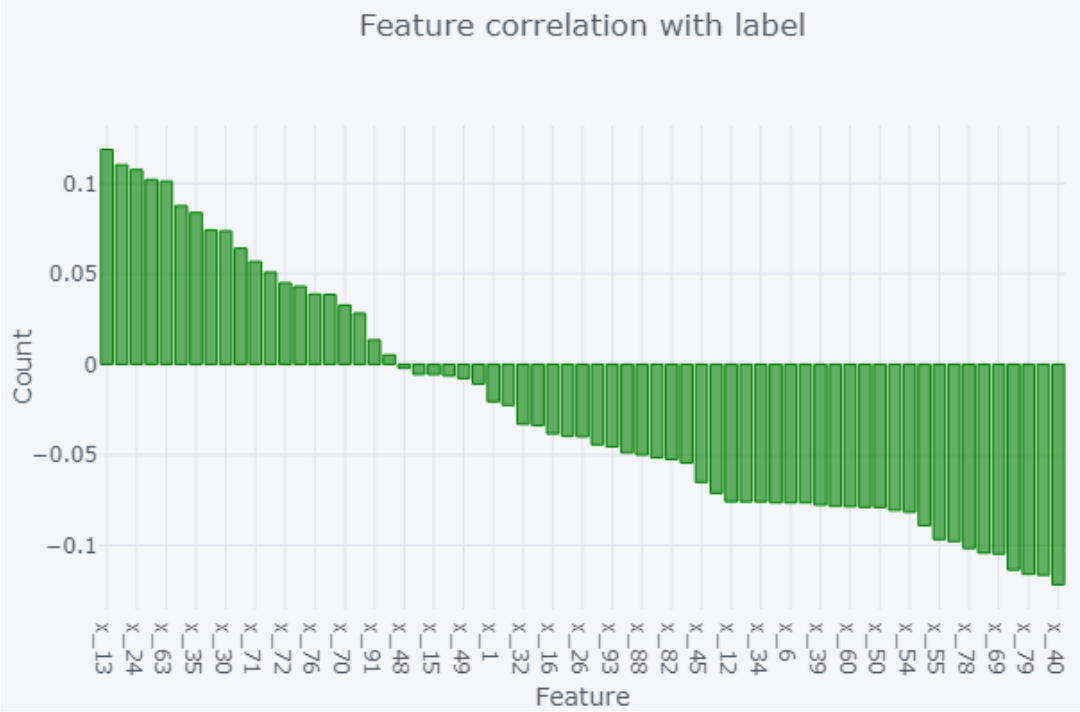

图 2.11 部分特征与标签之间的相关关系

由表我们可以看出，虽然特征与标签之间相关性不是很高（0.12-0.11），但这些特征可能对接下来的分析有帮助，因为这些特征的确对标签产生了影响。

之后作图观察选出来的四个特征与标签之间的相关系数，如图 2.7

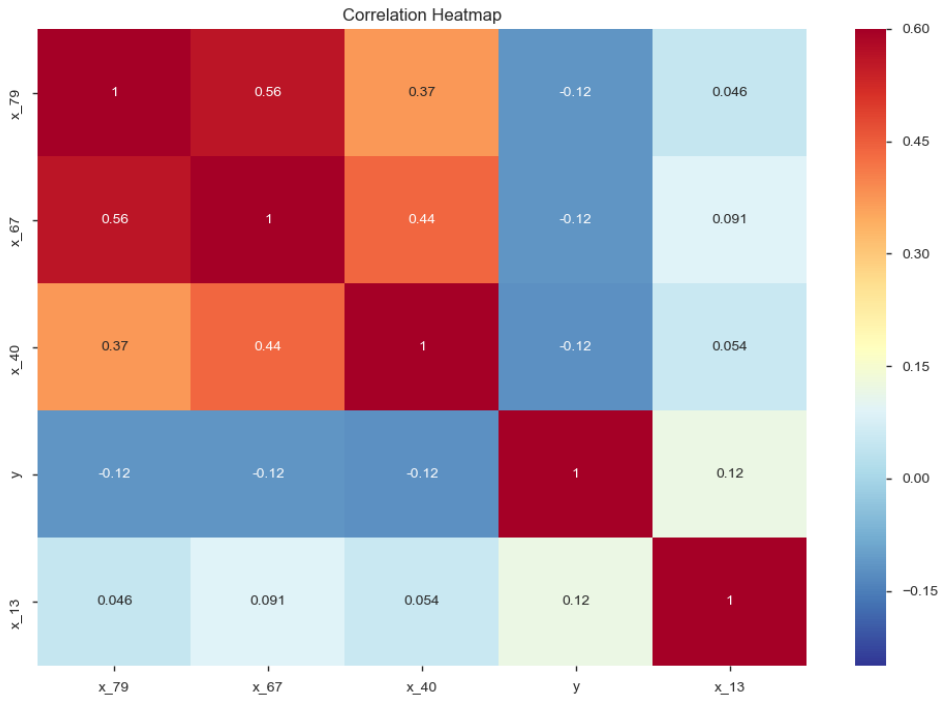

图 2.12 特征与标签之间的相关系数热力图

由图我们可以看出，x_79， x_67， x_40 三个特征之间依然存在较高的相关性。四个变量的统计分布，如表 2.1 所示

	x_79	x_67	x_40	x_13
count	14872	14872	14872	7655
mean	3.810987	5.576318	4.405594	0.911822
std	2.494856	3.695921	2.873027	1.51861
min	0	0	0	0
25%	2	1	2	0
50%	3	6	4	0
75%	6	10	7	1
max	9	10	10	52
Missing_value_count	128	128	128	7345
Missing_Percent	0.008533	0.008533	0.008533	0.489667
correlativa	0.115943	-0.11643	-0.12176	0.118604
Kurtosis	-0.81102	-1.6487	-0.9764	221.87
Skewness	0.62789	-0.073376	0.54674	8.7843

表 2.1　四个变量的统计分布

由表 2.1 得出，x_79，x_67，x_40 三个特征统计分布之间没有太大的差别，这与之前相关系数热力图所反应的结果类似。

最后我们作图研究一下变量与变量之间，变量与标签之间的关系图，如图 2.8 所示

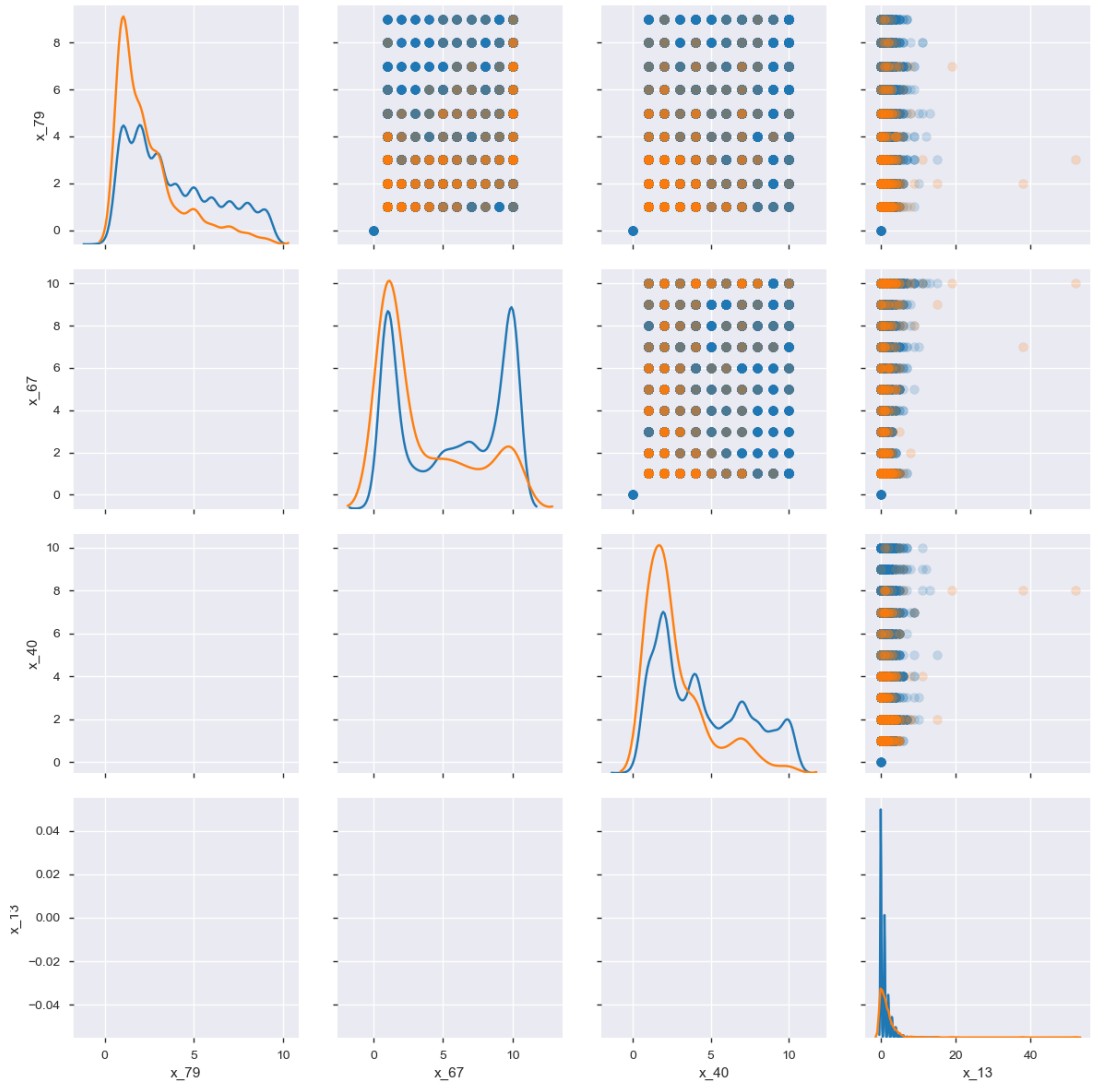

图 2.13 特征核密度估计图以及散点图

其中，橙色代表高风险客户，蓝色代表低风险客户，上三角形区域为特征之间的散点图，对角线区域为特征在不同标签下的核密度估计图。由图可见，不同标签的核密度估计图存在较为明显的差别，散点图也存在较为明显的区域分布，这证明了这四个特征的有效性。在接下来的特征工程中我们可以针对这四个特征做出更细致的分析。

第三章 数据预处理与特征工程

3.1 数据预处理

根据赛题任务，本次竞赛提供的数据包括用户 id，157 项脱敏的属性/行为特征，以及是否属高风险用户的标签项。特征变量名称以“x_”开头，其中，特征变量 x_1-x_95 是数值型变量，x_96-x_157 是类别型变量，x 变量的缺失值统一以-99 表示。但在前期数据可视化的过程中，发现，在原有特征的分类中，数值型变量大多是经过离散化处理的，仅有 6 个变（x_1,x_2,x_80,x_81,x_93,x_95）经过标准化处理，取值为位于[0,1]的连续性变量。具体的变量分布见表 3.1

表 3.1 每个特征的取值范围

变量名（数值）	取值范围	变量名（类别）	取值范围
x_1	[0,0.9]	x_96	1,2
x_2	[0,0.99997]	x_97	1,2,3,-99
x_3	0,1,2,5,-99,	x_98	1,2,3,-99
x_4	int[0,6],-99	x_99	1,2,-99
x_5	0,1,2,3,-99	x_100	1,2,-99
x_6	0,-99	x_101	1,2,-99
x_7	int[0,10],-99	x_102	2,-99
x_8	int[0,9],-99	x_103	0,1,-99
x_9	0,1,2,3,5,-99	x_104	0,1,2,-99
x_10	0,1,2,3,4,5,6,-99	x_105	0,1,2,-99
x_11	0,1,2,3,4,-99	x_106	1,2,-99
x_12	0,-99	x_107	0,-99
x_13	int[0,13],15,19,38,52,-99	x_108	0,1,2,-99
x_14	int[0,11],14,22,31,-99	x_109	1,2,-99
x_15	int[0,4],8,-99	x_110	-99
x_16	int[0,6],8,-99	x_111	0,1,2,-99
x_17	int[0,4],-99	x_112	-99
x_18	0,-99	x_113	0,-99
x_19	int[0,15],-99	x_114	2,-99
x_20	int[0,10],12,-99	x_115	1,2,-99
x_21	int[0,6],-99	x_116	-99
x_22	int[0,6],-99	x_117	1,2,-99
x_23	0,-99	x_118	0,-99
x_24	int[0,16],18,20,21,27,31,32,65,71,-99	x_119	0,1,-99
x_25	0,1,2,3,4,8,-99	x_120	1,2,-99
x_26	int[0,8],-99	x_121	0,1,2,-99
x_27	0,1,2,3,4,-99	x_122	0,1,2,-99
x_28	0,-99	x_123	0,2,-99
x_29	int[0,20],22,24,25,-99	x_124	1,2,-99
x_30	int[0,16],-99	x_125	0,2,-99
x_31	int[0,6],-99	x_126	1,-99
x_32	int[0,6],8,9,-99	x_127	0,1,2,-99
x_33	int[0,5],-99	x_128	1,2,-99
x_34	0,-99	x_129	-99

x_35		x_130	0,1,-99
x_36	int[0,15],17,18,19,22,24,28,33,34,40	x_131	0,-99
变量名（数值）	取值范围	变量名（类别）	取值范围
x_37	int[0,4],-99	x_132	-99
x_38	int[0,12],14,15,18,21,33,40,-99	x_133	2,-99
x_39-x_40	int[0,10],-99	x_134	-99
x_41-x_42	int[0,15],-99	x_135	2,-99
x_43-x_46	int[0,10],-99	x_136	1,2,-99
x_47	int[0,15],-99	x_137	2,-99
x_48	int[0,10],-99	x_138	1,-99
x_49	int[0,4],-99	x_139	1,2,-99
x_50	int[0,10],-99	x_140	1,2,-99
x_51-x_57	int[0,15],-99	x_141	int[1,6],-99
x_58-x_60	int[0,10],-99	x_142	1,2,-99
x_61-x_63	int[0,5],-99	x_143	1,2,-99
x_64-x_69	int[0,10],-99	x_144	1,2,3,-99
x_70	int[0,2],4,5,6,9,-99	x_145	1,2,-99
x_71	int[0,7],-99	x_146	1,2,-99
x_72	int[0,6],9,10,-99	x_147	1,2,4,-99
x_73	int[0,7],-99	x_148	1,2,3,-99
x_74	int[0,7],9,10-99	x_149	1,3,4,-99
x_75	int[0,8],-99	x_150	1,2,4,-99
x_76	int[0,7],9,10-99	x_151	1,-99
x_77	int[0,8],10,-99	x_152	1,2,-99
x_78	int[1,8],-99	x_153	1,2,3,-99
x_79	int[0,9],-99	x_154	1,2,3,-99
x_80	0,[0.645,1]	x_155	1,2,3,-99
x_81	[0.501,1],-99	x_156	1,2,3,-99
x_82	1,3,4,5,-99	x_157	1,2,3,4,10,11-99
x_83	int[1,10],-99		
x_84	int[1,16],-99		
x_85	1,3,4,5,6,-99		
x_86	int[1,10],-99		
x_87	int[1,6],-99		
x_88	int[1,10],-99		
x_89	int[1,15],17,-99		
x_90	int[1,3],-99		
x_91	1,2,5,-99		
x_92	0,-99		
x_93	[0,1],-99		
x_94	0,-99		
x_95	[0,1],-99		

由于考虑到数值型变量与类别型变量在树模型划分类别的时候的效果并不相同，对于连续性无意义的数值型变量来说，当作类别型变量处理可以有效地降低树模型分类的难度，就以年龄为例，年龄越大和越小可能对违约的影响并不大，那这个时候对年龄划分区间离散化后作为类别变量可能会有新的效果。

由于所有的特征都经过脱敏处理，无法判断被离散化的数值型变量究竟有何意义，所以在后续的模型处理中，我们尝试了两种数值型变量与连续型变量的划分方式：

方式一：维持赛题原有的划分方式不变，依旧是特征变量 x1-x95 是数值型变量，x96-x157 是类别型变量

方式二：将 x_1 ， x_2 ， x_80 ， x_81 ， x_93 ， x_95 这 6 个特征认定为数值型变量，剩下的特征都划分为类别型变量

对于现有的数据来说，大量特征存在缺失值，在应用到树模型中可能问题不大，但如果采用逻辑回归模型，或 SVM 模型等则需要对缺失值进行处理才能运行模型。但实际上，在树模型内添加缺失信息特征之后，树模型在线上的结果并没有提升，所以尝试通过缺失值处理的方法来提升树模型的结果。所以本文在此先讨论了多种进行缺失值处理的方法。

缺失值处理技术原理：缺失值常见的方法主要有缺失值的删除和缺失值的插补两种方式。

删除方法是指简单地删除缺失的样本，因为简单性，这种方法被广泛使用，许多统计包都默认使用这种方法去掉缺失值，但是除非数据缺失模式是完全随机缺失数据（MCAR），否则这种方法可能会导致数据的偏差。因此，这种方法主要运用于处理缺失数据非常少的情况。

插补方法是一类方法，它是对缺失值进行一些合理的猜测，用猜测值进行填补后得到完整的数据，然后在完整的数据集上做进一步的研究。常见的缺失值插补方法包含均值插补法、众数插补法、回归插补法、随机回归插补法、knn 插补法等多种方法。

均值插补与众数插补：其中均值插补方法采用特征的非缺失的样本均值来进行插补。众数插补方法采用特征的非缺失的样本众数来进行插补。但是当数据是 MAR 或者 MNAR 时，得到的结果是有偏的，并且因为所有的缺失值都是用的同样的数据填补，这样就低估了该变量的差异程度，所以无论哪种情况下都会低估方差和协方差。

Knn 插补：会找 k 个与缺失样本最相似的完整样本集 sin(k)，其距离衡量标准也一般采用马式距离。该方法能够处理混合类型的数据缺失状况，离散型变量缺失项取 sin(k) 的众数，连续型变量取 sin(k)距离倒数的加权平均值。k-NN 填补是一种非参数处理方法，不需要对各缺失项单独建立模型，思想比较简单，容易操作，但是在取 sin(k) 的过程中遍历实例空间，面对大数据集计算复杂，易产生维数灾难，且计算结果受 k 的取值以及距离度量标准的影响。

对于有标签的训练集，我们尝试了以上多种方法，但在模型的现象表现差异并不明显。

3.2 特征构建与筛选

3.2.1 特征构建

在风控领域中，客户的信息缺失往往与最后的违约情况有关。虽然关于所有特征的信息已经经过了数据脱敏处理，我们不能再知晓数字到底对应了什么经济意义，但信息的缺失可能代表着客户对于该特征的信息填写的意愿程度较低或者无法获取到客户某方面的信息。

所以，本文对客户的特征缺失值做统计。对缺失信息特征的构建主要包含三个板块，分别是用户每个特征是否缺失，客户缺失特征的总个数以及客户缺失特征占总特征的比率。其中用户每个特征是否缺失相应的保留了原数据缺失值的信息，缺失信息构造特征如表 3.2 所示

$$ii = \begin{cases} 1 & x_i\text{不为 nan} \\ 0 & x_i\text{为 nan} \end{cases}$$

$$ic = \sum_{i=1}^{157} i_i$$

$$ir = ic/157$$

表 3.2 每个特征的取值范围

特征名	特征意义
is_x_1	x_1 特征是否缺失
is_x_2	x_2 特征是否缺失
is_x_3	x_3 特征是否缺失
.....	.....
is_x_157	X_157 特征是否缺失
nan_count	用户缺失特征总个数
nan_ratio	用户特征缺失占比

由于数据的特征字段已进行匿名操作，难以从业务上构造特征，于是采取暴力特征组合的方式，通过特征间的加减乘除，以及自己的平方和开方运算构建特征。通过特征之间的相互组合形成新的特征来改善机器学习算法的学习能力。

$$P_{ij} = x_i + x_j \quad i >= j$$

$$s_{ij} = x_i - x_j$$

$$m_{ij} = x_i + x_j$$

$$d_{ij} = xi/xj$$

$$qi = xi \text{ ^2}$$

ei = xi^0.5

3.2.2 特征筛选

IV 值筛选。当在用决策树，逻辑回归等模型方法构建分类模型时，总是需要对自变量进行挑选处理。挑选入模变量过程是个比较复杂的过程，我们需要考虑较多的因素，比如：变量的结果预测能力，变量之间的相关性强弱，变量是否容易生成和使用，变量意义的可解释性等等。但是，其中最主要和最直接的衡量标准是变量的预测能力。一变量的预测能力‖是一个很笼统的说法，在衡量时我们需要一些具体的量化指标来衡量其中每一个自变量的预测能力，并通过比较这些量化指标数值的大小，来确定可以进入模型的变量。而 IV 就是这样一种可以用来衡量自变量的预测能力的指标。因此，在本次模型的建立中，我们也尝试了计算 IV 值。而 IV 的计算又是以 WOE 为基础的。

经过特征工程后，数据维数过高，往往会大大增加计算复杂度并且在小数据集中，十分容易导致过拟合的问题。为了减少变量的个数，同时使得变化趋势尽量平滑再计算每个变量属性的 WOE(weight of evidence)，公式为：

$$WOE = \ln\left(\frac{py_i}{pn_i}\right) = \ln\left(\frac{{}^{\#y_i}/_{\#y_T}}{{}^{\#n_i}/_{\#n_T}}\right) = \ln\left(\frac{{}^{\#y_i}/_{\#n_i}}{{}^{\#y_T}/_{\#n_T}}\right)$$

其中，py_i是这个组中响应客户（在本题的风险模型中，对应的就是违约客户，代表的是模型中预测变量取值为1的个体）占所有样本中所有响应客户的比例，pn_i则是这个组中未响应客户占样本中所有未响应客户的比例，p_i是这个组中违约客户的数量，n_i是这个组中未违约客户的数量，#yT是样本中所有违约客户的数量，nT是样本中所有未违约客户的数量。

在我们的计算过程中，WOE 表示的是违约与未违约的比值的差异。这个差异是通过这两个比值的比值，再取对数来得出的。WOE 越大，这种差异越大，WOE 越小，差异越小，就反映出这个分组里的样本响应的可能性就越小。进一步我们运用公式计算出IV 值：

$$IV_i = (py_i - pn_i) \times WOE_I$$

在计算出IV 值之后，我们将其排序，并绘制出特征重要性图，如图3.1

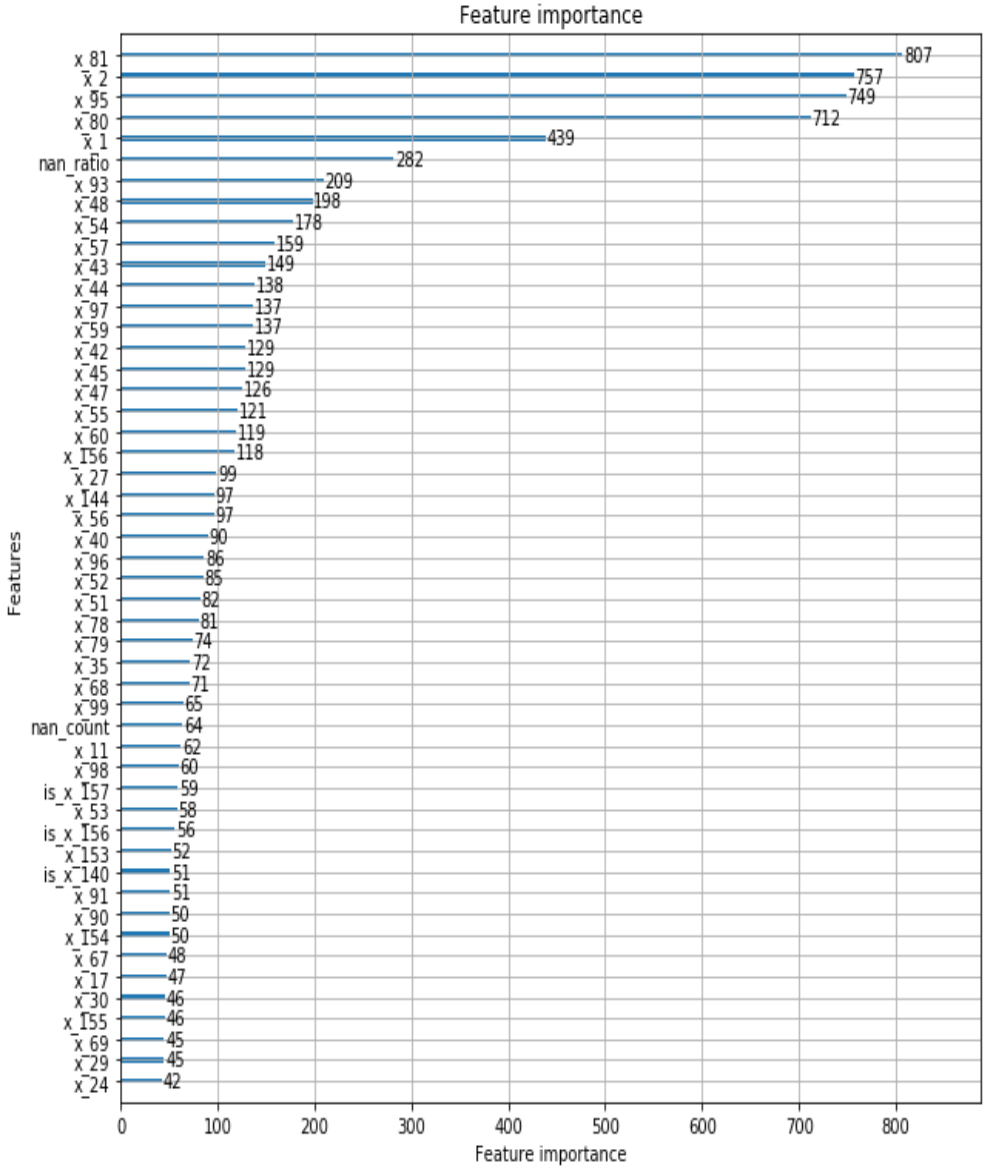

图 3.1 特征重要性统计图

考虑到本题数据较小，且数据分布不稳定，故也可采用贪心法进行特征选择。在数据原特征的基础上，每次加入一个新特征计算整体的 auc 值，不断更新最优值，算法框架如下：

```
If    new_score_with_new_feature< the_last_best:
            New_feature.pop()
Else
            The_last_best = new_score_with_new_feature
```

显然，这种选择方法是具有一定的盲目性的，贪心法陷入的是局部最优解，可能该组特征向量只是近似最优解，故可以考虑引入模拟退火机制，Random 一个数满足某个条件则改变最优值。

第四章 模型原理

模型、参数和目标函数是有监督机器学习的主要逻辑组成部分。模型即为给定输入量x_i如何去预测输出量y_i。模型和参数代表了给定输入后如何进行预测，但是需要目标函数来确定如何进行参数的选择。一般的目标函数包含以下两项：误差函数$L(\Theta)$与正则化项$\Omega(\Theta)$：$Obj(\Theta) = L(\Theta) + \Omega(\Theta)$。误差函数$L(\Theta) = \sum_i l(y_i, \hat{y}_i)$用于考虑模型对于数据的拟合程度，误差越小代表拟合度越好。常见的误差度量方法有残差平方和(SSR)以及 Logistic 残差等。本文采用 Logistic 残差作为误差函数，形式如下：

$$l(y_i, \hat{y}_i) = y_i \ln[1 + \exp(-\hat{y}_i)] + (1 - y_i)\ln[1 + \exp(\hat{y}_i)]$$

同时本文选择以 AUC 为拟合结果的指标，可以在 R 语言中直接输出 AUC。正则化项$\Omega(\Theta) = \lambda\|w\|_p$,则用于衡量模型的复杂程度。复杂程度越低，模型预测的方差越小，预测结果越稳定。

本文中用到了多个模型进行求解，这里对模型的原理作出了详细的表述。

4.1 梯度提升树原理

在分类问题中，提升方法（boosting）通过改变训练样本的权重（增加分错样本的权重，减小分对样本的的权重），学习多个分类器，并将这些分类器线性组合，提高分类器性能。boosting 数学表示为：

$$f(x) = \omega_0 + \sum_{m=1}^{M} \omega_m \phi_m(x) \qquad (4.1)$$

其中ω是权重，ϕ是弱分类器的集合，可以看出最终就是基函数的线性组合。提升树模型可以表示为决策树的加法模型，它主要的思想是，每一次建立模型是在之前建立模型损失函数的梯度下降方向。损失函数是评价模型性能（一般为拟合程度+正则项），认为损失函数越小，性能越好。而让损失函数持续下降，就能使得模型不断改性提升性能，其最好的方法就是使损失函数沿着梯度方向下降。

Freidman 提出了 Gradient Boosting 算法，其利用了损失函数的负梯度在当前模型的值

$$-\left[\frac{\partial L(y, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f_{m-1}(x)} \qquad (4.2)$$

GBDT 是基于 CART 回归树的梯度提升方法，可以用来做回归预测，调整后

也可以用于分类。它是一个加法模型：$f_m(x)$是每一次迭代学习得到的树模型：

$$\hat{F}(x) = F_M(x) = \sum_{m=1}^{M} f_m(x) \quad (4.3)$$

最终$f_m(x)$学习的是损失函数在函数空间上的负梯度。对于权重ρ_m通过线性搜索求解：

$$\rho_m = \underset{\rho}{\operatorname{argmin}} E_{y,x} L(y, F_{m-1}(x) - \rho * g_m(x)) \quad (4.4)$$

每一次迭代可以看做是采用梯度下降法对最优分类器$F^*(x)$的逐渐比较，每一次学习的模型$f_m(x)$是梯度，进行 M 步迭代之后，最后加出来的模型就是最优分类器的一个逼近模型，所以$f_m(x_i)$使用单步修正方向$-g_m(x_i)$：

$$-g_m(x_i) = g_m(x) = [\frac{\partial L(F(x))}{\partial F(x)}]_{F(x)=F_{m-1}(x)} \quad (4.5)$$

这里的梯度变量是函数，是在函数空间上求解，更新函数通过当前函数的负梯度方向来修正模型，使它更优，最后累加的模型近似最优函数。

4.2　XGboost 模型原理

XGboost 是华盛顿大学的陈天奇博士在 2014 年为改进 Gradient Boosting Machine 算法的计算精度与速度而开发的新的集成机器学习模型，通过对目标函数的巧妙处理，极大地优化了算法的迭代次数与准确度。

对于训练误差函数，采用 boosting 的方式，每一次保留原来的模型不变，加入一个新的函数f到模型中。每一轮选取使得误差函数尽量最大减小的f来优化这一目标。XGboost 方法更一般地采用二阶泰勒展开来定义一个近似误差函数。二阶泰勒展开公式如下：

$$f(x + \Delta x) \cong f(x) + f'(x)\Delta x + \frac{1}{2} f''(x)\Delta x^2 \quad (4.6)$$

则近似目标函数为：

$$Obj \cong \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + \Omega(f_t) + constant \quad (4.7)$$

其次定义树的复杂度。将函数参数f进一步细化为结构部分q和叶子权重部分w。结构函数q把输入映射到叶子的索引号上面，而w给定了每个索引号对应的叶子分数。复杂度包含了一棵树里面节点的个数T，以及每个树叶子节点上面输出分数的 2-范数。假设树的结构q已知，通过这个目标函数来求解出最好的w，以及对应目标函数最大的增益。结果如下：

$$Obj^* = -\frac{1}{2}\sum_{j=1}^{T}\frac{{G_j}^2}{H_j+\lambda} + \gamma T \quad (4.8)$$

通过不断地枚举不同树的结构，利用上述打分函数来寻找出一个最优结构的树，加入到模型中，再重复这样的操作。不过实际中做到穷举所有回归树的结构是十分困难的，所以这里使用贪婪法，每一次尝试去对已有的叶子加入一个分割。对于一个具体的分割方案，可以获得的增益可以由如下公式计算：

$$Gain = \frac{1}{2}\left[\frac{G_L}{H_L+\lambda} + \frac{G_R}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma \quad (4.9)$$

对于每次分割，依然要枚举所有可能的分割方案。假设要枚举所有$x < a$的条件，对于某个特定的分割a要计算a左边和右边的导数和。对于所有的a，只要做一遍从左到右的扫描就可以枚举出所有分割的梯度和G_L、G_R，然后用上面的公式计算每个分割方案的分数。

4.3 LightGBM 算法原理

4.3.1 LightGMB 对 GBM 的改进

LightGBM （Light Gradient Boosting Machin）是 2016 年微软亚洲研究院公布的一个开源、快速、高效的基于决策树算法的提升框架，被用于排序、分类、回归等多种机器学习的任务，支持高效率的并行训练。

LightGBM 中的决策树子模型是采用按叶子分裂的方法分裂节点的，因此它的计算代价比较小，也正是因为选择了这种分裂方式，需要控制树的深度和每个叶子节点的最小数据量，从而避免过拟合现象的发生。LightGBM 选择了基于 Histogram 的决策树算法，将特征值分为很多个小 “桶”，进而在这些 “桶” 上寻找分裂，这样可以减小储存成本和计算成本。另外，类别特征的处理，也使得 LightGBM 在特定数据下有比较好的提升。

LightGBM 具有以下优点：（1）更快的训练速度；（2）更低的内存消耗；（3）更好的模型精度；（4）支持并行学习；（5）可以快速处理海量数据。

4.3.2 GBDT+LR 原理

设y是 0-1 型变量，x是与y相关的确定性变量。为了使y的预测值$\hat{y}$总是介于[0，1]之间，在给定x的情况下，

$$P(y=1|\boldsymbol{x}) = F(\boldsymbol{x},\ \boldsymbol{\beta}) = \Lambda(\boldsymbol{x}'\boldsymbol{\beta}) \equiv \frac{\exp(\boldsymbol{x}'\boldsymbol{\beta})}{1-\exp(\boldsymbol{x}\prime\boldsymbol{\beta})} \quad (4.10)$$

该模型称为“Logit”.这也是一个广义的线性模型：

$$\ln\frac{P(y=1|\boldsymbol{x})}{1-P(y=1|\boldsymbol{x})}=\boldsymbol{x}'\boldsymbol{\beta} \tag{4.11}$$

根据逻辑回归模型的定义建立模型。首先定义y为一名客户的违约情况。如果y = 0，表示客户为正常客户；如果y = 1，表示客户为违约客户。显然这是一个非线性模型，可采用极大似然法（MLE）进行参数估计，整个样本的对数似然函数为

$$\ln L(\boldsymbol{\beta}|\boldsymbol{y},\ \boldsymbol{x})=\sum_{i=1}^{n}\{y_i\ln[\Lambda(\boldsymbol{x_i'\beta})]+(1-y_i)\ln[1-\Lambda(\boldsymbol{x_i'\beta})]\} \tag{4.12}$$

基于上述对数似然函数可以用数值计算中 Newton-Raphson 迭代法计算出参数β的极大似然值，可以通过 R 中的数据挖掘过程得到。Logistic 系数表示对应变量的优势比率的变化[5]，其显著性检验采用 Wald 检验统计量：

$$\text{Wald}=(\frac{\boldsymbol{\beta}}{\sqrt{D(\boldsymbol{\beta})}})^2 \tag{4.13}$$

gbdt+lr 融合了两种基本模型的模型，如图 4.1

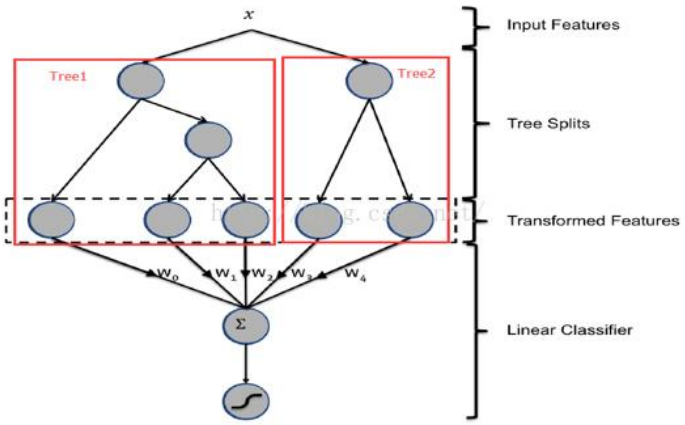

图 4.1 GBDT+LR 模型示意图

在 gbdt+lr 模型中，我们用已有特征训练 GBDT 模型，然后利用 GBDT 模型学习到的树来构造新特征，最后把这些新特征加入原有特征一起训练模型。构造的新特征向量是取值 0/1 的，向量的每个元素对应于 GBDT 模型中树的叶子结点。当一个样本点通过某棵树最终落在这棵树的一个叶子结点上，那么在新特征向量中这个叶子结点对应的元素值为 1，而这棵树的其他叶子结点对应的元素值为 0。新特征向量的长度等于 GBDT 模型里所有树包含的叶子结点数之和。

通过 gbdt+lr，我们可以有效地处理高维度稀疏特征，降低了对数据预处理的需求。

4.4　半监督模型原理

在许多 ML 的实际应用中，很容易找到大量的无标签样例，人们尝试将大量

的无类标签的样例加入到有限的有类标签的样本中一起训练来进行学习。半监督学习避免了数据和资源的浪费，同时解决了监督学习的模型泛化能力不强和无监督学习的模型不精确等问题。

半监督支持向量机中最著名的是 TSVM，它先利用有标记样本学得一个 SVM，利用这个 SVM 对未标记数据进行标记指派。将其代入即得到一个标准 SVM 问题，可求解出新的划分超平面和松弛向量，接下来，TSVM 找出两个标记指派为异类且很可能发生错误的未标记样本，交换它们的标记，再重新求解出更新后的划分超平面和松弛向量，然后再找出两个标记指派为异类且很可能发生错误的未标记样本，逐渐增大 Cu 以提高未标记样本对优化目标的影响，进行下一轮标记指派调整，直 Cu=Cl 为止。

图半监督学习是另一种常用方法。给定一个数据集，我们可将其映射为一个图。若两个样本之间的相似度很高，则对应的结点之间存在一条边的″强度″ 正比于样本之间的相似度。我们可将有标记样本所对应的结点想象为染过色，而未标记样本所对应的结点尚未染色。于是，半监督学习就对应于″颜色″在图上扩散或传播的过程。经典的图半监督学习算法是标记传播算法，详解如下：

输入： 有标记样本集$D_l=\{(x_1,y_1),(x_2,y_2),\dots,(x_l,y_l)\}$;
未标记样本集$D_u=\{x_{l+1},x_{l+2},\dots,x_{l+u}\}$;
构图参数σ;
折中参数α.

过程：

1：基于亲和矩阵和参数σ得到W;
2：基于W构造标记传播矩阵$S=D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$;
3：根据式（13.18）初始化$F(0)$;
4：$t=0$;
5：repeat
6: $F(t+1)=\alpha SF(t)+(1-\alpha)Y$;
7: $t=t+1$
8: until 迭代收敛至F^*
9: for $i=l+1,l+2,\dots,l+u$ do
10: $y_i=\mathrm{argmax}_{1\le j\le|\mathcal{Y}|}(F^*)_{ij}$
11: end for

输出： 未标记样本的预测结果：$\hat{y}=(\hat{y}_{l+1},\hat{y}_{l+2},\dots,\hat{y}_{l+u})$

图半监督学习方法在概念上相当清晰，且易于通过对所涉矩阵运算的分析来探索算法性质。在接收到新样本时，或是将其加入原数据集对图进行重构并重新进行标记传播，或是需引入额外的预测机制。

第五章 模型求解

5.1　XGB 模型求解

5.1.1 XGB 参数选择

一、通用参数设置

（1）booster：选择每次迭代的模型，有两种选择：gbtree：基于树的模型；gbliner：线性模型，因变量违约标志（Y）为 0-1 分类变量，所以不适合做线性模型，这里选择 booster=－gbtree。

（2）silent：当这个参数值为 1 时，静默模式开启，不会输出任何信息。这个参数就保持默认的 0，因为这样能帮我们更好地理解模型。

（3）nthread：这个参数用来进行多线程控制，应当输入系统的核数。本文不输入这个参数，由算法自动检测。

二、Booster 参数设置

（1）eta：典型值为 0.01-0.2，初步选取为 0.04，之后根据模型表现在进行调整。

（2）max_depth：这个值为树的最大深度。这个值也是用来避免过拟合的。max_depth 越大，模型会学到更具体更局部的样本。典型值为 3-10，初步选取为 9，之后根据模型表现在进行调整。

（3）subsample：这个参数控制对于每棵树，随机采样的比例。减小这个参数的值，算法会更加保守，避免过拟合。但是，如果这个值设置得过小，它可能会导致欠拟合。典型值：0.5-1，初步选取为 0.9，之后根据模型表现在进行调整。

（4）colsample_bytree：用来控制每棵随机采样的列数的占比(每一列是一个特征)。典型值：0.5-1，本文选取为 0.8。

（5）min_child_weight：在这里选了一个较小的值 5，因为这是一个极不平衡的分类问题。因此，某些叶子节点下的值会比较小。在后续可以为了解决不平衡分类问题继续降低该参数。

（6）n_estimators：boosting 迭代计算次数，本文取值为 230。

（7）max_delta_step：限制每棵树权重改变的最大步长。当各类别的样本十分不平衡时，它对逻辑回归是很有帮助的。本文的数据集中违约和不违约的用户

差距很大，所以该值初步设置为 8。

三、学习目标参数设置

（1）objective：这个参数定义需要被最小化的损失函数。因变量 Y 是二分类变量，且最后要返回预测的概率，所以选择二分类的逻辑回归：binary:logistic，在这种情况下，需要多设一个参数：num_class(类别数目)。返回的是每个数据属于各个类别的概率，因为题目要求预测用户违约的概率，所以设置 num_class=1。

（2）eval_metric：对于有效数据的度量方法。对于分类问题，默认值是 error。由于最后按照 AUC 来评价模型好坏，所以设置 eval_metric = "auc"。

5.1.2 模型构建与调整

由于测试集的标签未知，为了便于模型的比较，采用 5 折交叉验证的方法进行线下模型的评估。从数据集中随机抽取 80%的观测作为训练集，剩余 20%的观测作为测试集。利用 交叉验证得到的线下 AUC 指标进行预测评估。首先，尝试了基础的 XGBoast 的默认参数进行训练，并且采用了栅格化调参的方式选用线下交叉验证效果最好的训练结果提交，进行线上验证。线下的 cv 值如下表所示：

表 5.1 XGB 模型线下 CV 值

max_depth	7	8	9	10	选取结果
线下 cv 值	0.75489	0.75268	0.77596	0.76584	9
subsample	0.6	0.7	0.8	0.9	选取结果
线下 cv 值	0.76945	0.77017	0.77215	0.77953	0.9
learning_rate	0.03	0.05	0.1	0.3	选取结果
线下 cv 值	0.77648	.0.78124	0.78261	0.77925	0.1

所以我们最后选择了线下 cv 值最高的参数的模型结果进行线上提交，得到 auc 在 0.70 的范围（由于排名不在个人第一，无法看到具体的 auc 值）。

5.2 LightGBM 模型求解

5.2.1 参数选择

一、核心参数

（1）boosting， 选择每次迭代的模型，有四种选择：其中’gbdt’代表传统的

梯度提升决策树；‘rf’代表 Random Forest（随机森林）；‘dart’代表 Dropouts meet Multiple Additive Regression Trees：‘goss’代表 Gradient-based One-Side Sampling（基于梯度的单侧采样）。默认值为 gbdt。

（2）learning_rate：指收缩率，在 dart 中，它还影响了 dropped trees 的归一化权重，默认值为 0.1。

（3）tree_learner, default=serial, type=enum, options=serial, feature, data, voting, alias=tree

其中’serial’指单台机器的 tree learner；’ feature, alias=feature_parallel ’ 指特征并行的 tree learner；‘data, alias=data_parallel’指数据并行的 tree learner；’ voting, alias=voting_parallel’指投票并行的 tree learner。

（4）num_leaves, 指一棵树上的叶子数，默认值为 31（调整时只能为整数）

二、学习控制参数

（1）max_depth, 限制树模型的最大深度. 这可以在 #data 小的情况下防止过拟合. 树仍然可以通过 leaf-wise 生长，当 max_depth< 0 意味着没有限制.默认值为-1，调参只能选择整数

（2）min_data_in_leaf, 一个叶子上数据的最小数量. 可以用来处理过拟合. default=20, type=int, alias=min_data_per_leaf , min_data, min_child_samples

（3）feature_fraction, 如果 feature_fraction 小于 1.0, LightGBM 将会在每次迭代中随机选择部分特征. 例如, 如果设置为 0.8, 将会在每棵树训练之前选择 80% 的特征。这个参数也可以用来加速训练，也可以用来处理过拟合。default=1.0, type=double, 0.0 < feature_fraction < 1.0, alias=sub_feature, colsample_bytree

（4）bagging_fraction/ subsample, default=1.0, 0.0 < bagging_fraction < 1.0, 类似于 feature_fraction, 但是它将在不进行重采样的情况下随机选择部分数据，这个参数可以用来加速训练，也可以用来处理过拟合，但要注意的是，为了启用 bagging, bagging_freq 应该设置为非零值

（5）bagging_freq/ subsample_freq,这个参数指 bagging 的频率,默认值 0 意味着禁用 bagging. k 意味着每 k 次迭代执行 bagging，在使用中要注意的是：为了启用 bagging, bagging_fraction 设置适当

（6）lambda_l1, L1 正则项，默认值为 0 ，alias=reg_alpha

（7）lambda_l2, L2 正则项，默认值为 0，alias=reg_lambda

三、I0 参数

（1） is_sparse, 用于 enable/disable 稀疏优化. 设置 false 就禁用稀疏优化，默认值为 true, alias=is_enable_sparse, enable_sparse。由于训练集样本其实是稀疏

特征，所以后续调整该参数

（2）categorical_feature, 指定分类特征。用数字做索引，e.g. categorical_feature=0,1,2 意味着 column_0, column_1 和 column_2 是分类特征；为列名添加前缀 name:, e.g. categorical_feature=name:c1,c2,c3 意味着 c1, c2 和 c3 是分类特征

Note: 只支持分类与 int type. 索引从 0 开始. 同时它不包括标签栏；并且负值的值将被视为 missing values

default="", type=string, alias=categorical_column, cat_feature, cat_column

四、目标参数

is_unbalance, default=false, 这个参数用于 binary 分类，如果培训数据不平衡设置为 true

5.2.2 模型构建与调整

由于测试集的标签未知，为了便于模型的比较，首先采用 4 折交叉验证的方法进行线下模型的评估。从数据集中随机抽取 75%的观测作为训练集，剩余 25%的观测作为测试集。利用 4 折交叉验证得到的线下 AUC 指标进行预测评估。首先，尝试了基础的 lightGBM 的默认参数进行训练。并依据前文特征工程，缺失值处理等的不同处理方式，建立了多种模型求解的方案。

方案一：LightGBM 基模型，在基模型的基础上不断进行参数的优化，保留最优的线下交叉验证的 AUC 均值。

方案二：类别变量处理+ LightGBM 基模型，将将 x_1，x_2，x_80，x_81，x_93，x_95 这 6 个特征认定为数值型变量，剩下的特征都划分为类别型变量，然后再运行 LightGBM 基模型

方案三：特征工程（缺失特征）+ LightGBM 基模型，将特征工程生成的筛选后的缺失特征加入 LightGBM 基模型，运行求解结果，输出线下交叉验证的 AUC 均值。

方案四：特征工程（暴力特征）+ LightGBM 基模型，将暴力生成并经过筛选的特征加入到 LightGBM 基模型，运行求解结果，输出线下交叉验证的 AUC 均值。

方案五:数据预处理（特征的删除）+ LightGBM 基模型，将经过降维和特征删除后的特征数量代入到 LightGBM 基模型，并输出线下交叉验证的 AUC 均值

方案六:分客群+ LightGBM 基模型，将样本按照客户来源不同划分为三个子样本，分别运行 LightGBM 基模型，采用不同客户群的 AUC 得分的加权平均值与原基础结果比较。

所以以上几个方案的运行结果的 AUC 如下表所示。

表 5.2 LGB 模型线下 CV 值

	方案一	方案二	方案三	方案四	方案五	方案六
线下 AUC	0.816964	0.81116	0.92032	0.80267	0.79593	0.78069
线上 AUC	0.74982	0.74	0.73	0.73	0.74	0.72

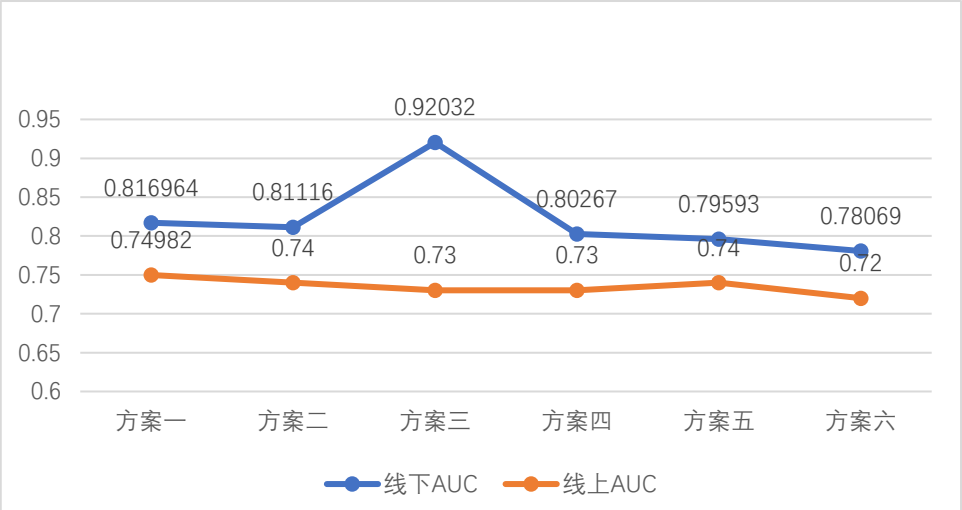

图 5.1 LightGBM 不同方案表现

其中方案一经过栅格调参得到最优调参，其中栅格调参主要调整了 max_depth、num_leaves、reg_alpha、 reg_lambda、subsample、learning_rate 的几个参数。

5.3　GBDT+LR 模型求解

5.3.1 参数选择

在 GBDT 模型中，我们选择了 lightGBM 作为梯度提升树。由于 GBDT+LR 可以看作是 boosting 模型与线性模型的组合，所以在参数选择方面我们分为两部分来介绍

一、lightGBM 相关参数

（1）boosting_type:　提升类型，由于我们采用的是 GBDT+LR 模型，故这里选择 “gbdt”.

（2）Objective :　由于目标问题是二分类问题，故选择“binary”

（3）metric:　根据赛题评价标准，我们选择 auc 作为参数

（4）num_leaves:　叶节点数量，在这个模型中，叶节点数目过多不仅在

lightGBM 上有过拟合的风险，而且会增加输入逻辑斯蒂回归模型特征的维度，故选择较小的值能防止模型过拟合，在此我们选择的值是 64

（5）reg_alpha L1 正则化系数 在这里选择 1

（6）reg_lambda L2 正则化系数 在这里选择 1

（7）learning_rate 学习率， 这里采用 0.01

（8）n_estimators boosting 迭代计算次数，本文取值为 5000

（9）feature_fraction 如果 feature_fraction 小于 1.0, LightGBM 将会在每次迭代中随机选择部分特征，可以用来加速训练与预防过拟合，在这里我们选择 0.8

（10）bagging_fraction 类似于 feature_fraction, 但是它将在不进行重采样的情况下随机选择部分数据，可以用来加速训练与预防过拟合，在这里我们选择 0.8

二、logistic regression

（1）penalty 正则化类型，有 L1，L2 两种正则化选择，在这里选择 L2 正则化

（2）C 正则化系数，在这里选择默认系数 1

5.3.2 模型构建与求解

由于测试集的标签未知，为了便于模型的比较，采用 5 折交叉验证的方法进行线下模型的评估。从数据集中随机抽取 80%的观测作为训练集，剩余 20%的观测作为测试集。利用 交叉验证得到的线下 AUC 指标进行预测评估。首先，尝试了基础的默认参数进行训练，并且采用了栅格化调参的方式选用线下交叉验证效果最好的训练结果提交，进行线上验证。

由于线下验证差异不大且表现不佳（0.5-0.55）所以没有做过多的模型改进。

所以我们最后选择了线下 cv 值最高的参数的模型结果进行线上提交，得到 auc 在 0.69 的范围（由于排名不在个人第一，无法看到具体的 auc 值）

5.4 半监督模型求解

5.4.1 自训练算法求解模型

在自训练算法中，选择之前在验证集上表现相对较好的 Lightgbm 模型作为自训练的基学习器。并且考虑到训练集数据相对较小且无标签数据分布与训练数据有差异，所以将无标签数据批量投入训练模型。以投入训练的无标签数据比例作为横轴，在验证集上的表现 AUC 作为纵轴绘制成图像如下

图 5.2 自训练模型不同数据比例时的 AUC 值

从图像中可以看出当投入 70%的无标签数据时，模型的训练效果最好。取投入 70%的无标签数据绘制 ROC 曲线图如下：

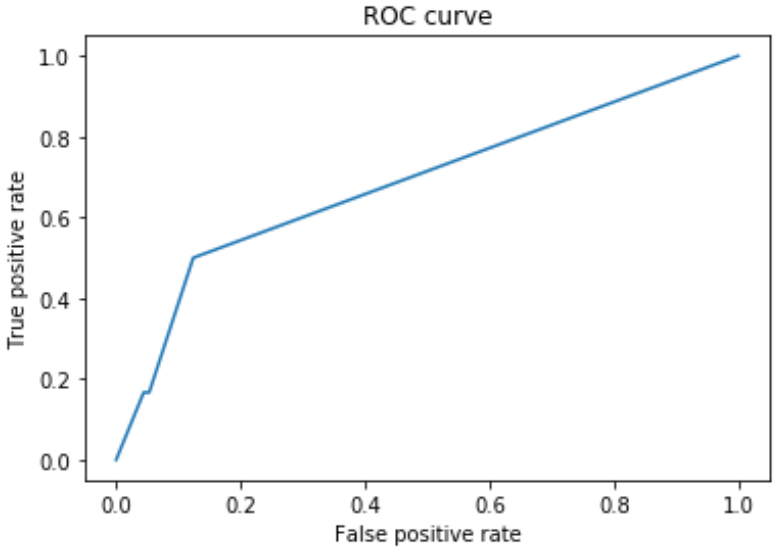

图 5.3 70%无标签数据时的 ROC 图像

5.4.2 图模型 Label Propagation 算法求解模型

Label Propagation 算法实现调用 sklearn 的 LabelPropagation API 接口，选用 KNN 作为核函数。同样分批量投入无标签数据进入模型训练。以投入训练的无标签数据比例作为横轴，在验证集上的表现 AUC 作为纵轴绘制成图像如下：

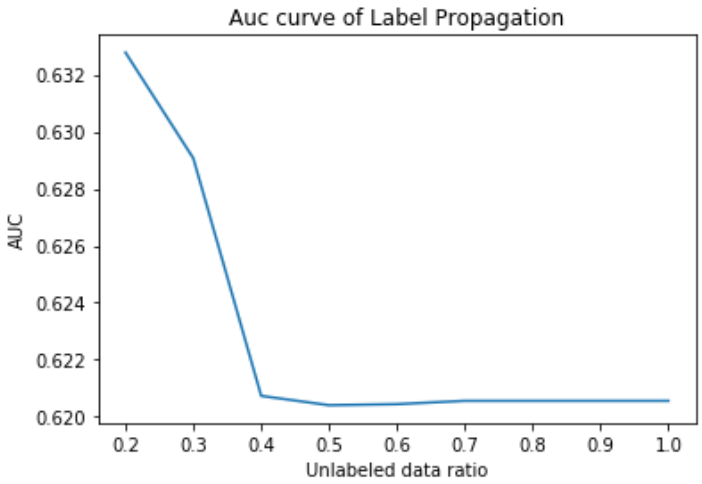

图 5.4 图模型不同数据比例时的 AUC 值

LabelPropagation 算法在验证集上总体表现较差，故不对此算法继续研究，也不在最高点画出 ROC 曲线图。

5.4.3 TSVM 模型求解

Tsvm 算法调用 github 的开源半监督学习框架 frameworks，使用默认参数。同样分批量投入无标签数据进入模型训练。以投入训练的无标签数据比例作为横轴，在验证集上的表现 AUC 作为纵轴绘制成图像如下：

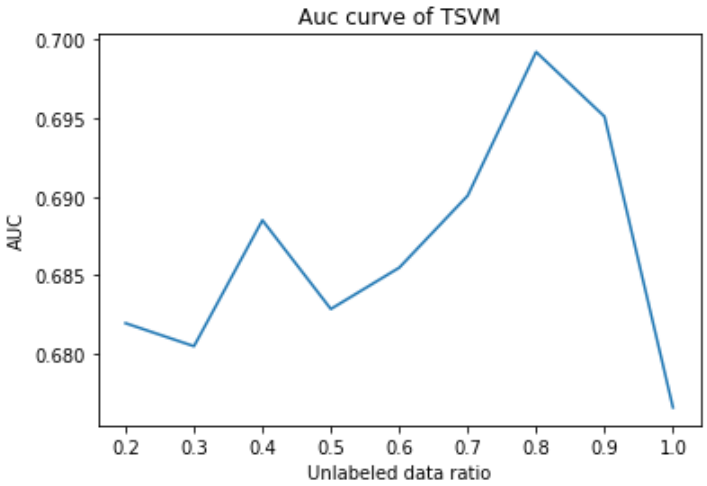

图 5.5 TSVM 模型不同数据比例时的 AUC 值

综上所述，可以得出，在本文尝试的三种半监督学习算法，自训练算法，LabelProgation 算法和 TSVM 算法中，最简单的自训练算法表现最为优异，也说明了在数据量较小，数据分布不稳定的情况下，简单的算法泛化能力更强。

5.5　方案验证

由于本题所给数据，数据量较小且分布不稳定，我们所尝试的方案中只有基于 lightgbm 的 baseline 调参方案在线上提交分数上有着相对稳定的提升，其余尝试特征工程，半监督学习以及复杂模型的方案在线上评测上均没有很稳定的表现。鉴于这种情况，我们团队在 kaggle 平台上 InClass Prediction Competition 中的贷款违约数据，并依次对我们上文中提到的方案进行尝试。

（1）Baseline 方案

我们首先对原数据进行了基础的数据处理以可以作为机器学习算法输入，我们选择了 lightgbm 模型作为 Baseline 方案的分类器。Lightgbm 的超参数如下：

表 5.3　LGB 模型参数选取

boosting_type	**gbdt**
max_depth	5
num_leaves	24
n_estimators	400
subsample	0.9
subsample	0.8
subsample_freq	1
learning_rate	0.05

（2）特征工程方案

在 baseline 方案的基础上，我们在原特征的基础上添加了缺失值信息特征，与少量的暴力组合特征并删除了缺失率超过 90%的特征。该方案的 lightgbm 分类器使用与 baseline 方案相同的超参数。

（3）半监督学习方案

在此方案中，我们添加了数据量大约为有标签训练数据 50%的无标签数据，选用基分类器为 lightgbm 的自训练算法，测试集测评结果如下：

表 5.4 测试集结果对比

方案	测试集 Auc
Baseline 方案	0.814572
特征工程方案	0.815743
半监督学习方案	0.818435

并绘制三种方案的 roc 曲线如下：

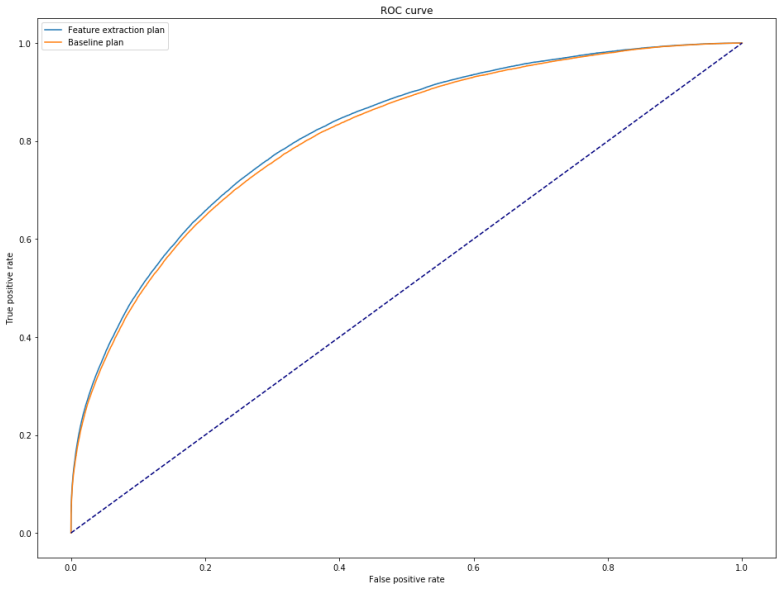

图 5.6　TSVM 模型不同数据比例时的 AUC 值

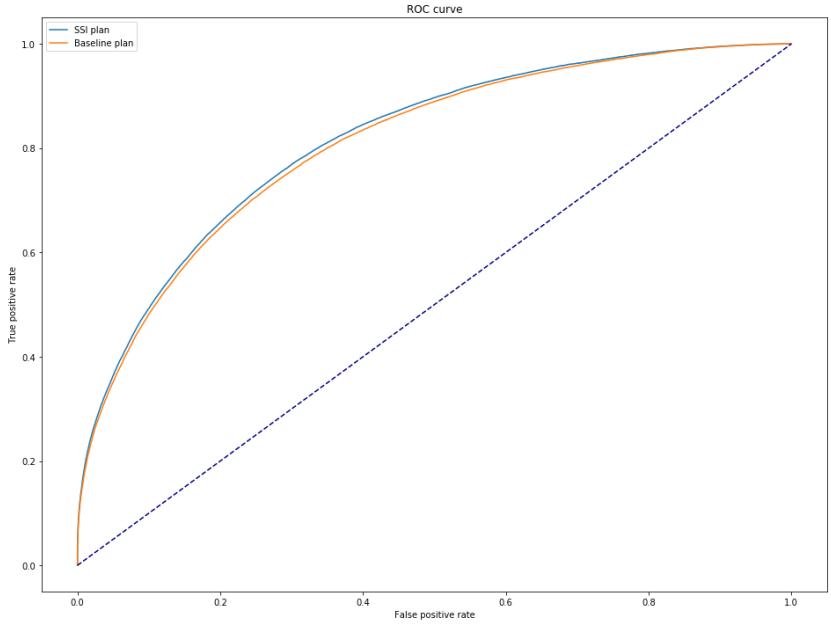

图 5.7　TSVM 模型不同数据比例时的 AUC 值

以上实验较好地说明前文中提到的多种方案在实际风控模型中具有一定使用价值，特征工程和对无标签数据的半监督学习方案可以提升风控模型对违约估计的准确性。

第六章 总结与展望

6.1 论文总结

本文为了预测用户的违约概率，使用了 GBDT+LR 模型、XGBoast 模型、LightGBM 模型和半监督模型进行了违约预测。并基于线下交叉验证的 CV 值与线上有限提交次数的 AUC 值进行模型的比较与评估，不断提高模型的可靠性。在可视化的基础上，采用不同的数据预处理方案（降维、缺失值处理、类别型变量转换等）与特征工程方案（缺失信息特征、特征组合）在最优的 LGB 基模型上进行试验，但最终还是选取了线上 AUC 最高的 LGB 基模型进行提交。

根据赛方提出数据存在的高维数据、稀疏数据问题、无标签样本的利用、多产品客群的不同模型构建、和好坏样本不平衡的问题，我们对整个过程中解决相应问题的举措进行了梳理如下：

（1）高维数据的解决方案（降维，特征的扩充与删除）

在本次的数据集中，15000 条有标签样本包含 157 个特征，在特征工程构建新特征后，特征数量更是急剧攀升。所以，在整个建模过程中利用各特征与标签 iv 值进行了特征筛选，同时，对归属同一类别，相似度高的特征进行了删除处理。

（2）稀疏数据的解决方案（缺失值处理，缺失数据的删除）

首先尝试去探寻稀疏数据产生的原因，稀疏数据的产生主要因为存在大量的缺失值与 0 值，在数据集内，最明显的稀疏数据的现象就是存在大量的缺失数据。因此我们主要采用树模型进行预测即 GBDT、XGboast 与 LightGBM 模型进行求解。在后期我们尝试了 GBDT+LR 模型，LR 模型中的特征组合很关键，但又无法直接通过特征笛卡尔积解决，只能依靠人工经验，耗时耗力同时并不一定会带来效果提升。为了自动发现有效的特征、特征组合，所以采用 GBDT 转换特征。

（3）无标签样本的解决方案（半监督）

对于无标签样本，我们引入了半监督学习模型进行处理。在自训练算法中，选择之前在验证集上表现相对较好的 Lightgbm 模型作为自训练的基学习器。并且考虑到训练集数据相对较小且无标签数据分布与训练数据有差异，所以将无标签数据批量投入训练模型。

（5）多产品客群的解决方案尝试

在前文我们在可视化阶段探究不同客群之间好坏样本的区别，并在模型处理阶段尝试用 LGB，模型分客群进行求解与不分客户群的结果进行比较，发现不

对客户群进行区分的模型 auc 表现更好，所以在初赛最终没有对客群进行区分。在后期反思的阶段，认为可能不区分客群能增大样本量，在高维数据和稀疏数据问题严重的情况下，能更好的捕捉数据信息

（6）好坏样本不平衡的解决方案尝试（模型参数与欠采样，过采样）

对于这个问题，在初赛阶段，我们引入 LGB 模型的样本不平衡参数进行调整，但调整后的模型线下 AUC 值远低于原来的基础模型，所以在最终没有采用。

为了试验我们的举措是能够解决这类问题的，选取 kaggle 平台上 InClass Prediction Competition 比赛的数据进行了半监督模型解决无标签问题的迁移验证。

6.2 问题与展望

由于题目给出的有标签训练集只有 15000 左右的样本，而每一个用户的特征共有 157 个（不包含违约标志），因此在模型拟合过程中很可能会出现模型概括性不好的情况。在后期进一步修正模型时，应尽可能增加训练集样本量，增加模型结果的稳定性。

由于题目给出的用户特征有很强的相关性且涉及的十分广，因此在最初构建模型时采取了一定的人工观察。但是在模型改进时还可以进行正向的逐步回归法与人工测试的结果进行比对，使最终结果更具可信性。

题目给出的无标签训练集在可视化阶段发现分布与有标签训练集差别很大，所以导致半监督的学习效果并不明显，反而低于只利用有标签数据的 LightGBM 基模型的 AUC 值，在模型改进阶段可以适当的改变无标签训练集的分布，以保证半监督模型训练的效果。

参考文献

[1] 巴曙松.巴塞尔新资本协议研究[M]. 北京：中国金融出版社, 2004
[2] 百度百科. http://baike.baidu.com/view/18754.htm?fr=ala0_1_1
[3] Beaver, W. Financial Ratios as Predictors of Failure[J]. Journal of Accounting Research, 1966,(4):71-111 [4] Altman, E. I. Financial Ratios Discriminant Analysis and the Prediction of Corporate Bankruptcy[J]. Journal of Finance, 1968,23(4): 589-609
[5] Altman, E. I., Haldeman, R., Narayanan, P. Zeta Analysis a New Model to Identify Bankruptcy Risk of Corporations[J]. Journal of Banking & Finance, 1977,1(1):29-54
[6] Altman, E. I., Saunders, A. Credit Risk Measurement: Developments over the Last 20 Years[J]. Journal of Banking & Finance, 1997, 21(11):1721-1742
[7] Shi, Y., Peng, Y., Xu, W., Tang, X. Data Mining via Multiple Criteria Linear Programming: Applications in Credit Card Portfolio Management[J]. International Journal of Information Technology and Decision Making, 2002,1(1):131-151
[8] Martin, D. Early Warning of Bank Failure: A Logit Regression Approach[J]. Journal of Banking & Finance, 1977,1(3):249-276
[9] Ohlson, J. A. Financial Ratios and the Probabilistic Prediction of Bankruptcy[J]. Journal of Accounting Research, 1980,18(1):109131
[10] Platt, H. D., Platt, M. B. A Note on the Use of Industry-relative Ratios in Bankruptcy Prediction[J]. Journal of Banking & Finance, 1991,15(6):1183-1194
[11] Lawrence, E. L., Smith, S., Rhoades, M. An Analysis of Default Risk in Mobile Home Credit[J]. Journal of Banking & Finance, 1992,16(2):299-312
[12] Coats, P. K., Fant, L. F. Recognizing Financial Distress Patterns Using a Neural Network Tool[J]. Financial Management, 1993,22 (3):142-155
[13] Zhang, G. P., Hu, M. Y., Patuwo, B. E., Indro, D. C. Artificial Neural Networks in Bankruptcy Prediction: General Framework and Cross-validation Analysis[J]. European Journal of Operational Research, 1999,116(1):16-32
[14]王春峰,万海晖,张维.组合预测在商业银行信用风险评估中的应用[J]. 管理工程学报, 1999,13(1):5-8
[15]王春峰,万海晖,张维.基于神经网络技术的商业银行信用风险评估[J].系统工程理论与实践, 1999,(9):25-32
[16]王春峰,赵欣,韩冬.基于改进蚁群算法的商业银行信用风险评估方法[J]. 天津大学学报, 2005,7(2):81-85
[17]张维,李玉霜,王春峰.递归分类树在信用风险分析中的应用[J]. 系统工程理论与实践, 2000,(3):50-55
[18]方洪全,曾勇.联机分析挖掘 OLAM 方法在银行信用风险评估中的应用[J]. 5 中国软科学, 2004,(10):126-130
[19]马晓君,沙靖岚,牛雪琪.基于LightGBM算法的P2P项目信用评级模型的设计及应用[J].数量经济技术经济研究,2018,35(05):144-160.
[20]沙靖岚. 基于 LightGBM 与 XGBoost 算法的 P2P 网络借贷违约预测模型的比较研究[D].

东北财经大学,2017.