

银行对中小微企业的信贷决策分析

摘要

本文研究银行如何根据中小微企业的实力、信誉对其信贷风险做出评估，然后依据信贷风险等因素来确定是否放贷及贷款额度、利率和期限等信贷策略。

针对问题一，对附件 1 中的数据进行特征提取，综合考虑了企业实力、发展潜力、上下游供求关系、企业抗风险能力四类指标，共提取出 20 个特征来衡量企业的信贷风险。本文以 Logistic 回归、Adaboost、GBDT、SVM 和随机森林为基分类器，建立 Soft Voting 集成学习算法，计算企业的违约风险。Soft Voting 在整个数据集上的综合准确率为 97.6%，ROC 曲线覆盖面积(AUC 值)达到 99.42%，接近完美模型，计算出的概率可信度高。根据附件 1 给出的信誉评级和得出的违约风险，根据银行的行为，建立一个利润最高，风险和潜在损失最小的多目标规划模型，求解得出本问的银行信贷策略。

针对问题二，由于本问为无信贷记录企业，银行需要根据贷款企业的已知信息和有信贷记录的信息对贷款企业进行信誉评级和违约风险判断。对附件 2 的数据做与附件 1 一样的处理，根据提取特征，使用问题一训练的 Soft Voting 模型对附件 2 企业进行违约风险预测。将违约风险加入已有特征，用有信贷记录的企业相关数据训练 XGBoost 模型，实现对信誉评级的多分类预测。XGBoost 在训练集上综合准确率为 87.8%。完成信誉评级和违约风险判断后，就可以进行多目标规划模型求解，得出银行的信贷策略。

针对问题三，本文以新冠疫情这个突发事件为例，综合考虑企业的信贷风险和疫情对不同企业的不同影响。从企业在疫情中面临的系统性风险和非系统性风险的角度，本文在附件 2 中提取行业风险、企业经营状况、利润同比增长率、废票占比变动率和交易企业数量变动率 5 个特征，反应新冠疫情对企业的综合影响。运用系统聚类将企业聚成 3 类，不同类别的企业具有不同的风险乘数 Q_i ，风险乘数描述了疫情对不同类别企业的违约风险的放大作用。将信誉评级与聚类结果综合考虑，银行将对企业建立新的评价分级体系，将风险乘数 Q_i 代入得到调整之后的多目标规划模型，求解可得出银行问题三情景下的信贷策略。

关键词：特征工程；集成学习算法；多目标规划；XGBoost；系统聚类



一、问题重述

1.1 问题背景

在实践中，由于中小企业规模相对较小，无较多抵押资产，银行通常根据信贷政策、企业交易票据信息以及上下游企业的影响力，向实力雄厚、供需关系稳定的企业提供贷款，对信誉好且信用风险底的企业可以给予优惠利率。银行首先根据中小企业的实力和信誉对其信用风险进行评估，然后根据信用风险等因素确定是否进行放贷和贷款额度、利率、期限等信贷策略。

1.2 问题提出

某银行对确定要放贷企业的贷款额度为 10-100 万元；年利率为 4%~15%；贷款期限为 1 年。该银行请你们团队根据 2019 年统计的有无信贷记录相关企业数据和贷款利率与客户流失率关系，通过建立数学模型，解决下列问题：

- (1) 请对有信贷记录的公司的信贷风险进行量化分析，给出信贷总额固定情况下的信贷策略。
- (2) 试对无信贷记录的公司的信贷风险进行量化分析，给出信贷总额为 1 亿元时的信贷策略。
- (3) 综合考虑无信贷公司的信贷风险与突发因素对企业的影响，给出信贷总额为 1 亿元时信贷策略的调整。

二、问题分析

2.1 问题一分析

本问要求对 123 家企业的信贷风险进行量化分析，并在信贷总额固定时，给出相应信贷策略。由于信贷策略主要包括是否放贷，贷款额度，利率和期限四个方面，且期限为一年已经由题目中给出，因此我们主要是从对不同信誉等级的企业提供相应的贷款额度、利率与是否放贷来制定策略。首先本文就企业实力、对上下游企业的影响能力、发展潜力、抗风险能力等四类指标先对数据进行特征提取，共提取出 20 个相关的特征。通过运用 Voting 集成学习算法，对提取出的特征进行训练，得到每个企业的违约风险值。之后，利用题目给出的信誉评级，算出每个等级的平均违约风险，以此代表该等级企业的违约风险。最后建立以银行利润最高，银行风险最低，潜在流失客户率最小为目标的多目标非线性规划模型。求解得出银行年度信贷总额为固定金额（以 5000 万为例）银行对于不同信誉等级的企业发放的相应的贷款额度与利率。

2.2 问题二分析

本问要求对附件 2 的 302 家企业进行信贷风险分析，并给出信贷策略。本问与问题一的主要差别在于本问的企业无信贷记录，即无信誉评级和是否违约的数据，需要银行通过附件 2 给出的数据进行自行判断。首先运用第一问训练出的的 Voting 集成学习算法，求出 302 家企业的违约风险，对违约风险大于 50% 的企业不予放贷。要确定企业的信誉评级是一个多分类问题，将企业的违约概率这一特征与之前提取 20 个特征组成新的 21 个特征，运用 Xgboost 模型，用附件 1 所提取的特征训练模型，然后使用



附件 2 数据对 302 家企业进行信誉评级，得出分类结果。最后求解多目标非线性规划模型，解出银行信贷总额为 1 亿元时对不同信誉等级的企业发放的相应贷款额度与利率。

2.3 问题三分析

本问考虑的突发事件以新冠疫情为例，不仅要考虑企业本身的信贷风险，还要考虑的疫情对企业的影响所导致企业经营困难，造成的“还贷难”问题。简而言之，新冠疫情扩大了企业的违约风险，具体的扩大倍数本文以风险乘数进行度量。考虑新冠疫情期间企业的所遭受的系统性风险和非系统性风险，在附件 2 中提取出行业风险、企业状况、利润同比增长率、废票占比变动率、交易企业数量变动率 5 个特征，反映新冠疫情对企业的影响。本文依据疫情对企业的影响程度对 302 家企业进行聚类分析，配合问题二所做出的信誉评级，建立一个新的综合分类评级。不同类别的企业具有不同的风险乘数 Q_i 。在之前的多目标规划模型中，加入风险乘数 Q_i ，对银行的风险目标进行修改，得到调整后的多目标规划模型，解出模型后得到银行在当期情况下的信贷策略。

三、模型假设

- 1、同一信誉评级下，本文认为所有企业的信贷风险一致。银行将给予相同信誉评级企业相同的贷款额度与年利率。
- 2、针对有信贷记录的企业，银行根据以往的信誉评级给贷款。若该企业之前就有违约记录，则今年不予以贷款。
- 3、针对无信贷记录的企业，银行根据已有信贷记录企业的数据对于无信贷记录企业进行信誉评级之后再考虑是否贷款。
- 4、银行给予贷款的目标为，所得利息收益最大，承受的风险与客户流失率最小，银行根据此目标对贷款企业发放贷款额度和给予利率优惠。
- 5、附件 1、2 所给交易票据数据为企业在所给时间内所有商业交易的总数据，不存在遗漏票据，可以反映企业在所给时间段内的所有可开票的商业交易活动。
- 6、附件中所给出的企业均为请求银行贷款的企业，并非潜在客户，即不论利率多高附件内企业都不会放弃贷款。

四、符号说明

符号	含义
π	企业利润
M_i	总金额, $i=1$ 时代表进项, $i=2$ 时代表销项



F_i	票据数量, $i=1$ 时代表进项, $i=2$ 时代表销项
Tal_i	税额, $i=1$ 时代表进项, $i=2$ 时代表销项
Tax	企业应交增值税
Num_{ij}	与某企业有 j ($j=1, \dots, 4$) 年交易关系的企业数量。 $i=1$ 时表示进项交易 (上游企业), $i=2$ 时表示销项交易 (下游企业)。
GU_i	当 $i=1$ 时, 表示进项发票中作废发票的比例; 当 $i=2$ 时, 表示销项发票中作废发票的比例。
An	企业利润增加的绝对数 (除 2020 年数据)
Rn	企业利润增长的相对数 (比例) (除 2020 年数据)
$ToPro$	企业是否扭亏为盈, 是为 1, 不是则为 0
$ToLoss$	企业是否变盈为亏, 是为 1, 不是则为 0
$Under$	企业是否为其他企业的下属部门, 是为 1, 不是则为 0
$Controlled$	企业是否为其他企业的分公司或子公司, 是为 1, 不是则为 0
$Independent$	企业是否为独立公司, 是为 1, 不是则为 0
$Individual$	企业是否为个体经营, 是为 1, 不是则为 0
$Grade$	企业信誉评级, 分 A, B, C, D 四个等级分别为 1, 2, 3, 4 四个值
$Break$	企业是否违约, 是为 1, 不是则为 0
$Accuracy_i$	第 i 个基分类器的综合准确率
W_i	第 i 个基分类器投票所占权重



$\bar{\sigma}_i$	第 <i>i</i> 类企业的平均风险
I_i	<i>i</i> 类企业的利率, $i=1, \dots, 3$
P_i	<i>i</i> 类企业的贷款额度, $i=1, \dots, 3$
C_i	<i>i</i> 类企业的数量, $i=1, \dots, 3$
$Lost(I_i)$	利率 I_i 下的 <i>i</i> 类企业的潜在客户流失率
Con_i	第 <i>i</i> 个企业在2020年1至2月的盈利状况, 若盈利则为1, 无交易为0, 亏损为-1, $i=1, \dots, 302$
YG_i	第 <i>i</i> 个企业的利润在2020年1至2月的同比增长率, $i=1, \dots, 302$
$Error_{ij}$	第 <i>i</i> 个企业2020年1至2月较2019年同期进项/销项作废和负数票据所占比例的变动率, $i=1, \dots, 302, j=1, 2$
Tp_{ij}	第 <i>i</i> 个企业2020年1至2月较2019年同期进项/销项交易对象数量的变动率, $i=1, \dots, 302, j=1, 2$
Ir_i	第 <i>i</i> 个企业所在行业的受疫情打击的程度, $i=1, \dots, 302$
CC_{ij}	第 <i>i</i> 个企业进项/销项交易的交易对象数量
Q_i	第 <i>i</i> 类企业的风险乘数

五、问题一建模与求解

5.1 数据预处理

我们对于附件1的数据, 运用Python的pandas库, 找出缺失值。若有缺失值,



可删除有缺失值的样本或特征，或对缺失值进行填补。检查数据是否存在缺失值或者NAN现象，通过简单的查看可知该题数据不存在缺失，无需进行缺失值处理。

5.2 数据特征提取

根据附件一所给出的数据，结合相关文献和题目，本文认为银行放贷额度与利润率的选择主要受企业自身实力、发展前景和对上下游企业的影响力这三个方面进行评估。根据附件一所给信息，本文就四个方面总共提取了20个特征，用以衡量银行对企业信誉等级的评判，以此来衡量企业贷款的信贷风险。

5.2.1 企业实力

由前文假设可知，附件所给数据完整，保护所给日期内企业所有的交易事项，故根据进项和销项发票的金额，我们可以得出企业在所给日期之内所得利润总额 π 。公式如下：

$$\pi = M_2 - M_1$$

利润总额越高，企业盈利能力越强，银行可提供贷款额度可相应提升，利率应该相应降低；根据有效进项与销项发票的税额，我们可以得出企业在所给日期之内的增值税 Tax ，公式如下：

$$Tax = Tal_2 - Tal_1$$

此处我们对增值税为负的情况置零；根据每个公司与上下游企业的无效票据与有效票据，我们可以得出在所给日期内的作废票据比 $GU_i (i=1, 2)$ ，公式如下：

$$GU_1 = \frac{F_1}{F_1 + F_2}$$

$$GU_2 = \frac{F_2}{F_1 + F_2}$$

作废票据比越高，即银行信贷风险越高，因此需要适当降低公司信誉等级，减少信贷额度，提高利率。

5.2.2 企业对上下游企业的影响力

根据附件1所给的开票日期判断各个企业与其上下游企业合作的时间长度由此来判断供应链的强度。强度与粘性越大，即代表企业对于上下游企业的影响力越大。对于这种企业银行可适当提高贷款额度，降低利率。据此，我们用 Num_{ij} 对其进行衡

量，其中 $i=1, 2$ ， $j=1, 2, 3, 4$ 。当 $i=1$ 时，表示该企业与上游企业的进项交易，

$i=2$ 时，表示该企业与下游企业的销项交易， j 表示与该企业有 j 年合作关系的企业数



量。

5.2.3 企业的发展潜力

对于企业的未来发展潜力，本文根据题目所提供的数据，分别求出每个企业的利润绝对数的变化 An 和相对数的变化 Rn 。因为考虑到 2020 年的数据并不为全年数据，所以在计算这两个指标的时候剔除了 2020 年的票据数据。

$$An = \pi_{\text{第一年}} - \pi_{\text{最后一年}}$$

$$Rn = \frac{\pi_{\text{第一年}}}{\pi_{\text{最后一年}}}$$

综合 An 和 Rn 两者可得出企业的经营是否发生最大变化，即得出企业是否扭亏为盈或者由盈转亏。当 Rn 小于 0 时，则代表企业利润发生了正负转变，这时候需要观察 An 。当 An 大于 0 时，说明企业最近年份利润为正而最早年份利润为负，企业扭亏为盈，即 $ToPro$ 为 1，反之为 0；当 An 小于 0 时，说明企业最近年份利润为负而最早年份利润正，企业由盈转亏，即 $ToLoss$ 为 1，反之为 0。针对扭亏为盈的企业，银行可适当提高贷款额度与降低利率，针对扭亏为盈的企业，银行可适当降低贷款额度与提高利率。

5.2.4 企业的抗风险能力

根据附件 1 所给数据可以明显看出不同企业可分为本身母公司，公司旗下子公司，下属部门与个体经营。每个不同企业的抗风险能力都不尽相同。因此我们将这四种情况分别作为四种特征值，将所有企业划分为四种情况内。当公司为独立公司时，*Independent* 为 1，当公司为个体经营，*Individual* 为 1，当公司为子公司时，*Controlled* 为 1，当公司为下属部门，*Under* 为 1。

5.3 信贷风险评分模型

5.3.1 模型选择

信用风险的度量方法主要有传统信用风险度量方法和现代信用度量方法，因为现代信用风险度量方法运用了大量财务数据，受本题条件所限，本文主要参考传统信用评分方法中的多元非线性回归模型。传统的风险测度中，一般使用 Logistics 回归或 Probit 回归计算企业的违约概率，以此来度量企业信贷风险^{[1][2]}。经过查询文献，亦有文献使用神经网络和机器学习的其他算法进行风险测量^[3]，所以本文拟采取集成多个机器学习算法的方式，对附件 1 中的 123 家企业进行信贷风险度量。



5.3.2 数据选择与处理

将企业的信誉评级数据进行数据编码，若信誉评级为 A，则 $Grade = 1$ ；若信誉评级为 B，则 $Grade = 2$ ；若信誉评级为 C，则 $Grade = 3$ ；若信誉评级为 D，则 $Grade = 4$ 。

模型将使用前文从附件一中提取出 20 个特征，共 123 个样本进行训练。使用 Python 将数据随机切分为训练集和测试集，并以原数据为标准，对训练集和测试集数据进行标准化，将数据处理至 0 到 1 之间。

5.3.3 模型建立

投票(voting)是在分类算法中广泛运用的集成学习算法之一。投票主要有硬投票(Hard voting)和软投票(Soft voting)两种。硬投票是一种特殊的软投票，即各基分类器权重相同的投票，其原理为多数投票原则：如果基分类器的某一分类结果超过半数，则集成算法选择该结果；若无半数结果则无输出。软投票的原理也为多数投票，但是各基分类器投票所占的权重可自己定义。当各基分类器分类效果差异比较大时，应当选择软投票，给予分类性能更好的基分类器更大的权重，以此优化分类结果。

本文所选择的 5 个基分类器分别为 Logistic 回归，Adaboost，GBDT，SVM 和随机森林。Logistic 回归为传统信贷风险评级中常用的模型，该模型的核心为 Logit 函数，

即： $g(z) = \frac{1}{1+e^{-z}}$ ，通过极大似然估计求解对应参数，将分类问题转化为概率问题映射至(0,1)区间。在传统的信用风险评分中，Logistic 回归的准确率能达到 54%—92%。

Adaboost，GBDT 和随机森林都是基于 boosting 算法的分类器，分类结果较为理想，模型具有较强的泛化能力。Adaboost 首先赋予 n 个训练样本相同的权重，从而训练出一个基分类器，之后进行预先设置的 T 次迭代，每次迭代将前一次分类器中分错的样本加大权重，使得在下一次迭代中更加关注这些样本，从而调整权重改善分类器，经过 T 次迭代得到 T 个基分类器，最终将这些基分类器线性组合得到最终分类器模型。^[4] GBDT 分类首先初始化一个弱分类器，计算损失函数的负梯度值值，再利用数据集拟合下一轮模型，重复计算负梯度值和拟合过程，利用 m 个基础模型，构建梯度提升树。^[5] 随机森林则选取了大量的决策树模型，各决策树独立的做出学习并进行分类，最后将分类结合为一个最终的结果，其优于单个决策树做出的分类结果。

SVM 是较为强大的传统机器学习算法。它将低维线性不可分的空间转换为高维线性可分的空间。本文主要应用非线性的 SVM 模型，其目标函数为：

$$\begin{cases} \min_a \left(\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j \left(\oint(x_i) * \oint(x_j) \right) - \sum_{i=1}^n a_i \right) \\ s.t. \quad \sum_{i=1}^n a_i y_i = 0 \\ 0 \leq a_i \leq C \end{cases}$$



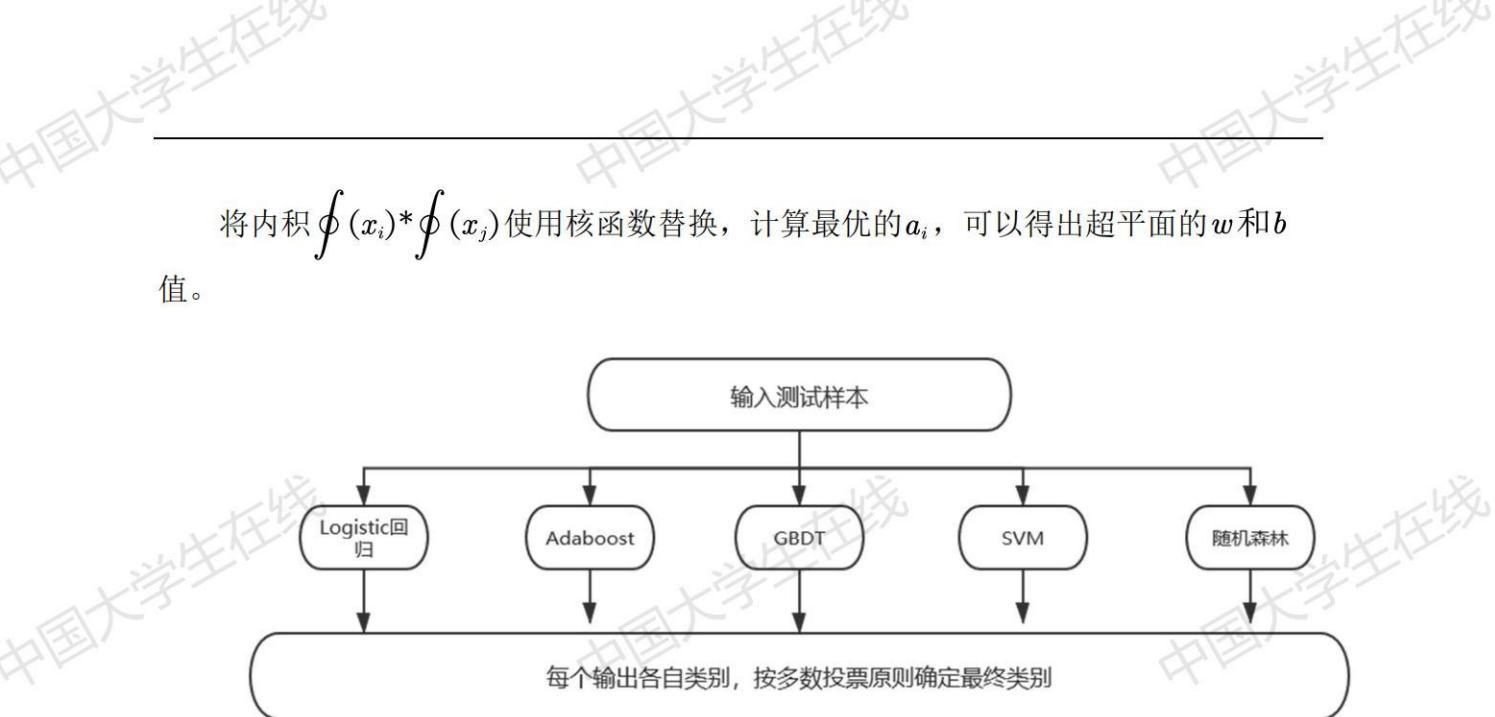


图 1 Voting 模型图解

5.3.4 模型求解

首先对 5 个基分类进行数据拟合和调参，选取较优参数，得出 5 个基分类器在改数据集上表现，如下表所示：

表 1 基分类器分类效果

分类器	测试集准确率	训练集准确率	综合准确率	AUC 值
Logistic 回归	0.839	0.902	0.886	0.754
Adaboost	0.903	0.913	0.911	0.822
GBDT	0.806	1	0.951	0.942
SVM	0.871	0.978	0.951	0.889
随机森林	0.871	1	0.967	0.953

从表 1 可以看出，训练集和测试集效果最接近的为 Adaboost，其余的分类器有程度不一样的过拟合现象。在整体上，Logistic 回归明显略逊一筹，后三个分类器的结果较为接近，主要是三个分类器在训练集上都有良好的表现，而训练集数据占到了整体数据的 7 成。

然而仅从准确率的角度判断分类模型的结果是有失偏颇的，并且本文的目的是，所以本文加入了 AUC 值进行参考。AUC 值为 ROC 曲线覆盖的面积，其含义可以综合的考虑召回率(recall)，精确率(precision)和准确率(accuracy)多种指标，一般 AUC 值达到 0.8 以上分类模型可以接受，从这个角度来看，Logistic 回归不符合要求。

因为各个基分类器的分类效果不一样，所以本文选择软投票，依据分类器在整体数据上的综合准确率来判断各基分类器的权重，公式为：

$$W_i = \frac{Accuracy_i}{\sum_{i=1}^5 Accuracy_i} \quad (1)$$

软投票分类结果如下表，

表 2 软投票结果



分类器	测试集准确率	训练集准确率	综合准确率	AUC 值
Soft Voting	0.903	1	0.976	0.994

可以看出，Soft Voting 在分类上的表现十分完美，非常接近完美分类器，说明集成学习算法在此数据集上有着优秀的表现。

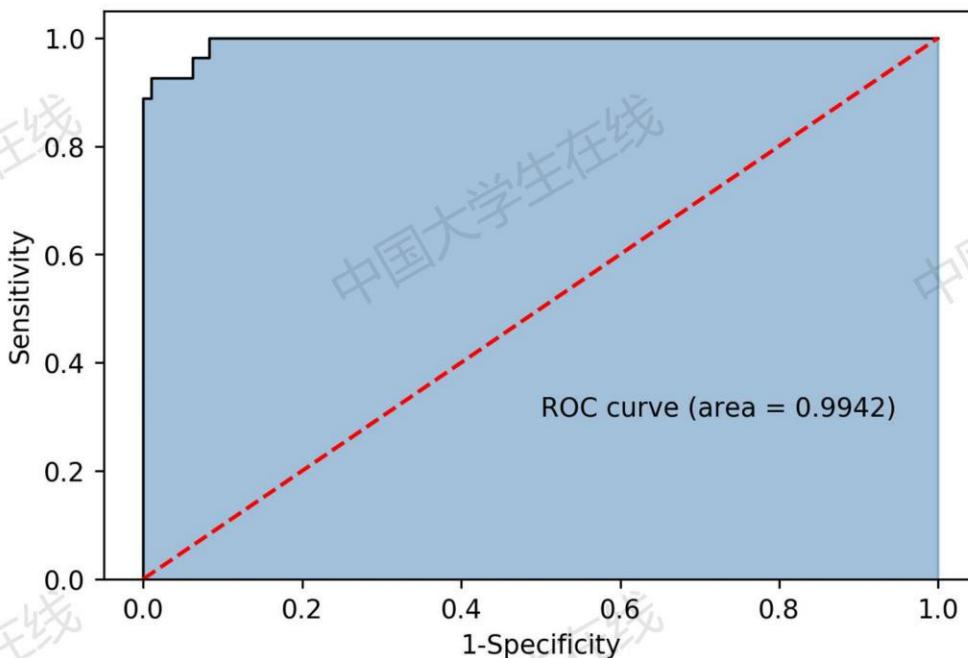


图 2 Soft Voting 的 ROC 曲线图

Voting 模型不仅可以判断出企业是否违约，并且可以给出企业违约的概率，本文将在后文以此作为对企业的信贷风险的度量。依据信誉评级分类，对不同类别企业的违约风险进行描述性统计分析，表格如下：

表 3 不同信誉评级下违约风险的描述性统计分析

信誉评级	Count	Mean	Std	Min	Max
A	27	0.145	0.068	0.079	0.435
B	38	0.182	0.108	0.101	0.757
C	34	0.209	0.147	0.088	0.638
D	24	0.710	0.170	0.237	0.882

5.4 信贷策略多目标规划模型

5.4.1 模型分析

根据假设 2，银行将对信誉评级在 C 及其以上并且之前没有违约记录的企业发放贷款，即符合条件的企业共有 96 家，其中 A 级企业 27 家，B 级企业 37 家和 C 级企业 32 家。银行的放贷决策需要考虑自身利益，即综合的考虑收益和风险，决定对各级企业的放贷额度和利率优惠，这将自然转化为一个多目标规划问题。

首先，由假设 1 可知银行认为同种信誉评级下的企业的违约风险一致，且在前文



中已经求出了每种类别的平均违约风险 $\bar{\sigma}_i$ ，故由每类企业的 $\bar{\sigma}_i$ 代表该类信誉评级下企业的违约风险。

5.4.2 模型建立与求解

收益目标：银行的目标之一为收益最大化，银行给予企业贷款得到的收益是利息收益，即

$$\max \sum_{i=1}^3 C_i * I_i * P_i \quad (2)$$

风险目标：银行的目标之一为风险最小，如果企业违约，即银行不仅收不回利息，也收不回本金，所以银行希望最小化损失，即

$$\min \sum_{i=1}^3 C_i * \bar{\sigma}_i * P_i * (1 + I_i) \quad (3)$$

潜在损失目标：银行会因为调高利率，而流失潜在客户，这些流失的客户也对银行造成了潜在损失，银行也希望吸引更多的企业来贷款，可以获得更多的利息收益。由假设 6 可知，已知附件来贷款的企业的数量，通过利率可以算出流失的潜在客户

数，即 $\frac{C_i * Lost(I_i)}{1 - Lost(I_i)}$ 。若流失了一个潜在客户，则银行就失去了贷款从而获得利息的机会，银行希望这笔损失最小，即

$$\min \sum_{i=1}^3 \frac{C_i * Lost(I_i)}{Lost(I_i)} * I_i * P_i \quad (4)$$

查阅资料，我们发现银行有对贷款额度进行分级的实践，不同信用等级的企业能贷的最大贷款和最低利率不同，所以银行应该对信用等级高的企业优先给予贷款，且提供数额较大，利率较低的贷款，以满足银行对收益和风险的追求。本文因此对不同信誉评级的企业贷款的额度和利率的上下限做出规定，以此来规范贷款行为，具体规则如下：

表 4 不同信誉等级下的贷款限制

信誉评级	利率下限	贷款额度上限
A	4%	100
B	7.5%	70
C	11%	40

综上所述，银行在贷款额度为固定数额的约束下（本问定为 5000 万），要使得满足上述三个目标，将上述三个目标转化为单一目标，列式可得：



$$\begin{aligned}
 & \min \sum_{i=1}^3 C_i * \bar{\sigma}_i * P_i * (1 + I_i) + I_i * P_i * \frac{C_i * \text{Lost}(I_i)}{1 - \text{Lost}(I_i)} - C_i * I_i * P_i \\
 & \text{s.t. } \left\{ \begin{array}{l} \sum_{i=1}^3 C_i * P_i = 50000000 \\ 0.04 \leq I_1 \leq 0.15 \\ 0.075 \leq I_2 \leq 0.15 \\ 0.11 \leq I_3 \leq 0.15 \\ 100000 \leq P_1 \leq 1000000 \\ 100000 \leq P_2 \leq 700000 \\ 100000 \leq P_3 \leq 400000 \end{array} \right. \quad (5)
 \end{aligned}$$

运用 MATLAB 求解可得：

$$\begin{aligned}
 I_1 &= 0.0585, I_2 = 0.075, I_3 = 0.11 \\
 P_1 &= 999905, P_2 = 535178, P_3 = 100030
 \end{aligned}$$

故对于附件一中队 123 家企业，银行的信贷策略是，对 D 级企业和有违约经历的企业拒绝提供贷款；剩下的企业中，对 A 级企业提供利率为 5.85%，额度为 999905 元的贷款；对 B 级企业提供利率为 7.5%，额度为 535178 元的贷款；对 C 级企业提供利率为 11%，额度为 100030 元的贷款。

六、问题二建模与求解

6.1 数据处理与编码

对附件 2 进行和附件 1 一样的数据处理方式，提取出问题一所列的 20 个特征。

将企业的信誉评级数据进行编码，若信誉评级为 A，则 $Grade = 1$ ；若信誉评级为 B，则 $Grade = 2$ ；若信誉评级为 C，则 $Grade = 3$ ；若信誉评级为 D，则 $Grade = 4$ 。

6.2 模型建立

6.2.1 违约风险预测

根据问题一所训练的 Voting 模型，运用附件 2 中所提取的 20 个特征进行预测附件 2 中 302 家企业的违约风险，判断企业是否会违约。

最终得到 302 个违约风险中，有 34 家企业的违约风险超过了 50%，即认为会违约，银行将不会向他们提供贷款。

6.2.2 信誉评级鉴定

在问题一中，我们已知 123 家企业的信誉评级并以此来确定贷款的发放，但本问的企业由于没有信贷记录，所以没有信誉评级。依据假设 3，银行将通过附件提供数据对企业进行信誉评级，然后据此做出信贷策略。基于 XGBoost 的在分类问题上的良好



表现，本问将使用 *XGBoost* 模型进行对企业进行信誉评级。

XGBoost 是一种 *Boosting* 型的树集成模型，在梯度提升决策树 *GBDT* 基础上扩展，能够进行多线程并行计算，通过迭代生成新树，即可将多个分类性能较低的弱学习器组合为一个准确率较高的强学习器。*XGBoost* 采用随机森林对字段抽样，将正则项引入损失函数中，从而防止模型过拟合，并降低模型计算量。具体算法步骤如下：

1、优化目标。假设模型具有 k 个决策树，即：

$$\hat{y}_i = \sum_{i=1}^k f_k(x_i), \quad f_k \in F$$

上式中， x_i 是第 i 个输入样本； y_i 为经过映射关系 f_k 计算出的预测值； F 为所有映射关系集合。优化目标以及损失函数为：

$$L(t) = \sum_{i=1}^n l[y_i, \hat{y}(t-1)_i + f_t(x_i)] + \Omega(f_t)$$

上式中， $L(t)$ 为第 t 次迭代时目标函数， n 为样本数量， I 为损失函数， $\hat{y}(t-1)_i$ 为第 $t-1$ 次迭代时模型的预测值， $f_t(x_i)$ 为新加入的函数， $\Omega(f_t)$ 为正则项。

2、对 $L(t)$ 进行二阶泰勒展开和移除常数项操作后可得：

$$\tilde{L}(t) \cong \sum_{j=1}^T \left[\left(\sum_{i \in I_j} d_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} g_i + \lambda \right) w_j^2 \right] + \gamma N$$

上式中， d_i 为 l 对 $\hat{y}(t-1)_i$ 的一阶导数； g_i 为对 $\hat{y}(t-1)_i$ 的二阶导数， N 为叶子节点个数， I_j 为每个叶子结点上样本集合， w_j^2 为每个叶子结点分数的 L_2 模的平方， λ 和 γ 则为比重系数，防止产生过拟合^[3]。

调用 Python 的 `xgboost` 库，对 21 个特征（附件 2 提取出的 20 个特征加上是否违约）进行训练。*XGBoost* 在附件 1 上的综合准确率为 0.878，在多分类问题中表现较好。

将信誉评级的分类结果和违约风险的分类结果进行比较验证，发现所有的 D 级企业均为预测会违约企业，除此之外只有两个 C 级企业预测会违约。两个不同模型训练的结果较为相近，进一步验证了本问分类的精确性。

详细分类结果见附录。



6.3 信贷策略目标规划

根据 *Xgboost* 算法对 302 家企业进行的信誉分类以及假设 2，银行将对符合标准的 268 家企业进行贷款发放。其中 A 类企业有 63 家，B 类企业有 103 家，C 类企业有 102 家。本问仍然使用问题一中对不同信誉评级的贷款限制，调整后的信贷策略目标规划模型如下：

$$\begin{aligned} \min \quad & \sum_{i=1}^3 C_i * \bar{\sigma}_i * P_i * (1 + I_i) + I_i * P_i * \frac{C_i * \text{Lost}(I_i)}{\text{Lost}(I_i)} - C_i * I_i * P_i \\ s.t. \quad & \left\{ \begin{array}{l} \sum_{i=1}^3 C_i * P_i = 100000000 \\ 0.04 \leq I_1 \leq 0.15 \\ 0.075 \leq I_2 \leq 0.15 \\ 0.11 \leq I_3 \leq 0.15 \\ 100000 \leq P_1 \leq 1000000 \\ 100000 \leq P_2 \leq 700000 \\ 100000 \leq P_3 \leq 400000 \end{array} \right. \end{aligned} \quad (6)$$

运用 MATLAB 求解可得：

$$\begin{aligned} I_1 &= 0.0705, I_2 = 0.0752, I_3 = 0.1101 \\ P_1 &= 999717, P_2 = 260320, P_3 = 100048 \end{aligned}$$

故对于附件二中的 302 家企业，银行的信贷策略是：对 D 级企业和有违约经历的企业拒绝提供贷款；剩下的企业中，对 A 级企业提供利率为 7.05%，额度为 999717 元的贷款；对 B 级企业提供利率为 7.52%，额度为 260320 元的贷款；对 C 级企业提供利率为 11.01%，额度为 100048 元的贷款。

七、问题三建模与求解

7.1 风险及其传导机制

本问加入了突发因素，需要考虑突发因素对企业的影响。根据题目所给数据，本问的突发事件确定为新冠疫情。本问需要综合考虑企业之前的信贷风险和新冠疫情对企业的冲击。

本文认为新冠疫情的爆发会影响企业的违约风险。新冠疫情首先对不同行业的企业有不同程度的影响，这属于系统风险的一种，是企业如果所处其行业必然会受到的影响，即疫情对行业的打击程度。

但是企业自身的特殊性，可能同一行业不同企业在疫情期间有不同的境遇，这属于非系统性风险，是由企业本身的特殊性所决定的，在数据上可以反映为企业在疫情期间的利润状况、较去年同期的同比增长率、废票所占比增长率和上下游供求企业数量的变化等等。

本文根据企业所受系统性风险（行业风险）和非系统性风险（自身表现）对企业



的进行聚类分析，同一类别的企业拥有相同的风险乘数 Q_i ，代表该类企业的违约风险较疫情未发生时增大的多少。银行将风险乘数加入信贷策略的目标规划模型，重新制定信贷策略。

7.2 数据处理和特征提取

7.2.1 企业状况

由于我们考虑疫情对企业的影响，因此我们主要对企业 2020 年 1、2 月份的盈利进行评估。用 2020 年 1、2 月份的销项金额减去进项金额即为盈利情况 π 。

当 $\pi > 0$ 时，公司盈利，令 Con_i 为 1；当 $\pi = 0$ 时，则公司没有在 2020 年进行交易，令 Con_i 为 0；当 $\pi < 0$ 时，公司亏损，令 Con_i 为 -1。

7.2.2 利润同比增长率

通过计算 2020 年 1、2 月份的企业利润 π 与 2019 年 1、2 月份的企业利润 π ，可以得出企业 2020 年 1、2 月份的利润同比增长率 $YG_i (i=1, \dots, 302)$ 。公式如下：

$$YG_i = \frac{\pi_{i, 2020} - \pi_{i, 2019}}{\pi_{i, 2019}}$$

7.2.3 废票占比变动率

废票指的时公司所开的无效票据与负数票据之和。由于负数票据是对冲掉企业之前已经入账的票据，一张负数票据其实代表了至少有两张票据是作废票据。废票占比变动率 $Error_{ij} (i=1, \dots, 302, j=1, 2)$ 即为 2020 年 1、2 月份的企业废票与总票据比例较 2019 年 1、2 月份企业废票与总票据比例的变动率，公式如下：

$$Error_{ij} = \frac{GU_{i, 2019} - GU_{i, 2020}}{GU_{i, 2019}}$$

7.2.4 交易企业数量变动率

交易企业数量变动率 $Tp_{ij} (i=1, \dots, 302, j=1, 2)$ 是指的公司在受疫情影响情况下，与公司交易企业数量的变动程度。公式如下：

$$Tp_{ij} = \frac{CC_{ij, 2020} - CC_{ij, 2019}}{CC_{ij, 2019}}$$



7.2.5 行业风险

通过国家对于行业的标准化分类方法，本文通过观察企业名称将 302 家企业一共分为了 13 个行业。针对其中 12 个行业，我们通过查询同花顺上的行业股票数据，得到了 12 个行业股票指数 2020 年 2 个月的增长率，用以衡量行业打击程度大小。额外的一个行业中的企业均为个体经营企业，根据第一财经发布的调研报告^[6]，个体经营企业预期降幅约为 1/3，本文以此数据作为对行业风险的度量。 Ir_i 计算公式如下：

$$Ir_i = \frac{2020\text{年}2\text{月底股票额} - 2020\text{年}1\text{月初股票额}}{2020\text{年}1\text{月初股票额}}$$

7.3 聚类模型

本文将上面计算的 5 种特征，作为系统聚类的依据。系统聚类首先将每一个样本都分为一类，然后不断计算子类与子类之间的距离，逐渐将所有的子类合并为一个大类。系统聚类算法的流程如下图所示。

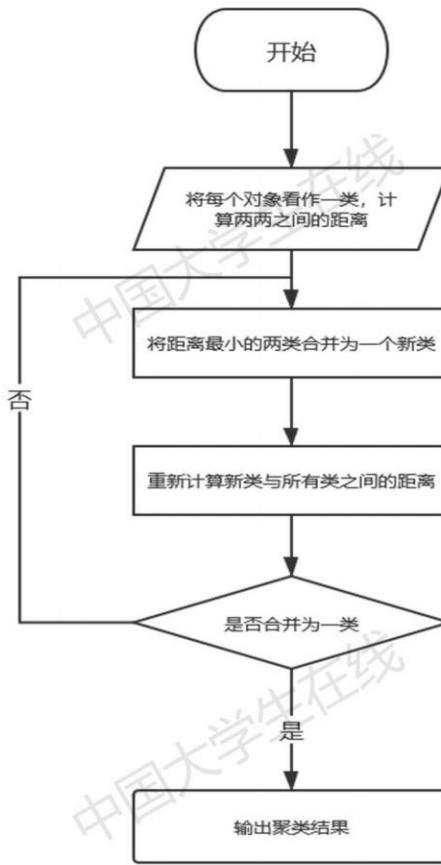


图 3 系统聚类流程图

将数据代入 SPSS，进行系统聚类，为了确定聚类类别 K ，画出聚类系数的折线图如下。



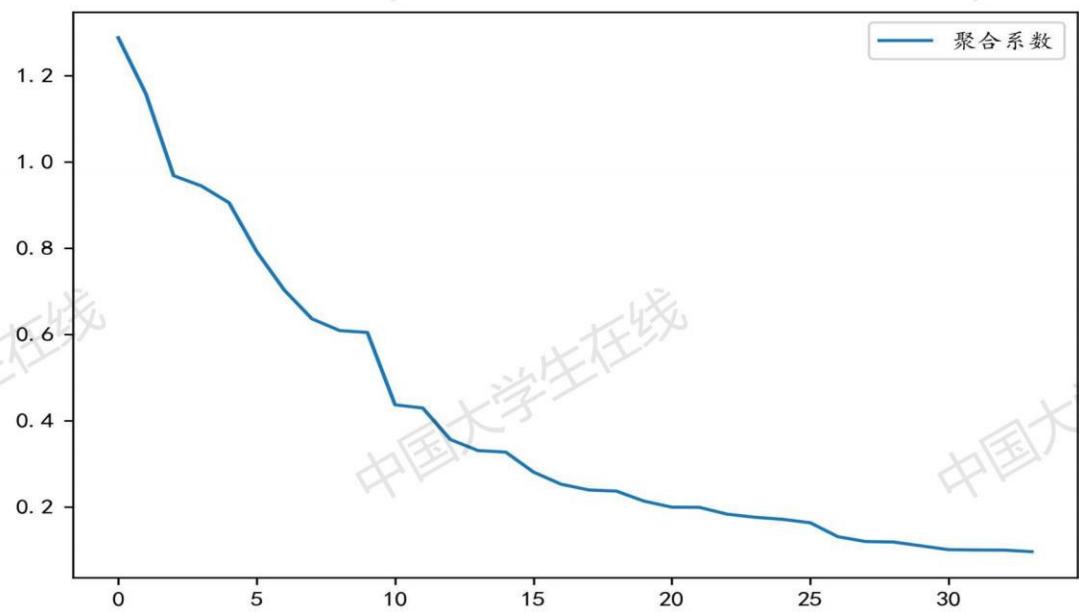


图 4 聚类系数折线图

根据聚合系数折线图, 可以看出, 令 $K=3$, 聚类效果最好。得出聚类结果后, 对聚类的结果分别进行描述性统计分析, 可得下面三表。

表 5 类别 1 企业特征描述性统计分析

指标	Con_i	YG_i	$Error_{i1}$	$Error_{i1}$	Tp_{i1}	Tp_{i2}	Ir_i
count	239	239	239	239	239	239	239
mean	0.996	-1.539	0.031	0.115	3.987	7.130	-0.120
min	0	-159.795	0	0	-1	-1	-0.33
max	1	33.787	0.25	0.839	116	101	0.126

表 6 类别 2 企业特征描述性统计分析

指标	Con_i	YG_i	$Error_{i1}$	$Error_{i1}$	Tp_{i1}	Tp_{i2}	Ir_i
count	59	59	59	59	59	59	59
mean	-0.932	0.301	0.033	0.116	1.458	6.677	-0.09
min	-1	-3.586	0	0	-1	-1	-0.33
max	0	9.020	0.125	0.734	27	93	0.126

表 7 类别 3 企业特征描述性统计分析

指标	Con_i	YG_i	$Error_{i1}$	$Error_{i1}$	Tp_{i1}	Tp_{i2}	Ir_i
count	4	4	4	4	4	4	4
mean	-1	1	0.047	0.934	-1	-1	-0.33
min	-1	1	0.019	0.909	-1	-1	-0.33



max	-1	1	0.067	0.955	-1	-1	-0.33
-----	----	---	-------	-------	----	----	-------

综合上三表数据，类别 1 的企业大部分为 2020 年依然盈利或者不盈不亏；平均 $Error_{i1}$ 较小， Tp_{ij} 大部分为正且数值较大，说明该类企业在疫情期间虽然盈利少于之前，但是供求关系仍然稳定甚至订单数较去年同期更多；类别 2 的企业大部分为亏损状态， $Error_{i1}$ 和 Tp_{ij} 的表现差于类别 1 的企业；类别 3 的企业只有 4 个，且全为高风险，差表现的个体经营企业。

根据以上聚类出的三类行业的表现，本文设置其对应的风险乘数为：

$$Q_1 = 1.2, Q_2 = 1.5, Q_3 = 2$$

7.4 信贷策略

在问题二中本文将企业依据信誉评级分为 4 类，本问中根据疫情对企业的影响将企业分为 3 类，理论来说企业被分为了 12 类。但在银行的信贷策略中，对认为将违约和 D 级企业不放贷款且聚类结果的类别 3 全为 D 即企业，对数据进行观察可知，数据被分为 6 类，银行也将根据信誉评级和聚类结果，建立一个新的评价分级体系，记为 $A_1, A_2, B_1, B_2, C_1, C_2$ 。将风险乘数 Q_i 加入信贷目标规划模型，调整企业风险，例如对

于 A_1 类企业，其风险为 $(1 + Q_1) * \bar{\sigma}_A$ 。建立调整后的多目标规划模型如下：

$$\begin{aligned} \min & \sum_{i=1}^3 \sum_j C_i * (1 + Q_j) \bar{\sigma}_i * P_i * (1 + I_i) + I_i * P_i * \frac{C_i * Lost(I_i)}{1 - Lost(I_i)} - C_i * I_i * P_i \\ \text{s.t.} & \left\{ \begin{array}{l} \sum_{i=1}^6 C_i * P_i = 10000000 \\ 0.04 \leq I_1 \leq 0.15 \\ 0.058 \leq I_2 \leq 0.15 \\ 0.076 \leq I_3 \leq 0.15 \\ 0.094 \leq I_4 \leq 0.15 \\ 0.112 \leq I_5 \leq 0.15 \\ 0.13 \leq I_6 \leq 0.15 \\ 100000 \leq P_1 \leq 1000000 \\ 100000 \leq P_2 \leq 850000 \\ 100000 \leq P_3 \leq 700000 \\ 100000 \leq P_4 \leq 550000 \\ 100000 \leq P_5 \leq 400000 \\ 100000 \leq P_6 \leq 250000 \end{array} \right. \end{aligned} \quad (7)$$

使用 MATLAB 的 fmincon 函数解出答案。银行在综合考虑企业信贷风险和疫情对企业影响后的信贷策略如下表：



表 8 银行信贷调整策略

企业类别	企业数量	贷款利率	贷款金额
A_1	54	7.60%	999988
A_2	9	6.25%	849863
B_1	82	11.20%	317698
B_2	21	9.40%	100017
C_1	87	11.20%	100002
C_1	15	13%	100006

八、模型评价

8.1 模型优点

- 1、数据特征提取时，考虑到企业实力、对上下游企业的影响能力、发展潜力、抗风险能力，较为全面；
- 2、使用强分类器，构建 Voting 集成学习算法，AUC 值接近完美模型，得到的违约风险可信度高；
- 3、多目标规划不仅考虑了常见的风险和收益，也考虑了潜在客户流失所造成的损失。
- 4、考虑疫情对企业影响时，综合考虑了系统风险和非系统风险的影响。

8.2 模型缺点

- 1、对于客户流失率选择的时候，一段区间内的贷款年利率用的是一个客户流失率，没有考虑小范围年利率变化时的客户流失率。
- 2、特征提取过多，有些变量互相有相关性。

8.3 模型改进方向

- 1、可以使用函数对附件 3 的数据进行拟合，将离散的客户流失率连续化，多目标规划的结果会更精确。
- 2、在不对分类效果有较大影响的前提下，可以考虑使用因子分析简化特征数量。

九、参考文献

- [1] 王晓燕.上市民营企业信用风险度量方法研究[D].山东大学,2020.
- [2] 刘琪.小额贷款公司个人贷款信用风险评估研究[D].扬州大学,2011.
- [3] 顾洲一.基于 XGBoost 模型的银行信贷高风险客户识别研究——以我国 Y 银行



为例[J].上海立信会计金融学院学报,2020,(1):17-28.

- [4] 孟叶,于忠清,周强.基于集成学习的股票指数预测方法[J].现代电子技术,2019,42(19):115-118. DOI:10.16652/j.issn.1004-373x.2019.19.027.
- [5] 王金柱,王翔.从零开始学 Python 数据分析与挖掘[M].北京:清华大学出版社,2018.
- [6] 祝嫣然,计亚.疫情影响调研报告:个体户、民企受影响最为严重.
<https://m.yicai.com/news/100570014.html.2020.9.13>

十、附录

第二问结果表

企业代号	是否违约	评级	企业代号	是否违约	评级
E124	0	B	E275	0	B
E125	0	B	E276	0	B
E126	0	A	E277	0	C
E127	0	B	E278	0	C
E128	0	B	E279	0	C
E129	0	C	E280	0	C
E130	0	B	E281	0	B
E131	0	C	E282	0	C
E132	0	A	E283	0	C
E133	0	B	E284	0	A
E134	0	B	E285	0	B
E135	0	B	E286	0	A
E136	0	B	E287	0	B
E137	0	C	E288	0	B
E138	0	A	E289	0	B
E139	0	B	E290	0	A
E140	0	B	E291	1	D
E141	0	A	E292	0	B
E142	0	A	E293	1	D
E143	0	A	E294	0	B
E144	0	A	E295	0	C
E145	0	A	E296	0	B
E146	0	B	E297	0	C
E147	0	B	E298	0	B
E148	0	A	E299	0	B
E149	0	B	E300	0	B
E150	0	A	E301	0	A
E151	0	B	E302	0	C
E152	0	B	E303	0	B
E153	0	C	E304	0	B



E154	0	B	E305	0	B
E155	0	C	E306	0	B
E156	0	B	E307	0	B
E157	0	C	E308	0	C
E158	0	B	E309	0	C
E159	0	C	E310	0	C
E160	0	A	E311	0	A
E161	0	B	E312	0	B
E162	0	A	E313	0	B
E163	0	B	E314	0	B
E164	0	B	E315	0	A
E165	0	A	E316	0	B
E166	0	B	E317	0	B
E167	0	A	E318	0	B
E168	0	C	E319	0	C
E169	0	C	E320	0	C
E170	0	B	E321	0	B
E171	0	A	E322	0	B
E172	0	A	E323	1	D
E173	0	A	E324	0	A
E174	0	B	E325	0	A
E175	0	A	E326	0	A
E176	0	A	E327	0	C
E177	0	B	E328	0	C
E178	0	A	E329	0	C
E179	0	A	E330	0	A
E180	0	A	E331	0	C
E181	0	B	E332	0	B
E182	0	C	E333	0	B
E183	0	B	E334	0	C
E184	0	B	E335	0	C
E185	0	A	E336	0	C
E186	0	B	E337	0	C
E187	1	D	E338	0	C
E188	0	A	E339	0	C
E189	0	C	E340	0	C
E190	0	B	E341	0	C
E191	0	B	E342	0	C
E192	0	C	E343	0	C
E193	0	C	E344	0	C
E194	0	A	E345	0	B
E195	0	A	E346	0	B



E196	0	A	E347	0	B
E197	0	A	E348	0	C
E198	0	B	E349	1	C
E199	0	B	E350	0	B
E200	0	B	E351	0	B
E201	0	A	E352	0	C
E202	0	C	E353	0	C
E203	0	B	E354	0	C
E204	0	A	E355	0	B
E205	0	C	E356	0	C
E206	0	B	E357	0	C
E207	0	B	E358	0	C
E208	0	B	E359	0	C
E209	0	A	E360	0	C
E210	0	B	E361	0	A
E211	0	B	E362	0	C
E212	0	A	E363	0	B
E213	0	A	E364	0	C
E214	0	B	E365	0	C
E215	0	B	E366	0	C
E216	0	A	E367	0	C
E217	1	D	E368	1	D
E218	1	D	E369	0	C
E219	0	B	E370	0	C
E220	0	A	E371	0	C
E221	0	B	E372	0	B
E222	0	A	E373	0	B
E223	0	A	E374	0	C
E224	0	B	E375	0	B
E225	0	A	E376	0	C
E226	0	B	E377	1	D
E227	0	B	E378	0	B
E228	0	A	E379	1	D
E229	0	C	E380	0	C
E230	0	A	E381	0	B
E231	1	D	E382	0	C
E232	0	C	E383	1	D
E233	0	B	E384	1	D
E234	0	A	E385	1	D
E235	0	A	E386	0	C
E236	0	C	E387	0	C
E237	0	B	E388	0	B



E238	0	C	E389	0	C
E239	0	C	E390	1	D
E240	1	D	E391	0	C
E241	0	B	E392	0	A
E242	1	D	E393	0	A
E243	0	B	E394	1	D
E244	0	C	E395	1	D
E245	0	B	E396	1	D
E246	0	B	E397	0	C
E247	0	C	E398	0	C
E248	0	A	E399	0	C
E249	1	C	E400	0	C
E250	0	A	E401	1	D
E251	0	B	E402	1	D
E252	0	C	E403	0	C
E253	0	A	E404	1	D
E254	0	C	E405	1	D
E255	0	B	E406	0	B
E256	0	A	E407	1	D
E257	0	C	E408	0	C
E258	0	A	E409	0	C
E259	0	C	E410	0	C
E260	0	A	E411	1	D
E261	0	A	E412	0	C
E262	0	C	E413	0	C
E263	0	B	E414	0	C
E264	1	D	E415	0	C
E265	1	D	E416	1	D
E266	0	B	E417	1	D
E267	0	C	E418	0	C
E268	0	B	E419	0	B
E269	0	C	E420	0	B
E270	0	A	E421	0	C
E271	0	C	E422	0	C
E272	0	C	E423	1	D
E273	0	C	E424	1	D
E274	0	C	E425	1	D



Matlab 代码

feixianxing.m

```
1. %27 37 32
2. % 分级之后的规划
3. clc;clear;
4. load data.mat
5.
6. x0=[0.10,0.10,0.10,50,50,50];
7. Aeq=[0,0,0,27,37,32];
8. beq=[5000];
9. lb=[0.04;0.075;0.11;10;10;10];
10. ub=[0.15;0.15;0.15;100;70;40];
11. %ub=[0.075;0.11;0.15;100;70;40];
12. [x,fval,ex]=fmincon(@fun,x0,[],[],Aeq,beq,lb,ub)
13.
14. 27*x(4)+37*x(5)+32*x(6)
15.
16.
17. 27+37+32
18.
19.
20. %% 第二问
21. % 总计 63 103 104 32
22. clc;clear;
23. load data.mat
24.
25. % 63+103+102 % 不违约
26.
27.
28. x0=[0.10,0.10,0.10,50,50,50];
29. Aeq=[0,0,0,63,103,102];
30. beq=[10000];
31. lb=[0.04;0.075;0.11;10;10;10];
32. ub=[0.15;0.15;0.15;100;70;40];
33. %ub=[0.075;0.11;0.15;100;70;40];
34. [x,fval,ex]=fmincon(@fun1,x0,[],[],Aeq,beq,lb,ub)
35.
36. %% 第三问
37. clc;clear;
38. load data.mat
```



```
39. fprintf("不违约的企业数:%d",54+9+82+21+87+15)
40. x0=[0.04,0.04,0.04,0.04,0.04,0.04,10,10,10,10,10,10];
41. Aeq=[0,0,0,0,0,54,9,82,21,87,15];
42. beq=[10000];
43. lb=[0.04;0.058;0.076;0.094;0.112;0.13;10;10;10;10;10];
44. ub=[0.15;0.15;0.15;0.15;0.15;0.15;100;85;70;55;40;25];
45. [x,fval,exitflag]=fmincon(@bbb,x0,[],[],Aeq,beq,lb,ub)
```

bbb.m

```
1. function f=bbb(x)
2. load data.mat
3. a1=0.14506157936837127 ; a2=0.1816959402725437 ; a3=0.2085706859586268 ;
4. q1=1.2 ;q2=1.5;
5. t1=54*x(1)*x(7)+9*x(2)*x(8)+82*x(3)*x(9)+21*x(4)*x(10)+87*x(5)*x(11)+15*x(6)*x(12);
6. t2=54*a1*q1*x(7)+9*a1*q2*x(8)+82*a2*q1*x(9)+21*a2*q2*x(10)+87*a3*q1*x(11)+15*a3*q2*x(12);
7.
8. for i=1:length(data)
9. if x(1)>data(i,1) && x(1)<=data(i+1,1)
10. m1=i;
11. break;
12. end
13. end
14. for j=1:length(data)
15. if x(2)>data(j,1) && x(2)<=data(j+1,1)
16. m2=j;
17. break;
18. end
19. end
20. for k=1:length(data)
21. if x(3)>data(k,1) && x(3)<=data(k+1,1)
22. m3=k;
23. break;
24. end
25. end
26. for l=1:length(data)
27. if x(4)>data(l,1) && x(4)<=data(l+1,1)
28. m4=l;
29. break;
30. end
31. end
32. for m=1:length(data)
33. if x(5)>data(m,1) && x(5)<=data(m+1,1)
34. m5=m;
```



```
35. break;
36. end
37. end
38. for n=1:length(data)
39. if x(6)>data(n,1) && x(6)<=data(n+1,1)
40. m6=n;
41. break;
42. end
43. end
44. t3=(data(m1,2)*54*x(1)*x(7))/(1-data(m1,2))+(data(m2,2)*9*x(2)*x(8))/(1-
    data(m2,2))+ (data(m3,3)*82*x(3)*x(9))/(1-data(m3,3))+ (data(m4,3)*21*x(4)*x(10))/(1-
    data(m4,3))+ (data(m5,4)*87*x(5)*x(11))/(1-data(m5,4))+ (data(m6,4)*15*x(6)*x(12))/(1-data(m6,4));
45.
46. f=t2+t3-t1;
47. end
```

fun.m

```
1. function f=fun(x)
2. load data.mat
3. a1=0.14506157936837127 ; a2=0.1816959402725437 ; a3=0.2085706859586268 ;
4. t1=27*x(1)*x(4)+37*x(2)*x(5)+32*x(3)*x(6);
5. t2=27*a1*x(4)*(1+x(1))+37*a2*x(5)*(1+x(2))+32*a3*x(6)*(1+x(3));
6.
7. for i=1:length(data)
8. if x(1)>data(i,1) && x(1)<=data(i+1,1)
9. m1=i;
10. break;
11. end
12. end
13. for j=1:length(data)
14. if x(2)>data(j,1) && x(2)<=data(j+1,1)
15. m2=j;
16. break;
17. end
18. end
19. for k=1:length(data)
20. if x(3)>data(k,1) && x(3)<=data(k+1,1)
21. m3=k;
22. break;
23. end
24. end
25. t3=(data(m1,2)*27*x(1)*x(4))/(1-data(m1,2))+(data(m2,3)*37*x(2)*x(5))/(1-
    data(m2,3))+ (data(m3,4)*32*x(3)*x(6))/(1-data(m3,4));
```



26.
27. f=t2-t1+t3;
28. end

fun1.m

```
1. function f=fun1(x)
2. % 63+103+102
3. load data.mat
4. q1 = 1.2;q2=1.5;
5. a1=0.14506157936837127 ;a2=0.1816959402725437 ;a3=0.2085706859586268 ;
6. t1=63*x(1)*x(4)+103*x(2)*x(5)+102*x(3)*x(6);
7. t2=63*a1*x(4)*(1+x(1))+103*a2*x(5)*(1+x(2))+102*a3*x(6)*(1+x(3));
8.
9. for i=1:length(data)
10. if x(1)>data(i,1) && x(1)<=data(i+1,1)
11. m1=i;
12. break;
13. end
14. end
15. for j=1:length(data)
16. if x(2)>data(j,1) && x(2)<=data(j+1,1)
17. m2=j;
18. break;
19. end
20. end
21. for k=1:length(data)
22. if x(3)>data(k,1) && x(3)<=data(k+1,1)
23. m3=k;
24. break;
25. end
26. end
27. t3=(data(m1,2)*63*x(1)*x(4))/(1-data(m1,2))+(data(m2,3)*103*x(2)*x(5))/(1-
data(m2,3))+(data(m3,4)*102*x(3)*x(6))/(1-data(m3,4));
28.
29. f=t2-t1+t3;
30. end
```

Python 代码

第一问

```
1. # 第一问集成学习
```



```
2. data = pd.read_csv('第一问所有特征.csv',encoding='gbk',index_col='企业代号')
3. for i in range(len(data)):
4.     a='E'+str(i+1)
5.     if data.loc[a,'是否违约']=='否':
6.         data.loc[a,'违约']=0
7.     else :
8.         data.loc[a,'违约']=1
9.
10. x = data.iloc[:, :-3].values
11. y = data.iloc[:, -1].values
12.
13. from sklearn.linear_model import LogisticRegression
14. from sklearn.model_selection import train_test_split
15. from sklearn.preprocessing import StandardScaler
16. from sklearn.model_selection import GridSearchCV
17. from sklearn import metrics
18. from sklearn.ensemble import AdaBoostClassifier as ada
19. from sklearn.ensemble import GradientBoostingClassifier
20. from sklearn.svm import SVC
21. from sklearn.ensemble import RandomForestClassifier as RF
22. from sklearn.model_selection import cross_val_score
23. from sklearn.metrics import roc_auc_score
24. from sklearn.ensemble import VotingClassifier
25.
26.
27. x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=30)
28.
29. tranfer = StandardScaler()
30. x = tranfer.fit_transform(x)
31. x_train = tranfer.transform(x_train)
32. x_test = tranfer.transform(x_test)
33.
34. LR = LogisticRegression(C=0.1, class_weight=None, dual=False, fit_intercept=True,
35.                         intercept_scaling=1, l1_ratio=None, max_iter=100,
36.                         multi_class='auto', n_jobs=None, penalty='l2',
37.                         random_state=None, solver='newton-cg', tol=0.0001, verbose=0,
38.                         warm_start=False)
39. Ada = ada(algorithm='SAMME', base_estimator=None, learning_rate=0.1,
40.             n_estimators=100, random_state=30)
41. GBDT = GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None
42.                                     ,
43.                                     learning_rate=0.7, loss='exponential', max_depth=3,
```



```
43.             max_features='auto', max_leaf_nodes=None,
44.             min_impurity_decrease=0.0, min_impurity_split=None,
45.             min_samples_leaf=1, min_samples_split=2,
46.             min_weight_fraction_leaf=0.0, n_estimators=25,
47.             n_iter_no_change=None, presort='deprecated',
48.             random_state=30, subsample=1.0, tol=0.0001,
49.             validation_fraction=0.1, verbose=0,
50.             warm_start=False)
51. svc = SVC(C=0.8, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
52.             decision_function_shape='ovr', degree=3, gamma=20, kernel='rbf',
53.             max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001,
54.             verbose=False)
55.
56. rf = RF(bootstrap=True, ccp_alpha=0.0, class_weight=None,
57.             criterion='gini', max_depth=None, max_features='auto',
58.             max_leaf_nodes=None, max_samples=None,
59.             min_impurity_decrease=0.0, min_impurity_split=None,
60.             min_samples_leaf=1, min_samples_split=2,
61.             min_weight_fraction_leaf=0.0, n_estimators=100,
62.             n_jobs=None, oob_score=False, random_state=30, verbose=0,
63.             warm_start=False)
64.
65. weight = []
66. for clf, label in zip([LR, Ada, GBDT, svc, rf, sclf],
67.                         ['LR',
68.                          'Ada',
69.                          'GBDT',
70.                          'svc',
71.                          'rf', 'StackingClassifier']):
72.     clf.fit(x_train, y_train)
73.     y_predict = clf.predict(x_test)
74.     print('{}在预测集模型的准确率为:\n'.format(label), metrics.accuracy_score(y_test, y_predict))
75.     print('{}在训练集模型的准确率为:\n'.format(label), metrics.accuracy_score(y_train, clf.predict(x_train)))
76.     print('{}的综合准确率为:\n'.format(label), metrics.accuracy_score(y, clf.predict(x)))
77.     tem = metrics.accuracy_score(y, clf.predict(x))
78.     weight.append(tem)
79.     print('{}的 ROC 面积为:\n'.format(label), metrics.roc_auc_score(y, clf.predict(x)))
80.     print()
```



```

81.
82. weight
83. del weight[-1]
84. # 软投票
85. w = weight/sum(weight)
86.
87. vote2= VotingClassifier(estimators=[('LR',LR),('Ada',Ada), ('GBDT',GBDT), ('SVC',svc
   ),('rf',rf)],

88.                               voting='soft',weights=weight)
89. vote2.fit(x_train,y_train)
90. y_predict = vote2.predict(x_test)
91. print('{}在预测集模型的准确率为:
   \n'.format('soft Voting'),metrics.accuracy_score(y_test,y_predict))
92. print('{}在训练集模型的准确率为:
   \n'.format('soft Voting'),metrics.accuracy_score(y_train,vote2.predict(x_train)))
93. print('soft voting 的综合表现:\n',metrics.accuracy_score(y,vote2.predict(x)))
94. print()
95.
96. P = vote2.predict_proba(x)[:,1]
97. df = pd.DataFrame(data={'违约概率':P})
98. df.to_csv('违约风险.csv',encoding='gbk')
99.
100.fpr,tpr,threshold = metrics.roc_curve(y,P)
101.roc_auc = metrics.auc(fpr,tpr)
102.
103.plt.figure(figsize=(6,4),dpi=250)
104.plt.stackplot(fpr,tpr,color='steelblue',alpha=0.5,edgecolor='black')
105.plt.plot(fpr,tpr,color='black',lw=1)
106.plt.plot([0,1],[0,1],color='red',linestyle='--')
107.plt.text(0.5,0.3,'ROC curve (area = %0.4f)' % roc_auc,fontsize=10)
108.plt.xlabel('1-Specificity')
109.plt.ylabel('Sensitivity')
110.plt.show()

```

第二问

```

1. from xgboost import XGBClassifier
2. new_da = pd.read_csv('第二问所有特征.csv',encoding='gbk')
3.
4. new_x = new_da.iloc[:,1:].values
5. new_x = tranfer.transform(new_x)
6. wieyue = vote2.predict(new_x)
7. sigma = vote2.predict_proba(new_x)[:,1]
8. new_da['是否违约']=wieyue

```



```

9. new_da['违约风险']=sigma
10.
11. x = data.drop(['是否违约','信誉评级'],axis=1).values # 21个特征
12. y = pd.read_csv('违约风险.csv',encoding='gbk')['评级'].values # 评价等级编码
13.
14. x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=30)
15.
16. tranfer = StandardScaler()
17. x = tranfer.fit_transform(x)
18. x_train = tranfer.transform(x_train)
19. x_test = tranfer.transform(x_test)
20.
21. other_params = {'learning_rate': 0.16, 'max_depth': 6, 'min_child_weight': 2, 'seed': 10,'estimator':60,
22.                 'subsample': 0.8, 'colsample_bytree': 0.9, 'gamma': 0.5, 'reg_alpha': 0.08, 'reg_lambda': 0.12}
23.
24. estimator = XGBClassifier(objective='multi:softmax',num_class=4,eval_metric='auc',**other_params)
25. y_predict = estimator.predict(x_test)
26. print('预测集模型的准确率为: \n', metrics.accuracy_score(y_test,y_predict))
27. print('训练集模型的准确率为: \n', metrics.accuracy_score(y_train,estimator.predict(x_train)))
28. print('综合准确率为: \n', metrics.accuracy_score(y,estimator.predict(x)))
29.
30. new = pd.read_csv('已经判断是否违约.csv',encoding='gbk')
31. new_x = new.iloc[:,1:-1].values
32. new_x = tranfer.transform(new_x)
33. predict_y = estimator.predict(new_x)
34. new['信誉评级'] = predict_y

```

第三问

```

1. import numpy as np
2. import pandas as pd
3. data01 = pd.read_csv('附件 2-1.csv',encoding='gbk',engine='python')
4. data021 = pd.read_csv('附件 2-21.csv',encoding='gbk',engine='python')
5. data031 = pd.read_csv('附件 2-31.csv',encoding='gbk',engine='python')
6.
7. ##2020/1/2
8. a=data021[((data021['开票日期']>'2020')&(data021['开票日期']<'2020/3'))]
9. a=a.reset_index()
10. tem1 = a['企业代号'].unique()
11. b=data031[((data031['开票日期']>'2020')&(data031['开票日期']<'2020/3'))]
12. b=b.reset_index()
13. tem2 = b['企业代号'].unique()
14. print("2020 年进项企业数: %d"%len(tem1))
15. print("2020 年销项企业数: %d"%len(tem2))

```



```

16. tmp1 = [val for val in list(tem1) if val in list(tem2)]
17. print("2020 年进项销项共有企业数: %d"%len(tmp1))
18. tmp2 = list(set(tem1).difference(set(tem2))) # 有进项无销项
19. tmp3 = list(set(tem2).difference(set(tem1))) # 有销项无进项
20.
21. #2019/1/2 销-进
22. a1=[]
23. a1=pd.DataFrame(a1)
24. A=data021[((data021['开票日期']>'2019')&(data021['开票日期']<'2019/3'))]
25. B=data031[((data031['开票日期']>'2019')&(data031['开票日期']<'2019/3'))]
26. for num in range(len(tmp01)):
27.     id1 = tmp01[num]
28.     a1.loc[num,'企业代号']=id1
29.     t1=A[A['企业代号']==id1]
30.
31.     a1.loc[num,'进项金额']=sum(t1['金额'])
32.
33.     t2=B[B['企业代号']==id1]
34.     a1.loc[num,'销项金额']=sum(t2['金额'])
35.
36. a1['销-进金额']=a1['销项金额']-a1['进项金额']
37.
38.
39. #2020/1/2 销-进
40. a2=[]
41. a2=pd.DataFrame(a2)
42. A=data021[(data021['开票日期']>'2020')]
43. B=data031[(data031['开票日期']>'2020')]
44. for num in range(len(tmp1)):
45.     id1 = tmp1[num]
46.     a2.loc[num,'企业代号']=id1
47.     t1=A[A['企业代号']==id1]
48.
49.     a2.loc[num,'进项金额']=sum(t1['金额'])
50.
51.     t2=B[B['企业代号']==id1]
52.     a2.loc[num,'销项金额']=sum(t2['金额'])
53.
54. a2['销-进金额']=a2['销项金额']-a2['进项金额']
55.
56.
57.

```



```
58. #2019/1/2
59. a1=[]
60. a1=pd.DataFrame(a1)
61. data021['开票日期']=pd.to_datetime(data021['开票日期'])
62. data031['开票日期']=pd.to_datetime(data031['开票日期'])
63. A=data021[((data021['开票日期']>'2019')&(data021['开票日期']<'2019/3'))]#进项 2019
64. B=data031[((data031['开票日期']>'2019')&(data031['开票日期']<'2019/3'))]#销项 2019
65. A=A.reset_index(drop=True)
66. B=B.reset_index(drop=True)
67. for num in range (302):
68.     id1 = data01['企业代号'][num]
69.     a1.loc[num,'企业代号']=id1
70.
71. for num in range (302):
72.     for i in range (len(A)):
73.         if A['企业代号'][i]==a1['企业代号'][num]:
74.             t1=A[A['企业代号']==A['企业代号'][i]]
75.             a1.loc[num,'进项金额']=sum(t1['金额'])
76.
77.     for j in range (len(B)):
78.         if B['企业代号'][j]==a1['企业代号'][num]:
79.             t2=B[B['企业代号']==B['企业代号'][j]]
80.             a1.loc[num,'销项金额']=sum(t2['金额'])
81. a1=a1.fillna(0)
82. a1['销-进金额']=a1['销项金额']-a1['进项金额']
83.
84.
85. #2020/1/2
86. a2=[]
87. a2=pd.DataFrame(a2)
88. data021['开票日期']=pd.to_datetime(data021['开票日期'])
89. data031['开票日期']=pd.to_datetime(data031['开票日期'])
90. A=data021[(data021['开票日期']>'2020')]#进项 2020
91. B=data031[(data031['开票日期']>'2020')]#销项 2020
92. A=A.reset_index(drop=True)
93. B=B.reset_index(drop=True)
94. for num in range (302):
95.     id1 = data01['企业代号'][num]
96.     a2.loc[num,'企业代号']=id1
97.
98. for num in range (302):
99.     for i in range (len(A)):
```



```
100.     if A['企业代号'][i]==a2['企业代号'][num]:  
101.         t1=A[A['企业代号']==A['企业代号'][i]]  
102.         a2.loc[num,'进项金额']=sum(t1['金额'])  
103.  
104.     for j in range (len(B)):  
105.         if B['企业代号'][j]==a2['企业代号'][num]:  
106.             t2=B[B['企业代号']==B['企业代号'][j]]  
107.             a2.loc[num,'销项金额']=sum(t2['金额'])  
108.  
109.a2=a2.fillna(0)  
110.a2['销-进金额']=a2['销项金额']-a2['进项金额']  
111.  
112.  
113.data01 = pd.read_csv('附件 2-1.csv',encoding='gbk',engine='python')  
114.a=[]  
115.a=pd.DataFrame(a)  
116.a['企业代号']=a2['企业代号']  
117.a['销-进金额']=a2['销-进金额']  
118.for i in range (302):  
119.    if a['销-进金额'][i]<0:  
120.        a.loc[i,'2020 企业状况']=-1  
121.    elif a['销-进金额'][i]== 0:  
122.        a.loc[i,'2020 企业状况']=0  
123.    else:  
124.        a.loc[i,'2020 企业状况']=1  
125.  
126.  
127.for i in range (302):  
128.    if a1['销-进金额'][i]==0:  
129.        if a2['销-进金额'][i]>0:  
130.            a.loc[i,'同比增长速度']=1  
131.        elif a2['销-进金额'][i]<0:  
132.            a.loc[i,'同比增长速度']=-1  
133.        else:  
134.            a.loc[i,'同比增长速度']=0  
135.    elif a1['销-进金额'][i]<0:  
136.        if a2['销-进金额'][i]==0:  
137.            a.loc[i,'同比增长速度']=1  
138.        else:  
139.            a.loc[i,'同比增长速度']=(a2['销-进金额'][i]-a1['销-进金额'][i])/a1['销-进金  
额'][i]  
140.    else:
```



```
141.         if a2['销-进金额'][i]==0:
142.             a.loc[i,'同比增长速度']=-1
143.         else:
144.             a.loc[i,'同比增长速度']=(a2['销-进金额'][i]-a1['销-进金额'][i])/a1['销-进金
145.               额'][i]
146.
147.
148. r_in=[]
149. for num in range(1,303):
150.     id1 = 'E'+str(num+123)
151.     tem = data02[data02['企业代号']==id1]
152.     all_num = tem.shape[0]
153.     fei_num = tem[tem['发票状态']=='作废发票'].shape[0]
154.     fu_num = tem[tem['发票状态']=='负数发票'].shape[0]
155.     ratio = (fei_num+fu_num*2)/all_num
156.
157.     r_in.append(ratio)
158.
159.
160. r_out=[]
161. for num in range(1,303):
162.     id1 = 'E'+str(num+123)
163.     tem = data03[data03['企业代号']==id1]
164.     all_num = tem.shape[0]
165.     fei_num = tem[tem['发票状态']=='作废发票'].shape[0]
166.     fu_num = tem[tem['发票状态']=='负数发票'].shape[0]
167.     ratio = (fei_num+fu_num*2)/all_num
168.
169.     r_out.append(ratio)
170. a['进项发票作废负数比例']=r_in
171. a['销项发票作废负数比例']=r_out
```

