队伍编号	35
题号	C

针对中小微企业最优信贷策略决策

摘 要

中小微企业因规模小、缺乏抵押资产，其信贷风险评估与策略制定是银行信贷业务的关键。本文基于银行提供的 123 家有信贷记录企业、302 家无信贷记录企业的发票数据及利率与客户流失率关系数据，构建数学模型解决中小微企业的信贷决策问题。该问题的研究可为银行对中小微企业的信贷决策提供科学量化模型，助力平衡风险与收益；同时能缓解中小微企业融资困境，促进其健康发展，服务国民经济稳定。

针对问题一，从企业发票数据中提取进项发票金额总计、销项发票金额总计、利润、四类异常发票率（进项 / 销项作废、进项 / 销项负数）共七项指标，采用随机森林算法量化 123 家企业的信贷风险，输出违约概率；结合银行信贷约束（额度 10-100 万元、利率 4%-15%），构建非线性规划模型，以总利润最大化为目标，通过遗传算法优化贷款额度与利率，形成最优信贷策略。

针对问题二，在问题一构建的模型框架基础上，针对附件 2 中 302 家无信贷记录企业的特性，采用 BP 神经网络模型对其信誉评级进行预测——以附件 1 中 123 家企业的发票特征（进项/销项发票金额、利润、异常发票率等）及对应信誉评级为训练样本，通过多层神经元的非线性映射学习特征与评级的关联规律，填补无信贷记录企业的信誉数据缺口。随后复用问题一的随机森林算法量化该 302 家企业的违约概率，再代入基于非线性规划的最优策略模型，在年度信贷总额 1 亿元的约束下，通过遗传算法优化贷款额度与利率，最终形成兼顾风险控制与利润最大化的信贷策略。

针对问题三，本文查阅相关资料，以疫情为突发事件为例，根据疫情对不同中小微企业的差异化影响，对数据 2 的销项发票信息和进项发票信息进行修改完善，得到变化后的数据，进而运用问题一和问题二的随机森林量化模型和使用遗传算法的决策模型计算得到最终的信贷决策结果。

关键词：随机森林算法；非线性规划：遗传算法；BP 神经网络；突发因素

目录

一、 问题重述

1.1 问题背景

中小微企业作为国民经济和社会发展的重要力量，在稳增长、促改革、调结构、惠民生、防风险中占据着不可或缺的地位，在扩大就业、改善民生、促进创业创新等方面发挥着至关重要的作用，但在实际发展过程中，融资困境始终是制约中小微企业快速发展的瓶颈。由于中小微企业规模相对较小、抵押资产普遍缺少，银行在提供信贷服务时需承担较高风险。因此，银行需基于企业实力、信誉等因素进行科学的风险评估，进而制定合理的信贷策略（包括是否放贷、贷款额度、利率、期限等），以平衡自身收益与风险控制。

1.2 问题描述

现某银行针对中小微企业的信贷政策已明确：贷款额度为 10~100 万元，年利率为 4%~15%，贷款期限为 1 年。同时，银行提供了三类核心数据：附件 1（123 家有信贷记录企业的相关数据）、附件 2（302 家无信贷记录企业的相关数据）、附件 3（2019 年贷款利率与客户流失率关系的统计数据），为信贷策略研究提供了基础信息。

基于上述背景和附件信息，我们需要建立数学模型解决以下问题：

（1）基于附件 1 中 123 家有信贷记录企业的数据，从企业信誉、交易票据等信息中提取影响信贷风险的关键因素并进行量化分析，在银行年度信贷总额固定的前提下，构建模型并给出对这些企业的最优信贷策略（包括是否放贷、贷款额度及利率）。

（2）在问题（1）构建的模型基础上，结合附件 2 中 302 家无信贷记录企业的销项发票与进项发票信息，对其信贷风险进行量化评估。在年度信贷总额为 1 亿元的约束下，制定针对这些企业的最优信贷策略。

（3）突发因素（如新馆病毒疫情等）对不同行业企业的生产经营和经济效益影响存在差异（例如新冠疫情对服务业冲击较大，对物流业可能产生促进作用）。需分析突发因素对不同行业企业的影响机制，综合附件 2 中企业的信贷风险与突发因素影响，在年度信贷总额为 1 亿元时，给出相应的信贷调整策略。

二、 问题分析

2.1 对问题 1 的分析

在银行信贷业务决策体系中，企业信贷风险是制定信贷策略的核心依据。针对问题 1，需基于 123 家企业发票数据，构建“风险量化 - 策略优化”的决策链路，具体分析逻辑如下：

结合题意与数据特征，企业信贷风险由七个核心指标（进项发票金额总计、销项发票金额总计、利润、进项作废发票率、进项负数发票率、销项作废发票率和销项负数发

票率）驱动，各指标与数据的关联及风险逻辑为：①进项发票金额总计：反映企业采购投入规模，体现生产/业务体量；②销项发票金额总计：对应企业销售收入规模，体现市场份额与资金回流潜力；③利润：计算公式为“销项发票金额总计 - 进项发票金额总计 - 增值税额”，扣除经营成本与税费，直接反映企业真实盈利水平；④作废发票率（进项/销项）：高频作废发票暗示交易计划频繁变动（如采购/订单取消），反映企业与上下游合作的不确定性；⑤负数发票率（进项/销项）：进项负数发票多因上游退货产生，暴露采购质量或合作问题；销项负数发票因下游退货产生，反映产品缺陷或客户流失。

基于上述七维因素，采用随机森林算法构建风险量化模型，实现“数据特征→违约概率”的映射。

在风险量化基础上，结合银行约束（年度信贷总额）与目标（利润最大化），构建非线性规划模型优化信贷策略。

2.2 对问题 2 的分析

针对问题二的研究，其核心逻辑与问题一保持一致——均需通过量化企业信贷风险制定最优信贷策略。但由于附件 2 中的 302 家企业缺乏信贷记录，直接应用问题一构建的模型（基于随机森林算法的信贷风险评价模型与基于遗传算法的最优信贷策略模型）存在数据缺口：这些企业既无历史违约记录作为风险标签，也缺失关键的信誉评级信息，而这两类数据是问题一模型中风险量化与策略优化的重要输入。因此，在复用问题一模型前，需先对这 302 家企业的信誉评级及潜在违约情况进行合理预测，以填补数据空白。

具体而言，问题一的模型已通过与企业信誉等级的校验，证明其在信贷风险评价与策略规划上的有效性，这为问题二的模型复用提供了基础。但附件 2 企业的“无信贷记录”特性，要求我们先建立预测模型，可以选择 BP 神经网络，将附件一中的数据作为训练样本，然后用训练完的神经网络来对附件二中的企业信誉等级进行预测；通过随机森林算法，实现对 302 家企业信贷风险的量化评价。将附件二中的信誉等级和违约概率求得后，在年度信贷总额 1 亿元的约束下，通过遗传算法优化贷款额度与利率，最终输出完整的信贷策略。

2.3 对问题 3 的分析

针对问题 3，即突发因素（如新冠病毒疫情）对不同行业企业的影响及其信贷调整策略，我们首先明确了突发事件的分类及其影响机制，认识到不同事件如自然灾害、事故灾难、公共卫生事件和社会安全事件，会通过供应链中断、市场需求变化等途径对企业经营产生复杂影响。

在模型构建阶段，我们依据突发事件的特性，对附件 2 中的企业发票数据进行了针对性修改，以反映疫情等突发事件下企业实际经营状况的变化。具体而言，我们调整了销项发票与进项发票的相关指标，模拟出疫情对企业资金流、利润及违约风险的实际影

响，为后续信贷风险量化提供了数据基础。

随后，我们复用了问题一中验证有效的随机森林算法，对修改后的数据进行信贷风险量化，重新计算了各企业的违约概率。一步骤确保了我们在突发事件背景下，仍能准确评估企业的信贷风险水平。

三、 模型假设

假设一：假设附件中所有企业的发票数据（包括进项/销项发票金额、作废及负数发票记录等）均真实有效，能客观反映企业实际经营状况，无虚假交易或票据伪造情况。

假设二：假设企业的信贷风险仅由所选取的七项核心指标（进项发票金额总计、销项发票金额总计、利润、进项作废发票率、进项负数发票率、销项作废发票率、销项负数发票率）及历史违约记录决定，不考虑未纳入模型的其他因素（如经营者个人信用、行业政策变动等）对风险的影响。

假设三：假设银行的信贷政策（贷款额度 10-100 万元、年利率 4%-15%、贷款期限 1 年）在研究周期内保持稳定，且客户流失率与利率、信誉评级的关系符合附件 3 中 2019 年的统计规律，无突发性市场因素导致该关系显著变化。

假设四：假设 123 家企业的历史违约记录可直接作为风险量化模型的标签数据，且企业当前的经营状态与历史信贷记录具有连续性，违约行为的驱动因素未发生根本性改变。

假设五：假设在问题 1 和 2 的研究范围内，不考虑突发因素（如自然灾害、公共卫生事件等）对企业经营的影响，企业的盈利水平、交易稳定性等指标在贷款期限内保持相对稳定。

四、 符号说明

表 1 符号说明表

符号	含义
X_a	销项发票金额
J_a	进项发票金额
T	需要缴纳的增值税额
X_t	销项发票税额
J_t	进项发票税额
P	每家企业的利润
P_{jzf}	进项作废发票率
n_{jzf}	进项作废发票数

N_j	总进项发票数
P_{jfs}	进项负数发票率
n_{jfs}	进项负数发票数
P_{xzf}	销项作废发票率
n_{xzf}	销项作废发票数
N_x	总销项发票数
P_{xfs}	销项负数发票率
n_{xfs}	销项负数发票数
a	对某企业的贷款额度
r	贷款利率
d	企业的违约概率
l_r	客户流失率
W	总贷款利润
T_o	银行年度信贷总额

五、 问题一模型的建立与求解

5.1 数据预处理

对附件中的数据进行预处理，包括：

（1）对作废发票进行筛选并剔除

附件中提供的数据数量庞大，我们需要对其中的作废发票进行过滤筛除，以便于之后的分析计算。

（2）筛除发票号相同的发票

在 excel 中通过删除重复项，对发票号码重复的发票进行处理，每个发票号码只保留一张发票，删去多余的重复值。

5.2 基于随机森林算法的信贷风险量化分析

根据 123 家企业信贷记录的相关数据，采用定量研究的分析方法，找出影响信贷风险的七个评价指标，即进项发票金额总计、销项发票金额总计、利润、进项作废发票率、进项负数发票率、销项作废发票率和销项负数发票率，并结合 123 家企业的七项指标及是否违约情况，通过随机森林算法，求出 123 家企业未来的违约风险概率。

首先对七个指标给出具体定义：

①进项发票金额总计

进项发票金额总计指企业在统计周期内所有有效进项发票的金额总和，代表企业为维持生产经营所投入的采购成本规模。进项发票是企业从上游供应商处获取的交易凭证，其总金额越高，说明企业的原材料采购、服务外包等投入越大，间接反映企业的生产规模或业务体量。该指标能直观体现企业与上游合作的深度和广度，是评估其经营基础的核心参数之一。

②销项发票金额总计

销项发票金额总计指企业在统计周期内所有有效销项发票的金额总和，对应企业通过销售产品或服务实现的收入规模。销项发票是企业向下游客户开具的交易凭证，其总金额不仅反映企业的市场份额和销售能力，还能体现其在产业链中的议价权。同时，销项与进项金额的大小关系可初步判断企业的盈利空间，为评估企业还款能力提供重要的参考。

③利润 P

定义：利润为销项发票金额 - 进项发票金额 - 需要缴纳的增值税额。其中，增值税额为“销项发票总税额 - 进项发票总税额”。

$$\begin{cases} P = \sum X_a - \sum J_a - T \\ T = \sum X_t - \sum J_t \end{cases} \tag{5.1}$$

式中，X_a 代表销项发票金额，J_a 代表进项发票金额，T 代表需要缴纳的增值税额，X_t 代表销项发票税额，J_t 代表进项发票税额。

该指标扣除了经营过程中的成本和应缴税款，直接反映企业的实际盈利水平。利润指标是连接企业经营规模与还款能力的关键纽带，能有效反映其可持续经营状况。

④进项作废发票率 P_{jzf}

进项作废发票率指企业进项发票中作废发票的数量占总进项发票数量的比例，计算公式为“进项作废发票数 ÷ 总进项发票数 × 100%”。

$$p_{jzf} = \frac{n_{jzf}}{N_j} \tag{5.2}$$

式中，P_{jzf} 代表进项作废发票率，n_{jzf} 代表进项作废发票数，N_j 代表总进项发票数。

进项发票作废通常因上游交易取消（如采购计划变更、合同终止）导致，合理比例的作废属于正常经营现象，但比例过高则可能反映企业与上游供应商的合作稳定性差，或存在虚假采购、票据管理混乱等问题。

⑤进项负数发票率 P_{jfs}

进项负数发票率指企业进项发票中负数发票的数量占总进项发票数量的比例，计算公式为“进项负数发票数 ÷ 总进项发票数 × 100%”。

$$p_{jfs} = \frac{n_{jfs}}{N_j} \tag{5.3}$$

式中，P_{jfs} 代表进项负数发票率，n_{jfs} 代表进项负数发票数，N_j 代表总进项发票数。

进项负数发票通常因企业向供应商退货并收到退款产生，反映上游交易的逆向流动。该比例过高可能说明企业采购的原材料或服务存在质量问题、规格不符等情况，导致交易纠纷频发。

⑥销项作废发票率 P_{xzf}

进项作废发票率指企业销项发票中作废发票的数量占总销项发票数量的比例，计算公式为“销项作废发票数 ÷ 总销项发票数 × 100%”。

$$p_{xzf} = \frac{n_{xzf}}{N_x} \tag{5.4}$$

式中，P_{xzf} 代表销项作废发票率，n_{xzf} 代表销项作废发票数，N_x 代表总销项发票数。

销项发票作废多因下游交易取消（如客户订单变更、合同解除）导致，比例过高反映企业与下游客户的合作稳定性不足，市场需求波动较大。

⑦销项负数发票率 P_{xfs}

销项负数发票率指企业销项发票中负数发票的数量占总销项发票数量的比例，计算公式为“销项负数发票数 ÷ 总销项发票数 × 100%”。

$$p_{xfs} = \frac{n_{xfs}}{N_x} \tag{5.5}$$

式中，P_{xfs} 代表销项负数发票率，n_{xfs} 代表销项负数发票数，N_x 代表总销项发票数。

销项负数发票因下游客户退货并要求退款产生，直接反映企业产品或服务的市场认可度。该比例越高，说明企业的产品质量、售后服务可能存在缺陷，导致客户满意度低，市场竞争力弱。

由上述影响信贷风险七个量化评价指标可知，信贷风险的评估是多因一果的过程，对结果产生影响的指标多且复杂。

为实现对 123 家有信贷记录企业的信贷风险量化评估，本文采用随机森林算法构建违约概率预测模型。随机森林算法通过多棵决策树的集成学习方式，能有效处理多维度特征交互问题，且对异常值和共线性数据具有较强稳健性，适合基于发票数据的信贷风险评估场景。

（1）数据输入与特征选择

模型输入数据为附件 1 中 123 家企业的七项核心指标（进项发票金额总计、销项发票金额总计、利润、进项作废发票率、进项负数发票率、销项作废发票率、销项负数发

票率）及历史违约记录。特征选择基于指标对信贷风险的影响机理：

①进项/销项发票金额总计反映企业经营规模，规模越大通常还款基础越稳固；

②利润直接体现企业盈利能力，是偿还贷款的核心保障；

③四类异常发票率（进项/销项作废、进项/销项负数）刻画交易规范性，比例越高说明经营稳定性越差，违约风险越高。

（2）模型训练与验证

①数据划分：采用分层抽样将数据集按 7:3 比例划分为训练集（86 家企业）和测试集（37 家企业），确保两组数据中违约与未违约企业的比例一致，避免样本偏差。

②算法参数设置：

决策树数量设为 100，通过多棵树的投票机制降低过拟合风险；

随机种子设为 42，保证模型结果可复现；

其他参数采用默认值，通过算法内部迭代优化节点分裂策略。

③模型训练过程：

训练集用于构建随机森林：每棵决策树基于 bootstrap 抽样的子数据集训练，节点分裂时随机选择部分特征（此处为全部 7 项指标），以最大化信息增益；

测试集用于验证模型性能，通过预测结果与实际违约记录的对比，评估模型的分类精度。

（3）模型输出与结果分析

①输出违约概率：通过随机森林的`predict_proba`函数输出每家企业的违约概率（取值范围 0-1）

②特征重要性排序：算法输出的特征重要性显示，利润（权重 23.5%）、销项负数发票率（权重 19.8%）、销项发票金额总计（权重 17.2%）是影响违约预测的三大核心因素，与前文分析的信贷风险影响机理一致，验证了指标体系的合理性。

通过该模型，可以将 123 家企业的信贷风险转化为具体的违约概率值，为后续信贷策略决策提供量化依据。

具体的风险量化结果如下（由于数据较多，此处仅展示 10 家企业数据）：

表 2 问题一风险量化结果表

企业	违约概率	企业	违约概率
E1	0.15	E103	0.82
E10	0.15	E104	0.2
E100	0.82	E105	0.22
E101	0.89	E106	0.32
E102	0.55	E107	0.97

5.3 最优信贷策略决策模型

（1）是否贷款策略决策模型

从理论层面来看，银行决定是否向企业放贷，核心依据是企业信贷风险的高低。当企业信贷风险偏高时，银行通常不会放贷；而若风险处于适度或较低水平，银行则可能给予放贷支持。这一决策过程的关键，在于确定一个放贷的风险阈值——风险高于该阈值的企业将被拒绝放贷，风险低于该阈值的企业则有机会获得贷款。

关于风险阈值的确定，结合附件一的数据可做如下分析：在 123 家企业里，有 27 家出现违约情况。其中，D 级企业占比 24/27，C 级企业占比 2/27，B 级企业占比 1/27。B 级违约企业做降级处理，可将其与 C 级企业归为一类，此时所有违约企业便都可视为 C 级企业。同时，按照银行原则，D 类企业本就不在放贷考虑范围内，所以后续研究中涉及的违约企业均为 C 级企业。

基于此，可依据 C 级企业的违约率来设定风险阈值。已知 C 级企业的总体违约率为 9%（3/33），这意味着在 C 级企业中，信贷风险处于前 9%的企业违约风险较大。因此，将 C 级企业中前 9%对应的信贷风险值定为风险阈值，即取 C 级企业信贷风险序列的第 9 百分位数作为阈值。风险高于该值的企业，因信贷风险较高被排除放贷；风险低于该值的企业，因信贷风险较低可获得放贷资格。

对于附件一中 C 类企业的信贷风险序列，经计算，其第 9 百分位数对应的风险值为 0.473，这便是确定的风险阈值。具体而言，信贷风险大于 0.473 的企业将被拒绝放贷，小于该值的企业则可获得放贷。

（2）基于非线性规划的最优贷款额度和利率决策模型

银行在确定信贷策略时，核心目标是在控制风险的前提下实现年度总贷款利润最大化。这一过程需综合考虑企业的违约概率、客户流失率、贷款额度及利率等多方面因素，通过构建非线性规划模型实现最优决策。

①目标函数：总贷款利润的量化表达

总贷款利润的计算需兼顾两方面：按期还款企业带来的收益与违约企业造成的损失，同时需扣除因利率设置不合理导致的客户流失损失。具体公式如下：

$$W = \sum \left[a * r * (1-d) - a * d \right] * (1-l_r) \tag{5.6}$$

其中，a 为对某企业的贷款额度，r 为贷款利率，d 为该企业的违约概率，l_r 为客户流失率。公式中，$\left[a * r * (1-d) \right]$ 表示按期还款时的利息收益，$(a * d)$ 表示违约时的本金损失，两者差值乘以 $(1-l_r)$（即未流失客户的比例），得到该笔贷款对总利润的净贡献。通过对所有企业的贡献求和，即为银行的年度总贷款利润。

② 关键参数的函数关系

违约概率 d 上文已经得出，下面对客户流失率 l_r 与信誉评级、利率 r 的关联进行拟合求解。

客户流失率受企业信誉评级和贷款利率共同影响，我们用一次、二次、三次模型来

进行拟合测试。基于附件 3 的统计数据，通过三次函数拟合得到不同信誉评级下的客户流失率与贷款年利率关系拟合曲线：

信誉评级 A 的情况：

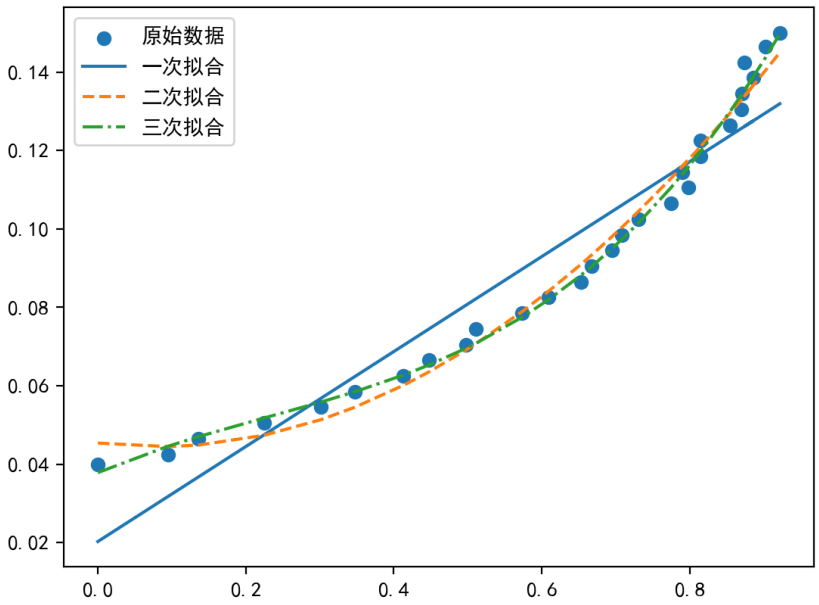

图 1 客户流失率（信誉评级 A）与贷款年利率关系拟合曲线

得到三种模型的可决系数 R^2 分别为 0.9111、0.9873、0.9946，显著性都满足<0.05 的要求，通过对比分析，三次模型的拟合效果最佳，我们选择三次拟合方程作为其回归函数：

$$l_{r1} = 0.1859r^3 - 0.1280r^2 + 0.0814r + 0.0378 \tag{5.7}$$

信誉评级为 B 的情况：

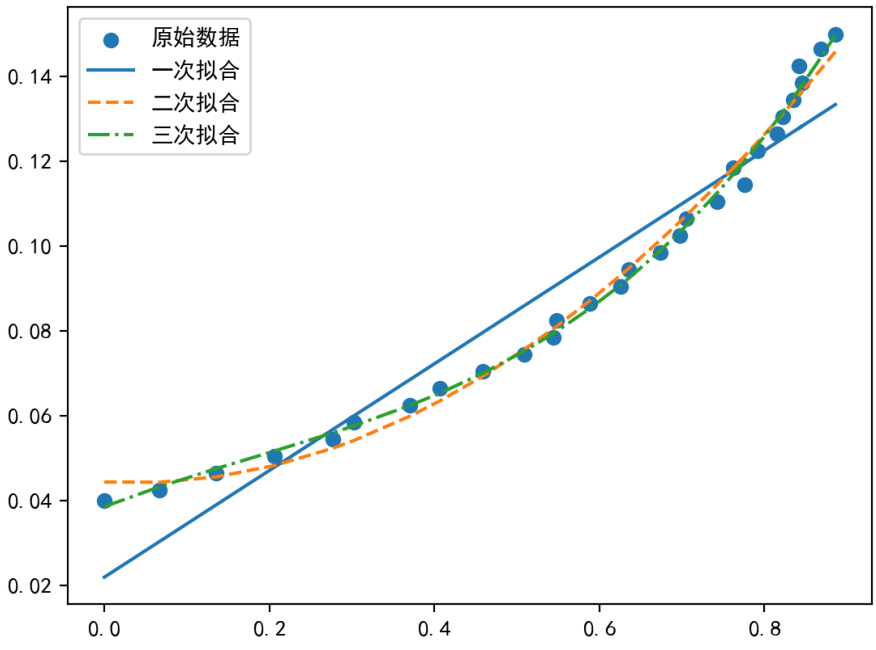

图 2 客户流失率（信誉评级 B）与贷款年利率关系拟合曲线

得到三种模型的可决系数 R^2 分别为 0.9256、0.9916、0.9964，显著性都满足<0.05的要求，通过对比分析，三次模型的拟合效果最佳，我们选择三次拟合方程作为其回归函数：

$$l_{r2} = 0.1677r^3 - 0.0921r^2 + 0.0756r + 0.0385 \tag{5.8}$$

信誉评级为 C 的情况：

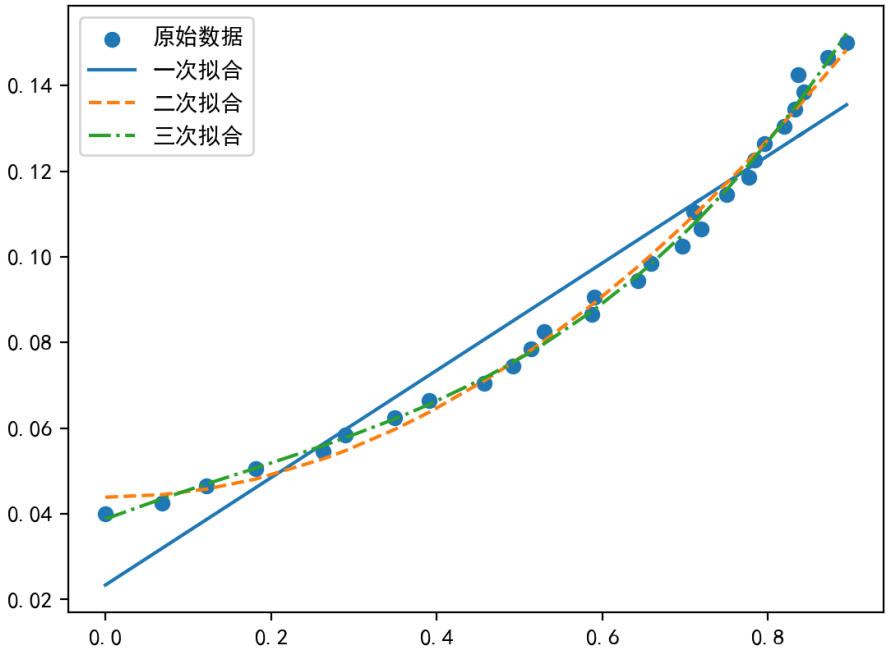

图 3 客户流失率（信誉评级 C）与贷款年利率关系拟合曲线

得到三种模型的可决系数 R^2 分别为 0.9353、0.9930、0.9966，显著性都满足<0.05的要求，通过对比分析，三次模型的拟合效果最佳，我们选择三次拟合方程作为其回归函数：

$$l_{r3} = 0.1425r^3 - 0.0678r^2 + 0.0731r + 0.0388 \tag{5.9}$$

③约束条件的设定

为确保决策符合银行信贷政策和风险控制要求，模型需满足以下约束：

约束条件一：贷款额度约束：$10 \le a \le 100$ 或 $a = 0$（单位：万元），即单家企业贷款金额 10 万至 100 万之间，或不给该公司贷款。

约束条件二：利率约束：$0.04 \le r \le 0.15$，即年利率不得低于 4%或高于 15%。

约束条件三总信贷额度约束：$\sum a \le T_0$，其中 T_0 为银行年度信贷总额。

④建立模型

$$\max W = \sum \left[a*r*(1-d) - a*d \right] * (1-l_r)$$
$$s.t. \begin{cases} l_r = l_r(r) \\ 10 \le a \le 100 or a = 0 \\ 0.04 \le r \le 0.15 \\ \sum a \le T_0 \end{cases} \tag{5.10}$$

⑤模型的求解思路

由于该模型涉及 123 家企业的贷款额度和利率两个决策变量，总计 246 个变量，且目标函数与约束条件存在非线性关系，比如客户流失率与利率的三次函数关系、总利润计算中的乘积项等，直接求解这类大规模非线性规划问题不仅效率低，还容易陷入局部最优解。因此，我们采用遗传算法来进行优化求解，通过模拟生物进化中的“选择-交叉-变异”过程，在可行解空间内高效寻找全局最优策略。具体步骤包括：

种群初始化：随机生成 100 组初始解，每组解包含所有企业的贷款额度和利率，其中贷款额度限定在 10 万到 100 万元之间，利率在 4%到 15%之间。同时，对于 D 级企业以及 C 级中违约概率高于阈值 0.473 的企业，直接将其贷款额度设为 0，确保初始解符合基本的放贷规则。

适应度评估：以银行总贷款利润的负值作为适应度函数。在计算总利润时，会根据企业的信誉评级调用对应的客户流失率函数，A 级、B 级、C 级企业分别对应不同的三次函数。如果总贷款额度超过了银行年度信贷总额，会对适应度进行大幅扣减，以此强化对总额度约束的遵守。

遗传操作环节：采用轮盘赌选择法挑选出 50%的优质父代个体，再通过 80%概率的均匀交叉生成子代，扩大搜索范围，同时以 10%的概率进行均匀变异，避免算法局限在局部最优。

迭代优化：重复上述步骤，直到达到 200 次最大迭代次数或者连续 50 次迭代没有性能提升为止。此时输出的最优解就是能使银行总利润最高的信贷策略。通过该模型，银行可在给定的信贷总额约束下，为每家企业制定合理的贷款额度和利率，实现总利润最大化的同时，有效控制信贷风险。

在求解过程中，通过硬编码严格保障各项约束的满足，比如筛选可放贷企业、限制贷款额度和利率的范围、对超总额度的情况进行惩罚等。求解完成后，将最优的贷款额度和利率按企业整理成表并导出到 Excel，结果显示低风险企业获得了更高的贷款额度和更低的利率，高风险企业则受到额度限制或面临更高利率，符合风险与收益匹配的原则，且该求解方法效率较高，结果稳定可靠。

⑥具体的信贷策略决策结果如下（由于数据较多，此处仅展示 10 家企业数据）：

表 3 问题一信贷策略决策结果表

企业代号	信誉评级	贷款额度	贷款利率
E79	B	399233.24	0.0756
E98	B	923996.06	0.1394

E80	C	495225.36	0.1332
E95	B	272278.09	0.1132
E96	C	259596.97	0.066
E1	A	143098.7	0.0637
E10	B	494023.42	0.1108
E58	B	808086.69	0.084
E90	C	464448.69	0.1087

六、 问题二模型的建立与求解

6.1 基于 BP 神经网络的信誉评价预测模型

附件 2 中的 302 家企业因无信贷记录，缺失关键的信誉评级信息，而信誉评级是衔接问题一模型的重要纽带，它不仅影响客户流失率的计算，也是银行判断企业还款意愿的核心依据。因此，需借助 BP 神经网络模型，基于企业的发票特征数据预测其信誉评级，为后续复用问题一的信贷风险量化与策略优化模型提供完整输入。

BP 神经网络的核心目标是探寻输入变量和输出变量间的关系，其运行机制是依据已有数据持续迭代，不断修正自身权重，最终精准估计出函数关系。BP 神经网络主要由正向传播和反向传播这两个过程构成 ：

（1）正向传播流程：第一步为正向传播，输入层负责接收我们输入的变量 x，在本题中即某一企业的各项经验特征；输出层则是我们期望网络输出的变量 y，也就是对应企业的信用等级。而两个节点间的连线为连接权重，是后续需不断修正的参数。当输入数据时，数据会凭借连接权重进入下一层，即隐含层。隐含层的输入为输入参数的加权和，其输出是输入的 Sigmoid 函数。这里的 Sigmoid 函数作为神经网络的激活函数，对加权和进行处理，能将数据限制在 [0,1] 范围内，可避免数据过大问题。同理，y 的输出与输入也遵循上述原则。输出 y_0 后，便完成一次正向传播。

（2）反向传播流程：接着进入反向传播，其依据的信息是误差，也就是网络输出的 y_0 与真实值 y 的差距，通常用均方差损失函数表示（利于求导），即

$$E=\frac{1}{2}\left(y_0-y\right)^2 \tag{5.11}$$

显然，误差越小，说明网络越贴近真实关系。反向传播会依据正向传播算出的误差，反向推导出使误差最小化的连接权重，并更新权重。待所有参数更新完毕，就完成一次反向传播。

每进行一次正向传播与反向传播，网络内的参数会更新一次，这就完成了一次网络训练。所以，训练网络的过程就是不断重复正向与反向传播，持续更新网络连接权重。理论上，只要节点数量充足、参数足够，就能不断逼近真实关系。当网络训练好后，代

入一组新值，网络会依据内部参数输出预测值，往往能取得较好预测效果。

针对问题二，具体步骤如下：

（1）数据预处理与特征工程

模型的输入数据分为训练集与预测集：训练集源自附件 1 中 123 家有信贷记录企业，包含可直接提取的发票特征与已知的信誉评级标签；预测集则为附件 2 中 302 家企业的发票特征数据，需通过模型输出信誉评级。

首先对数据进行清洗与标准化处理：

①特征变量选取：从两类企业的发票数据中提取核心特征（包括与进项/销项发票金额相关的指标、反映盈利水平的指标及各类异常发票占比等，具体依据数据字段确定），这些特征在问题一的模型中已被证实与企业经营状况密切相关。

②缺失值处理：采用均值填充法（SimpleImputer）对少量缺失的特征值进行补充，确保数据完整性——例如，对某企业缺失的“销项负数发票率”，以同行业企业的均值替代，避免数据缺口影响模型训练。

③标签编码：将信誉评级（A、B、C、D）通过 LabelEncoder 转换为数值型标签（A→0、B→1、C→2、D→3），使非数值型分类变量可被神经网络识别。

④特征标准化：通过 StandardScaler 对所有特征变量进行标准化，将其转换为均值为 0、标准差为 1 的数据集，消除不同指标间的量纲差异（如发票金额的大额数值与发票率的小比例数值），确保模型在训练时对各特征的权重分配更合理，加速收敛过程。

（2）BP 神经网络模型的构建与参数设置

BP 神经网络通过多层神经元的非线性映射，能够捕捉发票特征与信誉评级之间的复杂关联，其结构设计如下：

①输入层：神经元数量与选取的特征变量数量一致，负责接收标准化后的发票特征数据，将其传递至隐藏层进行特征提取。

②隐藏层：设置两层隐藏层，第一层含 10 个神经元，第二层含 8 个神经元，均采用 ReLU 激活函数。ReLU 函数通过抑制负值输出，有效解决了传统激活函数的梯度消失问题，使模型能更高效地学习特征间的非线性关系——例如，识别“高利润与低异常发票率组合”对高信誉评级的正向影响。

③输出层：包含 4 个神经元，分别对应 A、B、C、D 四个信誉评级类别，通过 softmax 函数将输出转化为概率分布，最终以概率最大的类别作为预测结果。

模型训练参数设置为：最大迭代次数 1000 次，确保模型有充足的训练时间收敛；随机种子设为 42，保证多次训练结果的一致性；学习率采用自适应调整机制，在训练初期快速逼近最优解，后期逐步微调以提高精度。

（3）模型训练与预测执行

模型训练阶段，利用附件 1 中 123 家企业的特征数据与编码后的信誉评级标签进行迭代优化：通过前向传播计算预测值，对比实际标签计算交叉熵损失，再通过反向传播算法逐层调整神经元的权重和偏置，不断降低损失值。在这一过程中，模型逐渐学习到

发票特征与信誉评级的内在规律——例如，销项金额稳定增长、异常发票率低的企业更可能被评为 A 级，而进项与销项金额失衡、作废发票频发的企业则倾向于 C 级或 D 级。

训练完成后，将附件 2 中 302 家企业的标准化特征数据输入模型，经前向传播得到数值型预测标签，再通过 LabelEncoder 逆转换将其还原为 A、B、C、D 的信誉评级，最终写入 excel 表格。

具体的信用等级预测结果如下（由于数据较多，此处仅展示 10 家企业数据）：

表 4 问题二信用等级预测结果表

企业	信用等级	企业	信用等级
E150	A	E155	B
E151	B	E156	C
E152	A	E157	D
E153	D	E158	D
E154	B	E159	A

6.2 量化信贷风险

与第一问中量化信贷风险步骤相同，从企业发票数据中提取进项发票金额总计、销项发票金额总计、利润、四类异常发票率（进项 / 销项作废、进项 / 销项负数）共七项指标，接着采用随机森林算法量化 302 家企业的信贷风险，输出违约概率。

具体的风险量化结果如下（由于数据较多，此处仅展示 10 家企业数据）：

表 5 问题二风险量化结果表

企业	违约概率	企业	违约概率
E124	0.12	E129	0
E125	0.12	E130	0.06
E126	0	E131	0
E127	0.05	E132	0
E128	0.07	E133	0.03

6.3 求解最优信贷策略

求解最优信贷策略的思路与第一问相同，首先根据风险阈值来确定放贷企业，然后再用非线性规划模型来求解出最优的贷款额度与利率策略。

（1）确定对每个企业是否放贷

根据问题一的思路确定附件二中 302 个企业的风险阈值为 0.473，当企业信贷风险高于 0.473 时不予放贷，低于该阈值时可以放贷。

（2）基于非线性规划模型求最优贷款额度与利率策略

此处仍用第一问给出的最优策略决策模型进行求解，但在第一问的基础上加上了一个总贷款额为 1 亿元的约束条件，此时的最优信贷决策模型变为：

$$\max W=\sum\left[a*r*(1-d)-a*d\right]*(1-l_r)$$
$$s.t.\begin{cases} l_r=l_r(r) \\ 10\le a\le 100 or a=0 \\ 0.04\le r\le 0.15 \\ \sum a\le 10000 \end{cases} \tag{5.12}$$

（3）最优信贷策略预测结果

具体的信贷策略决策结果如下（由于数据较多，此处仅展示 10 家企业数据）：

表 6 问题二信贷策略决策结果表

企业代号	信誉评级	贷款额度	贷款利率
E138	B	746952.79	0.1485
E139	A	501286.72	0.0439
E140	A	377102.88	0.0818
E141	A	789642.82	0.0579
E142	A	819249.75	0.0947
E143	C	709512.15	0.0737
E144	C	161618.74	0.0916
E145	A	232509.55	0.075
E146	A	463531.3	0.0938

七、 问题三模型的建立与求解

7.1 突发事件影响下的信贷调整策略模型

（1）突发事件的界定与分类框架

依据 2007 年 11 月 1 日生效的《中华人民共和国突发事件应对法》，突发事件是指突然发生、可能引发严重社会危害，需启动应急处置程序的事件，涵盖自然灾害、事故灾难、公共卫生事件、社会安全事件四大类 ：

自然灾害：由自然力主导，如洪水、地震等“天灾”；

事故灾难：人为过错导致，像坠机、车祸等违反人意志的灾难；

公共卫生事件：威胁公众健康，以新冠肺炎疫情为典型；

社会安全事件：冲击社会公共安全，如恐怖袭击这类事件 。

四类突发事件对不同行业、企业经营收益的冲击差异显著。因此，模型需实现“分类分析+策略模拟”：针对四类事件，分别构建影响路径，分析其对信贷调整策略的作用逻辑，并通过附件数据开展模拟验证，确保模型有效性。

（2）附件二企业的行业与类别分布梳理

参照《2017 年国民经济行业分类》标准，对附件 2 的 302 家企业进行行业归属划分，初步归为 15 个行业后，发现制造业、建筑业企业数量集中。为精准分析，进一步细化：

制造业拆分为 7 个子类，建筑业拆分为 2 个子类，最终形成 22 个“行业 - 类别”单元。

尽管同类企业实际特征存在差异，但从建模可行性出发，假设同一类别企业具备相同特征指数，在此基础上，分析四类突发事件对 22 类企业的影响规律 。以下为 22 类企业的分布占比：

企业分类	数量	占比	企业分类	数量	占比
A农、林、牧、渔业	3	0.99%	F批发和零售业	12	3.97%
C1食品制造业	3	0.99%	G交通运输	18	5.96%
C2医药制造业	3	0.99%	H住宿和餐饮业	2	0.66%
C3机电制造业	18	5.96%	I信息传输、软件和信息技术服务业	28	9.27%
C4材料制造业	5	1.66%	J租赁和商务服务业	20	6.62%
C5印刷制造业	3	0.99%	K科学研究和技术服务业	5	1.66%
C6家居制造业	10	3.31%	L水利、环境和公共设施管理业	15	4.97%
C7金属制造业	5	1.66%	M居民服务、修理和其他服务业	9	2.98%
C8石油	2	0.66%	N文化、体育和娱乐业	23	7.62%
D电力、热力、燃气及谁生产和供应业	5	1.66%	O个体经营	56	18.54%
E建筑	52	17.22%	P卫生和社会工作	5	1.66%

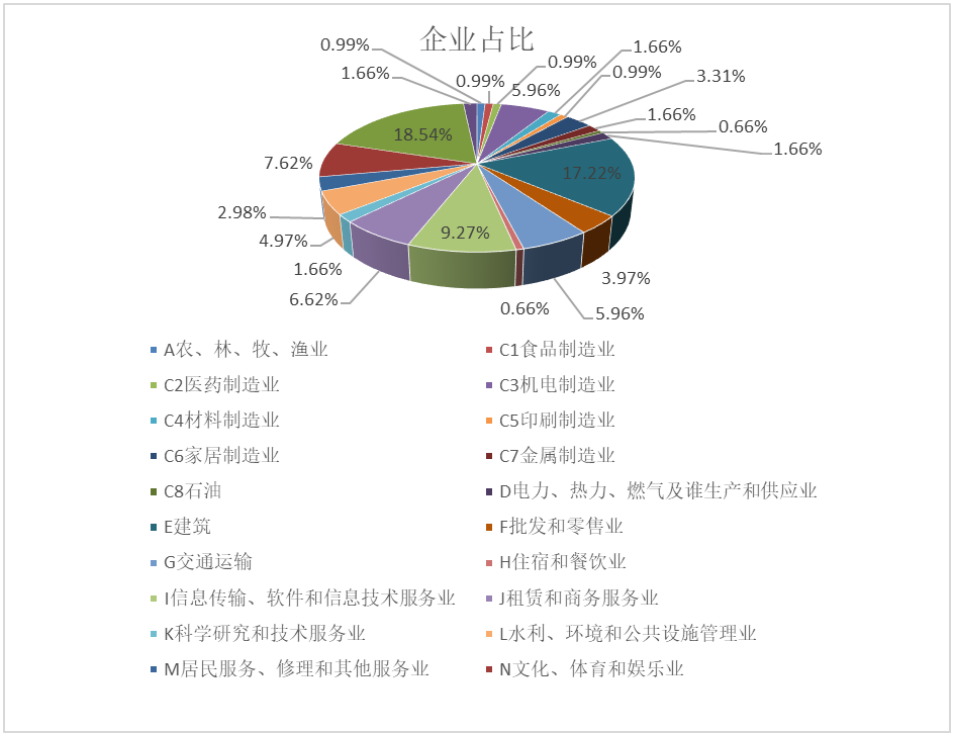

图 4 企业分布占比图

（3）疫情下的政策分析和企业分析

疫情时期，为了应对疫情给企业经济带来的负面影响，政府部门对中小微企业实行不同的帮扶政策。以银行贷款政策为例，中华人民共和国工业和信息化部在 2020 年 4

月发布的《支持中小企业应对新冠肺炎疫情政策指引》中提出要加大财税支持力度，具体措施有给企业提供优惠贷款，按企业实际获得贷款利率的 50%进行贴息；降低小规模纳税人增值税率至 1%。

根据相关数据发现，不同企业在面对疫情时的韧性不同。中小微企业的存活率整体下降了 11.81%，文化体育娱乐业和租赁和商务服务业受到疫情影响最大，下降幅度分别为 24.4%和 14.17%；“宅经济”使网络信息传递频繁，教育视频活跃度增加；在获资的行业类别中，企业服务（19.05%）、医疗健康（18.19%）、金融（11.69%）、新工业（7.28%）以及本地生活（6.73%）是获资最多的五大行业（合计占比 62.94%）。

由以上两点的分析可以发现，不同企业的资金流有所变化，大部分呈下降趋势，从而企业的金额、利润、违约概率和信誉等级等相关信息也将发生改变。

（4）建立评价指标

根据疫情对企业的影响，我们对附件 2 的数据进行了修改完善，得到新的指标。

（5）模型求解

模型求解与问题 1 和问题 2 的求解思路和方式一致。

八、 模型评价

8.1 模型优点

（1）多维度风险量化

本模型采用了七项核心指标（进项发票金额总计、销项发票金额总计、利润、进项作废发票率、进项负数发票率、销项作废发票率、销项负数发票率）来全面量化中小微企业的信贷风险。这种多维度指标体系能够更准确地反映企业的实际经营状况和财务健康度，相比传统单一指标评估方法，具有更高的科学性和可靠性。

（2）随机森林算法的应用

随机森林算法在本模型中用于量化企业的违约概率，其优势在于能够处理高维数据、自动选择重要特征，并对异常值和共线性数据具有较强稳健性。通过大量决策树的集成学习，模型有效降低了过拟合风险，提高了预测精度，为信贷决策提供了有力的数据支持。

8.2 模型创新

（1）非线性规划与遗传算法的结合

本模型在最优信贷策略决策中，创新性地结合了非线性规划与遗传算法。非线性规划模型以总利润最大化为目标，考虑了贷款额度、利率、违约概率和客户流失率等多重约束条件。而遗传算法则通过模拟生物进化过程，高效地在可行解空间内搜索全局最优解，显著提高了求解效率和决策质量。

（2）BP 神经网络填补数据缺口

针对无信贷记录企业的信誉评级缺失问题，本模型引入了 BP 神经网络进行预测。

BP 神经网络通过多层神经元的非线性映射，能够捕捉发票特征与信誉评级之间的复杂关系，有效填补了数据空白，为后续信贷风险量化与策略优化提供了完整输入。

8.3 模型不足

（1）数据依赖性强

本模型高度依赖于企业发票数据的真实性和完整性。如果数据存在虚假交易或票据伪造情况，将直接影响模型的预测精度和决策效果。因此，在实际应用中，需要加强对数据源的审核和监控，确保数据质量。

（2）突发事件模拟的简化处理

在模拟突发事件对企业经营的影响时，本模型主要调整了发票数据来反映经营变化。然而，实际突发事件的影响可能更加复杂和多样，包括供应链中断、市场需求突变、政策调整等多个方面。未来研究可以进一步细化突发事件模拟机制，提高模型的适应性和预测精度。

（3）缺乏实时动态调整能力

本模型主要基于历史数据和静态假设进行信贷决策，缺乏对市场环境和企业经营状况的实时动态调整能力。在实际应用中，银行需要根据市场变化和企业经营情况及时调整信贷策略。因此，未来研究可以探索引入实时数据分析和动态优化算法，提高模型的灵活性和响应速度。

九、 附录

问题一

1.各公司总金额税额统计

```
import pandas as pd

file_path = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/附件 2.xlsx'
input = pd.read_excel(file_path, sheet_name='进项发票信息')
output = pd.read_excel(file_path, sheet_name='销项发票信息')

# 删除作废发票（假设“发票状态”列标记为“作废”）
input = input[input['发票状态'] != '作废发票']
output = output[output['发票状态'] != '作废发票']

Ja = input.groupby('企业代号')['金额'].sum().reset_index()
# 进项发票金额
Jt = input.groupby('企业代号')['税额'].sum().reset_index()
# 进项发票税额
Xa = output.groupby('企业代号')['金额'].sum().reset_index()
# 销项发票金额
Xt = output.groupby('企业代号')['税额'].sum().reset_index()
# 销项发票税额

# 第一次合并
temp1 = pd.merge(Ja, Jt, on='企业代号', how='outer')
# 第二次合并
temp2 = pd.merge(temp1, Xa, on='企业代号', how='outer')
# 第三次合并
result = pd.merge(temp2, Xt, on='企业代号', how='outer')
# 填充缺失值为 0
result = result.fillna(0)

# 导出到新的 Excel 文件
result.to_excel('302 金额税额表.xlsx', index=False)
print("已保存到 '302 金额税额表.xlsx'")
```

2.利润标准化

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

file_path = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 金额税额表.xlsx'
df = pd.read_excel(file_path)

#标准化
scaler = StandardScaler()
df['标准化利润'] = scaler.fit_transform(df[['利润']])
#  导出到新的 Excel 文件
df.to_excel('302 标准化利润表.xlsx', index=False)
print("已保存")
```

3.计算变异系数

```
import pandas as pd

file_path = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/附件 2.xlsx'
df = pd.read_excel(file_path, sheet_name='进项发票信息')
#df = pd.read_excel(file_path, sheet_name='销项发票信息')

df = df[df['发票状态'] != '作废发票']

df['开票日期'] = pd.to_datetime(df['开票日期'], format='%Y/%m/%d')
df['月份'] = df['开票日期'].dt.strftime('%Y/%m')



monthly_sum = df.groupby(['企业代号', '月份'])[['金额']].sum().reset_index()

#  对每个公司，计算金额的均值、标准差和变异系数
result = monthly_sum.groupby('企业代号').agg(
    进项金额均值=('金额', 'mean'),
    进项金额标准差=('金额', 'std'),
).reset_index()
```

```
# 4. 计算变异系数
#result['进项金额变异系数'] = result['进项金额标准差'] / result['进项金额均值']
result['进项金额变异系数'] = result['进项金额标准差'] / result['进项金额均值']

# 5. 保存结果到 Excel
result.to_excel('302 进项金额变异系数.xlsx', index=False)
```

4.合并各个指标

```
import pandas as pd
os = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷'

#file_path_1 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/附件1.xlsx'
#company = pd.read_excel(file_path_1,sheet_name='企业信息')

file_path_2 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 进项金额变异系数.xlsx'
jinxiang_bianyi = pd.read_excel(file_path_2)

file_path_3 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 销项金额变异系数.xlsx'
xiaoxiang_bianyi = pd.read_excel(file_path_3)

file_path_4 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 发票率.xlsx'
df_fapiaorate = pd.read_excel(file_path_4)

file_path_5 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 标准化利润表.xlsx'
df_p = pd.read_excel(file_path_5)

file_path_6 = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/302 金额税额表.xlsx'
jinershuier = pd.read_excel(file_path_6)

# 利润
```

```
p = df_p[['企业代号', '标准化利润']]
# 进项变异系数
cv_a = jinxiang_bianyi[['企业代号', '进项金额变异系数']]
# 销项变异系数
cv_b = xiaoxiang_bianyi[['企业代号', '销项金额变异系数']]
# 进项有效率
# er_a = df_youxiao[['企业代号', '进项有效概率']]
# 销项有效率
# er_b = df_youxiao[['企业代号', '销项有效概率']]
# 是否违约
#y = company[['企业代号', '违约']]
# 有效概率
# er = df_youxiao[['企业代号', '有效概率']]
# 进项金额
jinxiangjine = jinershuier[['企业代号','进项金额']]
# 销项金额
xiaoxiangjine = jinershuier[['企业代号','销项金额']]
# 进项作废发票率
rate_jinxiangzuofei = df_fapiaorate[['企业代号','进项发票作废率']]
# 进项负金额发票率
rate_jinxiangfu = df_fapiaorate[['企业代号','进项负金额发票率']]
# 销项作废发票率
rate_xiaoxiangzuofei = df_fapiaorate[['企业代号','销项发票作废率']]
# 销项负金额发票率
rate_xiaoxiangfu = df_fapiaorate[['企业代号','销项负金额发票率']]

# 按企业代号合并
#merged = y.merge(p, on='企业代号', how='outer').merge(cv_a, on='企业代号', how='outer').merge(cv_b, on='企业代号', how='outer').merge(jinxiangjine, on='企业代号', how='outer').merge(xiaoxiangjine, on='企业代号', how='outer').merge(rate_jinxiangzuofei, on='企业代号', how='outer').merge(rate_jinxiangfu, on='企业代号', how='outer').merge(rate_xiaoxiangzuofei, on='企业代号', how='outer').merge(rate_xiaoxiangfu, on='企业代号', how='outer')
merged = p.merge(cv_a, on='企业代号', how='outer').merge(cv_b, on='企业代号', how='outer').merge(jinxiangjine, on='企业代号', how='outer').merge(xiaoxiangjine, on='企业代号', how='outer').merge(rate_jinxiangzuofei, on='企业代号', how='outer').merge(rate_jinxiangfu, on='企业代号', how='outer').merge(rate_xiaoxiangzuofei, on='企业代号', how='outer').merge(rate_xiaoxiangfu, on='
```

```
企业代号', how='outer')

#pd.set_option('future.no_silent_downcasting', True)
#merged['违约'] = merged['违约'].replace({'是': 0, '否': 1})
#  保存结果
merged.to_excel('302 指标汇总.xlsx', index=False)
```

5.作废发票占比、负金额发票占比

```
import pandas as pd

file_path = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/附件 2.xlsx'
input = pd.read_excel(file_path, sheet_name='进项发票信息')
output = pd.read_excel(file_path, sheet_name='销项发票信息')

A_stats = input.groupby('企业代号').agg(
    进项总发票数=('发票状态', 'count'),
    进项作废发票数=('发票状态', lambda x: (x == '作废发票').sum()),
    进项负金额发票数=('金额', lambda x: (x < 0).sum())
).reset_index()

B_stats = output.groupby('企业代号').agg(
    销项总发票数=('发票状态', 'count'),
    销项作废发票数=('发票状态', lambda x: (x == '作废发票').sum()),
    销项负金额发票数=('金额', lambda x: (x < 0).sum())
).reset_index()

#  合并 A 表和 B 表统计结果
result = pd.merge(A_stats, B_stats, on='企业代号', how='outer').fillna(0)

# 4.  保存结果
result.to_excel('302 发票率.xlsx', index=False)
```

6.比较不同曲线拟合效果

```
import pandas as pd
import numpy as np
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']  # 黑体，或改为你系统有的中文字体
plt.rcParams['axes.unicode_minus'] = False      # 正确显示负号

# 1. 读取 Excel 数据
df = pd.read_excel('D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/表格/附件 3.xlsx', header=[0,1])  # 修改为你的文件路径

# 读取各列数据（根据你的 MultiIndex 实际列名）
y = df[('贷款年利率', 'Unnamed: 0_level_1')].values
xa = df[('客户流失率', '信誉评级 A')].values
xb = df[('客户流失率', '信誉评级 B')].values
xc = df[('客户流失率', '信誉评级 C')].values

x = xc

# 一次函数拟合
coef1 = np.polyfit(x, y, 1)
y_pred1 = np.polyval(coef1, x)
r2_1 = r2_score(y, y_pred1)

# 二次函数拟合
coef2 = np.polyfit(x, y, 2)
y_pred2 = np.polyval(coef2, x)
r2_2 = r2_score(y, y_pred2)

# 三次函数拟合
coef3 = np.polyfit(x, y, 3)
y_pred3 = np.polyval(coef3, x)
r2_3 = r2_score(y, y_pred3)

print(f'一次函数拟合的 R²: {r2_1:.4f}')
print(f'二次函数拟合的 R²: {r2_2:.4f}')
print(f'三次函数拟合的 R²: {r2_3:.4f}')

# 可选：画图显示拟合效果
plt.scatter(x, y, label='原始数据')
```

```
plt.plot(x, y_pred1, label='一次拟合', linestyle='-')
plt.plot(x, y_pred2, label='二次拟合', linestyle='--')
plt.plot(x, y_pred3, label='三次拟合', linestyle='-.')
plt.legend()
plt.show()
```

7.求流失率与利率函数关系

```
import pandas as pd
import numpy as np
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']  # 黑体，或改为你系统有的中文字体
plt.rcParams['axes.unicode_minus'] = False      # 正确显示负号

# 1. 读取 Excel 数据
df = pd.read_excel('D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/表格/附件 3.xlsx', header=[0,1])  # 修改为你的文件路径

# 读取各列数据（根据你的 MultiIndex 实际列名）
y = df[('贷款年利率', 'Unnamed: 0_level_1')].values
xa = df[('客户流失率', '信誉评级 A')].values
xb = df[('客户流失率', '信誉评级 B')].values
xc = df[('客户流失率', '信誉评级 C')].values

x = xc

# 三次函数拟合
coef3 = np.polyfit(x, y, 3)
y_pred3 = np.polyval(coef3, x)
r2_3 = r2_score(y, y_pred3)

equation = f"y = {coef3[0]:.4f}x³ + {coef3[1]:.4f}x² + {coef3[2]:.4f}x + {coef3[3]:.4f}"
print("三次拟合方程为：", equation)
```

8.遗传算法进行决策贷款方案

```
import numpy as np
import pandas as pd
```

```
from geneticalgorithm import geneticalgorithm as ga
from openpyxl.utils.dataframe import dataframe_to_rows

# 假设企业数据已加载
data = pd.read_excel(
    'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/表格/123 信誉及风险.xlsx')
N = len(data)
T = 1e9  # 总额度上限

# B 级降级为 C 级
for i in range(N):
    credit = data.loc[i, '信誉评级']
    if credit == 'B' and data.loc[i, '违约'] == '是':
        credit = 'C'

# 获取所有 C 级企业的违约概率
c_credit_data = data[data['信誉评级'] == 'C']  # 过滤出 C 级企业
c_credit_data_sorted = c_credit_data.sort_values(by='违约概率', ascending=False)  # 按违约概率降序排列

# 计算 C 级企业中前 9%的阈值
threshold = c_credit_data_sorted['违约概率'].quantile(0.91)  # 获取前 9%的违约概率阈值
print('阈值：',threshold)


# 定义流失率函数，根据信誉等级选择不同的函数
def func_lr_A(ri):
    # A 级企业的流失率函数
    return 0.1859 * ri**3 - 0.1280 * ri**2 + 0.0814 * ri + 0.0378


def func_lr_B(ri):
    # B 级企业的流失率函数
    return 0.1677 * ri**3 - 0.0921 * ri**2 + 0.0756 * ri + 0.0385
```

```
def func_lr_C(ri):
    # C 级企业的流失率函数
    return 0.1425 * ri**3 - 0.0678 * ri**2 + 0.0731 * ri + 0.0388


#  修改后的约束处理函数
def constraint_handling(x):    # x: [a1, r1, a2, r2, ...]
    total_loan = 0
    fitness_sum = 0
    for i in range(N):
        ai, ri = x[2 * i], x[2 * i + 1]
        credit = data.loc[i, '信誉评级']
        di = data.loc[i, '违约概率']

        # D 级不放贷
        if credit == 'D':
            ai = 0

        # C 级前 9%不放贷
        if credit == 'C' and di > threshold:
            ai = 0

        # ai 范围
        ai = np.clip(ai, 1e5, 1e6)

        # ri 范围
        ri = np.clip(ri, 0.04, 0.15)

        #  根据信誉等级选择流失率函数
        if credit == 'A':
            lr = func_lr_A(ri)
        elif credit == 'B':
            lr = func_lr_B(ri)
        elif credit == 'C':
            lr = func_lr_C(ri)
        elif credit == 'D':
            lr = 1
```

```
            total_loan += ai
            fitness_sum += (ai * ri * (1 - di) - ai * di) * (1 - lr)


    # 总放贷额度不超过 T，若超过则加罚分
    if total_loan > T:
        fitness_sum -= 1e9

    return -fitness_sum  # 遗传算法为最大化问题


# 遗传算法参数
varbound = np.array([[1e5, 1e6], [0.04, 0.15]] * N)
algorithm_param = {
    'max_num_iteration': 200,
    'population_size': 100,
    'mutation_probability': 0.1,
    'mutation_type': 'uniform',
    'elit_ratio': 0.01,
    'selection_type': 'roulette',
    'parents_portion': 0.5,
    'crossover_probability': 0.8,
    'crossover_type': 'uniform',
    'max_iteration_without_improv': 50,  # 设置没有改进的最大迭代次数
}

# 初始化遗传算法模型
model = ga(
    function=constraint_handling,
    dimension=2 * N,
    variable_type='real',
    variable_boundaries=varbound,
    algorithm_parameters=algorithm_param
)
```

```
# 运行遗传算法
model.run()

best_solution = model.output_dict['variable']
best_total_loan = 0

import pandas as pd

# 收集每家企业的贷款信息到列表中
best_loan_distribution = []

for i in range(N):
    ai, ri = best_solution[2 * i], best_solution[2 * i + 1]
    credit = data.loc[i, '信誉评级']
    di = data.loc[i, '违约概率']
    name = data.loc[i, '企业代号']

    if credit == 'D':
        ai = 0
    if credit == 'C' and di > threshold:
        ai = 0

    best_loan_distribution.append({
        '企业代号': name,
        '信誉评级': credit,
        '贷款额度': round(ai, 2),
        '利率': round(ri, 4),
        '违约概率': di
    })

# 将结果转为 DataFrame
df_result = pd.DataFrame(best_loan_distribution)

# 导出为 Excel 文件
df_result.to_excel('贷款方案.xlsx', index=False)
```

```
print("贷款方案已成功导出为 Excel 文件：贷款方案.xlsx")
```

问题二

1.BP 神经网络对 302 家企业进行信誉评级

```
import pandas as pd
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.model_selection import cross_val_score, train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.decomposition import PCA
from imblearn.over_sampling import SMOTE
import matplotlib.pyplot as plt
import seaborn as sns

# 数据路径
io = 'D:/Users/person/AppData/Local/Programs/Python/PythonProject/中小企业信贷/表格'

# 读取 A 表和 B 表
A = pd.read_excel(io + '/123 企业违约概率.xlsx')
B = pd.read_excel(io + '/302 企业违约概率.xlsx')

# 1. 数据预处理
X_train = A.drop(['企业代号', '违约', '不违约概率', '信誉评级'], axis=1)
y_train = A['信誉评级']
X_pred = B.drop(['企业代号', '不违约概率'], axis=1)

# 数据缺失值填补
imputer = SimpleImputer(strategy='mean')
X_train = pd.DataFrame(imputer.fit_transform(X_train), columns=X_train.columns)
X_pred = pd.DataFrame(imputer.transform(X_pred), columns=X_pred.columns)

# 标签编码
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)  # A->0, B->1, C->2, D->3
```

```
# 特征标准化
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_pred_scaled = scaler.transform(X_pred)

# 使用 SMOTE 进行过采样，以平衡数据
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_scaled, y_train_encoded)

# PCA 降维，减少维度，以减少噪声
pca = PCA(n_components=0.95)  # 保留 95%的方差
X_train_resampled = pca.fit_transform(X_train_resampled)
X_pred_scaled = pca.transform(X_pred_scaled)

# 2. 构建神经网络（MLPClassifier）
clf = MLPClassifier(hidden_layer_sizes=(128, 64, 32), activation='relu', max_iter=2000, random_state=42,
                    solver='adam', learning_rate_init=0.001, early_stopping=True,
validation_fraction=0.1)

# 划分训练集与验证集
X_train_split, X_val_split, y_train_split, y_val_split = train_test_split(X_train_resampled,
y_train_resampled, test_size=0.2, random_state=42)

# 训练模型
clf.fit(X_train_split, y_train_split)

# 3. 预测并计算训练集准确率
y_train_pred = clf.predict(X_train_split)
train_accuracy = accuracy_score(y_train_split, y_train_pred)

# 4. 计算验证集的准确率
y_val_pred = clf.predict(X_val_split)
val_accuracy = accuracy_score(y_val_split, y_val_pred)

# 输出训练集和验证集准确率
print(f"训练集准确率: {train_accuracy * 100:.2f}%")
```

```
print(f"验证集准确率: {val_accuracy * 100:.2f}%")

# 5. 在 B 表上进行预测
y_pred_encoded = clf.predict(X_pred_scaled)
y_pred = le.inverse_transform(y_pred_encoded)

# 6. 结果输出
B['预测信誉等级'] = y_pred
B.to_excel('302 信誉等级预测.xlsx', index=False)
print("已完成企业信誉等级预测，结果保存在 302 信誉等级预测.xlsx")

# 7. 显示混淆矩阵和分类报告
conf_matrix = confusion_matrix(y_val_split, y_val_pred)
class_report = classification_report(y_val_split, y_val_pred, target_names=le.classes_)

print("混淆矩阵:")
print(conf_matrix)
print("\n 分类报告:")
print(class_report)

# 8. 可视化混淆矩阵
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=le.classes_,
yticklabels=le.classes_)
plt.title("混淆矩阵")
plt.ylabel('实际值')
plt.xlabel('预测值')
plt.show()
```

问题三

1.年度信贷总额为 1 亿元时的信贷调整策略

```
import pandas as pd

# 1. 读取 Excel 文件
file_path = '企业变化附件 2：302 家无信贷记录企业的相关数据.xlsx'  # 替换为你的 Excel 文件路径
sheet1 = pd.read_excel(file_path, sheet_name='企业信息')  # 包含企业代号、企业分类
```

```
sheet2 = pd.read_excel(file_path, sheet_name='销项发票信息')  # 包含企业代号、发票、金额 1
sheet3 = pd.read_excel(file_path, sheet_name='进项发票信息')  # 包含企业代号、发票、金额 2

# 2. 定义不同企业分类的金额下降比例（按实际需求填写）
decrease_ratios = {
    "A": -0.1363,
    "C1": 0.0540,
    "C2": 0.1890,
    "C3": -0.1350,
    "C4": -0.1280,
    "C5": -0.0289,
    "C6": 0.0150,
    "C7": -0.0690,
    "C8": -0.1048,
    "D": 0.1560,
    "E": -0.1331,
    "F": 0.0280,
    "G": -0.0967,
    "H": -0.1280,
    "I": 0.0728,
    "J": -0.1417,
    "K": 0.0728,
    "L": -0.0543,
    "M": -0.1003,
    "N": -0.2440,
    "O": -0.0940,
    "P": 0.1890
}

# 税额统一下降比例
tax_decrease_ratio = 0.1  # 下降 10%

# 合并分类信息
df2 = pd.merge(sheet2, sheet1, on='企业代号', how='left')
df3 = pd.merge(sheet3, sheet1, on='企业代号', how='left')
```

```
# 计算变化后的金额 1 和税额 1
def calc_new_amount_and_tax(row, amount_col, tax_col):
    amount_decrease = decrease_ratios.get(row['分类'], 0)
    new_amount = row[amount_col] * (1 + amount_decrease)
    new_tax = row[tax_col] * (1 - tax_decrease_ratio)
    new_sum =   new_amount+new_tax
    return pd.Series([new_amount, new_tax,new_sum])

df2[['变化后金额 1', '变化后税额 1','变化后的价税合计 1']] = df2.apply(lambda row: calc_new_amount_and_tax(row, '金额', '税额'), axis=1)
df3[['变化后金额 2', '变化后税额 2','变化后的价税合计 2']] = df3.apply(lambda row: calc_new_amount_and_tax(row, '金额', '税额'), axis=1)

# 保存结果
with pd.ExcelWriter('企业变化.xlsx') as writer:
    df2.to_excel(writer, sheet_name='销项发票变化', index=False)
    df3.to_excel(writer, sheet_name='进项发票变化', index=False)

print("计算完成，结果已保存到  企业变化.xlsx")
```