中小微企业的信贷决策

摘　要

本文通过机器学习算法，给出企业潜在违约风险，并对企业的信誉进行评估进一步确定贷款额度，最后通过建立最优化模型，得出各企业的最优利率。

针对问题一，为方便分析，首先我们将信贷风险的量化分析转化为信誉评级的量化分析，选取销售收入、销项交易失败率、销售收入增长率和平均每单退货金额共 4 个评价指标建立模型对信誉评级进行分析。首先利用机器学习方法，将 4 个评价指标作为输入变量，是否违约作为输出变量，建立 BP 神经网络的二分类模型，即该模型为是否为企业提供贷款的决策模型。进一步，我们引入了企业信誉评价指数，定义其值与 4 个指标存在线性关系，利用组合决策树模型，我们可以得出变量权重和信誉评级。销售收入、销项交易失败率、销售收入增长率和平均每单退货金额的权重以此为 23.48，22.63，28.02，25.87。接下来，我们定义某企业信誉评价指数与所有企业信誉指数之和的比例为贷款比率，比率与银行年度贷款总额相乘即可得到该企业贷款额度。针对利率，我们构建了银行收益的最优化模型，通过梯度下降法，即可求出银行收益最大时的利率。对信誉评价为 A、B、C 的企业的贷款利率分别为：0.053983983983983984，0.06004004004004004，0.05937937937937938。

针对问题二，我们将附件 2 中的数据代入计算，首先求出各个指标的值，去除异常值并标准化以后，代入问题一中的 BP 神经网络模型和组合决策树模型，对企业潜在违约风险进行评估，决定是否放贷；同时依据计算出的每个企业的信誉评价指数，在银行年度贷款总额为 1 亿元的条件下，得出贷款额度。结合问题一得出的利率，得出不同信誉评级企业的信贷策略：对评级为 A 的企业，贷款额度最高，利率最低，对评级为 B 的企业，贷款额度中等，利率最高，对评级为 C 的企业，贷款额度中等，利率中等。

针对问题三，假定本问的突发条件为新冠病毒疫情，发生疫情时，银行贷款的主要目标从银行获取收益最高转换为企业收入最高，所以我们建立企业收益的最优化模型，通过梯度下降法求出各企业所对应的最优利率集中在 0.060，0.054，0.040 附近，比较疫情间与正常情况下企业贷款额度，我们发现信誉度高的企业疫情期间贷款额度变少，信誉低的企业贷款额度增加。信贷调整策略首先包括降低贷款利率，部分企业的利率降低至 0.04，其次，减少信誉等级高的企业的贷款额度，增加信誉等级稍低的企业的贷款额度。

关键字：BP 神经网络 组合决策树 最优化模型

1 问题重述

1.1 问题背景

中小微企业在我国的经济发展中发挥了重要的作用。然而因为中小微企业规模小，缺少抵押资产，中小微企业在获取贷款方面存在一定困难。为此，2019 年，国家发展改革委、银保监会联合印发《关于深入开展“信易贷” 支持中小微企业融资的通知》，积极推进信用资产变现。

银行通常依据信贷政策、企业的交易票据信息和上下游企业的影响力，决定对企业的信贷策略。银行首先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。对信誉高、信贷风险小的企业可以给予利率优惠。企业的生产经营和经济效益可能会受到一些突发因素影响，而且突发因素往往对不同行业、不同类别的企业会有不同的影响。

1.2 问题提出

现有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。在上述背景下，研究信贷策略，解决如下问题：

(1) 对附件 1 中 123 家企业的信贷风险进行量化分析，给出该银行在年度信贷总额固定时对这些企业的信贷策略。

(2) 在问题 1 的基础上，对附件 2 中 302 家企业的信贷风险进行量化分析，并给出该银行在年度信贷总额为 1 亿元时对这些企业的信贷策略。

(3) 综合考虑附件 2 中各企业的信贷风险和可能的突发因素对各企业的影响，给出该银行在年度信贷总额为 1 亿元时的信贷调整策略。

2 问题分析

2.1 问题一分析

问题一需要我们对 123 家企业的信贷风险进行量化分析，然后给出在年度信贷总额固定时银行对企业的信贷策略。考虑到信贷风险与信誉评级存在负相关关系，信誉等级越高，信贷风险就越低，所以我们将对信贷风险的量化分析转化为对信誉评级的分析。为了衡量企业的信誉评级，我们考虑引入企业信誉评价指数，并分析哪些因素会对企业竞争力产生影响。最终选取了平均每单退货金额、销售收入、销售收入增长率和销项交易失败率共 4 个评价指标。判断企业信誉评级情况，首先建立二分类模型判断企业是否违约，若违约则信誉等级默认为 D，其次计算出企业信誉评价指数确定具体信誉等级。

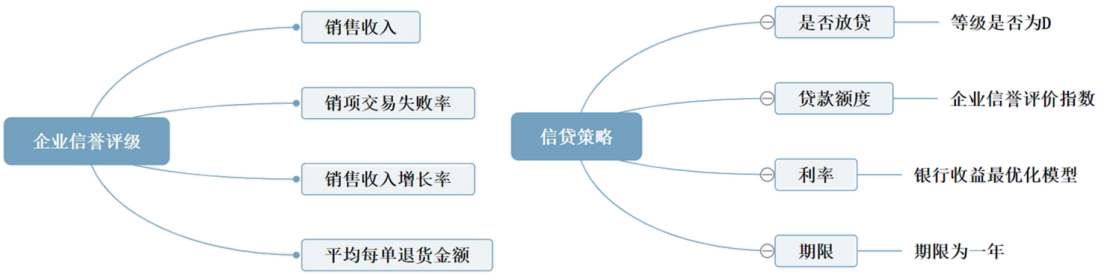

图 1 问题一思维导图

信贷策略主要包括是否放贷及贷款额度、利率和期限。题目给出期限为一年，不予讨论。针对是否放贷，若企业等级为 D 则不放贷，可以通过二分类模型预期预测。针对贷款额度，可以通过其与企业信誉指数的关系来确定。针对利率，我们考虑建立银行总收益优化模型来确定。

2.2 问题二分析

问题二要求我们对附件 2 中没有信贷记录的企业的信贷风险进行量化分析，并给出信贷策略。我们依旧将对信贷风险的量化转化为信誉评级的量化分析。在银行年度贷款总额为 1 亿元的条件下，将数据代入问题一中的模型，得出企业信誉评级及信誉评级指数，结合最优化模型得出的利率，最终可得出信贷策略。

2.3 问题三分析

问题三需要我们在考虑突发因素的情况下，对贷款策略进行调整。我们假定本问的突发条件为新冠病毒疫情，发生疫情时，银行贷款的主要目标从银行获取收益最高转换为企业收入最高，即通过调整信贷策略帮助企业减少利息支出。首先得出有突发因素影响下的企业信誉指数，得出贷款额度；接下来，建立企业收益最优化模型，求解企业收益最大时的贷款利率。

3 模型假设

1. 销项收入为企业主营业务收入，进项成本为主营业务成本。
2. 退货金额对企业售后服务质量有显著影响。
3. E1~E425 企业的抽样为随机抽样，服从正态分布。

4 符号说明

符号	说明	单位
R_i	第 i 个企业三年内平均每单退货金额	元/单
I_i	第 i 个企业的三年销售收入	元
G_i	第 i 个企业三年的销售收入增长率	%
F_i	第 i 个企业销售时的交易失败率	%
y_i	第 i 家企业信誉评价指数	
L_i	第 i 个企业的贷款额度	元

S	银行贷款总额	亿元
C_i	第 家企业进项有效发票的总金额	元

5 模型的建立与求解

5.1 问题一

首先我们确定了衡量企业信誉的 4 个指标，用以衡量信誉评级。接下来建立信誉等级的预测模型，运用 BP 神经网络建立二分类预测模型判断企业是否存在违约风险，若存在违约风险则信誉评级置为 D，再通过组合决策树模型确定企业信誉指数，确定具体等级。其次确定信贷策略，若信誉等级为 D，则不放贷。贷款额度可通过企业信誉评价指数确定。针对利率，我们考虑建立银行总收益优化模型，寻找总收益最大时的利率。

5.1.1 数据选取

对附件 1 中的进项和销项发票信息进行简要分析，我们发现交易信息主要集中在 2017 年、2018 年和 2019 年。利用 Python，我们绘制了 123 家企业每月总计的交易次数随时间变化的曲线图，如下图：

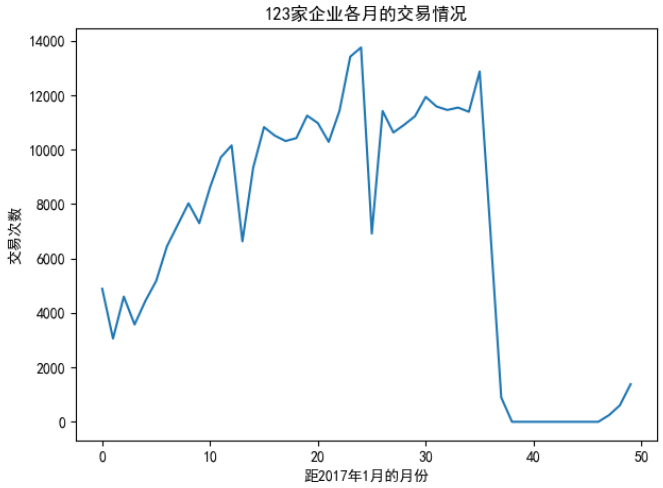

图 2 123 家企业各月交易次数总计

从图中看出 2019 年 12 月之后，123 家企业交易数据显著下降，其中大多数企业并没有 2020 年的相应交易数据，因此不妨认为数据的时间跨度认为 2017 年 1 月-2019 年 12 月，2020 年之后数据不予考虑。

5.1.2 模型建立

1. 确定评价指标

(1) 销售收入

收入是企业销售产品取得的经济利益。销售收入是直观衡量一个企业的销售状况的指标之一。企业销售的产品越多，收入就越多，其经营状况就越好，则企业能及时还债，违约概率低，所以该企业信誉度高，信贷风险低。反之，信誉度

低，信贷风险高。

我们定义第i个企业的销售收入为I_i。

作废发票、负数发票表示此次交易失败，不计入成本和收入，因此在计算过程中不予考虑，剔除作废发票、负数发票的数据之后进行计算。

(2)销项交易失败率

现实生活中，不仅存在有效发票，还存在作废发票和负数发票。这两类发票都代表这此次交易失败，所以我们将这两种情况合并考虑，统称为交易失败发票。交易失败率可以从供求关系的角度解释企业的销售是否稳定。交易失败率越高，企业的信用会受到影响，信誉度降低，信贷风险提高。若该企业交易失败率低，代表企业与客户会形成良好的契约关系，客户二次消费的概率也会提高，企业信誉也随之提高。此处我们只关注销售商品时交易失败的情况，不考虑进项时的退货情况。我们将失败率定义为交易失败次数与交易总次数之比，则第i个企业销售时的交易失败率F_i为：

$$F_i = \frac{n_i + v_i}{N_i} \tag{1}$$

其中，n_i表示第i个企业销售时的负数发票的数量，v_i表示第i个企业销售时的作废发票的数量，N_i表示第i个企业销售交易的总次数。

(3)销售收入增长率

销售收入增长率可以反应出企业在一段时间内销售收入的变化程度。增长率高，说明企业未来的运营情况比较乐观，企业的竞争力水平会提高，信贷风险会降低，所以我们选取第i个企业销售收入增长率G_i作为衡量指标，其定义如下：

$$G_i = \sqrt[3]{\frac{I_{i,j}}{I_{i,j-3}}} - 1 \tag{2}$$

其中，$I_{i,j}$表示第i个企业第j年销售总收入。

(4)平均每单退货金额

如今企业都在逐渐转型，不断优化自身的售后服务。企业能够接受大金额的退款，体现了企业以客户为中心的价值体系，希望给客户带来更好的体验，客户对企业的信赖程度便会提高，企业的信誉也会提高。所以我们进一步对退货金额进行研究，选取平均每单退货金额作为衡量企业的信誉的指标。我们定义第i个企业平均每单退货金额R_i为：

$$R_i = \frac{r_i}{n_i} \tag{3}$$

其中，r_i表示第i个企业销售时退货的总金额。

2.信誉评级

(1)潜在违约风险——划分 D 与非 D 评级

考虑到企业的信贷风险综合评价是一件十分复杂的事情，且我们需要从有限的数据当中分析出企业的信贷风险指标，鉴于神经网络具有强大的自我学习和改善的能力，能够从有限的数据当中发掘出隐藏的关系，我们首先使用前馈式多层感知机的 BP-反向传播网络模型对数据进行初步分析，这里我们首先对企业潜在的违约行为进行预测分析。

我们假设违约企业的信誉评级为D，若企业存在潜在违约风险不予放贷。

我们选用交互熵作为损失函数的形式，采用非线性激活函数模型，同时为减少迭代次数，尽量保证正确率我们选取一个3隐节点的一层隐层模型来建立反向传播网络模型。我们将销售收入、交易失败率、销售收入增长率和平均每单退货金额4个指标作为输入变量，是否违约作为输出变量，得出预测企业是否违约的模型。对于分类数据0表示“否”，即没有违约，分类数据1表示“是”，即违约。

(2)企业信誉评价指数——划分A、B、C评级

在此模型基础上，我们可以推测出所有可能违约的企业，并不予放贷，这里我们默认潜在违约企业的信誉评级为D，在此基础上我们进一步想对所有信誉评级进行划分，同时衡量4个指标对决策的影响程度，我们建立企业信誉评价指数函数作为衡量指标。企业信誉评价指数越高，则代表信誉越好，信贷风险小。企业信誉评价指数的定义如下：

$$y=\alpha x_1+\beta x_2+\gamma x_3+\theta x_4 \tag{4}$$

其中，y为企业信誉评价指标，x_1表示企业收入指标（销售收入I_i），x_2表示企业稳定性指标（销项交易失败率F_i），x_3表示企业经营指标（销售收入增长率G_i），x_4表示企业服务指标（销项平均退款金额R_i）。β应为负数，失败率越高，企业稳定性越差，竞争力越低，α,γ,θ均为正数。

3.贷款额度

企业贷款额度受企业信誉指数的影响，即信誉越高，风险越低，贷款额度变会提高。

计算出每个企业的竞争力指数后，我们可以依据每个企业的竞争力来确定每个企业的贷款比率，贷款比率为企业获得的贷款与银行年度贷款总额的比值。则第i个企业的贷款额度比率l_i为：

$$l_i=\frac{y_i}{\sum_1^n y_i} \tag{5}$$

其中，y_i为第i个企业的信誉度，n为放贷企业的数量（不包含评级为D的企业）。通过银行年度贷款额度，进一步我们计算出贷款额度。其中，信誉等级为D的企业不予放贷。第i个企业的贷款额度L_i（$L_i\in[10,100]$）为：

$$L_i=l_i\times S \tag{6}$$

其中，S为年度信贷总额。

4.收益最优模型确定利率

利率是一段时间内，应付利息与贷款本金的比率。银行需要在保证风险降低的同时，得到较高的收益，即应收利息。收益与贷款额度和贷款利率呈正相关，额度与利率越高，则银行收益越高，利润与客户流失率呈负相关，顾客流失率增高，向银行贷款的人会减少，银行收益会随之减少。则我们定义银行收益率Rr为：

$$Rr=(1-c_i)\times\varphi_i \tag{7}$$

其中，c_i为银行的顾客流失率。φ_i为第i家企业的贷款利率。

根据附件 3，我们可以得到 A、B 和 C 评级的顾客流失率与放贷利率之间的关系：

$$c_i = f(\varphi_i) \tag{8}$$

接下来，我们需要计算银行获得的总收益，得出收益最大时的贷款利率。不妨建立一个最优化模型，以收益为优化目标函数，以利率和贷款额度为限制条件，得到如下最优化模型：

$$Max \quad \sum_{i=1}^{123} L_i \times (1-c_i) \times \varphi_i$$

$$S.T. \begin{cases} \varphi_i \in [0.04, 0.15] \\ L_i \in [10, 100] \end{cases}$$

其中，$c_i = f(\varphi_i)$，L_i 为第 i 家企业获得的贷款。

因为 L_i 与 φ_i 无关，所以我们可以进一步将目标函数转换为：$Max \quad Rr = (1-c_i)\varphi_i$，则最优化模型为：

$$Max \quad Rr = (1-c_i)\varphi_i$$

$$S.T. \begin{cases} \varphi_i \in [0.04, 0.15] \\ L_i \in [10, 100] \end{cases} \tag{9}$$

银行收益率最大时对应的利率即为我们所求的贷款利率。

5.1.3 模型求解

全模型求解过程流程图如下所示

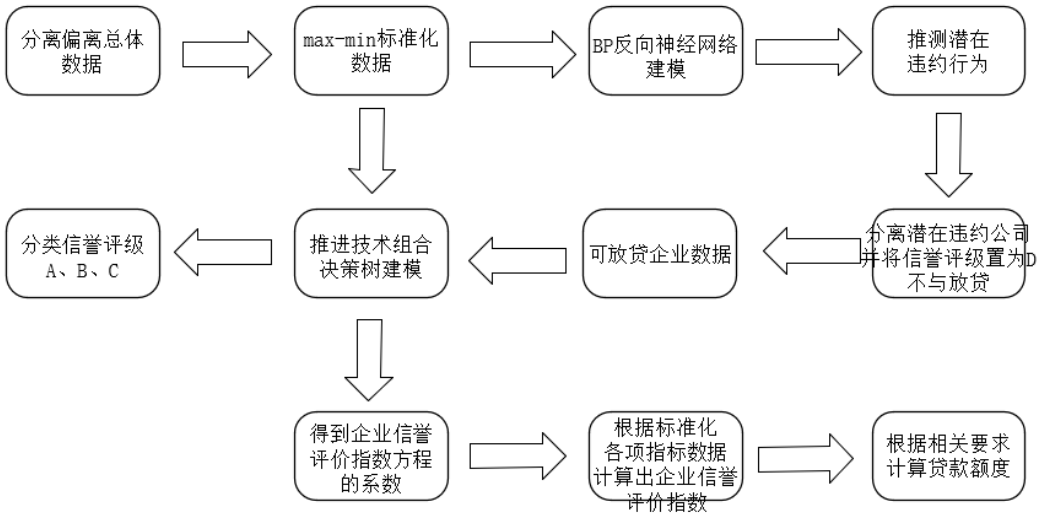

图 3-1 信誉评级、是否放贷及贷款额度的求解过程

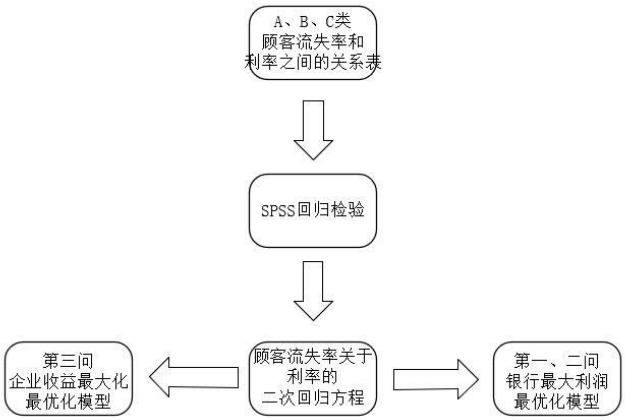

图 3-2 利率的求解过程

1. 数据准备

在初步数据分析阶段，我们发现 123 个企业中的各个变量当中存在一些偏离总体极大的数据项，考虑到样本是大样本，我们首先将这些数据使用 3σ 原则进行剔除。假设变量服从，均值为 μ，标准差为 σ 的正态分布，变量数值分布在 $(\mu-3\sigma,\mu+3\sigma)$ 中的概率为 99.74%，超过这个区间的数据即可认为是随机误差导致了偏差，将其剔除。

此外，将企业信誉评价等级和是否违约等标志进行数字化转换，例如 0 表示 A 类，1 表示 B 类；剔除偏离总体的样本以后，剩余 100 个样本可被用于进一步的数据分析，我们对着 100 个企业的每项指标进行 max-min 标准化，去除各项变量的量纲以待进一步分析处理工作。

2. 信誉评级

(1)潜在违约风险——划分 D 与非 D 评级

我们使用 R 语言 neuralnet 包下的 neuralnet 方法来进行建模，将数据代入模型，可得模型中参数和概率分割值，模型如下：

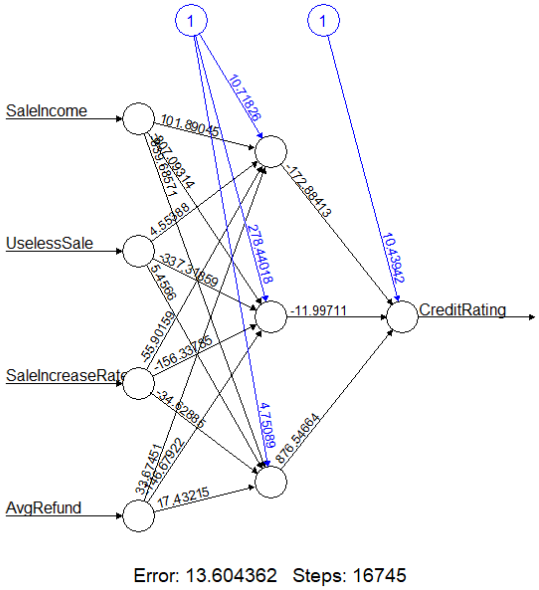

图 4 神经网络结果示意图

Min.	0.0000
1st Qu.	0.0000
Median	0.0101
Mean	0.2400
3rd Qu.	0.1774
Max.	1.0000

表 1 概率分割值分布

为了探寻临界概率分割值将两类进行分开，我们需要绘制 ROC 曲线进行分析，横轴代表负正类率，代表分类器预测的正类中实际负实例占所有负实例的比例；纵轴代表代表分类器预测的负类中实际负实例占所有负实例的比例。曲线下方面积越大，分类效果越好。这里采用 R 语言中的 ROCR 包下的 prediction 和 performance 方法绘制 ROC 曲线图，并将特定点绘制于 ROC 曲线中：

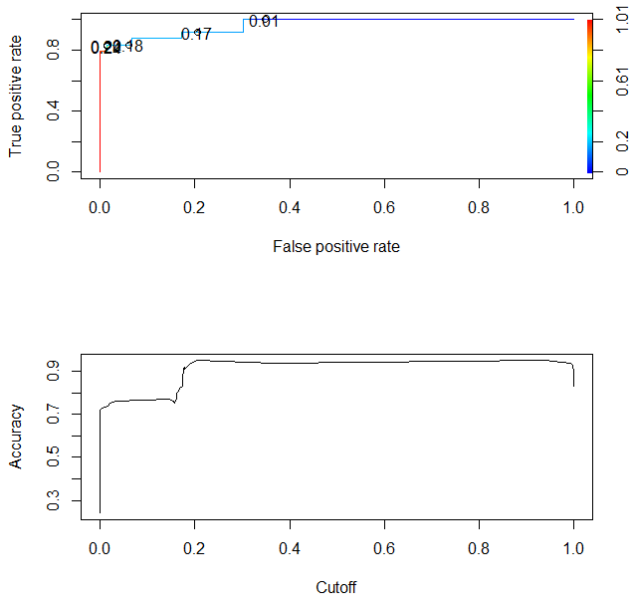

图 5 ROC 曲线及分割点

可以看出当概率分割值在 0.22 左右时模型精确度较高，这里我们取 0.22 为概率分割值，并利用模型对训练样本自身进行误差分析，发现错判率为 6%，模型精度较高，之后我们对被剔除出去的 23 个企业进行误差分析，发现错判率为 13.04%，模型精度较高，考虑到样本量较少，可能会导致误差增大，实际精度可能会高于测试值。所以用此模型预测企业是否违约是合理的。

(2)企业信誉评价指数

- **数据处理**

我们带入各个指标的 max-min 标准化数据，进而去除量纲对最终指标的影响，同时我们去除每一类中偏离总体的值，这类值可能会导致标准化指标极小从而使得指标可信度下降，这些偏离总体的企业我们不把他们列入系数确定时的样本。

对于偏离总体数据的的处理：

这里，由于我们采用的是 max-min 标准化，故数据的标准化仅受最大值最小值和自身的影响，因此偏离总体的值的变量中必然有一个变量会极大或者极小，这个变量会直接大幅度影响最终企业的信誉评价指数，故我们单独对这些偏离总体的企业进行标准化处理，并带入数值计算出企业信誉评价指数，一般来说这样的处理方式势必会带来一定的误差，但是在存在极端值的情况下，这样的误差将被有效的减少。

- **组合决策树模型建立**

这里我们为了确定各个变量的系数，我们需要寻找一个可行的方法来计算。我们为了同时将企业的信誉评级为 A，B，C 的企业划分出来，我们考虑使用袋装

技术的组合决策树模型，在给出各项权重的同时确定企业的信誉评级。

考虑到所有潜在违约行为的企业都默认被评级为 D 级，为了确定企业信誉评价指标相关变量的系数、并最终分类，我们初次建模时将 A~C（0~2）类所有不偏离总体的数据带入模型中进行学习，这里我们采用 R 语言中的 adabag 包下的 rpart.control 和 bagging 方法进行推进技术的组合决策树模型建立，这里我们复杂度参数选择默认值 0.01，采用 10 折交叉验证，经过反复测试我们发现当结点最小样本量为 8，树最大深度为 8 时模型精度较优，错误率仅为 8.86%。

模型运行完毕后得到的销售收入、交易失败率、销售收入增长率和平均每单退货金额的权重分别为 23.48，22.63，28.02，25.87。

这里在神经网络模型推算出潜在违约企业之后，我们使用组合决策树模型进行优化，进一步筛选出模型预测结果为 D 类的企业并不予放贷，剩余的 A，B，C 类企业可参与下一步信贷策略的指定。

这里将各变量权重作为系数带入方程中，得到最终的企业信誉评价指标计算公式如下：

$$y = 23.48x_1 - 22.63x_2 + 28.02x_3 + 25.87x_4$$

针对 123 家企业，我们首先剔除出 23 个偏离总体的样本，并针对偏离总体和未偏离总体的样本分别进行标准化，再进行合并，我们将 123 家企业的标准化指标数据带入方程得到全部企业的信誉评价指标如图所示：

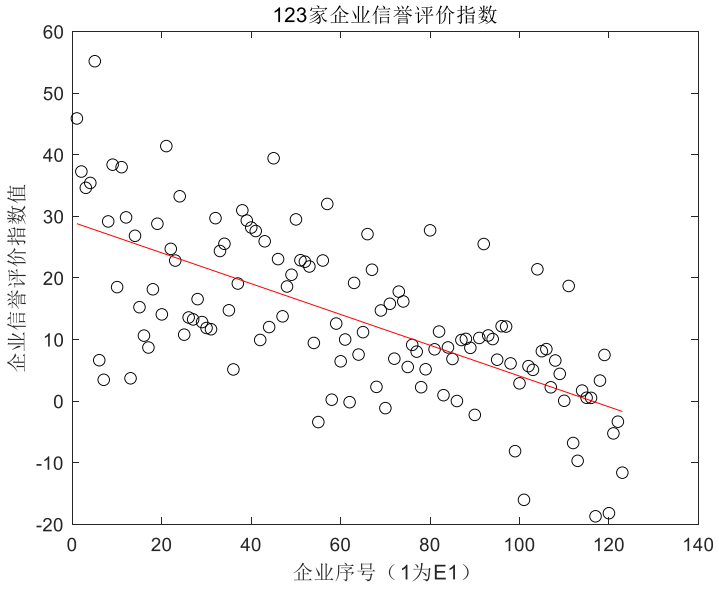

图 6 企业信誉评价指数

绝大多数低信誉评级的企业序号都靠后，整体可以看出企业信誉评价基本随着企业信誉等级降低而降低。

3. 贷款额度

本问中，银行银行年度贷款额度固定，故我们只分析了各企业的贷款比率。在筛选出所有 27 家信誉评级为的 D 级企业后，我们将所有 96 个可放贷企业的信誉评价指数 y_i 进行标准化 y_i'，然后将数据代入式(5)得出所有可放贷企业的放贷比率，银行放贷情况如下：（企业序号 1 代表 E1，空白表示不予贷款）

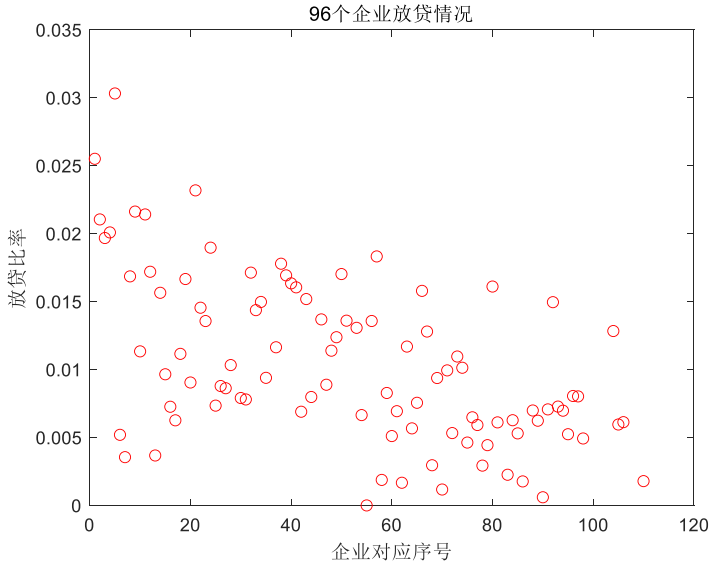

图 7 对不同企业的放贷比率

从图中我们可以看出，大部分企业的贷款比率集中在 0.5%至 2%，少数企业的贷款比率在 0.5%以下，个别企业贷款比率在 2%以上。

4. 利率

首先我们对客户流失率与利率的关系求解。以等级为 A 为例，B、C 等级的关系求解同理。步骤如下：

- 划出等级为 A 时的客户流失率与利率的散点图，用 SPSS 分别进行一次函数和二次函数的拟合。
- 比较两种拟合的判定系数 R^2，选择判定系数高的模型
- 对所选模型进行关系显著性进行检验。

最终我们选择了二次模型，判定系数为 0.993，通过显著性检验，得出的关系表达式为：$y=-73.748\varphi^2+21.549\varphi-0.681$。同理可得出其他等级的关系模型。

客户流失率与利率的关系 $y_i=f(\varphi_i)$ 如下：

$$y_i=\begin{cases}-73.748\varphi_i^2+21.549\varphi_i-0.681, \text{当企业}i\text{为A类}\\ -65.709\varphi_i^2+19.843\varphi_i-0.673, \text{当企业}i\text{为B类}\\ -61.485\varphi_i^2+19.167\varphi_i-0.625, \text{当企业}i\text{为C类}\end{cases}$$

接下来，我们对收益最优化模型进行求解，首先我们可以绘制收益率随利率变化的趋势，收益率随着利率的上升，先上升，后下降，再上升，收益率最大时，利率在 6%左右，如下图：

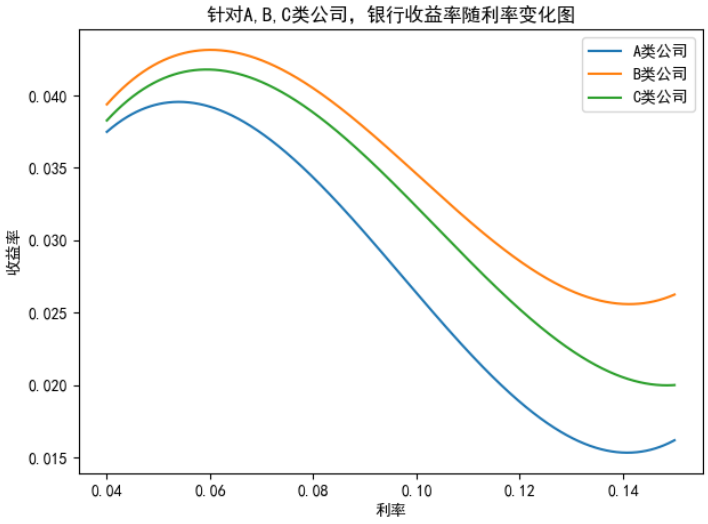

图 8 银行收益率随利率变化的曲线图

利用梯度下降法对目标函数在 $\varphi_i \in [0.04, 0.15]$ 进行求解银行收益率最大时，对 A、B、C 等级企业的贷款利率如下表：

表 2 各评级企业贷款系数

φ_A	0.053983983983983984
φ_B	0.06004004004004004
φ_C	0.05937937937937938

5.2 问题二

在问题一的基础上，我们将附件 2 中没有信贷记录的企业相关数据代入模型，即可判断企业的信誉评级，进而求出贷款额度，得出贷款策略。

5.2.1 数据处理

我们将附件 2 中的数据代入问题一中的模型，求解出各个企业对应的指标数据，并作为初始数据处理，首先对初始数据进行初步筛选，剔除 3sigma 以外的偏离总体的数据，并分两组分别计算各项指标的 max-min 标准化值，再进行合并。

5.2.2 模型求解

模型求解的步骤如下：

*Step*1:将标准化指标带入 BP-反向传播模型中对企业的潜在违约行为进行评测，剔除出预测标志为 1，即可能违约的企业，并将这些企业的信誉评级置为 D 级。

*Step*2:针对剩下的可贷款企业使用组合决策树模型预测出 A、B、C 类评级。

*Step*3:计算每个企业的企业信誉评价指数，并将企业信誉评价指数进行标准化处理。

*Step*4：将银行年度贷款总额 $S=1$（亿元）代入，可得每个企业的贷款利率和额度，贷款超过 100 万的企业按 100 万处理，小于 10 万的企业按 10 万处理。

5.2.3 信贷策略

针对是否放贷，若企业信誉评级为 A、B、C，则可以放贷，若评级为 D，则不予放贷。

针对放贷额度，企业信誉越高，贷款额度越高，带入标准化信誉评价指数，根据式（5）计算出各企业贷款比例，贷款额度即可看出评价指数变化。

进一步，通过第一问得知银行借贷只与企业的信誉等级相关，当贷款企业信誉评级为 A 类时，贷款收益率最大时的利率 $\varphi_A = 0.053983983983983984$；当贷款企业信誉评级为 B 类时，借贷收益率最大时的利率 $\varphi_B = 0.06004004004004004$；当贷款企业信誉评级为 C 类时，借贷收益率最大时的利率 $\varphi_C = 0.05937937937937938$。

结合放贷额度和利率，我们绘制 E124~E425 企业的信贷策略气泡图，其中气泡的大小表示该家企业的贷款利率，纵坐标表示贷款额度，没有数据表示不予贷款；气泡图绘制如下（蓝色气泡代表等级为 A 的企业，黄色气泡代表等级为 B 的企业，灰色气泡代表等级为 C 的企业）

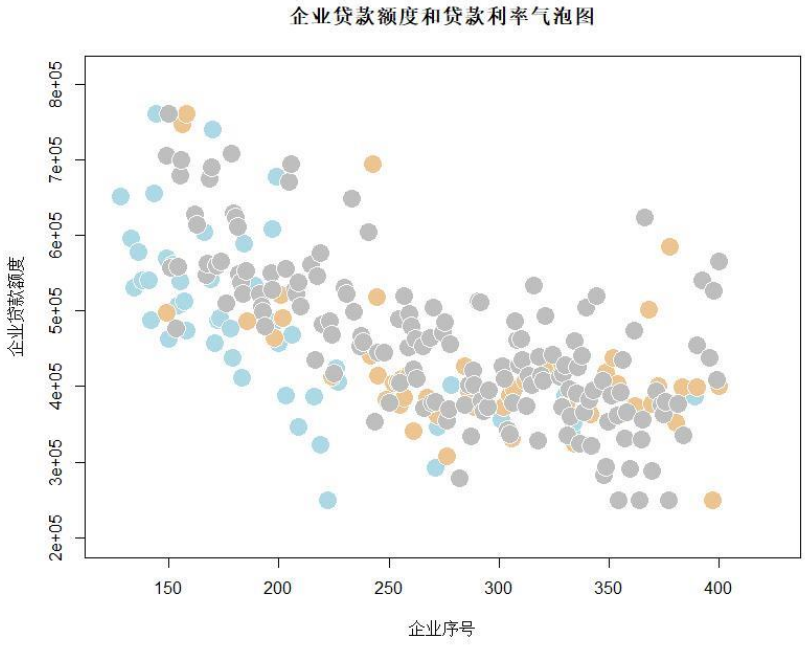

图 9 信贷额度和贷款利率气泡图

由于利率差距较小，气泡大小并不能很好的看出利率的变化，但是可以看出信誉评级为 A 的企业的贷款额度普遍高于等级 B 和 C 的贷款额度，而评级为 B 和 C 的贷款额度相差不多，在银行进行放贷时可以给 A 的贷款额度相对高而 B 和 C 的额度相对低，给 A 的利率相对低而给 B 和 C 的利率相对高。

5.3 问题三

本问研究突发因素为新冠病毒疫情时，企业信贷风险的变化情况。此时，银行的目标是帮助企业度过难关，尽可能降低企业的负担，而并非以获取最大收益为目标。

5.3.1 背景假设

我们站在银行层面提出假设：

1.疫情从 1 月持续到 10 月，10 月以后不会发生大型公共卫生问题。

2.从发现第一例死亡患者 3 个月之后，国家可以有效地控制疫情，实现企业的全面复工。

3.第一例死亡患者发现的 3 个月内各行业的 GDP 同期增长率符合 2020 年新冠疫情的在第一季度的 GDP 增长率[1]。

4.住宿和餐饮业、租赁和商业服务业以外需要一整年的时间来恢复到过去同期的 GDP 增长速度，按照 2020 年第二季度的增长率增长[2]，其余行业可以在 3 个月之后恢复到过去同期 GDP 增长速度。

5.无突发因素影响时，各季度 GDP 同期增长率按照 2019 年度的固定速度增长[3]。

6.大型公共卫生问题不会影响到企业的每单退款金额和无效交易率的变动。

5.3.2 模型建立

我们对问题一种企业信誉评价指数的表达式进行变化，得到：

$$y = 23.48I_i - 22.63F_i + 28.02G_i + 25.87R_i$$

若第 j 个月发生大型公共卫生问题，则企业销售收入和平均销售增长率会发生变化,其中销售收入为：

$$I_i = \begin{cases} profit_{1,i} + profit_{2,i} + profit_{2,i} \times \dfrac{j}{12}(1+\eta_i) + profit_{2,i} \times \dfrac{3}{12}(1+\eta_i^{'}) + profit_{2,i} \times \dfrac{12-3-j}{12}(1+\eta_i), \\ 企业i属于住宿和餐饮业、租赁和商业服务业 \\ profit_{1,i} + profit_{2,i} + profit_{2,i} \times (1+\eta_i^{''}), \\ 企业i不属于住宿和餐饮业、租赁和商业服务业 \end{cases}$$

其中，$profit_{t,i}$ 为第 t 年，企业 i 的销售收入，η_i 为企业 i 的正常情况下销售收入增长率，$\eta_i^{'}$ 企业 i 的疫情情况下销售收入增长率，进一步，可转化为：

$$I_i = \begin{cases} profit_{1,i} + profit_{2,i} + profit_{2,i}(1+\dfrac{9}{12}\eta_i + \dfrac{3}{12}\eta_i^{'}), \\ 企业i属于住宿和餐饮业、租赁和商业服务业 \\ profit_{1,i} + profit_{2,i} + profit_{2,i} \times (1+\eta_i^{''}), \\ 企业i不属于住宿和餐饮业、租赁和商业服务业 \end{cases} \tag{10}$$

销售收入增长率为：

$$G_i = \sqrt[3]{\frac{I_i - profit_{1,i} - profit_{2,i}}{profit_{1,i}}} - 1 \tag{11}$$

计算得出各个企业信誉指数后，通过贷款比率，可以得出各个企业所对应的贷款金额。

为使企业第三年获得最大收益，建立优化模型，确定企业收益最大时的贷款利率：

$$\max \sum_{i=1}^{302}[profit_3 - E(C_i)]$$
$$S.T.\begin{cases} 0.04 \le \varphi_i \le 0.15 \\ 10 \le L_i \le 100 \end{cases} \tag{12}$$

$$E(C_i) = (1 - f(\varphi_i))(L_i\varphi_i + C_i) + f(\varphi_i)C_i \tag{13}$$

其中，$f(\varphi_i)$为银行顾客流失率与利率的函数关系，φ_i为利率，C_i为第i家企业的成本，$E(C_i)$表示第i家企业的成本的期望值，L_i为第i家企业的贷款，y_i为第i家企业的企业信誉指数。

观察目标函数，可以看出目标函数的最优解必定为各个企业的最优解的集合，即目标函数可转换为求出每个企业的最优解：

$$\sum_{i=1}^{302}\max[profit_3 - E(C_i)]$$
$$S.T.\begin{cases} 0.04 \le \varphi_i \le 0.15 \\ 10 \le L_i \le 100 \end{cases} \tag{14}$$

5.3.3 模型求解

将附件 2 中处理过的数据代入上述最优化模型，利用梯度下降法进行求解，得出各个企业的对应的利率，将其做成散点图，如下图所示，不难看出 E124~E425 公司的利率散点图集中在 0.060，0.054，0.040 附近。

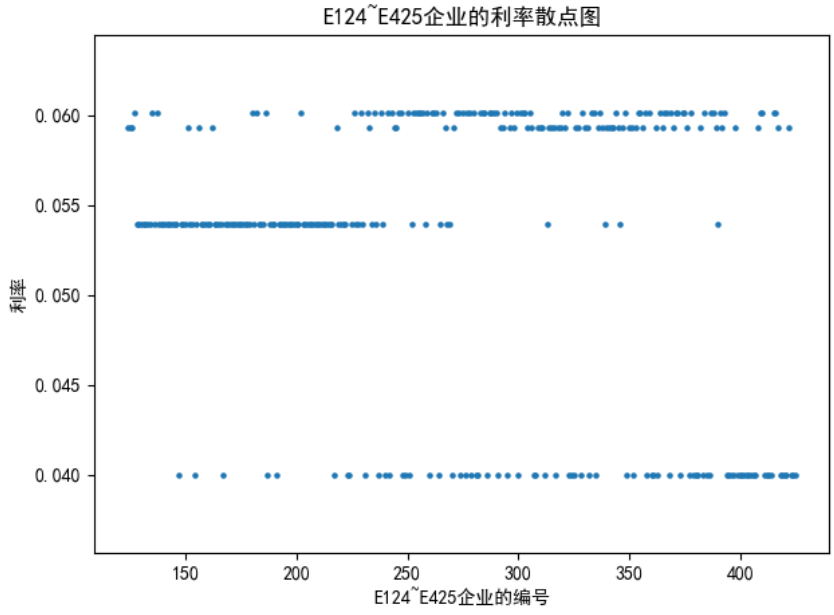

图 10 企业贷款利率散点图

我们对结果进行可视化，绘制企业贷款额度和利率的气泡图，其中横坐标表示第i家企业的编号，气泡的大小表示第i家企业的利率，纵坐标表示第i家企业的贷款额度。（蓝色气泡代表等级为 A 的企业，黄色气泡代表等级为 B 的企业，

灰色气泡代表等级为 C 的企业）

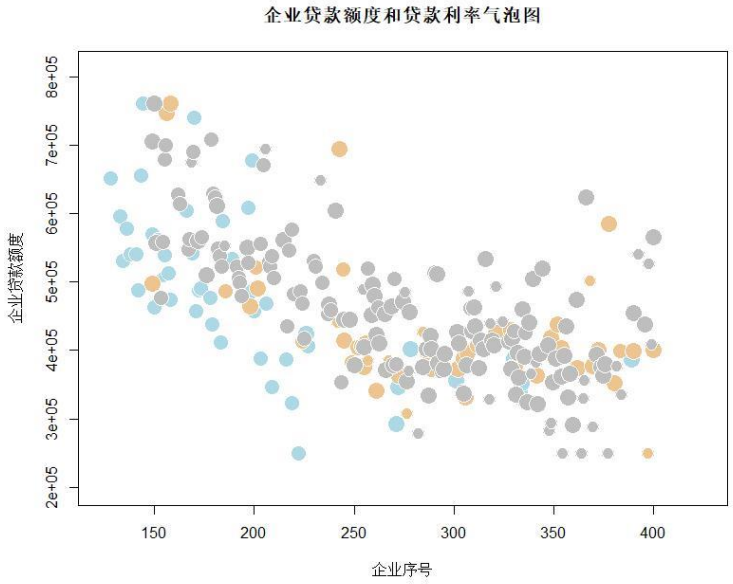

图 11 疫情下企业贷款额度和贷款利率气泡图

可以看出，企业之间贷款利率与额度并不相等，从总体来看，A 级企业贷款额度偏高，B、C 级企业的贷款额度大致相同，由于单从气泡图难以分辨企业贷款额度在疫情期间和正常时期的改变，进一步我们绘制图像对疫情下与正常情况下贷款的差异：

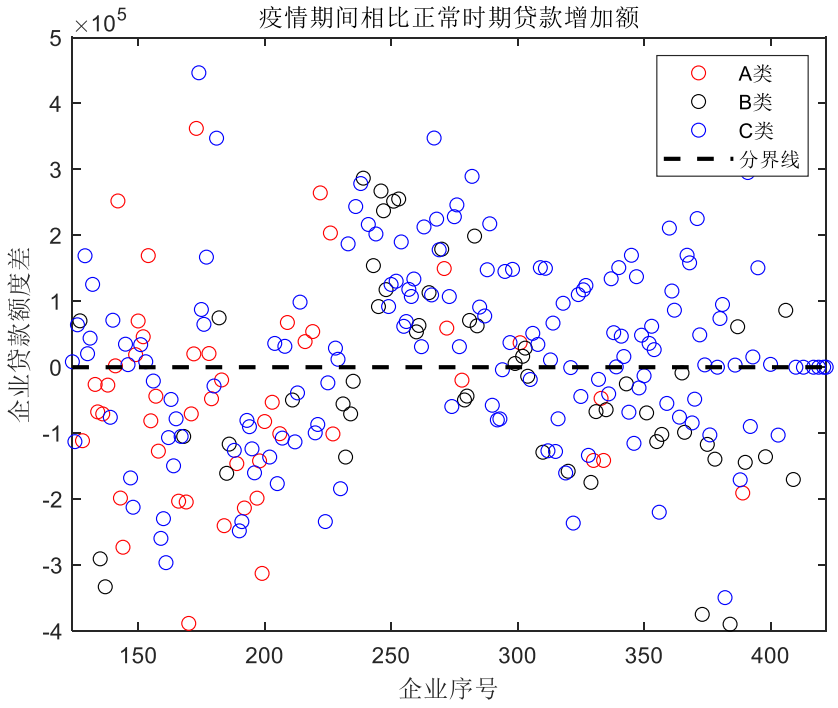

图 12 疫情期间比正常时期贷款的增加额

观察图像，我们可以看出，A 级企业的贷款额度总体减少，B、C 级企业的贷款额度总体呈现增加的态势。

5.3.4 策略调整

综上所述，在发生疫情时，为了帮助中小企业度过难关，银行首先应该降低对企业的贷款利率，减少企业的利息支出，另一方面，针对信誉等级高的企业，减少贷款，针对信誉等级稍低的企业，应提高贷款额度，帮助其资金周转，度过经济寒冬，此外不同行业的企业在不同的情况下能得到的银行利率也会有所变化，例如在疫情期间的医疗企业的贷款额度会明显地增加。

6 模型优缺点

6.1 优点

1. 信誉评级分类模型采用二段分类模型，在误差更小小的情况下进行一分类，进一步多分类减小误差。
2. 分类器误差较小，可信度高。
3. 剔除以及标准化数据使得数据处理过程中误差减少。
4. 在不同的情况下，银行在不同角度建立最优化模型，适应环境变化，使模型贴切实际。

6.2 缺点

1. 除了已经提出的指标之外，可能存在某些尚未发掘的指标与结果相关。
2. 在第一问中的分类模型建立中由于样本量少的原因可能会导致一定的误差。

参 考 文 献

[1] 中国统计局.2020 年一季度国内生产总值（GDP）初步核算结果[E].http://www.stats.gov.cn/tjsj/zxfb/202004/t20200417_1739602.html . 2020-04-18

[2] 中国统计局. 2020 年二季度和上半年国内生产总值(GDP)初步核算结果[E]. http://www.stats.gov.cn/tjsj/zxfb/202007/t20200717_1776516.html . 2020-07-17

[3] 中国统计局. 2019 年四季度和全年国内生产总值（GDP）初步核算结果[E]. http://www.stats.gov.cn/tjsj/zxfb/202001/t20200117_1723591.html . 2020-01-18

[4] GB/T 4754—2017,国民经济行业分类[S].中华人民共和国:中华人民共和国国家质量监督检验检疫总局，中国国家标准化管理委员会,2017-10-01.

[5] 毕马威中国.新冠肺炎疫情的行业影响和未来发展趋势[Z]. 中国：毕马威中国, 2020-03

[6] 川财证券.疫情对资本市场各行业影响相关性分析[Z].中国:川财证券,2020/2/4.

[7] 薛薇.R 语音数据挖掘[M].中国人民大学出版社:北京,2016.4:1-.

附　录

附件清单：

1.代码文件列表
(1)Python 文件
第一问利率最优化模型求解.py
前 123 家企业的数据处理.py
梯度下降法求解第三问利率最优化模型.py
第二问气泡图.py
处理第二问相关数据.py
(2)matlab 文件
cal_index.m
div_class.m
in_3sigma.m
in_3sigma2.m
max_min.m
pre_dataprocess.m
data_process_11.m
data_process_12.m
data_process_13.m
data_process_21.m
data_process_22.m
data_process_31.m
data_process_32.m
plots.m
(3)R 语言文件
prob1.R
prob2.R
prob3.R

2.数据文件
(1)Python 数据文件
123 家企业指标.csv
123 家指标.csv
302 公司.csv
302 公司名.csv
302 家企业行业划分.csv
302 进项发票.csv
302 销项发票.csv
A 类顾客流失率与利率.spv
B 类顾客流失率与利率.spv
C 类顾客流失率与利率.spv
企业信息.csv

分类后.csv
拟合顾客流失率与利率的关系.sav
第三问 final.csv
第三问标准化评价指数.csv
进项发票信息.csv
销项发票信息.csv

(2)matlab 数据文件
%% 注意每一问用到的数据文件都不同（名字可能相同）
%%第一问文件
all_data.mat
Category.mat
class.mat
data.mat
ind_data.mat
index.mat
is_lend.mat
norm_data.mat
processed_data.mat
prop.mat

%%第二问文件
all_index.mat
all_prop.mat
comb_data.mat
data1.mat
division.mat
index.mat
loan.mat
loan_situation.mat
predict_class.mat
stand_data.mat

%%第三问文件
all_index.mat
all_prop.mat
comb_data.mat
data1.mat
dis.mat
division.mat
index.mat
loan.mat
loan_situation.mat

predict_class.mat
stand_data.mat

(3) R 语言数据文件
##第一问
test_data.txt
train_data.txt
train_data2.txt

##第二问
plot.txt
test_data1.txt
test_data2.txt
输出预测分割值.txt
输出预测类别值.txt

##第三问
plot2.txt
test_data1.txt
test_data2.txt
输出预测分割值.txt
输出预测类别值.txt

****
Python 代码如下
****

#前 123 家企业的数据处理.py

```
import csv
from datetime import datetime
import numpy as np
import matplotlib.pyplot as plt

file_name1 = "进项发票信息.csv"
file_name2 = "销项发票信息.csv"
file_name3 = "企业信息.csv"
input_invoice = [[] for _ in range(123)]
output_invoice = [[] for _ in range(123)]
input_time = [[] for _ in range(123)]
output_time = [[] for _ in range(123)]
```

```
output_fail_count = [0 for _ in range(123)]
output_final_count = [0 for _ in range(123)]
input_fail_count = [0 for _ in range(123)]
input_final_count = [0 for _ in range(123)]
profit = [0 for _ in range(123)]
cost = [0 for _ in range(123)]
name = []
fail_money = [0 for _ in range(123)]
output_rate = [0 for _ in range(123)]

with open(file_name1, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            input_invoice[int(row[0][1:]) - 1].append(float(row[6]))
            input_time[int(row[0][1:])                                    -
1].append(datetime.strptime(row[2], "%m/%d/%Y"))
        elif row[7] == "有效发票" and float(row[6]) < 0:
            input_fail_count[int(row[0][1:]) - 1] += 1
        elif row[7] == "作废发票":
            input_fail_count[int(row[0][1:]) - 1] += 1
        input_final_count[int(row[0][1:]) - 1] += 1
with open(file_name2, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            output_invoice[int(row[0][1:]) - 1].append(float(row[6]))
            output_time[int(row[0][1:])                                   -
1].append(datetime.strptime(row[2], "%m/%d/%Y"))
        elif row[7] == "有效发票" and float(row[6]) < 0:
            output_fail_count[int(row[0][1:]) - 1] += 1

        # 等会可能删除
        elif row[7] == "作废发票":
            output_fail_count[int(row[0][1:]) - 1] += 1
        # 计算每单退货金额先注释掉
        output_final_count[int(row[0][1:]) - 1] += 1

        if float(row[6]) < 0:
            fail_money[int(row[0][1:]) - 1] += abs(float(row[6]))
with open(file_name3, encoding='utf-8') as f:
    reader = csv.reader(f)
```

```
    head_row = next(reader)
    for row in reader:
        name.append(row[1])
# for i in range(123):
#     for item in output_invoice[i]:
#         profit[i] += item
# for i in range(123):
#     for item in input_invoice[i]:
#         profit[i] -= item
for i in range(123):
    for item in output_invoice[i]:
        profit[i] += item
for i in range(123):
    for item in input_invoice[i]:
        cost[i] += item

input_fail_rate = []  # 作废发票率
for i in range(len(input_final_count)):
    input_fail_rate.append(input_fail_count[i] / input_final_count[i])
output_fail_rate = []  # 销项退货率+作废发票率
for i in range(len(output_final_count)):
    output_fail_rate.append(output_fail_count[i]                    /
output_final_count[i])

# 观测2017年1月到2019年12月的数据，多余数据delete
upstream_data = [[0 for _ in range(36)] for _ in range(123)]  # 进项
相关数据

for i in range(123):
    for j in range(len(input_time[i])):
        if    (input_time[i][j].year    -    2017)    *    12    +
input_time[i][j].month - 1 < 36:
            upstream_data[i][(input_time[i][j].year - 2017) * 12 +
input_time[i][j].month - 1] += input_invoice[i][j]
output_data = [[0 for _ in range(3)] for _ in range(123)]
for i in range(123):
    for j in range(len(output_time[i])):
        if    (output_time[i][j].year    -    2017)    *    12    +
output_time[i][j].month - 1 < 36:
            output_data[i][output_time[i][j].year    -    2017]    +=
output_invoice[i][j]

# for i in range(123):
#     for j in range(len(input_time[i])):
```

```
#                     if  (input_time[i][j].year  -  2017)  *  12  +
input_time[i][j].month - 1 < 36:
#                     output_data[i][input_time[i][j].year  -  2017]  -=
input_invoice[i][j]

    # 观测2017年1月到2019年12月的数据，多余数据delete
downstream_data = [[0 for _ in range(36)] for _ in range(123)]  # 销
项相关数据
for i in range(123):
    for j in range(len(output_time[i])):
        if     (output_time[i][j].year     -     2017)     *     12     +
output_time[i][j].month - 1 < 36:
            downstream_data[i][(output_time[i][j].year - 2017) * 12 +
output_time[i][j].month - 1] += output_invoice[i][
                j]
# 计算下游的方差
down_sigma2 = [0 for _ in range(123)]
for i in range(123):
    temp = []
    for j in range(12):  # 计算标准离差率
        average = (downstream_data[i][j] + downstream_data[i][j + 12]
+ downstream_data[i][j + 24]) / 3
        if average != 0:
            temp_aver  =  ((downstream_data[i][j]  -  average)  **  2  +
(downstream_data[i][j + 12] - average) ** 2 +
                           downstream_data[i][j  +  24]  -  average)  /
(average * 3)
        #       temp_aver       =       np.var([downstream_data[i][j],
downstream_data[i][j + 12], downstream_data[i][j + 24]]) / \
        #                                 np.mean([downstream_data[i][j],
downstream_data[i][j + 12], downstream_data[i][j + 24]])
        else:
            temp_aver = 0

        temp.append(temp_aver)
    down_sigma2[i] = np.mean(temp)

up_sigma2 = [0 for _ in range(123)]
for i in range(123):
    temp = []
    for j in range(12):  # 计算标准离差率
        average  =  (upstream_data[i][j]  +  upstream_data[i][j  +  12]  +
upstream_data[i][j + 24]) / 3
        if average != 0:
```

```
            temp_aver = ((upstream_data[i][j] - average) ** 2 +
(upstream_data[i][j + 12] - average) ** 2 +
                         upstream_data[i][j + 24] - average) /
(average * 3)
        else:
            temp_aver = 0
        temp.append(temp_aver)
    up_sigma2[i] = np.mean(temp)
# 单位原材料成本下的销售收入
res = [0 for _ in range(123)]

for i in range(123):
    res[i] = (profit[i] - cost[i]) / cost[i]
fail_money_rate = [0 for _ in range(123)]
for i in range(123):
    fail_money_rate[i] = fail_money[i] / sum(output_invoice[i])
# 测试
# a_dict = {}
# for i in range(123):
#     a_dict[res[i]] = i
# res.sort(reverse=True)
# for i in range(123):
#     print("E", a_dict[res[i]] + 1, name[a_dict[res[i]]], ":", res[i])


# plt.rcParams['font.sans-serif'] = ['SimHei']
# plt.xlabel("企业标号")
# plt.ylabel("单位原材料成本下销售收入")
# plt.title("123 家企业单位原材料的收益率情况")
# plt.plot([i for i in range(1, 124)], res)
# plt.show()

# for item in output_fail_rate:
#     print(item)


# 看这些公司的月份分布的情况
# month = [0 for _ in range(50)]
# for i in range(123):
#     for j in range(len(output_time[i])):
#                 month[(output_time[i][j].year - 2017) * 12 +
output_time[i][j].month - 1] += 1
# for i in range(123):
#     for j in range(len(input_time[i])):
```

```
#                   month[(input_time[i][j].year  -  2017)  *  12  +
input_time[i][j].month - 1] += 1
# plt.rcParams['font.sans-serif'] = ['SimHei']
# plt.xlabel("距 2017 年 1 月的月份")
# plt.ylabel("交易次数")
# plt.title("123 家企业各月的交易情况")
# plt.plot(month)
# plt.show()
#
growth_rate = []
for i in range(123):
    if output_data[i][0] != 0:
        growth_rate.append(pow(output_data[i][2] / output_data[i][0],
1 / 3) - 1)
    else:
        growth_rate.append(pow(output_data[i][2] / output_data[i][1],
1 / 2) - 1)
nb = [0 for _ in range(123)]
for i in range(123):
    if output_fail_count[i] == 0:
        nb[i] = 0
    else:
        nb[i] = fail_money[i] / output_fail_count[i]
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title("进项金额")
plt.plot(cost[1:])
plt.show()
for item in growth_rate:
    print(item)


#第一问利率最优模型求解
import numpy as np
from matplotlib import pyplot as plt
from scipy.optimize import fminbound


# 求 x 在(4%,15%)之间的极大值
def f1(x):
    return 1.681 * x - 21.549 * x ** 2 + 73.748 * x ** 3


def f2(x):
    return 1.673 * x - 19.843 * x ** 2 + 65.709 * x ** 3
```

```
def f3(x):
    return 1.625 * x - 19.167 * x ** 2 + 61.485 * x ** 3


x = np.linspace(0.04, 0.15, 100000)
max_of_A = 0
max_of_B = 0
max_of_C = 0

for i in range(100000):
    if max_of_A < f1(x[i]):
        max_of_A = f1(x[i])
        index_a = x[i]
    if max_of_B < f2(x[i]):
        max_of_B = f2(x[i])
        index_b = x[i]
    if max_of_C < f3(x[i]):
        max_of_C = f3(x[i])
        index_c = x[i]
print("A类最大:", index_a)
print("B类最大:", index_b)
print("C类最大:", index_c)
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.xlabel("利率")
plt.ylabel("收益率")
plt.title("针对A,B,C类公司，银行收益率随利率变化图")
plt.plot(x, f1(x))
plt.plot(x, f2(x))
plt.plot(x, f3(x))
plt.legend(["A类公司", "B类公司", "C类公司"])
plt.show()
```

```
#第二问相关数据
# 第二问处理数据
import csv
from datetime import datetime
import numpy as np
```

```
file_name1 = "302 进项发票.csv"
file_name2 = "302 销项发票.csv"
file_name3 = "302 公司.csv"
input_invoice = [[] for _ in range(302)]
input_time = [[] for _ in range(302)]
output_invoice = [[] for _ in range(302)]
output_time = [[] for _ in range(302)]
output_fail_count = [0 for _ in range(302)]
output_final_count = [0 for _ in range(302)]
profit = [0 for _ in range(302)]
cost = [0 for _ in range(302)]
name = []
fail_money = [0 for _ in range(302)]
with open(file_name1, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            input_invoice[int(row[0][1:]) - 124].append(float(row[6]))
            input_time[int(row[0][1:])                                  -
124].append(datetime.strptime(row[2], "%m/%d/%Y"))

with open(file_name2, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            output_invoice[int(row[0][1:])                              -
124].append(float(row[6]))
            output_time[int(row[0][1:])                                 -
124].append(datetime.strptime(row[2], "%m/%d/%Y"))
        elif row[7] == "有效发票" and float(row[6]) < 0:
            output_fail_count[int(row[0][1:]) - 124] += 1

        # 等会可能删除
        elif row[7] == "作废发票":
            output_fail_count[int(row[0][1:]) - 124] += 1
        # 计算每单退货金额先注释掉
        output_final_count[int(row[0][1:]) - 124] += 1

        if float(row[6]) < 0:
            fail_money[int(row[0][1:]) - 124] += abs(float(row[6]))
with open(file_name3, encoding='utf-8') as f:
```

```
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        name.append(row[1])

for i in range(302):
    for item in output_invoice[i]:
        profit[i] += item
for i in range(302):
    for item in input_invoice[i]:
        cost[i] += item

output_fail_rate = []  # 销项退货率&报废发票率
for i in range(len(output_final_count)):
    output_fail_rate.append(output_fail_count[i]                    /
output_final_count[i])

# 观测 2017 年 1 月到 2019 年 12 月的数据，多余数据 delete
upstream_data = [[0 for _ in range(36)] for _ in range(302)]  # 进项
相关数据

for i in range(302):
    for j in range(len(input_time[i])):
        if    (input_time[i][j].year    -    2017)    *    12    +
input_time[i][j].month - 1 < 36:
            upstream_data[i][(input_time[i][j].year - 2017) * 12 +
input_time[i][j].month - 1] += input_invoice[i][j]

    # 观测 2017 年 1 月到 2019 年 12 月的数据，多余数据 delete
downstream_data = [[0 for _ in range(36)] for _ in range(302)]  # 销
项相关数据
for i in range(302):
    for j in range(len(output_time[i])):
        if    (output_time[i][j].year    -    2017)    *    12    +
output_time[i][j].month - 1 < 36:
            downstream_data[i][(output_time[i][j].year - 2017) * 12 +
output_time[i][j].month - 1] += output_invoice[i][
                j]
# 计算下游的方差
down_sigma2 = [0 for _ in range(302)]
for i in range(302):
    temp = []
    for j in range(12):  # 计算标准离差率
        average = (downstream_data[i][j] + downstream_data[i][j + 12]
```

```
+ downstream_data[i][j + 24]) / 3
        if average != 0:
            temp_aver = ((downstream_data[i][j] - average) ** 2 +
(downstream_data[i][j + 12] - average) ** 2 +
                         downstream_data[i][j + 24] - average) /
(average * 3)
        #       temp_aver       =       np.var([downstream_data[i][j],
downstream_data[i][j + 12], downstream_data[i][j + 24]]) / \
        #                                 np.mean([downstream_data[i][j],
downstream_data[i][j + 12], downstream_data[i][j + 24]])
        else:
            temp_aver = 0

        temp.append(temp_aver)
    down_sigma2[i] = np.mean(temp)

up_sigma2 = [0 for _ in range(302)]
for i in range(302):
    temp = []
    for j in range(12):  # 计算标准离差率
        average = (upstream_data[i][j] + upstream_data[i][j + 12] +
upstream_data[i][j + 24]) / 3
        if average != 0:
            temp_aver = ((upstream_data[i][j] - average) ** 2 +
(upstream_data[i][j + 12] - average) ** 2 +
                         upstream_data[i][j + 24] - average) /
(average * 3)
        else:
            temp_aver = 0
        temp.append(temp_aver)
    up_sigma2[i] = np.mean(temp)
# 单位原材料成本下的销售收入
res = [0 for _ in range(302)]

for i in range(302):
    res[i] = (profit[i] - cost[i]) / cost[i]
# 测试
# a_dict = {}
# for i in range(123):
#     a_dict[res[i]] = i
# res.sort(reverse=True)
# for i in range(123):
#     print("E", a_dict[res[i]] + 1, name[a_dict[res[i]]], ":", res[i])
```

```
import matplotlib.pyplot as plt

# plt.rcParams['font.sans-serif'] = ['SimHei']
# plt.xlabel("企业标号")
# plt.ylabel("单位原材料成本下销售收入")
# plt.title("123 家企业单位原材料的收益率情况")
# plt.plot([i for i in range(1, 124)], res)
# plt.show()

# for item in down_sigma2:
#     print(item)


# 看这些公司的月份分布的情况
# month = [0 for _ in range(50)]
# for i in range(123):
#     for j in range(len(output_time[i])):
#                   month[(output_time[i][j].year - 2017) * 12 +
output_time[i][j].month - 1] += 1
# for i in range(123):
#     for j in range(len(input_time[i])):
#                   month[(input_time[i][j].year - 2017) * 12 +
input_time[i][j].month - 1] += 1
# plt.rcParams['font.sans-serif'] = ['SimHei']
# plt.xlabel("距 2017 年 1 月的月份")
# plt.ylabel("交易次数")
# plt.title("123 家企业各月的交易情况")
# plt.plot(month)
# plt.show()

output_data = [[0 for _ in range(3)] for _ in range(302)]
for i in range(302):
    for j in range(len(output_time[i])):
        if    (output_time[i][j].year    -    2017)    *    12    +
output_time[i][j].month - 1 < 36:
            output_data[i][output_time[i][j].year    -    2017]    +=
output_invoice[i][j]

growth_rate = []
for i in range(302):
    if output_data[i][0] != 0:
        growth_rate.append(pow(output_data[i][2] / output_data[i][0],
1 / 3) - 1)
    else:
```

```
        growth_rate.append(pow(output_data[i][2] / output_data[i][1],
1 / 2) - 1)
nb = [0 for _ in range(302)]
for i in range(302):
    if output_fail_count[i] == 0:
        nb[i] = 0
    else:
        nb[i] = fail_money[i] / output_fail_count[i]
for item in growth_rate:
    print(item)
plt.rcParams["font.sans-serif"] = ["SimHei"]
plt.title("302 家企业的销售增长率")
plt.plot(growth_rate)
plt.show()
```

```
#梯度下降法求解利率最优模型
# 第 3 问处理数据
import csv
from datetime import datetime
import numpy as np
import matplotlib.pyplot as plt


def fA(n):
    return -73.748 * n ** 2 + 21.549 * n - 0.681


def fB(n):
    return -65.709 * n ** 2 + 19.843 * n - 0.673


def fC(n):
    return -61.485 * n ** 2 + 19.167 * n - 0.625


def ans(loan, r, cost, ty_co, target):
    if ty_co == "A":
        return fin_profit[i] - output_data[i][1] - output_data[i][2]
- (1 - fA(r)) * (cost - loan * (1 + r)) - fA(r) * (
                    cost - loan)
    elif ty_co == "B":
```

```
        return fin_profit[i] - output_data[i][1] - output_data[i][2]
- (1 - fB(r)) * (cost - loan * (1 + r)) - fB(r) * (
                    cost - loan)
    elif ty_co == "C":
        return fin_profit[i] - output_data[i][1] - output_data[i][2]
- (1 - fC(r)) * (cost - loan * (1 + r)) - fC(r) * (
                    cost - loan)
    else:
        return -1


file_name2 = "302 销项发票.csv"
file_name3 = "302 公司.csv"
file_name4 = "第三问 final.csv"
output_invoice = [[] for _ in range(302)]
output_time = [[] for _ in range(302)]
output_fail_count = [0 for _ in range(302)]
output_final_count = [0 for _ in range(302)]
profit = [0 for _ in range(302)]
cost = [0 for _ in range(302)]
name = []
profit_by_month = [[0 for _ in range(3)] for _ in range(302)]
fail_money = [0 for _ in range(302)]
trade = []
com_type = []
loans = []
input_invoice = [[] for _ in range(302)]
targets = []
colors = ["black" for _ in range(302)]
with open(file_name2, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            output_invoice[int(row[0][1:])                              -
124].append(float(row[6]))
            output_time[int(row[0][1:])                                 -
124].append(datetime.strptime(row[2], "%m/%d/%Y"))
        elif row[7] == "有效发票" and float(row[6]) < 0:
            output_fail_count[int(row[0][1:]) - 124] += 1

        # 等会可能删除
        elif row[7] == "作废发票":
            output_fail_count[int(row[0][1:]) - 124] += 1
```

```
        # 计算每单退货金额先注释掉
        output_final_count[int(row[0][1:]) - 124] += 1

        if float(row[6]) < 0:
            fail_money[int(row[0][1:]) - 124] += abs(float(row[6]))
with open(file_name3, encoding='utf-8') as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        name.append(row[1])
with open("302家企业行业划分.csv", encoding='utf-8') as f:
    reader = csv.reader(f)
    header = next(reader)
    for row in reader:
        trade.append(row[2])

with open(file_name4, encoding='utf-8') as f:
    reader = csv.reader(f)
    header = next(reader)
    i = 0
    for row in reader:
        if row[2] == "0":
            com_type.append("A")
            colors[i] = "red"
        elif row[2] == "1":
            com_type.append("B")
            colors[i] = "green"
        elif row[2] == "2":
            com_type.append("C")
            colors[i] = "blue"
        elif row[2] == "3":
            com_type.append("D")
        i += 1
        loans.append(float(row[1]))
with open("302进项发票.csv", encoding="utf-8") as f:
    reader = csv.reader(f)
    head_row = next(reader)
    for row in reader:
        if row[7] == "有效发票" and float(row[6]) > 0:
            input_invoice[int(row[0][1:]) - 124].append(float(row[6]))
with open("第三问标准化评价指数.csv", encoding='utf-8') as f:
    reader = csv.reader(f)
    header = next(reader)
    for row in reader:
```

```
        targets.append(float(row[1]))

for i in range(302):
    for item in output_invoice[i]:
        profit[i] += item
for i in range(302):
    for item in input_invoice[i]:
        cost[i] += item
# 疫情发生的三个月增长率
three_add = {"农、林、牧、渔业": -0.028, "工业": -0.085,
             "制造业": -0.102, "建筑业": -0.175,
             "批发和零售业": -0.178, "交通运输、仓储和邮政业": -0.14,
             "住宿和餐饮业": -0.353, "金融业": 0.06,
             "房产业": -0.061, "信息传输、软件和信息技术服务业": 0.132,
             "租赁和商务服务业": -0.094, "其他服务业": -0.018}

fin_add = {"农、林、牧、渔业": 0.032, "工业": 0.057,
           "制造业": 0.057, "建筑业": 0.056,
           "批发和零售业": 0.057, "交通运输、仓储和邮政业": 0.071,
           "住宿和餐饮业": 0.063, "金融业": 0.072,
           "房产业": 0.03, "信息传输、软件和信息技术服务业": 0.187,
           "租赁和商务服务业": 0.087, "其他服务业": 0.059}
one_year_decr = {"住宿和餐饮业": -0.18, "租赁和商务服务业": -0.08,
                 "其他服务业": -0.009}
output_fail_rate = []  # 销项退货率&报废发票率
for i in range(len(output_final_count)):
    output_fail_rate.append(output_fail_count[i]                    /
output_final_count[i])

output_data = [[0 for _ in range(3)] for _ in range(302)]
for i in range(302):
    for j in range(len(output_time[i])):
        if    (output_time[i][j].year    -    2017)    *    12    +
output_time[i][j].month - 1 < 36:
            output_data[i][output_time[i][j].year    -    2017]    +=
output_invoice[i][j]

growth_rate = []
for i in range(302):
    if output_data[i][0] != 0:
        growth_rate.append(pow(output_data[i][2] / output_data[i][0],
1 / 3) - 1)
    else:
        growth_rate.append(pow(output_data[i][2] / output_data[i][1],
```

```
1 / 2) - 1)

# 平均每单退款金额
nb = [0 for _ in range(302)]
for i in range(302):
    if output_fail_count[i] == 0:
        nb[i] = 0
    else:
        nb[i] = fail_money[i] / output_fail_count[i]

fin_profit = [0 for _ in range(302)]
sales_revenue_growth_rate = [0 for _ in range(302)]
for i in range(302):

    if trade[i] in one_year_decr.keys():
        fin_profit[i] = output_data[i][1] + output_data[i][2] +
output_data[i][2] * (1 + one_year_decr[trade[i]])
    else:
        fin_profit[i] = output_data[i][1] + output_data[i][2] +
output_data[i][2] * (
                1 + 9 * fin_add[trade[i]] / 12 + 3 * three_add[trade[i]]
/ 12)
    if output_data[i][1] == 0:
        if trade[i] in one_year_decr.keys():
            sales_revenue_growth_rate[i] = pow(output_data[i][2] * (1
+ one_year_decr[trade[i]]) / output_data[i][2],
                                               1 / 2) - 1
        else:
            sales_revenue_growth_rate[i] = pow(
                output_data[i][2] * (1 + 9 * fin_add[trade[i]] / 12 +
3 * three_add[trade[i]] / 12) / output_data[i][2],
                1 / 2) - 1
    else:
        if trade[i] in one_year_decr.keys():
            sales_revenue_growth_rate[i] = pow(output_data[i][2] * (1
+ one_year_decr[trade[i]]) / output_data[i][1],
                                               1 / 3) - 1
        else:
            sales_revenue_growth_rate[i] = pow(
                output_data[i][2] * (1 + 9 * fin_add[trade[i]] / 12 +
3 * three_add[trade[i]] / 12) / output_data[i][1],
                1 / 3) - 1

rate = []
```

```
x = np.linspace(0.04, 0.15, 10000)

for i in range(302):
    max_por = 0
    r = 0.04
    for j in range(10000):
        temp = ans(loans[i], x[j], cost[i] / 3, com_type[i], targets[i])

        if max_por < temp:
            max_por = temp
            r = x[j]
    rate.append(r)
# loans 贷款金额 rate  利率
nt = []
for item in rate:
    nt.append(item * 2000)
number = list(range(124, 426))
plt.rcParams['font.sans-serif'] = "SimHei"
plt.scatter(number, loans, s=nt, c=colors,alpha=0.6)
plt.text(370, 1000000, "红色代表 A 类", c="red")
plt.text(370, 900000, "绿色代表 B 类", c="green")
plt.text(370, 800000, "蓝色代表 C 类", c="blue")
plt.text(370, 700000, "黑色代表 D 类", c="black")
plt.xlim([124, 426])

plt.xlabel("企业序号")
plt.ylabel("贷款额度")
plt.title("基于 E124~E405 企业的贷款额度和利率的气泡图")

plt.show()
for item in rate:
    print(item)
```

****
matlab 代码如下
****

```
%%% 函数部分

%%%cal_index.m
function [index] = cal_index(norm_data)
% 计算评价指数
len = size(norm_data);
index = zeros(len(1),1);
```

```
for i=1:len(1)
    index(i) = ...
        23.48*norm_data(i,1) - 22.63*norm_data(i,2)...
        + 28.02*norm_data(i,3) + 25.87*norm_data(i,4);
end
end

%%%div_class.m
function [A,B,C,D] = div_class(input)
% 将 ABCD 类分开（0，1，2，3 类）
% 第一列和最后两列存放标志和序号
A = [];
B = [];
C = [];
D = [];
for i=1:size(input,1)
    class = input(i,1);
    row = input(i,:);
    if class == 0
        A = [A;row];
    elseif class == 1
        B = [B;row];
    elseif class == 2
        C = [C;row];
    else
        D = [D;row];
    end
end

%%%max_min.m
function [norm_data] = max_min(init_data)
% 最大最小标准化函数
len = size(init_data);
norm_data = zeros(len(1),len(2));
for i=1:len(2)
    max_num = max(init_data(:,i));
    min_num = min(init_data(:,i));
    for j=1:len(1)
        norm_data(j,i) = ...
            (init_data(j,i) - min_num)/(max_num - min_num);
    end
end
end
```

```
%%%in_3sigma.m
function [output1,output2] = in_3sigma(input)
% 去除 3sigma 以外的数据,输出分开的数据
% 第一列和最后一列是类别标志，不动
output1 = [];
output2 = [];
% 计算每列 std
std_list = [];
for i=2:size(input,2)-2
    std_list = [std_list std(input(:,i))];
end
% 逐个比较是否去除该行
for i=1:size(input,1)
    is_out = 0;
    for j=2:size(input,2)-2
        if input(i,j) > 3*std_list(j-1)
            is_out = 1;
        end
    end
    if is_out == 0
        output1 = [output1;input(i,:)];
    else
        output2 = [output2;input(i,:)];
    end
end


%%%in_3sigma_2.m
function [output1,output2] = in_3sigma_2(input)
% 问题 2 的 去除 3sigma 以外的数据,输出分开的数据
% 第一行为序号不动
output1 = [];
output2 = [];
% 计算每列 std
std_list = [];
for i=2:size(input,2)
    std_list = [std_list std(input(:,i))];
end
% 逐个比较是否去除该行
for i=1:size(input,1)
    is_out = 0;
    for j=2:size(input,2)
        if input(i,j) > 3*std_list(j-1)
```

```
            is_out = 1;
        end
    end
    if is_out == 0
        output1 = [output1;input(i,:)];
    else
        output2 = [output2;input(i,:)];
    end
end

%%%data_process_11.m
% 计算 123 企业竞争力评价指数
% 数据最大最小标准化
clc;clear;
load processed_data;
% 提取中间数据项
adata = fin_data(:,2:size(fin_data,2)-2);
% 两类总体标准化
norm_data = [fin_data(:,1),max_min(adata),...
    fin_data(:,size(fin_data,2)-1:size(fin_data,2))];
out_norm_data = [out_data(:,1),max_min(out_data(:,2:size(out_data,2)-
2)),...
    out_data(:,size(out_data,2)-1:size(out_data,2))];
save('norm_data','norm_data','out_norm_data');

% 计算评价指数
index_norm = cal_index(norm_data(:,2:size(norm_data,2)-2));
index_norm = [fin_data(:,1),index_norm,...
    fin_data(:,size(fin_data,2)-1:size(fin_data,2))];
index_out_norm  =  cal_index(out_norm_data(:,2:size(out_norm_data,2)-
2));
index_out_norm = [out_data(:,1),index_out_norm,...
    out_data(:,size(out_data,2)-1:size(out_data,2))];
save ('index','index_norm','index_out_norm');

% 合并所有数据
all_data = [fin_data;out_data];
index = [index_norm;index_out_norm];
all_norm_data = [norm_data;out_norm_data];
all_norm_data = sortrows(all_norm_data,size(all_norm_data,2));
save('all_data','all_data','index','all_norm_data');


%%%data_process_12.m
```

```
load all_data
index = sortrows(index,4);
plot(index(:,4),index(:,2),'ko');
hold on;
% 基于 excel 绘制的大致趋势图
x = 1:size(index,1);
y = -0.24994*x + 29.068;
plot(x,y,'r');
title('123 家企业信誉评价指数');
xlabel('企业序号（1 为 E1）');
ylabel('企业信誉评价指数值');


%%%data_process_13.m
% 计算每个企业比率
clc,clear;
load all_data
all = [index(:,3),index(:,1),index(:,2),index(:,4)];
effect = [];
% 筛选出可以放贷且信誉评级不为 D 的数据
for i=1:size(all,1)
    if all(i,1) == 0 && all(i,2) ~= 3
        effect = [effect;[all(i,4) index(i,2)]];
    end
end
% 标准化指标
effect(:,2) = max_min(effect(:,2));
% 计算每个公司比率
prop = zeros(size(effect,1),size(effect,2));
sum_ABC = sum(effect(:,2));
for i=1:size(effect,1)
    prop(i,1) = effect(i,1);
    prop(i,2) = effect(i,2) / sum_ABC;
end
plot(prop(:,1),prop(:,2),'ro');
title("96 个企业放贷情况");
xlabel("企业对应序号");
ylabel("放贷比率");
save('prop','prop')

%%%pre_dataprocess.m
% 是否放贷类别转换
load is_lend
a = find(is_lend == '否');
```

```
b = find(is_lend == '是');
Category = zeros(size(is_lend,1),1);
Category(a) = 0;
Category(b) = 1;
save('Category','Category')

% 将评级数据转化成数字类别
load data
a = find(data == 'A');
b = find(data == 'B');
c = find(data == 'C');
d = find(data == 'D');

class = zeros(size(data,1),1);
class(a) = 0;
class(b) = 1;
class(c) = 2;
class(d) = 3;
save('class','class');

% 去除偏离总体值
% 导入初始指标数据
% 指标顺序为销售收入、销项交易失败率
% 销售收入增长率、平均退款金额
load ind_data;
load Category;
% ABCD 分类数据
load class;
% 合并信誉评级以及是否违约标志
wait_data = [class ind_data Category [1:size(class,1)]'];
[A,B,C,D] = div_class(wait_data);

% 处理后数据
[A,out_A] = in_3sigma(A);
[B,out_B] = in_3sigma(B);
[C,out_C] = in_3sigma(C);
[D,out_D] = in_3sigma(D);
% 组合
fin_data = [A;B;C;D];
out_data = [out_A;out_B;out_C;out_D];
save('processed_data','fin_data','out_data');


%%%data_process_21.m
```

```
clc;clear;
load data1
temp = 124:123+size(data1,1);
temp = temp';
data1 = [temp,data1];

% 3sigma 去除偏离总体的样本并加上企业序号
[ava_data,una_data] = in_3sigma_2(data1);
stand_ava = [ava_data(:,1),max_min(ava_data(:,2:size(ava_data,2)))];
stand_una = [una_data(:,1),max_min(una_data(:,2:size(una_data,2)))];
comb_stand = [stand_ava;stand_una];
comb_stand = sortrows(comb_stand,1);
save('stand_data','stand_ava','stand_una','ava_data','una_data');

% 根据 BP 反向传播网络的分割值确定是否放贷
% 0 代表没有违约即可以放贷 1 代表违约不能放贷
% 被归为 1 类默认信誉为 D
load division
is_lend = zeros(size(division,1),1);
for i=1:size(division,1)
    if division(i) >= 0.22
        is_lend(i) = 1;
    end
end
% 末尾放是否违约标志 0 表示不会违约 1 表示违约
comb_data = [comb_stand,is_lend];
save('comb_data','comb_data');

% 先剔除不能放贷的企业
load comb_data
is_loan = [];
no_loan = [];
for i=1:size(comb_data,1)
    flag = comb_data(i,size(comb_data,2)) ;
    if flag == 0
        is_loan = [is_loan;comb_data(i,1:size(comb_data,2)-1)];
    else
        no_loan = [no_loan;comb_data(i,1:size(comb_data,2)-1)];
    end
end

% 代入可贷款企业的数据，使用决策树推测出每个企业的类别
load predict_class
is_loan = [is_loan,predict_class];
```

```
lin = linspace(3,3,size(no_loan,1));
no_loan = [no_loan,lin'];
save('loan','is_loan','no_loan');


%%%data_process_22.m
% 计算可放贷企业评价指标
clc;clear;
load loan
index = [is_loan(:,1),...
    cal_index(is_loan(:,2:size(is_loan,2)-1)),...
    is_loan(:,size(is_loan,2))];

% 标准化评价指标
stand_index = [index(:,1),max_min(index(:,2)),index(:,3)];
save('index','index','stand_index');

% 考虑所有贷款额度>100 万并且<10 万的企业并规定放贷为 100 万和 10 万
load index
% 求各企业贷款比例
all_prop = [];
sum_all = sum(stand_index(:,2));
for i=1:size(stand_index,1)
    all_prop = [all_prop;[stand_index(i,1),...
        stand_index(i,2)/sum_all],stand_index(i,3)];
end
save('all_prop','all_prop');

% 划分贷款
load all_prop
max_loan = [];
min_loan = [];
norm_loan = [];
total = 1E8;
for i=1:size(all_prop,1)
    loan = all_prop(i,2) * total;
    if loan > 1E6
        max_loan = [max_loan;[all_prop(i,1),1E6,all_prop(i,3)]];
    elseif loan < 1E5
        min_loan = [min_loan;[all_prop(i,1),1E5,all_prop(i,3)]];
    else
        norm_loan = [norm_loan;stand_index(i,:)];
    end
end
```

```
rest = total - size(max_loan,1)*1E6 - size(min_loan,1)*1e5;
sum_prop = sum(norm_loan(:,2));
new_loan = [];
for i=1:size(norm_loan,1)
    new_loan = [new_loan;...
        [norm_loan(i,1),...
        rest * norm_loan(i,2)/sum_prop,...
        norm_loan(i,3)]];
end
% 合并全部贷款情况
all_loan = [max_loan;min_loan;new_loan];
save('loan_situation','max_loan','min_loan','new_loan','all_loan');

% 企业贷款分布（无数据表示不贷款）
plot(all_loan(:,1),all_loan(:,2),'ro');
xlim([124 124+301]);
title('普通企业贷款额度分布图');
xlabel('企业序号');
ylabel('贷款额度');

% 计算潜在违约企业的信誉评价指数
load loan
no_loan_index = [no_loan(:,1),...
    cal_index(no_loan(:,2:size(no_loan,2)-1)),...
    no_loan(:,size(no_loan,2))];

% 标准化信誉评价指标
stand_no_index = [no_loan_index(:,1),...
    max_min(no_loan_index(:,2)),...
    no_loan_index(:,3)];
all_index = [stand_index;stand_no_index];
all_index = sortrows(all_index,1);
save('all_index','all_index');

%%%data_process_31.m
clc;clear;
load data1
temp = 124:123+size(data1,1);
temp = temp';
data1 = [temp,data1];

% 3sigma 去除偏离总体的样本并加上企业序号
[ava_data,una_data] = in_3sigma_2(data1);
stand_ava = [ava_data(:,1),max_min(ava_data(:,2:size(ava_data,2)))];
```

```
stand_una = [una_data(:,1),max_min(una_data(:,2:size(una_data,2)))];
comb_stand = [stand_ava;stand_una];
comb_stand = sortrows(comb_stand,1);
save('stand_data','stand_ava','stand_una','ava_data','una_data');

% 根据 BP 反向传播网络的分割值确定是否放贷
% 0 代表没有违约即可以放贷 1 代表违约不能放贷
% 被归为 1 类默认信誉为 D
load division
is_lend = zeros(size(division,1),1);
for i=1:size(division,1)
    if division(i) >= 0.22
        is_lend(i) = 1;
    end
end
% 末尾放是否违约标志 0 表示不会违约 1 表示违约
comb_data = [comb_stand,is_lend];
save('comb_data','comb_data');

% 先剔除不能放贷的企业
load comb_data
is_loan = [];
no_loan = [];
for i=1:size(comb_data,1)
    flag = comb_data(i,size(comb_data,2)) ;
    if flag == 0
        is_loan = [is_loan;comb_data(i,1:size(comb_data,2)-1)];
    else
        no_loan = [no_loan;comb_data(i,1:size(comb_data,2)-1)];
    end
end

% 代入可贷款企业的数据，使用决策树推测出每个企业的类别
load predict_class
is_loan = [is_loan,predict_class];
lin = linspace(3,3,size(no_loan,1));
no_loan = [no_loan,lin'];
save('loan','is_loan','no_loan');

%%%data_process_32.m
% 计算可放贷企业评价指标
clc;clear;
load loan
index = [is_loan(:,1),...
```

```
    cal_index(is_loan(:,2:size(is_loan,2)-1)),...
    is_loan(:,size(is_loan,2))];

% 标准化评价指标
stand_index = [index(:,1),max_min(index(:,2)),index(:,3)];
save('index','index','stand_index');

% 考虑所有贷款额度>100 万并且<10 万的企业并规定放贷为 100 万和 10 万
load index
% 求各企业贷款比例
all_prop = [];
sum_all = sum(stand_index(:,2));
for i=1:size(stand_index,1)
    all_prop = [all_prop;[stand_index(i,1),...
        stand_index(i,2)/sum_all],stand_index(i,3)];
end
save('all_prop','all_prop');

% 划分贷款
load all_prop
max_loan = [];
min_loan = [];
norm_loan = [];
total = 1E8;
for i=1:size(all_prop,1)
    loan = all_prop(i,2) * total;
    if loan > 1E6
        max_loan = [max_loan;[all_prop(i,1),1E6,all_prop(i,3)]];
    elseif loan < 1E5
        min_loan = [min_loan;[all_prop(i,1),1E5,all_prop(i,3)]];
    else
        norm_loan = [norm_loan;stand_index(i,:)];
    end
end
rest = total - size(max_loan,1)*1E6 - size(min_loan,1)*1e5;
sum_prop = sum(norm_loan(:,2));
new_loan = [];
for i=1:size(norm_loan,1)
    new_loan = [new_loan;...
        [norm_loan(i,1),...
        rest * norm_loan(i,2)/sum_prop,...
        norm_loan(i,3)]];
end
% 合并全部贷款情况
```

```
all_loan = [max_loan;min_loan;new_loan];
save('loan_situation','max_loan','min_loan','new_loan','all_loan');

% 企业贷款分布（无数据表示不贷款）
plot(all_loan(:,1),all_loan(:,2),'ro');
xlim([124 124+301]);
title('普通企业贷款额度分布图');
xlabel('企业序号');
ylabel('贷款额度');

% 计算潜在违约企业的信誉评价指数
load loan
no_loan_index = [no_loan(:,1),...
    cal_index(no_loan(:,2:size(no_loan,2)-1)),...
    no_loan(:,size(no_loan,2))];

% 标准化信誉评价指标
stand_no_index = [no_loan_index(:,1),...
    max_min(no_loan_index(:,2)),...
    no_loan_index(:,3)];
all_index = [stand_index;stand_no_index];
all_index = sortrows(all_index,1);
save('all_index','all_index');

%%%plots.m
% 第二问和第三问贷款额度差距
load dis
dis = [dis(:,2),dis(:,3),dis(:,1)];
[A,B,C] = div_class(dis);

plot(A(:,3),A(:,2),'ro')
hold on;
plot(B(:,3),B(:,2),'ko')
plot(C(:,3),C(:,2),'bo')
xlim([min(dis(:,3)) max(dis(:,3))]);
xlabel('企业序号');
ylabel('企业贷款额度差');
title('疫情期间相比正常时期贷款增加额');
plot([min(dis(:,2)) max(dis(:,2))],[0 0],'k--','LineWidth',1.8);
legend('A类','B类','C类','分界线');
```

****

***
R 语言 代码如下
****
***
第一问

```
# GB2312 编码
setwd("D:/R_WorkingPlace")
# BP 反向传播网络
detach("package:ROCR")
library("neuralnet")
set.seed(12345)
data1 <- read.table(file="train_data.txt",header=TRUE)
BPnet<-
neuralnet(CreditRating~.,data=data1,hidden=3,err.fct="ce",linear.outp
ut = FALSE,stepmax = 10000000)
plot(BPnet)


# ROC 曲线
detach("package:neuralnet")
library("ROCR")
summary(BPnet$net.result[[1]])
pred              <-              prediction(predictions              =
as.vector(BPnet$net.result),labels=BPnet$response)
par(mfrow=c(2,1))
options(digits=4)
perf <- performance(pred,measure="tpr",x.measure = "fpr")
plot(perf,colorize=TRUE,print.cutoffs.at=c(0.01,0.22,0.24,0.17,0.18))
perf <- performance(pred,measure="acc",)
plot(perf)
out <- cbind(BPnet$response,BPnet$net.result[[1]])
out <- cbind(out,ifelse(out[,2]>0.22,1,0))
(ConfM.BP<-table(out[,1],out[,3]))
(Err.BP<-(sum(ConfM.BP)-sum(diag(ConfM.BP)))/sum(ConfM.BP))

# 测试样本检验
# 使用 BP 模型预测 23 个值
detach("package:ROCR")
library("neuralnet")
sample = read.table(file="test_data.txt",header = TRUE)
new.output <- compute(BPnet,covariate = sample)
write.table (new.output$net.result[[1]], file ="输出预测分割值.txt",
             sep ="", row.names =FALSE, col.names =TRUE, quote =TRUE)
```

```
out <- cbind(sample[,5],new.output$net.result[[1]])
out <- cbind(out,ifelse(out[,2]>0.22,1,0))
(ConfM1.BP<-table(out[,1],out[,3]))
(Err1.BP<-(sum(ConfM1.BP)-sum(diag(ConfM1.BP)))/sum(ConfM1.BP))


# 决策树
library("adabag")
set.seed(12345)
train_data <- read.table(file="train_data2.txt",header = TRUE)
train_data$CreditRating <- as.factor(train_data$CreditRating)
ctl      <-      rpart.control(minsplit=8,maxcompete=4,maxdepth      =
8,cp=0.01,xval=10)
BagM <- bagging(CreditRating~.,data=train_data,control=ctl,mfinal=25)
BagM$importance
CFit <- predict.bagging(BagM,train_data)
CFit$confusion
CFit$error


## 第二问
# GB2312 编码
setwd("D:/R_WorkingPlace")
# 使用 BP 模型预测 302 个值
detach("package:ROCR")
library("neuralnet")
set.seed(12345)
sample = read.table(file="test_data1.txt",header = TRUE)
new.output <- compute(BPnet,covariate = sample)
write.table (new.output$net.result, file ="输出预测分割值.txt",
             sep ="", row.names =FALSE, col.names =FALSE, quote =TRUE)

# 决策树预测可贷款企业的信誉评级
set.seed(12345)
test_data2 <- read.table(file="test_data2.txt",header=TRUE)
CFit2 <- predict.bagging(BagM,test_data2)
write.table (as.numeric(CFit2$class), file ="输出预测类别值.txt",
             sep ="", row.names =FALSE, col.names =FALSE, quote =TRUE)


# 绘图
plot_data2 <- read.table(file="plot.txt",header=FALSE)
# 数据分类
data_A2 = subset(plot_data2,V3==0)
```

```
data_B2 = subset(plot_data2,V3==1)
data_C2 = subset(plot_data2,V3==2)

# 气泡图
symbols(data_A2$V1,data_A2$V2,circle=data_A2$V4,inches=0.1,
        xlab="企业序号",ylab="企业贷款额度",main="企业贷款额度和贷款
利率气泡图",
        fg="white",bg="lightblue",xlim =c(124,124+301))
par(new=TRUE)
symbols(data_B2$V1,data_B2$V2,circle=data_B2$V4,inches=0.1,
        xlab="",ylab="",fg="white",bg="burlywood2",axes=FALSE)
par(new=TRUE)
symbols(data_C2$V1,data_C2$V2,circle=data_C2$V4,inches=0.1,
        xlab="",ylab="",fg="white",bg="grey",axes=FALSE)


## 第三问
# GB2312 编码
setwd("D:/R_WorkingPlace")
# 使用 BP 模型预测 302 个值
detach("package:ROCR")
library("neuralnet")
set.seed(12345)
sample = read.table(file="test_data1.txt",header = TRUE)
new.output <- compute(BPnet,covariate = sample)
write.table (new.output$net.result, file ="输出预测分割值.txt",
             sep ="", row.names =FALSE, col.names =FALSE, quote =TRUE)

# 决策树预测可贷款企业的信誉评级
set.seed(12345)
test_data2 <- read.table(file="test_data2.txt",header=TRUE)
CFit2 <- predict.bagging(BagM,test_data2)
write.table (as.numeric(CFit2$class), file ="输出预测类别值.txt",
             sep ="", row.names =FALSE, col.names =FALSE, quote =TRUE)

# 绘图
plot_data <- read.table(file="plot2.txt",header=FALSE)
# 数据分类
data_A = subset(plot_data,V4==0)
data_B = subset(plot_data,V4==1)
data_C = subset(plot_data,V4==2)

# 气泡图
symbols(data_A$V1,data_A$V3,circle=data_A$V2,inches=0.1,
```

```
        xlab="企业序号",ylab="企业贷款额度",main="企业贷款额度和贷款利率气泡图",
        fg="white",bg="lightblue",xlim =c(124,124+301))
par(new=TRUE)
symbols(data_B$V1,data_B$V3,circle=data_B$V2,inches=0.1,
        xlab="",ylab="",fg="white",bg="burlywood2",axes=FALSE)
par(new=TRUE)
symbols(data_C$V1,data_C$V3,circle=data_C$V2,inches=0.1,
        xlab="",ylab="",fg="white",bg="grey",axes=FALSE)
```