基于熵权法的中小微企业银行信贷模型

摘要

基于中小企业规模小，资金不足的特点，银行依据中小企业的经营状况，给出贷款策略。本文首先利用多元线性回归的方法，以利润率等五个指标作为自变量，建立了还款能力评估模型，随后利用熵权法建立贷款数量界定模型，最后使用多目标规划法，平衡收益和用户流失，确定了企业贷款利率。除此之外，使用 K 近邻算法对缺失信誉等级进行补充，并针对新冠疫情带来的突发影响，引入系数进行调整。

针对问题一，为了解决银行的放贷问题，我们将问题化解为向谁贷款，贷款数目，贷款利率三个子问题。基于多变量线性回归建立企业还款能力评估模型，依据熵权法建立贷款数量分配模型，根据多目标规划建立了最优的利率设置，分别解决了提出的三个问题。使用的方法基于利润率，有效交易占比，供应链丰富度，信誉等级，平均单价，资金缺口六个要素，丰富全面挖掘了附件的信息。

针对问题二，我们在信誉等级缺失的情况下，利用 K 近邻算法预测附件二中的信誉等级。随后使用问题一所建立的银行借贷模型进行贷款分配。

针对问题三，面对新冠疫情等可能的突发因素影响，我们对银行信贷模型做出调整。首先搜寻相关资料，通过企业名称来推断企业所处类别，随后整理不同行业下受新冠影响的程度。最后对模型中利润率和资金缺口两项进行调整，做到了针对不同行业的企业都有对应策略。

该模型建立在较多的资料收集和数据分析基础上，客观可靠，具有较好的推广性。

关键词: 多元线性回归　熵权法　多目标规划　K 近邻算法

一、问题重述

1.1 问题背景

为了给予中小企业现金支持，银行针对中小企业的特点，推出一系列不同的信用贷款。对于实力较强信用较好的企业，银行倾向于提供更加优惠的利率。我们需要针对小微企业的开票情况，建立数学模型来给出银行的信贷策略。

1.2 问题重述

经过分析整理，我们需要解决以下问题：

1. 对附件 1 给出有信贷记录的 123 家企业的信贷风险进行量化分 ǹ 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 中没有信贷记录的 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（例如：新冠病毒疫情）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。

二、问题分析

2.1 问题一的分析

为了解决问题 1，需要利用附件中的小微企业发票数据和信誉评级来构建数学模型，以给出银行在年信贷额固定时的信贷策略。因此，我们利用企业发票数据估计营业额，资金缺口和稳定情况，且结合信用评级来给出信贷策略。因此，我们需要构建模型以解决贷款给谁、贷款多少以及利率多少的问题。

2.2 问题二的分析

为了解决问题 2，需要对问题一中提出的银行信贷模型进行补充。由于不知道企业的信誉评级，我们可以利用附件一中已知的数据，构建 KNN 等机器学习算法，对缺失的用户标签进行预测。随后利用银行信贷模型组织信贷分配。

2.3 问题三的分析

为了能够定量地呈现出突发情况下银行地信贷调整策略，我们先按照企业的名称将企业进行行业分类。然后以新冠疫情为例，搜集疫情对各行业地影响程度和方面，然后将这些影响带入到响应自变量中，重新组织和实现之前的信贷策略模型。

三、模型假设

1. 假设企业发票明细完整无误，没有瞒报漏报。

   原因：依据企业的发票开具情况，可以直观显示出一个企业的收入支出。因此通过完整的发票记录，可以得到企业的经营特点。

2. 假设企业都在同一时间办理信贷业务。

   原因：为了简化模型，做出此假设后可以灵活分配银行的贷款资金。且在实际生活中，银行在工作日都有资金流动，可以将一年期的信贷行为统一处理。

3. 各企业经济特征不会有较大变化。

   原因：我们的模型是建立在已有数据基础之上的，有一定程度的“预测性”，若想要获得较好的“预测结果”，我们需要数据具有一定的“保持性”。

4. 企业名称符合一般的起名习惯。

   原因：我们在对企业做行业划分时，是根据企业名称进行的，所以需要企业在名称中体现企业的行业属性。

5. 企业仅从事一方面经营。

   原因：题目中的企业都为中小企业，很难具备多项业务同时处理的能力，同时不考虑复合型企业，也可以简化模型，提高模型的效率。

四、符号说明

4.1 符号说明

以下是本文使用的符号以及含义：

符号	说明	单位
x_1	利润率	元
x_2	有效交易占比	/
x_3	供应链丰富度	/
x_4	信誉等级	/
x_5	平均单价	元
x_6	资金缺口	元
T, t	发票集合，单张发票金额	/
P	企业还款概率	/
C	贷款额度	元
I	贷款利率	元/年
R	信贷收益	元
L	流失率	/

五、模型的建立与求解

以下将对提出的三个问题进行建模求解。

5.1 基于熵权法的银行信贷模型

银行信贷的过程主要是解决三个问题：贷给谁，贷多少钱以及贷款利率多少。针对借贷对象而言，需要评估其还款能力来做出贷款选择，我们将在 5.1.1 一节中详细说明一种基于多元线性回归的方法来界定。针对贷款数量而言，我们利用利润率，有效交易占比，供应链丰富度，信誉等级，平均单价，资金缺口六个要素，使用熵权法确定了每个企业的贷款数目，具体内容在 5.1.2 节中体现。针对贷款利率而言，我们在 5.1.3 节中提出一种综合贷款利率和用户流失的多目标规划模型。一批用户经由还款能力评估，贷款数量界定，贷款利率确定后，便可以完成借贷工作。

5.1.1 企业还款能力评估

银行放出贷款后是否能够获得收益，取决于贷款是否能够连本带息如数收回。银行信贷盈利的前提是对企业的还款能力进行有效评估，只有向还款能力较好的企业投放信贷，才会有较低的风险，保证收益来源。

为了衡量企业的还款能力，我们提出以下指标：

1. 利润率

   利润率[1] 在经济学中被解释为总所得和总成本的差额同总成本之间的比值。在题目所给的条件中，结合假设条件，我们认为进项发票的票值代表购买产品的成本，而销项发票的票值代表卖出商品的销售额。利用这一指标，可以判断企业的经营情况，根据统计局的相关数据[?] 显示，2019 年中小企业营业收入利润率为 5.6%，对于高于这一水平的企业，可视为其有较高的利润水平。我们给出利润率 x_1 的计算公式

$$x_1 = \frac{\sum\limits_{t \in T_{\text{销项}}} t - \sum\limits_{t \in T_{\text{进项}}} t}{\sum\limits_{t \in T_{\text{进项}}} t} \tag{1}$$

其中 T 代表发票集合，$T_{\text{进项}}, T_{\text{销项}}$ 分别代表进项和销项发票，t 代表某一发票的数额，下同。

2. 有效交易占比

在购买商品时，如果出现质量问题，双方无法协商一致时可以选择退货，这是消费者的权益之一。一家店铺的退货数量较多，可以从侧面反映出其存在问题，无论是商品质量，还是服务是否周全，都可在这一指标中体现，所以我们认为开票金额为正的有效发票是有效交易，计算有效交易的占比以判断经营状况的好坏，给出下面的计算公式：

$$x_2 = \frac{card(T_{\text{有效},t>0})}{card(T)} \tag{2}$$

其中分子分母中 T 都来自于同一公司的发票记录。

3. 供应链丰富度

一个企业不能脱离于其他的企业而孤立存在，每个企业都或多或少向其他企业购买产品或者服务，并向其他企业出售。当企业具有丰富的上下游关系时，其抗风险的能力较高，同样反映出企业的组织管理水平较好。供应链丰富度这个指标，我们定义为与企业发生资金往来的企业数目，可以使用单位代号进行标识统计。给出供应链丰富度 x_3 的计算方法：

$$x_3 = card(c_{\text{购方}}) + card(c_{\text{销方}}) \tag{3}$$

其中 c 代表企业的集合。

4. 信誉等级

信用评级[2]（信誉等级）的目的是显示受评对象信贷违约风险的大小，一般由某些专门信用评估机构进行。对于已经有信贷记录的企业而言，银行已经具有信用评级，可以作为参考，由于使用 A、B、C、D 四个字母代表不同的信誉评级，为此将其量化为：

$$x_4 = \begin{cases} 1, A \\ 0.75, B \\ 0.5, C \\ \text{不予放贷}, D \end{cases} \tag{4}$$

其中信用等级为 D 的不予放贷，仅为了完整性罗列于此。

5. 平均单价

平均单价反映了企业流水的规模，银行更偏向于向流水规模更高的公司提供信贷。在所给条件下，我们可以使用进销项的平均金额来计算其平均单价。给出计算式：

$$x_5 = \frac{\sum\limits_{t \in T_{\text{销项}}} t + \sum\limits_{t \in T_{\text{进项}}} t}{card(T_{\text{进项}}) + card(T_{\text{销项}})} \tag{5}$$

企业的还款能力需要综合上述的五个因素来看，因此需要为五个指标赋予权重。为此，我们借鉴 Chesser 模型[3] 的思想，提出了求解权重的方法。首先基于五个自变量列出多元线性回归[4] 判别法的一般公式：

$$Y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 \tag{6}$$

其中 α 代表常系数，β 代表比例系数，Y 代表最终输出结果，即企业是否还款。各项 x 的数值已在前文说明，总结如下：

$$\begin{cases} x_1 = \dfrac{\sum\limits_{t \in T_{\text{销项}}} t - \sum\limits_{t \in T_{\text{进项}}} t}{\sum\limits_{t \in T_{\text{进项}}} t} \\ x_2 = \dfrac{card(T_{\text{有效},t>0})}{card(T)} \\ x_3 = card(c_{\text{购方}}) + card(c_{\text{销方}}) \\ x_4 = \begin{cases} 1, A \\ 0.75, B \\ 0.5, C \\ \text{不予放贷}, D \end{cases} \\ x_5 = \dfrac{\sum\limits_{t \in T_{\text{销项}}} t + \sum\limits_{t \in T_{\text{进项}}} t}{card(T_{\text{进项}}) + card(T_{\text{销项}})} \end{cases}$$

我们输出的结果应当是企业是否还贷款，只有“是”和“否”两个选项。是一个典型的二分类问题，认为 0 为不还贷款，1 为还贷。这样待求出的系数可以使用多元线性回归的工具求出。

在得到多元线性回归方程（6）后，为了进行预测还贷情况，需要将 Y 映射到 $[0, 1]$ 区间内，以符合数理统计的规律。所以在多元线性回归的基础上引入 Logit 变换，其步骤如下：

1. 首先引入比例数 (Odds) 的概念，企业可以还款的概率为 P，则其比例数定义为下式：

$$Odds = \frac{P}{1 - P} \tag{7}$$

2. 对其取对数，得到 θ。

$$\theta = \ln Odds = \ln \frac{P}{1 - P} \tag{8}$$

3. 对式（8）变形后，得到概率 P 与 θ 之间的关系：

$$P = \frac{1}{1 - e^{-\theta}} \tag{9}$$

Logit 模型事实上就是将线性回归的输出 Y 视作与 θ 等同，这样最终输出结果在 0-1 之间，符合要求。因此估计一家企业的还款的概率 P 由下式计算：

$$\begin{aligned} P &= \frac{1}{1 - e^{-Y}} \\ Y &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 \end{aligned} \tag{10}$$

综上所述，在针对一名用户判断是否放贷时，首先判断其信用评级是否为 D 若是，则不予放贷，若不是，则利用式（10）来求解，得到概率 P 若大于 0.5 则认为可以贷款，若反之，则不予放贷。流程如下图所示：

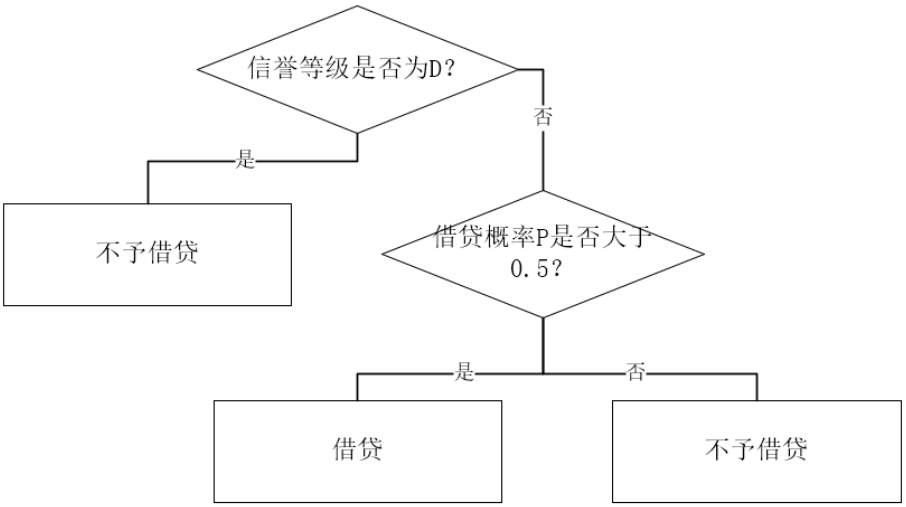


图 1: 银行信贷判断图

5.1.2 贷款数量界定

中小企业由于其经营策略的不同，需要周转的资金数量也有差异。界定中小企业的贷款需求，精准投放有限的资金储备，是银行需要解决的问题。在完成发放贷款与否的界定后，我们将使用熵权法来给出企业贷款数额的界定。

为了更好的判断借款数额，不仅需要利用式（1-5）所定义的数值，还需要引入企业的资金缺口 x_6 作为附加的变量。企业在经营的过程中，遵循着先投入后产出的规律，只有首先购进材料产品等，随后售卖盈利，在这一过程中，存在一段时间企业投入资金大于收入所得。在这段时间内产生的支出与获利之差我们定义为资金缺口。当企业获得贷款能够补齐资金缺口时，便可以顺利保证经营的正常进行。我们记录资金缺口为:

$$x_6 = \sum_{t \in T_{\text{缺口进项}}} t - \sum_{t \in T_{\text{缺口销项}}} t \tag{11}$$

对于许多小企业而言，需要使用同一套标准来计算其贷款额度。计算的依据是 $x_1, \cdots, x_6$ 这六个指标，分别代表利润率，有效交易占比，供应链丰富度，信誉等级，平均单价，资金缺口六个要素。这六个要素量纲不同，数值不同，需要利用熵权法来给定各个因素在贷款分配策略中所占有的权重。其具体步骤如下:

1. 利用附件中的数据，计算 i 用户（$i = 1, 2, 3, \cdots, m$）的 j 项指标 x_{ij}（$j = 1, 2, 3, \cdots, 6$）数

值，构成用户指标矩阵 $X_{m\times n}$，每项指标的计算方法如下：

$$\begin{cases} x_1 = \dfrac{\sum\limits_{t\in T_{销项}} t - \sum\limits_{t\in T_{进项}} t}{\sum\limits_{t\in T_{进项}} t} \\ x_2 = \dfrac{card(T_{有效,t>0})}{card(T)} \\ x_3 = card(c_{购方}) + card(c_{销方}) \\ x_4 = \begin{cases} 1, A \\ 0.75, B \\ 0.5, C \\ 不予放贷, D \end{cases} \\ x_5 = \dfrac{\sum\limits_{t\in T_{销项}} t + \sum\limits_{t\in T_{进项}} t}{card(T_{进项}) + card(T_{销项})} \\ x_6 = \sum\limits_{t\in T_{缺口进项}} t - \sum\limits_{t\in T_{缺口销项}} t \end{cases}$$

2. 利用 X, 按照熵权法的步骤进行分配权重的计算。

$$p_{ij} = \frac{x_{ij}}{\sum\limits_{i=1}^{n} x_{ij}} \tag{12}$$

$$e_j = -\frac{1}{\ln n}\sum_{i=1}^{n} p_{ij} \tag{13}$$

$$g_j = 1 - e_j \tag{14}$$

$$w_j = \frac{g_i}{\sum\limits_{j=1}^{m} g_j} \tag{15}$$

式（13）利用各个指标出现的概率 p 计算熵值，并且在式（15）中计算各项指标的权重 w_{ij}。该方法由数据自身给出结果，较为客观有效。

3. 利用得出的权重 $w_j, j = 1, 2, \cdots, 6$, 可以计算每个企业的贷款评分数值 s。

$$s_i = \sum_{j=1}^{m} w_j \cdot x_{ij} \tag{16}$$

4. 在计算出每一企业的贷款评分数值 s_i 后，根据数值在当年贷款企业总评分 S 之间的占比来分配银行贷款总额 C_0。

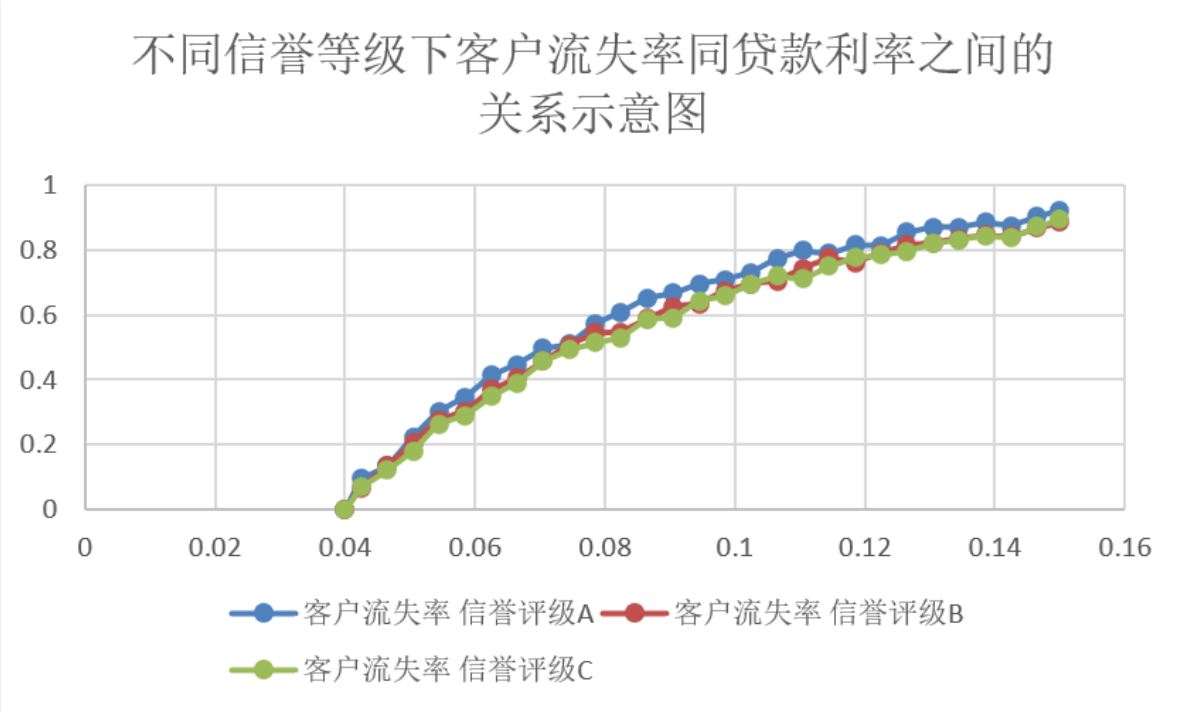
$$\begin{aligned} S &= \sum_{i=1}^{n} s_i \\ C_i &= C_0 \cdot \frac{s_1}{S} \end{aligned} \tag{17}$$

至此，我们便计算出在有能力偿还贷款的企业中，各企业分配的贷款金额 C_i。

5.1.3 贷款利率确定

银行的贷款利率会影响客户的续借情况，过高的贷款利率给企业带来负担，最终导致企业不再继续向该行借款。这种不再续借的现象称为用户流失。银行要做到收益最大化，需要平衡利率和流失率之间的关系。

分析附件三中的数据，并做可视化，得到图（2）：



注：横轴代表贷款利率，纵轴代表用户流失率。

图 2: 不同信誉等级下客户流失率同贷款利率之间的关系示意图

可见当贷款利率上升时，用户流失率也随之上升，基本反映了客观规律。且横向对比不同信誉评级的客户，具有较高信誉评级的客户，面对同一借款利率是较容易流失的。这是由于信誉较好的企业能够在同等条件下获得更佳的利率优惠，银行承担的违约风险较小，因而面对较高的贷款利率时可以做出其他选择。

银行借贷的策略，不仅需要保证当期的贷款能够获得较高利润，还要确保未来还有稳定的客户来源，因此需要保持较低的流失率。为此，我们提出了一个兼顾二者的方法，将这一问题转换为多目标规划的问题。银行考虑取得最佳收益，这要求对每个用户都给出最佳的利率 I_i, 以兼顾收益和发展。

给出两个指标：信贷收益 R 和流失率 L。

- 信贷收益 R

  即银行借款，按照利率 I 收回本息后，可以获得的利润。对于用户 i 而言，若其利率为 I_i, 银行贷款额度（5.1.2 节）为 C_i, 则银行在该用户上的信贷收益 R 计算如下：

$$R_i = C_i \cdot I_i \tag{18}$$

- 流失率 L

  流失率定义为次年用户不再续借人数同当年借款人数的比例，其关于贷款利率和信誉等级的关系已经由附件三给出，可以使用拟合的方法给出关系式 $f(I, grade)$ 来求解。

$$L_i = f(I_i, grade) \tag{19}$$

为了综合考虑这两个指标，还需要对得到的数据做进一步处理，在计算出每个用户的信贷收益 R 和流失率 L 后，计算两个指标的最大值最小值，做归一化处理。

$$\begin{aligned} R_i^* &= \frac{R_i - R_{min}}{R_{max} - R_{min}} \\ L_i^* &= \frac{L_i - L_{min}}{L_{max} - L_{min}} \end{aligned} \tag{20}$$

利用归一化后的指标，分别赋权，作为目标函数 F:

$$\underset{I_i}{argmax} F = \omega_1 \cdot R_i^* + \omega_2 \cdot L_i^* \tag{21}$$

, 其中 R_i^* 和 L_i^* 分别由式（18）式（19）以及式（20）得出，如下所示:

$$\begin{aligned} R_i^* &= \frac{C_i \cdot I_i - R_{min}}{R_{max} - R_{min}} \\ L_i^* &= \frac{f(I_i, grade) - L_{min}}{L_{max} - L_{min}} \end{aligned} \tag{22}$$

至此便得出针对每一用户的最佳利率。

5.1.4 模型求解

5.2 没有借贷记录下的银行信贷模型

对比附件一和附件二中数据，可以观察到附件二中的企业没有借贷记录，在数据上体现为缺失信誉评级 x_4 和违约记录。我们需要对其可能的信用评级进行预测。因此，我们不能直接使用 5.1 中建立的银行信贷模型，而需要先通过 K 近邻法[5] 对用户的信用评级进行预测，随后使用银行信贷模型组织信贷分配。

5.2.1 基于 K 近邻算法的信誉评级预测

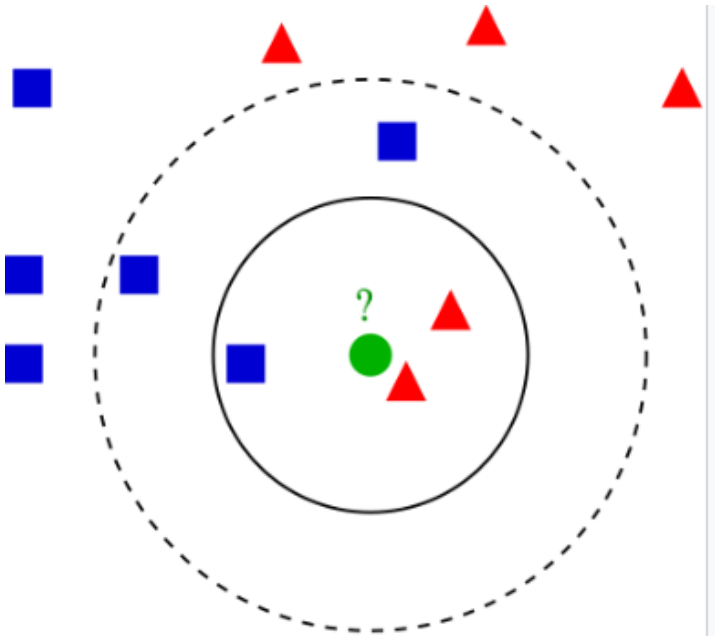
K 近邻算法是一种非参数的监督学习方法，其最大的特点是不需要训练。依赖已有的训练集数据进行投票，从而确定样本的类别。在前面的工作中，我们提出了一系列关于信贷偿还能力的指标，包括利润率，有效交易占比，供应链丰富度，平均单价等四个要素。一般而言，具有良好信用评级的企业都拥有相似的特质，每个企业的“特性”可以由一个四元向量 $\vec{a}(x_1, x_2, x_3, x_5)$ 构成，各项分别代表利润率，有效交易占比，供应链丰富度，平均单价四个要素。

在得到企业经营特点的向量表示后，使用向量距离 d 来度量两个企业之间的相似度:

$$d = \left\| \vec{a} - \vec{b} \right\| \tag{23}$$

,这里 $\vec{a},\vec{b}$ 代表两个企业的特征向量，距离度量可以使用欧几里得距离，曼哈顿距离等，依据问题的不同来选定。

要对一个企业的信誉等级进行预测，可以由其周围 k 个最近的企业投票决定。如图（3）所示：



注：图中彩色物体代表企业特征向量在二维上的可视化示意图。红色代表信誉等级为 A，蓝色表示信誉等级为 B，绿色表示待归类的企业数据。（其余信誉等级企业未列出）

图 3: k 近邻预测示意图

若取 k 值等于 3，则记录信誉等级为 A 的票数为 2，等级为 B 的票数为 1，按少数服从多数的原则，待分类企业的信用等级为 A。

依照上述步骤，可以对附件二中未知评级的企业数据依次进行处理，得到评级预测结果之后，使用银行信贷模型进行信贷资金分配。

5.2.2 模型求解

5.3 企业行业类型的划分

5.3.1 企业行业类型的划分

我们首先进行标准行业特征字、词的确定。通过对行业名称的观察和研究，我们发现企业名称能够很大程度上表明企业所在的行业。所以我们按照国民经济行业分类划分出 16 大类行业（鉴于医药业在疫情中有十分重要的地位，制造业分细分成了其他和医药制造业）。并依据国家行业标准[?] 等资料和文献将每类行业中企业名称的关键字和词提取出来。

在得到样本词和企业关键词后，我们将样本企业划分到所属的标准行业。因为能够表明企业所属行业的关键词均为 1 个和 2 个字的词语，所以我们只需将各企业的名称划分成单个字和两个相邻字为一组的词语，并将这些从企业名称中提取的字和词语与各行业的特征字、词进行匹配和对应，最终将该企业归到特征字、词重复数量最多的行业中。

5.3.2 疫情对各类行业影响的确定

在划分企业类型后，我们进行受疫情影响的相关自变量的确定。经查阅相关资料和文献[6-7]，发现疫情对企业经济生产最重要的影响就在经营利润和资金缺口上。所以我们主要还会更改企业还款能力评估模型中的利润率和贷款数量界定模型中的年平均利润和投资资金缺口。

随后判断疫情对各行业影响程度。经过查阅新冠疫情对国民经济的影响等资料，我们得到了疫情对各类行业的影响程度，如表（1）所示。

表 1: 新冠病毒疫情对不同行业的影响程度

行业分类	受疫情影响程度	特征关键字、词
农、林、牧、渔业	提高 15%	农、林、牧、渔
采矿业	下降 20%	金属、材料、矿
制造业（其他）	下降 20%	食品、饮料、酒、烟草、服饰、家具、纸、汽车、电气、电器
制造业（医药）	提高 40%	医药、医疗
电力、热力、燃气及水生产和供应业	下降 40%	电力、热力、燃气、水
建筑业	下降 10%	土木、房屋、建筑
批发和零售业	下降 20%	批发零售
交通运输、仓储和邮政业	下降 20%	物流、运输、邮政、交通
住宿和餐饮业	下降 40%	住宿、餐饮
信息传输、软件和信息技术服务业	提高 30%	电信、广播、网络、软件、信息、电子
金融业	提高 20%	货币、金融、资本、保险
房地产业	下降 21%	地产、房产
租赁和商务服务业	下降 50%	商业、服务、租赁
教育	提高 20%	教育
文化、体育和娱乐业	提高 20%	新闻、出版、广播、电视、电影、录音、文化、艺术、体育、娱乐

5.3.3 信贷策略的调整

设疫情对行业的影响程度为 α, 规定其含义如下:

$$\alpha \begin{cases} > 0, \text{疫情有利于行业发展} \\ < 0, \text{疫情不利于行业发展} \end{cases}$$

利用 α 这个调整因子，将企业还款能力评估模型修改如下，对利润率 x_1 指标进行系数调整，其余保持不变。

$$\begin{aligned} P &= \frac{1}{1-e^{-Y}} \\ Y &= \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 \end{aligned} \tag{24}$$

$$
\begin{cases}
x_1 = \dfrac{\sum\limits_{t\in T_{销项}} t - \sum\limits_{t\in T_{进项}} t}{\sum\limits_{t\in T_{进项}} t}(1+\alpha) \\
x_2 = \dfrac{card(T_{有效,t>0})}{card(T)} \\
x_3 = card(c_{购方}) + card(c_{销方}) \\
x_4 = \begin{cases} 1, A \\ 0.75, B \\ 0.5, C \\ 不予放贷, D \end{cases} \\
x_5 = \dfrac{\sum\limits_{t\in T_{销项}} t + \sum\limits_{t\in T_{进项}} t}{card(T_{进项}) + card(T_{销项})}
\end{cases}
$$

随后对贷款数量界定模型进行修改，修改利润率 x_1 以及资金缺口 x_6, 其余保持不变。

$$
\begin{aligned}
S &= \sum_{i=1}^{n} s_i \\
C_i &= C_0 \cdot \frac{s_1}{S}
\end{aligned}
\tag{25}
$$

$$
\begin{cases}
x_1 = \dfrac{\sum\limits_{t\in T_{销项}} t - \sum\limits_{t\in T_{进项}} t}{\sum\limits_{t\in T_{进项}} t}(1+\alpha) \\
x_2 = \dfrac{card(T_{有效,t>0})}{card(T)} \\
x_3 = card(c_{购方}) + card(c_{销方}) \\
x_4 = \begin{cases} 1, A \\ 0.75, B \\ 0.5, C \\ 不予放贷, D \end{cases} \\
x_5 = \dfrac{\sum\limits_{t\in T_{销项}} t + \sum\limits_{t\in T_{进项}} t}{card(T_{进项}) + card(T_{销项})} \\
x_6 = \left(\sum\limits_{t\in T_{缺口进项}} t - \sum\limits_{t\in T_{缺口销项}} t\right)\cdot(1+\alpha)
\end{cases}
$$

七、模型的评价

6.1 模型的优点

1. 此模型充分的利用了所给数据，并且对数据都进行了一定处理，将初始数据变为具有经济意义的指标。

2. 企业还款能力模型采用多元线性回归，可以使各指标不仅与因变量 P 具有简单的线性关系，避免了线性拟合带来的弊端。

3. 贷款数量的界定模型使用了熵权法，可以在没有储备相关知识的情况下做出客观的评价，提高了模型的适用性。

4. 针对突发情况下的贷款策略调整模型仅在原有模型上进行修饰，可以整体上简化模型。

6.2 模型的缺点

- 突发情况下的贷款策略调整模型中对企业类型的划分可以使用更先进的文本聚类，但因时间有限未能完成，会降低模型的适用性。

参考文献

[1] 利润 [M/OL]//维基百科，自由的百科全书. 2022[2022-08-27]. https://zh.wikipedia.org/w/index.php?title=&oldid=72812261.

[2] 信用评级 [M/OL]//维基百科，自由的百科全书. 2021[2022-08-27]. https://zh.wikipedia.org/w/index.php?title=%E4%BF%A1%E7%94%A8%E8%AF%84%E7%BA%A7&oldid=64636182.

[3] 楚泽昊. 基于税务信息的中小企业信用评价体系研究 [EB/OL]. 河南财经政法大学(2020)[2022-08-26]. http://kns-cnki-net-s.vpn.uestc.edu.cn:8118/kcms/detail/detail.aspx?dbcode=CMFD&dbname=CMFD202002&filename=1020625957.nh&uniplatform=NZKPT&v=bneTNuR0p-f2nePl-N5teHl5IpnrziB_HGComTgr7cp8v2P90Gw2aFsVoDYHNG3E. DOI: 10.27113/d.cnki.ghncc.2020.000188.

[4] General linear model[M/OL]//Wikipedia. 2022[2022-08-27]. https://en.wikipedia.org/w/index.php?title=General_linear_model&oldid=1102725280.

[5] *k*-nearest neighbors algorithm[M/OL]//Wikipedia. 2022[2022-08-28]. https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=1091525121.

[6] 赵汉伟, 朱津锐, 胡晓忠. 新冠肺炎疫情对于中小企业的影响程度研究——基于财务报告分析基础上 [J/OL]. 齐鲁珠坛(4): 9-12[2022-08-28]. http://kns-cnki-net-s.vpn.uestc.edu.cn:8118/kcms/detail/detail.aspx?dbcode=CJFD&dbname=CJFDLAST2020&filename=QLZT202004004&uniplatform=NZKPT&v=ei8Yx5lrDSx_DeGzwvIQL3R-PDcdrjaxYlcla7v0KMLwwgOqRE8ywqR2eVAfv1Xd.

[7] 赵楷文, 周呈妍, 钱艺萱, 等. 新冠疫情对服务行业的影响研究——以南京市为例 [J/OL]. 现代商业(34): 27-30[2022-08-28]. http://kns-cnki-net-s.vpn.uestc.edu.cn:8118/kcms/detail/detail.aspx?dbcode=CJFD&dbname=CJFDLAST2022&filename=XDBY202134008&uniplatform=NZKPT&v=mxUAR9zyeekPYhcAdngiYX5XdeoIsqH7mqhc8FXoMZTtf3EROuLrRhJDffq1kjL8. DOI: 10.14097/j.cnki.5392/2021.34.008.

附件

附件清单：

- 主函数

```
%% 主函数
%进行多元线性拟合

clear;clc;
data_path = 'C:\Users\ASUS\Desktop\第四次模拟赛\data\data1.mat';
output_path = 'C:\Users\ASUS\Desktop\第四次模拟赛\output data';
load(data_path) % 获得包含123个有贷款历史的用户信息的元胞数组data
answer = zeros(length(data),8); % Y P YY x1 x2 x3 x4 x5
X = zeros(length(data),6);
X4 = zeros(length(data),1);

for i = 1:length(data)
    user = data{i};
    answer(i,:) = [y(user),1/(1+exp(-y(user))),YY(user),x1(user),x2(
        user),x3(user),x4(user),x5(user)];
    X(i,:) = [x1(user),x2(user),x3(user),x4(user),x5(user),YY(user)];
    X4(i,:) = x4(user);
    user.x1 = x1(user);
    user.x2 = x2(user);
    user.x3 = x3(user);
    user.x4 = x4(user);
    user.x5 = x5(user);
    data{i} = user;
end
index = find(X4 == 0);
X(index,:) = [];
answer(index,:) = [];

vector = answer;
save('data.mat','data')
save(strcat(output_path,'\vector.mat'),"vector");
```

- 数据预处理代码

```
%% 数据预处理
% 元胞数组 元胞数组内部存储结构体
```

```
3
4  clear;clc;
5  data_path_1 = 'C:\Users\ASUS\Desktop\第四次模拟赛\data\data1_1.csv';
6  data_path_2 = 'C:\Users\ASUS\Desktop\第四次模拟赛\data\data1_2.csv';
7  data_path_3 = 'C:\Users\ASUS\Desktop\第四次模拟赛\data\data1_3.csv';
8  output_path = 'C:\Users\ASUS\Desktop\第四次模拟赛\output data';
9  data_1 = readmatrix(data_path_1,'OutputType','string'); % 公司基本信息
10 data_out = readmatrix(data_path_2,'OutputType','string'); % 进项发票信
       息
11 data_in = readmatrix(data_path_3,'OutputType','string'); % 销项发票信
       息
12
13 data = cell(1,height(data_1)-1); % 目标元胞数组
14
15 for n = 2:height(data_1) %获取公司数
16     user = struct;
17     user.num_str = data_1(n,1);
18     num_str = data_1(n,1);
19     user.num = n - 1; % 代号
20     user.name = data_1(n,2); % 公司名
21     user.level = data_1(n,3); % 公司评级
22     if data_1(n,4) == '否'
23         user.breach = 0; % 违约词条
24     else
25         user.breach = 1;
26     end
27     index_out = find(data_out(:,1)==num_str);
28     index_in = find(data_in(:,1)==num_str);
29     out_mat = zeros(length(index_out),7); % 进项发票也就是出去的钱款
30     in_mat = zeros(length(index_in),7); % 销项发票也就是进账的钱款
31     out_mat(:,1) = str2double(data_out(index_out,2)); % 发票编号
32     out_mat(:,2) = datenum(data_out(index_out,3)); % 日期序列号
33     out_mat(:,3) = str2double(strrep(data_out(index_out,4),'A','1'));
34     out_mat(:,4:6) = str2double(data_out(index_out,5:7));
35     index = data_out(index_out,8) == '有效发票';
36     out_mat(:,7) = index;
37     user.out = out_mat;
38
39     in_mat(:,1) = str2double(data_in(index_in,2)); % 发票编号
```

```
    in_mat(:,2) = datenum(data_in(index_in,3)); % 日期序列号
    in_mat(:,3) = str2double(strrep(data_in(index_in,4),'B','2'));
    in_mat(:,4:6) = str2double(data_in(index_in,5:7));
    index = data_in(index_in,8) == '有效发票';
    in_mat(:,7) = index;
    user.in = in_mat;
    data{n-1} = user;
    disp(n-1);
end
save(strcat(output_path,'\data1.mat'),"data")
```

- 多元线性回归相关代码，各变量处理

```
function y = x1(user)
    % 求解企业的利润率
    index_in = find(user.in(:,7) == 1); %获得有效发票的index
    index_out = find(user.out(:,7) == 1);
    y = (sum(user.in(index_in,6))-sum(user.out(index_out,6)))/(sum(
        user.out(index_out,6)));
end
```

```
function y = x2(user)
    % 有效正交易占比
    num = height(user.in) + height(user.out); % 发票总数
    num_2 = sum((user.in(:,6) > 0) & (user.in(:,7) == 1)) + sum((user.
        out(:,6) > 0) & (user.out(:,7) == 1));
    y = num_2 / num;
end
```

```
function y = x3(user)
    %上下游企业数量
    y = length(unique(user.in(:,3))) + length(unique(user.out(:,3)));
end
```

```
function y = x4(user)
    % 信誉等级的量化
    if user.level == 'A'
        y = 1;
    elseif user.level == 'B'
        y = 0.75;
```

```
    elseif user.level == 'C'
        y = 0.5;
    elseif user.level == 'D'
        y = 0; % 原则上不借款
    end
end
```

```
function y = x5(user)
    % 每笔销进项的平均金额
    in = (user.in(:,6) > 0) & (user.in(:,7) == 1);
    out = (user.out(:,6) > 0) & (user.out(:,7) == 1);
    N = sum(in) + sum(out);
    n = sum(user.in(:,6).*in) + sum(user.out(:,6).*out);
    y = n / N;
end
```

```
function Y = y(user)
    % 根据是否违约确定Y
    if user.breach == 0
        Y = 2.944;
    elseif user.breach == 1
        Y = -2.944;
    end
end
```

```
function Y = YY(user)
    % 根据是否违约确定Y
    if user.breach == 0
        Y = 1;
    elseif user.breach == 1
        Y = -1;
    end
end
```

```
clear;clc;

data_path = 'C:\Users\ASUS\Desktop\第四次模拟赛\output data';

load(strcat(data_path,'\vector.mat'));
```

```
[b,bint,r,rint,stats] = regress(vector(:,1),[ones(height(vector),1),
    vector(:,4:8)]);
```

```
function [trainedClassifier, validationAccuracy] = trainClassifier(
    trainingData)
% [trainedClassifier, validationAccuracy] = trainClassifier(
    trainingData)
% 返回经过训练的分类器及其 准确度。以下代码重新创建在分类学习器中训练
    的分类模型。您可以使用
% 该生成的代码基于新数据自动训练同一模型，或通过它了解如何以程序化方式
    训练模型。
%
%  输入:
%      trainingData: 一个与导入 App 中的矩阵具有相同列数和数据类型的矩
    阵。
%
%  输出:
%      trainedClassifier: 一个包含训练的分类器的结构体。该结构体中具有
    各种关于所训练分
%       类器的信息的字段。
%
%      trainedClassifier.predictFcn: 一个对新数据进行预测的函数。
%
%      validationAccuracy: 包含准确度百分比的双精度值。在 App 中，"模
    型" 窗格显示每
%       个模型的总体准确度分数。
%
% 使用该代码基于新数据来训练模型。要重新训练分类器，请使用原始数据或新
    数据作为输入参数
% trainingData 从命令行调用该函数。
%
% 例如，要重新训练基于原始数据集 T 训练的分类器，请输入:
%   [trainedClassifier, validationAccuracy] = trainClassifier(T)
%
% 要使用返回的 "trainedClassifier" 对新数据 T2 进行预测，请使用
%   yfit = trainedClassifier.predictFcn(T2)
%
% T2 必须是仅包含用于训练的预测变量列的矩阵。有关详细信息，请输入:
%   trainedClassifier.HowToPredict

```

```
% 由 MATLAB 于 2022-08-27 21:25:37 自动生成


% 提取预测变量和响应
% 以下代码将数据处理为合适的形状以训练模型。
%
% 将输入转换为表
inputTable = array2table(trainingData, 'VariableNames', {'column_1', 'column_2', 'column_3', 'column_4', 'column_5', 'column_6'});

predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', 'column_5'};
predictors = inputTable(:, predictorNames);
response = inputTable.column_6;
isCategoricalPredictor = [false, false, false, false, false];

% 训练分类器
% 以下代码指定所有分类器选项并训练分类器。
% 对于逻辑回归，必须将响应值转换为 0 和 1，因为响应被假定为遵循二项分布。
% 1 或 true = '成功' 类
% 0 或 false = '失败' 类
% NaN - 缺失响应。
successClass = double(1);
failureClass = double(-1);
% 计算主响应类。如果 fitglm 的预测中包含 NaN，则会将 NaN 转换为这一主类标签。
numSuccess = sum(response == successClass);
numFailure = sum(response == failureClass);
if numSuccess > numFailure
    missingClass = successClass;
else
    missingClass = failureClass;
end
successFailureAndMissingClasses = [successClass; failureClass; missingClass];
isMissing = isnan(response);
zeroOneResponse = double(ismember(response, successClass));
zeroOneResponse(isMissing) = NaN;
```

```
% 为 fitglm 准备输入参数。
concatenatedPredictorsAndResponse = [predictors, table(zeroOneResponse
    )];
% 使用 fitglm 进行训练。
GeneralizedLinearModel = fitglm(...
    concatenatedPredictorsAndResponse, ...
    'Distribution', 'binomial', ...
    'link', 'logit');

% 将预测概率转换为预测类标签和分数。
convertSuccessProbsToPredictions = @(p)
    successFailureAndMissingClasses( ~isnan(p).*( (p<0.5) + 1 ) + isnan
    (p)*3 );
returnMultipleValuesFcn = @(varargin) varargin{1:max(1,nargout)};
scoresFcn = @(p) [1-p, p];
predictionsAndScoresFcn = @(p) returnMultipleValuesFcn(
    convertSuccessProbsToPredictions(p), scoresFcn(p) );

% 使用预测函数创建结果结构体
predictorExtractionFcn = @(x) array2table(x, 'VariableNames',
    predictorNames);
logisticRegressionPredictFcn = @(x) predictionsAndScoresFcn( predict(
    GeneralizedLinearModel, x) );
trainedClassifier.predictFcn = @(x) logisticRegressionPredictFcn(
    predictorExtractionFcn(x));

% 向结果结构体中添加字段
trainedClassifier.GeneralizedLinearModel = GeneralizedLinearModel;
trainedClassifier.SuccessClass = successClass;
trainedClassifier.FailureClass = failureClass;
trainedClassifier.MissingClass = missingClass;
trainedClassifier.ClassNames = {successClass; failureClass};
trainedClassifier.About = '此结构体是从分类学习器 R2022a 导出的训练模
    型。';
trainedClassifier.HowToPredict = sprintf('要对新预测变量列矩阵 X 进行
    预测，请使用: \n yfit = c.predictFcn(X) \n将 ''c'' 替换为作为此结构
    体的变量的名称，例如 ''trainedModel''。\n \nX 必须包含正好 5 个列，
    因为此模型是使用 5 个预测变量进行训练的。\nX 必须仅包含与训练数据具
    有完全相同的顺序和格式的\n预测变量列。不要包含响应列或未导入 App 的
```

```
    任何列。\n \n有关详细信息，请参阅 <a href="matlab:helpview(fullfile
    (docroot, ''stats'', ''stats.map''), ''
    appclassification_exportmodeltoworkspace'')">How to predict using
    an exported model</a>。');
91
92  % 提取预测变量和响应
93  % 以下代码将数据处理为合适的形状以训练模型。
94  %
95  % 将输入转换为表
96  inputTable = array2table(trainingData, 'VariableNames', {'column_1', '
        column_2', 'column_3', 'column_4', 'column_5', 'column_6'});
97
98  predictorNames = {'column_1', 'column_2', 'column_3', 'column_4', '
        column_5'};
99  predictors = inputTable(:, predictorNames);
100 response = inputTable.column_6;
101 isCategoricalPredictor = [false, false, false, false, false];
102
103 % 执行交叉验证
104 KFolds = 5;
105 cvp = cvpartition(response, 'KFold', KFolds);
106 % 将预测初始化为适当的大小
107 validationPredictions = response;
108 numObservations = size(predictors, 1);
109 numClasses = 2;
110 validationScores = NaN(numObservations, numClasses);
111 for fold = 1:KFolds
112     trainingPredictors = predictors(cvp.training(fold), :);
113     trainingResponse = response(cvp.training(fold), :);
114     foldIsCategoricalPredictor = isCategoricalPredictor;
115
116     % 训练分类器
117     % 以下代码指定所有分类器选项并训练分类器。
118     % 对于逻辑回归，必须将响应值转换为 0 和 1，因为响应被假定为遵循二
            项分布。
119     % 1 或 true = '成功' 类
120     % 0 或 false = '失败' 类
121     % NaN - 缺失响应。
122     successClass = double(1);
```

```
    failureClass = double(-1);
    % 计算主响应类。如果 fitglm 的预测中包含 NaN, 则会将 NaN 转换为这
        一主类标签。
    numSuccess = sum(trainingResponse == successClass);
    numFailure = sum(trainingResponse == failureClass);
    if numSuccess > numFailure
        missingClass = successClass;
    else
        missingClass = failureClass;
    end
    successFailureAndMissingClasses = [successClass; failureClass;
       missingClass];
    isMissing = isnan(trainingResponse);
    zeroOneResponse = double(ismember(trainingResponse, successClass))
       ;
    zeroOneResponse(isMissing) = NaN;
    % 为 fitglm 准备输入参数。
    concatenatedPredictorsAndResponse = [trainingPredictors, table(
       zeroOneResponse)];
    % 使用 fitglm 进行训练。
    GeneralizedLinearModel = fitglm(...
        concatenatedPredictorsAndResponse, ...
        'Distribution', 'binomial', ...
        'link', 'logit');

    % 将预测概率转换为预测类标签和分数。
    convertSuccessProbsToPredictions = @(p)
       successFailureAndMissingClasses( ~isnan(p).*( (p<0.5) + 1 ) +
       isnan(p)*3 );
    returnMultipleValuesFcn = @(varargin) varargin{1:max(1,nargout)};
    scoresFcn = @(p) [1-p, p];
    predictionsAndScoresFcn = @(p) returnMultipleValuesFcn(
       convertSuccessProbsToPredictions(p), scoresFcn(p) );

    % 使用预测函数创建结果结构体
    logisticRegressionPredictFcn = @(x) predictionsAndScoresFcn(
       predict(GeneralizedLinearModel, x) );
    validationPredictFcn = @(x) logisticRegressionPredictFcn(x);

```

```
    % 向结果结构体中添加字段

    % 计算验证预测
    validationPredictors = predictors(cvp.test(fold), :);
    [foldPredictions, foldScores] = validationPredictFcn(
        validationPredictors);

    % 按原始顺序存储预测
    validationPredictions(cvp.test(fold), :) = foldPredictions;
    validationScores(cvp.test(fold), :) = foldScores;
end

% 计算验证准确度
correctPredictions = (validationPredictions == response);
isMissing = isnan(response);
correctPredictions = correctPredictions(~isMissing);
validationAccuracy = sum(correctPredictions)/length(correctPredictions
    );
```

```
function P = class1(x1,x2,x3,x4,x5)
    % 输入五个变量，进行逻辑判别是否借贷
    Y = 7.0939+1.4956e-05*x1+-5.4457*x2+5.3433e-05*x3+0.9045*x4+3.0582
        e-08*x5;
    P = 1/(1+exp(-Y));
end
```