队伍编号	MC2303198
题号	A

基于 QUBO 模型的信用评分卡组合优化分析

摘 要

本题是要求采用 *QUBO* 模型求解的信用评分卡组合优化问题。如何选择最合理的信用评分卡组合以及其阈值，使得银行最终收入最多以及如何将基本模型转化为便于求解的 *QUBO* 形式成为本文的核心。

针对问题一，我们对题目中的 1000 个阈值按照顺位排列进行单独编号，用 1000 个 0—1 决策变量去控制阈值是否被选取；接着，我们建立一个简单的 0—1 整数规划模型，将其转化为带约束的 *QUBO* 模型，此时我们将约束条件转变为惩罚函数与目标函数整合成基础的 *QUBO* 模型；然后，我们对设定不同的惩罚系数进行测试，对运行结果，运行时间，运行效率进行分析，我们发现惩罚系数越小，运行时间越短，运行时间与惩罚系数呈线性关系。我们将要求解的 *QUBO* 看成 *MIQP* 问题的特例；最后利用 matlab 和 SCIP 求解器得出第 **49** 张信用评分卡的阈值设置为 **1** 时，最大收入为 **61172** 元。

针对问题二，我们需要通过不同的组合方式来选择不同的阈值，以达到最大化贷款利息收入和最小化坏账损失的目标。首先，我们明确有三大类组合方式，对选定一张信用评分卡和选定两张信用评分卡的情况我们都可以化归为问题 1 的模型进行求解，具体结果可以参考表 5，表 6；然后，讨论三张信用评分卡均选定的情况，先对阈值进行顺位编号，建立 0—1 整数规划模型，此时转化成的 *QUBO* 模型含有三次多项式，我们利用 *Rosenberg* 多项式通过增加变量和约束的方法对其进行降维处理转化为基础的 *QUBO* 模型；最后利用 matlab 和 SCIP 求解器得出三张信用评分卡阈值分别设置为 **8，1，2** 时，最大收入为 **27915** 元。通过比较可知单独设置选定信用评分卡 **3** 的阈值为 **2** 时最终收入最大为 **54087** 元。与此同时，我们对合适的惩罚系数选取进行了试验，并利用问题 1 的 *QUBO* 模型对问题 2 建模，验证了结果的一致性并进行比较分析。

针对问题三，本题需要对任意的三张信用评分卡设定阈值以获得最大收入。首先，对任意的三张信用评分卡，都通过问题 2 的解决思路求出最优的阈值组合和最大收入；然后，我们直接计算所有的三张信用评分卡组的最大收入，记录各个数据至数据库，在数据库中找出最大的最终收入对应的信用评分卡及其最优阈值组合；最后，我们发现选定信用评分卡 **8** 阈值设置为 **2**，信用评分卡 **33** 阈值设置为 **6**，信用评分卡 **49** 阈值设置为 **3** 时最终收入最大为 **43881** 元。

关键词：**QUBO 模型　信用评分卡　组合优化　SCIP 求解器**

目录

一、问题重述

1.1 问题背景

1.1.1 信用评分卡

在银行信用卡或相关的贷款等业务中，对客户授信之前，需要先通过各种审核规则对客户的信用等级进行评定，通过评定后的客户才能获得信用或贷款资格。规则审核过程实际是经过一重或者多重组合规则后对客户进行打分，这些规则就被称为信用评分卡，每个信用评分卡又有多种阈值设置（但且只有一个阈值生效），这就使得不同的信用评分卡在不同的阈值下，对应不同的通过率和坏账率，一般通过率越高，坏账率也会越高，反之，通过率越低，坏账率也越低。对银行来说，通过率越高，通过贷款资格审核的客户数量就越多，相应的银行获得的利息收入就会越多，但高通过率一般对应着高坏账率，而坏账意味着资金的损失风险，选择不同的信用评分卡，不同的阈值组合，会给银行带来不同的收入与损失，决定银行最终收入。因此，选择最合理的信用评分卡组合以及其阈值，使得最终收入最多成为银行的研究重点之一。

1.1.2 *QUBO* 模型

QUBO（*Quadratic Unconstrained Binary Optimization*）模型是指二次无约束二值优化模型，它是一种用于解决组合优化问题的数学模型。在 *QUBO* 模型中，需要将问题转化为一个决策变量为二值变量，目标函数是一个二次函数形式优化模型。*QUBO* 模型可以运行在量子计算机硬件上，通过量子计算机进行毫秒级的加速求解。这种模型和加速方式在未来各行业中将得到广泛的实际应用。因此现阶段研究基于 *QUBO* 模型的量子专用算法十分有应用价值。例如典型的图着色、旅行商问题、车辆路径优化问题等，都可以转化为 *QUBO* 模型并借助于量子计算机求解。

1.2 目标任务

根据上面的说明及附件 1 中的数据，假设贷款资金为 1000000 元，银行贷款利息收入率为 8%, 请你们团队通过建立数学模型完成如下问题 1 至问题 3。

问题 1： 在 100 个信用评分卡中找出 1 张及其对应阈值，使最终收入最多，请针对该问题进行建模，将该模型转为 *QUBO* 形式并求解。

问题 2： 假设赛题说明 3 目前已经选定了数据集中给出的信用评分卡 1、信用评分卡 2、信用评分卡 3 这三种规则，如何设置其对应的阈值，使最终收入最多，请针对该问题进行建模，将模型转为 *QUBO* 形式并求解。

问题 3：：从所给附件 1 中 100 个信用评分卡中任选取 3 种信用评分卡，并设置合理的阈值，使得最终收入最多，请针对该问题进行建模，并将模型转为 *QUBO* 形式并求解。

二、问题分析

不同的信用评分卡会有不同的阈值设置，不同的阈值会对应不同的通过率和坏账率。银行的最终收入由贷款利息收入和坏账损失共同决定，现在需要使用量子计算机对信用评分卡进行优化，以最大化银行的最终收入。这个问题需要建立一个优化模型，进而转化为 *QUBO* 形式解决。为了建立模型，我们需要明确一些问题，例如评分卡如何打分，阈值对通过率和坏账率的影响如何描述，以及如何建立银行收入与阈值之间的联系等等。

2.1 针对问题一

问题一要求建立一个在 100 个信用评分卡中找出 1 张及其对应阈值，使最终收入最多的数学模型，并将该模型转为 *QUBO* 形式并求解。首先，我们根据附件 1 中的数据可知，每张信用评分卡均有十个待设置的阈值且数据完整，结合下图，本质上我们只需要找到一个最优阈值，使得最终收入最多；然后，将所有的 1000 个阈值有序编号，通过引入决策变量，我们将该问题建模为一个 0—1 整数规划问题；最后，我们将其转化为有约束优化问题的 *QUBO* 模型并用 matlab 和 SCIP 求解器进行求解。

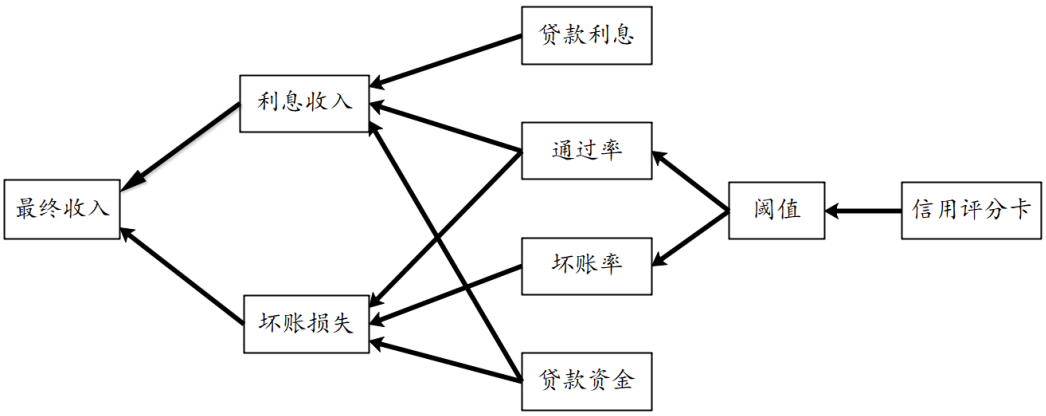

图 1　最终收入关系图

2.2 针对问题二

问题二要求给已确定的信用评分卡 1、信用评分卡 2、信用评分卡 3 在规则内选择性的设置阈值，使最终收入最多，对其进行建模并将模型转为 *QUBO* 形式并求解。在这个问题中，需要通过不同的组合方式来选择不同的阈值，以达到最大化贷款利息收入和最小化坏账损失的目标。首先，对于信用评分卡而言，我们可以确定一共有 7 种组合方式；然后，我们根据组合方式中信用评分卡的数量将其分为三类进行讨论，结合问题 1 分别建立 0—1 整数规划模型；最后，我们将其转化为有约束优化问题的 *QUBO* 模型并用 matlab 和 SCIP 求解器进行求解。

2.3 针对问题三

本题需要对任意的三张信用评分卡设定阈值以获得最大收入。对任意的三张信用评分卡，都通过问题 2 的解决思路求出最优的阈值组合和最大收入，我们直接计算所有的三张信用评分卡组的最大收入，记录各个数据至数据库，在数据库中找出最大的最终收入对应的信用评分卡及其最优阈值组合。

三、模型的假设及符号说明

3.1 模型假设

为了便于考虑问题，我们在不影响模型准确性的前提下，根据题意作出以下假设：

（1）假设信用评分卡组合后的总通过率为所有信用评分卡通过率相乘，总坏账率为所有信用评分卡对应坏账率的平均值。

（2）假设贷款利息收入为：贷款资金 × 利息收入率 × 总通过率 ×（1 - 总坏账率）。

（3）假设坏账带来的坏账损失为：贷款资金 × 总通过率 × 总坏账率。

（4）假设银行的最终收入为：贷款利息收入 - 坏账损失。

（5）认为附件中的所有数据均真实可靠。

3.2 主要的符号说明

注：此为本文的主要符号说明，其它符号解释详见正文部分。

符号	意义
M	贷款资金
p	贷款利息的收入率
Q	$QUBO$ 矩阵
α_i	编号为 i 所对应的通过率
β_i	编号为 i 所对应的坏账率
λ	惩罚系数
x_i	决策变量
B	常数项
A_i	常系数
q_{ij}	矩阵 Q 第 i 行第 j 列的元素
$g(x_1, x_2, \ldots, x_n)$	惩罚函数
$f(x_1, x_2, \ldots, x_n)$	目标函数

四、模型一的建立与求解

4.1 对阈值编号

首先，根据附件 1 中数据，我们将 100 张信用评分卡的通过率和坏账率重新进行编号，如下表所示：

表 1　信用评分卡通过率编号

编号	t_1	编号	t_2	编号	t_3	……	编号	t_{100}
1	0.76	11	0.72	21	0.80	……	991	0.71
2	0.77	12	0.73	22	0.82	……	992	0.78
3	0.78	13	0.76	23	0.83	……	993	0.80
4	0.80	14	0.77	24	0.86	……	994	0.81
5	0.82	15	0.79	25	0.89	……	995	0.84
6	0.84	16	0.82	26	0.92	……	996	0.86
7	0.87	17	0.86	27	0.93	……	997	0.87
8	0.93	18	0.87	28	0.94	……	998	0.88
9	0.94	19	0.91	29	0.97	……	999	0.90
10	0.96	20	0.92	30	0.98	……	1000	0.91

表 2　信用评分卡坏账率编号

编号	h_1	编号	h_2	编号	h_3	……	编号	h_{100}
1	0.013	11	0.032	21	0.012	……	991	0.016
2	0.015	12	0.038	22	0.013	……	992	0.024
3	0.017	13	0.050	23	0.024	……	993	0.028
4	0.024	14	0.053	24	0.040	……	994	0.040
5	0.026	15	0.065	25	0.042	……	995	0.043
6	0.028	16	0.074	26	0.057	……	996	0.052
7	0.030	17	0.076	27	0.068	……	997	0.053
8	0.036	18	0.079	28	0.069	……	998	0.064
9	0.043	19	0.080	29	0.070	……	999	0.069
10	0.052	20	0.084	30	0.073	……	1000	0.084

由上表可知，编号为 k 的通过率（坏账率）表示的是信用评分卡 i 阈值设置为 j 时所对应的通过率（坏账率）。其中

$$i = [\frac{k-1}{10}] + 1 \tag{1}$$

$$j = k - 10 \cdot [\frac{k-1}{10}] \tag{2}$$

我们现在仅需要在 1000 组通过率（坏账率）中找出一组最优的通过率（坏账率），使得最终收入最多。

4.2 信用评分卡的 0—1 整数规划模型

我们定义如下 0—1 决策变量：

$$x_i = \begin{cases} 1, \text{编号为 } i \text{ 的阈值是最优的} \\ 0, \text{其他} \end{cases} \tag{3}$$

明显的，我们只从 1000 组通过率（坏账率）中取一组作为我们的最优解，那么：

$$\sum_{i=1}^{1000} x_i = 1 \tag{4}$$

由题目信息知，贷款利息收入为：

$$W_1 = \sum_{i=1}^{1000} Mp\alpha_i(1-\beta_i)x_i \tag{5}$$

由坏账带来的坏账损失为：

$$W_2 = \sum_{i=1}^{1000} M\alpha_i\beta_i x_i \tag{6}$$

其中，M 表示贷款资金，p 表示贷款利息的收入率，α_i，β_i 分别表示编号为 i 的通过率和坏账率。

我们的目标是使得最终收入最多，目标函数：

$$\max Z = W_1 - W_2 \tag{7}$$

综上所述，建立如下的 0—1 整数规划模型：

目标函数：

$$\max Z = W_1 - W_2 = \sum_{i=1}^{1000} Mp\alpha_i(1-\beta_i)x_i - \sum_{i=1}^{1000} M\alpha_i\beta_i x_i \tag{8}$$

$$s.t. \begin{cases} \sum\limits_{i=1}^{1000} x_i = 1 \\ x_i \in (0,1), i = 1,2,\ldots,1000 \end{cases} \tag{9}$$

4.3 *QUBO* 模型的建立

QUBO（*Quadratic Unconstrained Binary Optimizatoin*），无约束二次二进制优化模型是现在量子计算中应用最广泛的优化模型，它统一了丰富多样的组合优化问题。随着问题规模的增加，利用传统方法求解该问题，求解时间会很长，但利用 *QUBO* 模型可以通过量子计算机加速，高效求解组合优化问题。具体形式如下：

4.3.1 基础的 $QUBO$ 模型[2]

$$\min/\max y = x^T Q x \tag{10}$$

Q 为 $QUBO$ 矩阵，x 为二进制变量组成的向量，每个变量取值均为 0 或 1，$QUBO$ 的目标为找到使得 y 最小或最大的 x。

其中，Q 矩阵的形式有 2 种：1）对称形式　2）上三角形式

4.3.2 有约束的 $QUBO$ 模型

基础 $QUBO$ 的模型要求 x 是 0—1 变量外，是没有其他约束的，然而在实际例子中，往往有其他的约束限制，我们需要对这些约束进行转换，转换的思想很简单，假设我们的目的是计算目标函数的最小值，那么我们可以在目标函数上增加一个非负的惩罚函数（设计为关于决策变量的二次函数形式），该函数满足下面两点要求：

（1）如果当前解满足约束，那么惩罚函数的值等于 0；

（2）如果当前解不满足约束，那么惩罚函数是一个很大的正数。

在此基础上，正常有约束的优化问题会变换成下面的形式：

$$\min y = f(x_1, x_2, \ldots, x_n) + \lambda g(x_1, x_2, \ldots, x_n) \tag{11}$$

其中，f 是目标函数，g 是惩罚函数，λ 是调整惩罚函数影响的权重值，被称为惩罚系数，惩罚系数越大，要满足约束的希望越强烈。之后再将 y 转化为基础的 $QUBO$ 形式进行求解。

4.3.3 问题 1 模型的 $QUBO$ 形式

我们结合有约束的 $QUBO$ 模型对问题 1 的 0—1 整数规划模型进行转换。首先我们将目标函数转化为求最小值，然后将约束条件转化为惩罚函数。

目标函数：

$$\min y = -\sum_{i=1}^{1000} A_i x_i \tag{12}$$

其中 A_i=Mpα_i(1-β_i)-M$\alpha_i\beta_i$,i=1,2,…,1000, 根据附件 1 中的数据，我们可以算出每一个 A_i 的值，最后得出 $QUBO$。

根据约束条件：

$$\sum_{i=1}^{1000} x_i = 1 \tag{13}$$

我们设惩罚函数为：

$$g(x_1, x_2, \ldots, x_{1000}) = (\sum_{i=1}^{1000} x_i - 1)^2 \tag{14}$$

最终化为的 $QUBO$ 形式[2] 如下：

$$\min H = -\sum_{i=1}^{1000} A_i x_i + \lambda(\sum_{i=1}^{1000} x_i - 1)^2 \tag{15}$$

注意到 ${x_i}^2 = x_i$，将上式展开后：

$$\min H = \sum_{i=1}^{1000} (-\lambda - A_i){x_i}^2 - 2\lambda \sum_{i \leqslant j} x_i x_j + B \tag{16}$$

将其转化为基础的 $QUBO$ 模型：

$$\min y = x^T Q x \tag{17}$$

$$Q = \begin{bmatrix} -\lambda - A_1 & \lambda & \cdots & \lambda & \lambda \\ \lambda & -\lambda - A_2 & \ddots & \vdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \lambda \\ \lambda & \lambda & \cdots & \lambda & -\lambda - A_{1000} \end{bmatrix}$$

其中，$q_{ii} = -\lambda - A_i, q_{ij} = \lambda, i, j = 1,2,\ldots,1000, B$ 是常数项。

4.3.4 问题 1 的 $QUBO$ 模型求解

由于 MATLAB 没有求解 $QUBO$ 的内置函数，因此我们借助第三方求解器:OPTI Toolbox。OPTI Toolbox 集成了一系列优化求解器，安装好后在 Matlab 命令窗口输入 optiSolver，即可看到自带的求解器。对于 $QUBO$ 优化问题，工具箱中自带的求解器很弱，容易求出局部最优解，因此我们我们需要补充安装一个更强大的 SCIP 求解器。

(1)SCIP 求解器介绍[4]

SCIP(Solving Constraint Integer Programs) 一款非常成熟的开源混合整数规划 (MILP) 框架，支持自定义搜索树中的各个模块。SCIP 之所以命名成整数约束规划求解器，是因为他的技术结合了整数线性规划和约束规划的技术，现在运筹学应用中，基本以调库为主。学术版本的 SCIP 可以免费使用，只需要提交一个邮箱即可下载 scip.mexw64，下载好后需要把文件放到 OPTI 的 Solvers 文件夹。结合 OPTI Toolbox 可以高效简便的解决 LP,QP,$MILP$,$MIQP$,$QCQP$,SDP 等一系列问题。

(2)$MIQP$ 问题的特例[4]

标准的 $MIQP$（混合整数二次规划问题）形式如下：

$$\min y = \frac{1}{2} x^T H x + f^T x \tag{18}$$

其中 H 是包含二次目标函数的 $n \times n$ 矩阵（二次项和双线性项），f 是包含二次目标函数的 $n \times 1$ 向量（线性项）。目标是通过选择也满足所有约束的 x 值来最小化

目标函数, 受到以下约束:

$$s.t.\begin{cases} Ax \le B \\ A_{eq}x = b_{eq} \\ u_b \le x \le v_b \\ x_i \in \mathbb{Z} \\ x_j \in (0,1), i \ne j \end{cases} \tag{19}$$

线性不等式约束中 A 是 $m \times n$ 矩阵，b 是 $m \times 1$ 向量；线性等式约束中 A_{eq} 是 $k \times n$ 矩阵, b_{eq} 是 $k \times 1$ 向量; 决策变量边界 u_b 和 v_b 是 $n \times 1$ 向量, 其中 $-inf$ 或 inf 分别表示无界下限或上限；整数约束 x_i 是决策变量，必须是整数 (…,-2,-1,0,1,2,…)；二进制约束 x_j 是决策变量，必须是二进制数 $(0,\ 1)$，其中 $i \ne j$。

当我们认为 u_b=$(0,0,\ldots,0)^T$,v_b=$(1,1,\ldots,1)^T$ 时，结合 x_i^2=x_i, 我们要求解的 $QUBO$ 可以看成 $MIQP$ 问题的特例，即:

$$\min y = x^T Q x \tag{20}$$

满足:

$$q_{ii} = \frac{1}{2}h_{ii} + f_{i1}, i = 1,2,\ldots,n \tag{21}$$

$$q_{ij} = q_{ji} = \frac{1}{2}h_{ij}, 1 \le i < j \le n \tag{22}$$

(3)$QUBO$ 模型求解

一方面，我们利用加强后的求解器多次更改惩罚系数后发现求得的最优阈值为 481 且不改变，即当第 49 张信用评分卡的阈值设置为 1 时，最终收入最大为 61172 元。另一方面，为了研究求解速度与惩罚系数的关系，我们对惩罚系数 λ 的值进行不断试验，尝试找到较优的惩罚系数，结果如下表所示:

表 3　问题 1 不同惩罚系数运行时间

惩罚系数 λ	运行时间 (s)	惩罚系数 λ	运行时间 (s)	惩罚系数 λ	运行时间 (s)
0	3.428	900	4.335	9000	5.888
100	2.566	1000	5.214	10000	6.339
200	3.706	2000	5.275	20000	6.040
300	3.704	3000	4.866	30000	6.104
400	3.849	4000	4.722	40000	6.168
500	3.957	5000	4.590	50000	6.052
600	3.961	6000	4.683	60000	6.228
700	4.099	7000	4.630	70000	6.161
800	4.226	8000	4.647	80000	6.189

通过分析上表我们发现在运行时间在整体上随着惩罚系数的增大而增大，在局部上有先增后减趋势。根据上表我们可以认为惩罚系数 λ 在区间 [0,300] 内最优，在区间 [0,8000] 内较优。

五、模型二的建立与求解

问题 2 已经选定了数据集中给出的信用评分卡 1、信用评分卡 2、信用评分卡 3 这三种规则，我们根据组合方式中信用评分卡的数量将其分为三大类进行讨论，如下表所示：

表 4　问题二信用评分卡组合

序号	组合方式
1	信用评分卡 1
2	信用评分卡 2
3	信用评分卡 3
4	信用评分卡 1，信用评分卡 2
5	信用评分卡 1，信用评分卡 3
6	信用评分卡 2，信用评分卡 3
7	信用评分卡 1，信用评分卡 2，信用评分卡 3

5.1 第一类：选定一张信用评分卡

当我们在信用评分卡 1、信用评分卡 2、信用评分卡 3 中仅选定一张信用评分卡设置阈值时，只需要考虑这张卡的十个阈值所分别带来的最终收入大小，可以划归为问题 1 的简化版，在此我们步过多进行讨论，结合问题 1 可以求出结果如下表所示：

表 5　选定一张信用评分卡的最优规则

选定的信用评分卡	设置的阈值	最大收入 (元)
信用评分卡 1	1	50130
信用评分卡 2	1	32717
信用评分卡 3	2	54087

5.2 第二类：选定两张信用评分卡

我们不妨先假设选定了信用评分卡 1 和信用评分卡 2，将两张信用卡的阈值进行两两配对共有 100 种不同的组合方式，当一个组合被选定时，其对应的最终收入也随之确定，我们期望选出一个最优的组合使得最终收入最大。

5.2.1 第二类情况的化归

我们将这些组合进行编号，类似于问题 1 的处理，编号为 k 的组合表示下信用评分卡 1 的第 i 个阈值和信用评分卡 2 的第 j 个阈值两两配对，其中

$$i = [\frac{k-1}{10}] + 1 \tag{23}$$

$$j = k - 10 \cdot [\frac{k-1}{10}] \tag{24}$$

这样处理之后就与问题 1 的建模一致，我们直接将其转化为 $QUBO$ 形式。

5.2.2 第二类的 $QUBA$ 模型

我们定义如下 0—1 决策变量：

$$x_k = \begin{cases} 1, \text{编号为 } k \text{ 的阈值组合是最优的} \\ 0, \text{其他} \end{cases} \tag{25}$$

我们只从 100 组中取一组作为我们的最优解，那么约束条件：

$$\sum_{k=1}^{100} x_k = 1 \tag{26}$$

值得注意的是，两张信用评分卡的贷款利息收入和坏账带来的坏账损失与问题一的计算有所差别，由题目信息知，两张信用评分卡的贷款利息收入为：

$$W_1 = \sum_{k=1}^{100} Mp\alpha_i\alpha_j(1 - \frac{\beta_i + \beta_j}{2})x_k \tag{27}$$

由坏账带来的坏账损失为：

$$W_2 = \sum_{k=1}^{100} M\alpha_i\alpha_j\frac{\beta_i + \beta_j}{2}x_k \tag{28}$$

其中 i，j，k 的数量关系如 (18)(19) 式所示，α_i，β_i 分别表示编号为 $[\frac{k}{10}]+1$ 的通过率和坏账率，α_j，β_j 分别表示编号为 $k - 10 \cdot [\frac{k}{10}]$ 的通过率和坏账率。

目标函数：

$$\min y = -\sum_{k=1}^{100} A_k x_k \tag{29}$$

其中 A_k=M$\alpha_i\alpha_j$(p-$\frac{(1+p)(\beta_i+\beta_j)}{2}$),i=1,2,...,100, 根据附件 1 中的数据，我们可以算出每一个 A_k 的值，最后得出 $QUBO$。

最终化为的 $QUBO$ 形式如下：

$$\min H = -\sum_{k=1}^{100} A_k x_k + \lambda(\sum_{k=1}^{100} x_k - 1)^2 \tag{30}$$

将其转化为基础的 $QUBO$ 模型：

$$\min y = x^T Q x \tag{31}$$

$$Q=\begin{bmatrix} -\lambda-A_1 & \lambda & \cdots & \lambda & \lambda \\ \lambda & -\lambda-A_2 & \ddots & \vdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \lambda \\ \lambda & \lambda & \cdots & \lambda & -\lambda-A_{100} \end{bmatrix}$$

其中，$q_{ii}=-\lambda-A_i, q_{ij}=\lambda, i,j=1,2,\ldots,100, B$ 是常数项。

5.2.3 第二类情况的模型求解

我们将 $QUBO$ 矩阵的系数输入求解器中，设置惩罚系数 $\lambda=100$，再分别求其他两种组合方式的最终收入可以得到以下结果：

方式	选定的信用评分卡	设置的阈值	最大收入 (元)
组合 1	信用评分卡 1	1	30479
	信用评分卡 2	1	
组合 2	信用评分卡 1	1	41106
	信用评分卡 3	2	
组合 3	信用评分卡 2	1	32885
	信用评分卡 3	2	

5.3 第三类：选定三张信用评分卡

5.3.1 第三类的 0—1 整数规划模型

对于三张信用评分卡均被选定的情况，我们还是先建立 0—1 整数规划模型，将 30 个阈值重新编号 1，2，…，30，信用评分卡 i 的第 j 个阈值编号为 $(10i+j)$，定义如下 0—1 决策变量：

$$x_k=\begin{cases} 1, \text{编号为 } k \text{ 的阈值被选定} \\ 0, \text{其他} \end{cases} \tag{32}$$

一张信用评分卡有且仅有一个阈值会被选中，即：

$$\sum_{i=1}^{10} x_i=1 \quad \sum_{i=11}^{20} x_i=1 \quad \sum_{i=21}^{30} x_i=1 \tag{33}$$

对于三张信用评分卡而言，由题目可知，贷款利息收入为：

$$W_1=\sum_{i=1}^{10}\sum_{j=11}^{20}\sum_{k=21}^{30} Mp\alpha_i\alpha_j\alpha_k\left(1-\frac{\beta_i+\beta_j+\beta_k}{3}\right)x_ix_jx_k \tag{34}$$

由坏账带来的坏账损失为：

$$W_2 = \sum_{i=1}^{10}\sum_{j=11}^{20}\sum_{k=21}^{30} M\alpha_i\alpha_j\alpha_k(\frac{\beta_i+\beta_j+\beta_k}{3})x_ix_jx_k \tag{35}$$

我们的目标是使得最终收入最多，目标函数：

$$\max Z = W_1 - W_2 \tag{36}$$

综上所述，建立如下的 0—1 整数规划模型：

目标函数：

$$\max Z = W_1 - W_2 \tag{37}$$

$$s.t.\begin{cases} \sum_{i=1}^{10} x_i = 1 \\ \sum_{i=11}^{20} x_i = 1 \\ \sum_{i=21}^{30} x_i = 1 \\ x_k \in (0,1), k = 1,2,\ldots,30 \end{cases} \tag{38}$$

5.3.2 第三类的 *QUBO* 初始模型

结合有约束的 *QUBO* 模型对上述 0—1 整数规划模型进行转换。首先，我们将目标函数转化为求最小值，然后将约束条件转化为惩罚函数。

目标函数：

$$\min y = -\sum_{i=1}^{10}\sum_{j=11}^{20}\sum_{k=21}^{30} A_{ijk}x_ix_jx_k \tag{39}$$

其中 A_{ijk}=$M\alpha_i\alpha_j\alpha_k(p-\frac{(1+p)(\beta_i+\beta_j+\beta_k)}{3})$,$i$=1,2,…,10;$j$=11,12,,20;$k$=21,22,…,30. 对确定的 i，j，k，A_{ijk} 是可以算出的已知常数。

设惩罚函数[2] 分别为：

$$g_1(x_1,x_2,\ldots,x_{10}) = (\sum_{i=1}^{10} x_i - 1)^2 \tag{40}$$

$$g_2(x_{11},x_{12},\ldots,x_{20}) = (\sum_{i=11}^{20} x_i - 1)^2 \tag{41}$$

$$g_3(x_{21},x_{22},\ldots,x_{30}) = (\sum_{i=21}^{30} x_i - 1)^2 \tag{42}$$

整理上述式子得到的 *QUBO* 形式如下：

$$\min H = -\sum_{i=1}^{10}\sum_{j=11}^{20}\sum_{k=21}^{30} A_{ijk}x_ix_jx_k + \lambda(\sum_{i=1}^{10} x_i - 1)^2 + \lambda(\sum_{i=11}^{20} x_i - 1)^2 + \lambda(\sum_{i=21}^{30} x_i - 1)^2 \tag{43}$$

5.3.3 *QUBO* 模型降维处理

上述 *QUBO* 模型中含有三次多项式，我们为了将其化简为最基本的 *QUBO* 形式，我们需要对这个式子进行降维处理。我们的主要思路是将三阶或更高阶中包含的两个变量的乘积替换为新变量后添加约束，并将公式作为 *QUBO* 执行。

(1)*Rosenberg* 多项式[1]

$$g(x_1,x_2,y)=3y+x_1x_2-2x_1y-2x_2y \tag{44}$$

满足：

$$\begin{cases} g(x_1,x_2,y)>0 \Longleftrightarrow \ \mathrm{y}\neq x_1x_2 \\ g(x_1,x_2,y)=0 \Longleftrightarrow \ \mathrm{y}=\mathrm{x}_1x_2 \end{cases} \tag{45}$$

对于一个含三次多项式的 *QUBO* 式子 $f(x_1,x_2,x_3)$ 求最小值，令 $y=x_1x_2$，可以表示为：

$$f(x_1,x_2,x_3,y)=Ax_3y+f'(x_1,x_2,x_3) \tag{46}$$

其中 $f'(x_1,x_2,x_3)$ 是一个只含一次多项式和二次多项式的三元函数，A 是系数。

这样处理后 $f(x_1,x_2,x_3)$ 由一个含三次多项式的三元函数变成了一个只含一次多项式和二次多项式的四元函数 $f(x_1,x_2,x_3,y)$。但同时也增加了一个约束条件：

$$y=x_1x_2 \tag{47}$$

将上面的约束条件对应的惩罚函数设为：

$$g(x_1,x_2,y)=ay+bx_1+cx_2+dx_1x_2+ex_1y+fx_2y \tag{48}$$

我们需要寻找合适的系数 (a,b,c,d,e,f) 使得当 $y=x_1x_2$ 时，$g(x_1,x_2,y)=0$；当 $y\neq x_1x_2$ 时，$g(x_1,x_2,y)>0$.

1. 当 $x_1x_2=y$ 时，期望 $g(x_1,x_2,y)=0$.

表 6　惩罚函数系数关系一

x_1	x_2	y	实际 g 值	期望 g 值
0	0	0	0	0
1	0	0	b	0
0	1	0	c	0
1	1	1	$a+b+c+d+e+f$	0

2. 当 $x_1x_2\neq y$ 时，期望 $g(x_1,x_2,y)>0$.

表 7 惩罚函数系数关系二

x_1	x_2	y	实际 g 值	期望 g 值
0	0	1	a	s_1
1	0	1	$a+b+e$	s_2
0	1	1	$a+c+f$	s_3
1	1	0	$b+c+d$	s_4

其中，s_1,s_1,s_1,s_1 都是大于 0 的常数。

综上所述可知：

$$\begin{cases} b=0 \\ c=0 \\ a+b+c+d+e+f \\ a=s_1 \\ a+b+e=s_2 \\ a+c+f=s_3 \\ b+c+d=s_4 \end{cases} \tag{49}$$

最后我们可以求得 $(s_1,s_1,s_1,s_1)=(3,1,1,1)$ 是一组可行解，在此基础上可以求得 $(a,b,c,d,e,f)=(3,0,0,1,-2,-2)$, 即

$$g(x_1,x_2,y)=3y+x_1x_2-2x_1y-2x_2y \tag{50}$$

在求得惩罚函数的具体系数后我们可以将只含一次多项式和二次多项式的四元函数 $f(x_1,x_2,x_3,y)$ 写出 *QUBO* 形式：

$$\min H=f(x_1,x_2,x_3,y)+\lambda(3y+x_1x_2-2x_1y-2x_2y) \tag{51}$$

(2) 对第三类的 *QUBO* 模型降维

利用 *Rosenberg* 多项式，可以通过增加变量和约束的方法对含三次多项式的 *QUBO* 模型进行降维，下面我们使用该方法对问题 2 第三类的 *QUBA* 初始模型进行降维处理，具体步骤如下：

***Step*1**. 变量替换

令 $x_l=x_ix_j$,$i=1,2,\cdots,10$;$j=11,12,\cdots,20$, 下标关系如下：

$$l=10i+j+10 \tag{52}$$

$$i=[\frac{l-1}{10}]-2 \tag{53}$$

$$j=l+10-10\cdot[\frac{l-1}{10}] \tag{54}$$

新增的 100 个变量依次为 x_{31},x_{32},$\cdots$,x_{130}.

***Step*2**. 加入约束条件

Step3. 寻找合适的惩罚函数系数

Step4. 获得确定系数的惩罚函数

将这三个步骤合并根据 *Rosenberg* 多项式，我们新增的约束有 100 个分别为:

$$g(x_i, x_j, x_l) = 3x_l + x_i x_j - 2x_i x_l - 2x_j x_l \tag{55}$$

Step5. 获得最高次为二次的 *QUBA* 模型[3]:

$$\min H = -\sum_{l=31}^{130}\sum_{k=21}^{30} A_{ijk} x_l x_k + \lambda_1 \sum_{i=1}^{3} (\sum_{j=10i-9}^{10i} x_j - 1)^2 + \lambda_2 \sum_{l=31}^{130} (3x_l + x_i x_j - 2x_i x_l - 2x_j x_l) \tag{56}$$

(3) 确定 *QUBO* 矩阵

我们将 (51) 式展开后结合 $x_i^2 = x_i$ 合并同类项化简为:

$$H = \sum_{i=1}^{130} q_{ii} x_i^2 + \sum_{i<j} q_{ij} x_i x_j + B \tag{57}$$

QUBO 矩阵可以用 q_{ij} 表示:

$$Q = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{(1)(30)} & q_{(1)(31)} & \cdots & \cdots & q_{1(130)} \\ q_{21} & q_{22} & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & q_{(30)(30)} & \ddots & \ddots & \ddots & q_{(30)(130)} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & q_{(31)(130)} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ q_{(130)1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & q_{(130)(130)} \end{bmatrix}$$

注意到 Q 是一个 130 阶的对称矩阵，$q_{ij} = q_{ji}(1 \le i < j \le 130)$, 其中

$$q_{ij} = \begin{cases} \lambda_1, 1 \le i = j \le 30 \\ 3\lambda_2, 31 \le i = j \le 130 \\ -\lambda_1, 1 \le i < j \le 10; 11 \le i < j \le 20; 21 \le i < j \le 30 \\ -\lambda_2, 1 \le i \le 20, 31 \le j \le 130 \\ \frac{1}{2}\lambda_2, 1 \le i \le 10, 11 \le j \le 20 \\ -\frac{1}{2}A_{([\frac{j-1}{10}]-2)(j+10-10\cdot[\frac{j-1}{10}])i}, 21 \le i \le 30, 31 \le j \le 130 \\ 0, \text{其他} \end{cases} \tag{58}$$

5.3.4 惩罚系数试验

借助 SCIP 求解器，经过多次求解，我们得到当 x_8=x_{11}=x_{22} 时有最大收入，即三张信用评分卡阈值分别设置为 8，1，2 最大收入为 27915 元。

在求解过程中发现，惩罚系数 λ_1,λ_2 的取值会直接影响求解的时间和结果，并非每一次都可以得到正确结果。我们希望找到合适的 λ_1,λ_2 使得求解结果精确的同时提高求解的速度。

下面分两种情况讨论：

(1) 讨论 λ_1=λ_2 的情况

测试结果如下表所示：

表 8　第三类模型惩罚系数测试表

惩罚系数 λ	运行时间 (s)	运行结果	惩罚系数 λ	运行时间 (s)	运行结果
0	11.302	(8,1,1)	75000	17.994	(8,1,6)
5000	12.611	(8,1,5)	80000	17.247	(8,1,2)
10000	11.965	(8,1,4)	85000	40.192	(8,1,2)
15000	15.647	(8,1,9)	90000	17.379	(8,1,9)
20000	15.624	(8,1,7)	95000	33.714	(8,1,7)
25000	41.401	(8,1,5)	100000	41.824	(8,1,9)
30000	52.937	(8,1,6)	105000	38.756	(8,1,4)
35000	31.548	(8,1,3)	110000	10.37	(8,1,7)
40000	29.221	(8,1,1)	115000	39.854	(8,1,9)
45000	29.088	(8,1,3)	120000	21.113	(8,1,9)
50000	60.304	(8,1,1)	200000	13.04	(8,1,2)
55000	45.405	(8,1,4)	300000	21.175	(8,1,8)
60000	17.407	(8,1,2)	400000	22.622	(8,1,9)
65000	22.183	(8,1,1)	500000	16.834	(8,1,2)
70000	30.32	(8,1,6)	100000	32.319	(8,1,3)

根据上表可知，当 λ 小于 60000 时，测试值中没有任何一个值能够求出正确结果；当 λ 取 60000，80000，85000，200000，500000 时，我们可以得出正确的结果；而当 λ 取 85000 时运行时间显著高于其他三个取值，故排出其较优惩罚系数的可能。

(2) 讨论 λ_1≠λ_2 的情况

测试结果如下表所示：

表 9　第三类模型惩罚系数测试表

惩罚系数 λ_1	惩罚系数 λ_2	运行时间 (s)	运行结果
0	0	11.302	(8,1,1)
0	20000	68.976	(8,1,9)
0	40000	59.409	(8,1,7)
0	60000	40.439	(8,1,4)
0	80000	31.132	(8,1,6)
20000	0	2.388	(8,1,1)
20000	20000	11.666	(8,1,7)
20000	40000	15.175	(8,1,3)
20000	60000	35.811	(8,1,5)
20000	80000	19,916	(8,1,1)
40000	0	6.674	(8,1,9)
40000	20000	11.966	(8,1,5)
40000	40000	22.937	(8,1,1)
40000	60000	13.07	(8,1,9)
40000	80000	24.593	(8,1,6)
60000	0	2.679	(8,1,3)
60000	20000	13.986	(8,1,2)
60000	40000	13.172	(8,1,7)
60000	60000	17.407	(8,1,2)
60000	80000	27.409	(8,1,3)
80000	0	6.441	(8,1,9)
80000	20000	39.417	(8,1,7)
80000	40000	12.303	(8,1,7)
80000	60000	25.056	(8,1,3)
80000	80000	17.247	(8,1,2)

根据上表可知，当 λ_1 小于 60000 时，无论 λ_2 取何值都不能得出正确结果，说明惩罚函数 g_1 的惩罚强度远小于 g_2，在惩罚系数的选取上应该满足 λ_1 大于 λ_2; 而当 (λ_1,λ_2) 取 (60000,20000),(60000,60000),(80000,80000) 时，我们可以得出正确的结果。

考虑到当惩罚值过大会阻碍解决过程，因为惩罚项压倒了原始目标函数的信息，使得难以区分一个解和另一个解的质量，不利于区分不同信用评分卡的不同阈值的组合之间的差异；当惩罚值太小会危及寻找可行解的过程，导致最终无法找到不同信用评分卡的不同阈值的组合[3]。

综上所述，我们可以认为较优的惩罚系数为:(λ_1,λ_2)=(60000,20000)。

5.3.5 第三类模型的检验

(1) 仿照第二类的模型

考虑第二类建立的 QUBO 模型，这是以阈值组合为主体设定的决策变量，我们以同样的方法对第三类进行建模，验证结果的准确性并比较两种方。从每张信用卡的阈值中选取分别一个阈值组成一个新的三元组 $card(i,j,k)$, 其中 i，j，k 分别表示信用评分卡 1 的阈值 i，信用评分卡 2 的阈值 j，信用评分卡 3 的阈值 k。这样的 $card(i,j,k)$ 有 1000 组，将其编号为 $card_1$,$card_2$,…,$card_{1000}$, 我们只需要选出一个最优的 $card$，使得收入最大。定义 0—1 决策变量：

$$x_l = \begin{cases} 1, card_l \text{ 是最优的} \\ 0, \text{其他} \end{cases} \tag{59}$$

其中

$$l = 100(i-1) + 10(j-1) + k, i, j, k = 1, 2, \ldots, 10 \tag{60}$$

我们借鉴第二类的解法可以求出目标函数：

$$\min y = -\sum_{l=1}^{1000} A_l x_l \tag{61}$$

其中 A_l=M$\alpha_i\alpha_j\alpha_k$(p-$\frac{(1+p)(\beta_i+\beta_j+\beta_k)}{3}$),$i$,$j$,$k$=1,2,…,10, 根据附件 1 中的数据，我们可以算出每一个 A_l 的值，最后得出 $QUBO$。

$QUBO$ 形式如下：

$$\min H = -\sum_{l=1}^{1000} A_l x_l + \lambda(\sum_{l=1}^{1000} x_l - 1)^2 \tag{62}$$

将其转化为基础的 $QUBO$ 模型：

$$\min y = x^T Q x \tag{63}$$

$$Q = \begin{bmatrix} -\lambda - A_1 & \lambda & \cdots & \lambda & \lambda \\ \lambda & -\lambda - A_2 & \ddots & \vdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \lambda \\ \lambda & \lambda & \cdots & \lambda & -\lambda - A_{100} \end{bmatrix}$$

其中，q_{ii}=$-\lambda - A_i$,i=1,2,…,1000,q_{ij}=q_{ji}=λ,$1 \le i \le 1000$,B 是常数项。

(2) 模型对比分析

我们仿照问题 1 的求解过程可以解出 $card_{702}$ 是最优的，由 (60) 式的转化关系可知 $card(8,1,2)$ 对应的收入最大，而由问题 1 的分析我们知道该 $QUBO$ 模型下惩罚函数的大小对结果没有影响，所以我们认为当信用评分卡 1，信用评分卡 2，信用

评分卡 3 的阈值分别设定为 8，1，2 时，最终收入最大为 27915 元，这与第三类的 *QUBO* 模型求解结果是一致的。

通过对比我们发现第二类的 *QUBO* 模型是以阈值组合为主体进行建模，其优点在于可以用较少的约束条件和较低的维度去控制 *QUBO* 模型，惩罚系数对结果的影响较小；但其缺点也非常明显，决策变量的数量倍数级增长，大大降低了运行速度。第三类的 *QUBO* 模型是以单个阈值为主体进行建模，其优点在于决策变量的数目容易控制，但是 *QUBO* 可能含有三次甚至三次以上的多项式，需要进行降维处理；与此同时，惩罚系数的选取需要通过大量的检验。在需要设置的阈值较少时，以阈值组合为主体进行建模有很高的准确度和简便性；而在需要设置的阈值较多时，以单个阈值为主体进行建模会有更好的适用度和可行性。

六、模型三的建立与求解

6.1 化归为问题 2 第三类

对附件 1 中的 100 张信用评分卡,我们求出所有的三张信用评分卡组合 $group(i,j,k)$, 其中 $1 \le i < j < k \le 100$, 通过组合计数原理这样的组合有 C_{100}^{3}=161700 组。

对固定的 i，j，k 而言，我们将其化归为问题 2 的第三类:

$$group(i,j,k) \Longrightarrow group(1,2,3) \tag{64}$$

考虑到以单个阈值为主体的 *QUBO* 模型需要设置合适的惩罚系数来确保结果的准确性。所以我们采用以阈值组合为主体的 *QUBO* 模型，并设定惩罚系数 $\lambda = 100$ 进行求解，这在可行范围内最大程度减少单次运算的时间。借助 matlab 和 SCIP 求解器算出的结果我们用一个 7×1 数组表示：

$$Outcom = (i,j,k,y_i,y_j,y_k,r_{ijk}) \tag{65}$$

其中 y_i,y_j,y_k 分别表示 $group(i,j,k)$ 下三张卡的最优阈值,r_{ijk} 表示 $group(i,j,k)$ 下的最大收入。

最后，我们只需要比较 r_{ijk} 的大小就可以求出最优的信用评分卡和阈值组合。

6.2 问题 3 的模型求解

我们在 matlab 和 SCIP 求解器中插入一个求解循环 l=1，2，...,161700.

计算顺序以此类推：$group(1,1,1) \Rightarrow group(1,1,2) \Rightarrow \ldots \Rightarrow group(1,1,100) \Rightarrow group(1,2,3) \Rightarrow group(1,2,4) \Rightarrow \ldots \Rightarrow group(1,100,100) \Rightarrow group(2,3,4) \Rightarrow group(2,3,5) \Rightarrow \ldots \Rightarrow group(97,98,99) \Rightarrow group(97,98,100) \Rightarrow group(97,99,100) \Rightarrow group(98,99,100)$.

根据附件 1 中的数据，*QUBO* 矩阵不断更新得出新的 *Outcom* 并保存到数据库，最后求得 $Outcom = (8,33,49,2,6,3,43881)$，即信用评分卡 8 阈值设置为 2，信用评分卡 33 阈值设置为 6，信用评分卡 49 阈值设置为 3 时最终收入最大为 43881 元。

七、模型的评价与改进

7.1 模型的优点

1、对惩罚系数的设置进行了分析，通过大量试验得到了最优惩罚系数的大致区间，同时增加了惩罚函数的数量，提高了模型的灵活度与求解的精确度。

2、利用 *Rosenberg* 多项式进行降维处理，有效的减小了 *QUBO* 模型逻辑上的复杂性。

3、借助 SCIP 求解器进行求解，只需要求出 *QUBO* 矩阵各个元素的值即可求出结果，简便快捷，普及性较好。

4、我们对以单个阈值为主体以及以阈值组合为主体两种方法分别进行 *QUBO* 模型的建立并比较，发现前者的 *QUBO* 矩阵规模比后者多了一个数量级，后者求解速度更快，在实际情况中，信用评分卡的阈值组合会变得更加的复杂，由此可知我们对单个阈值为主体的 *QUBO* 模型的适用性比较好。

7.2 模型的缺点

1、*QUBO* 矩阵的确定对代数要求较高，需要精确计算出每个元素的值。

2、对含三次多项式 *QUBO* 模型，需要降维而引入新的变量，从而增大 *QUBO* 矩阵的规模，降低了求解的速度，当信用评分卡数量较大时，降维产生的变量数目会难以控制。

3、合适的惩罚系数需要大量的试验确定，对惩罚系数的分析设定没有充分的数据支撑。

4. 当要求所选定的信用评分卡阈值数量以及目标信用评分卡数量变复杂时，我们计算所有组合的时间会变得十分漫长甚至不可控制。

7.3 模型的改进

1、可以通过引入松弛变量减少分类，提高模型的适用性和整体性。

2、对于惩罚系数进行更多次的试验，找到最优的惩罚系数，保证求解的准确性。

3、对信用评分卡组合实际场景过于简化，可以通过增加新的影响因素来考虑更复杂的情形，进而将 *QUBO* 模型改进为 *MIQP* 模型来满足实际信用评分卡的组合优化场景。

参考文献

[1] 久保 · 穗高.《使用优化问题和 Wildqat 进行量子退火计算的介绍。》[M]. 第二版. 日本共立出版社, 2020.

[2] Lewis, M.; Glover, F. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. Networks 2017, 70, 79–97.

[3] Papalitsas, C.; Andronikos, T.; Giannakis, K.; Theocharopoulou, G.; Fanarioti, S. A QUBO Model for the Traveling Salesman Problem with Time Windows. Algorithms 2019, 12, 224.

[4] OPTI Toolbox.Mixed Integer Quadratic Program (MIQP)[EB/OL].[2023-4-16].https://www.controlengineering.co.nz/.

附 录

附录 1：问题 1 的 MATLAB 代码

```
clear,clc
M=1000000;
p=0.08;
T=readtable('附件1：data_100.csv',"VariableNamingRule","preserve");
data=[T{:,:}];
Flag=[];
Fval=[];
for k=1:2:size(data,2)
    % 目标函数系数
    for i=1:size(data,1)
        W(i)=-M*(p*data(i,k)-p*data(i,k)*data(i,k+1)-data(i,k)*data(i,k+1));
    end
    n=length(W);
    A = ones(1,n);
    b = 1;
    lb=zeros(1,n);
    % 惩罚项系数（需要根据实际问题进行调整）
    penalty = 80000;
    % 将整数规划问题转换为QUBO问题
    Q = zeros(n);
    c = sum(W);
    m = numel(W);
for ii =1:n
    for jj = ii:n
        if ii == jj
            Q(ii,jj) = W(ii)+penalty*(A(ii)*A(ii)-2*A(ii)*b);
        else
             Q(ii,jj) = penalty*A(ii)*A(jj);
              Q(jj,ii) = penalty*A(ii)*A(jj);
        end
    end
end
H = Q;
f=zeros(1,n);
xtype = char(join( repmat("B",1,n),""));
Opt = opti('qp',H,f,'ineq',A,b,'lb',lb,'xtype',xtype);
% Solve the MIQP problem
[x,fval,exitflag,info] = solve(Opt) ;
fval=-1*fval;
Fval=[Fval,fval];
```

```
flag=1;
for i=1:n
    if(x(i)==1)
        flag=i;
        break
```

附录 2: 问题 2 的 MATLAB 代码

(1)以阈值组合为主体的QUBO模型代码:

```
clear,clc
M=1000000;
q=0.08;
T=readtable('附件1: data_100.csv',"VariableNamingRule","preserve");
data=[T{:,:}];
Flag=[];
Fval=[];
%设定惩罚系数
penalty=30000;
%计算Q矩阵
Q=penalty*ones(1000);
n=1000;
for i=1:10
    for j=1:10
        for k=1:10
            Q((i-1)*100+(j-1)*10+k,(i-1)*100+(j-1)*10+k)=-penalty-M*data(i,1)*
                data(j,3)*data(k,5)*(q-(1+q)*(data(i,2)+data(j,4)+data(k,6))/3);
        end
    end
end
% Solve the MIQP problem
for t=1:10
A = ones(1,n/10);
H=zeros(n/10);
b = 1;
lb=zeros(1,n/10);
for i=1:100
    for j=1:100
        H(i,j)=Q((t-1)*100+i,(t-1)*100+j);
    end
end
f=zeros(1,n/10);
xtype = char(join( repmat("B",1,n/10),""));
Opt = opti('qp',H,f,'ineq',A,b,'lb',lb,'xtype',xtype);
% Solve the MIQP problem
```

```
[x,fval,exitflag,info] = solve(Opt);
fval=-1*fval;
Fval=[Fval,fval];
flag=1;
for i=1:n
    if(x(i)==1)
        flag=i;
        break
    end
end
flag=t*100+flag+10;
Flag=[Flag,flag];
end
t=max(Fval);
index=find(max(Fval)==Fval);
c_i=index;
c_j=fix((Flag(index)-index*100)/10);
c_k=Flag(index)-c_i*100-c_j*10;
z=M*q*data(c_i,1)*data(c_j,3)*data(c_k,5)*(1-1/3*(data(c_i,2)+data(c_j,4)+data(
    c_k,6)))-M*data(c_i,1)*data(c_j,3)*data(c_k,5)*(1/3*(data(c_i,2)+data(c_j
    ,4)+data(c_k,6)));
disp('选择信用卡1阈值为：');
disp(c_i);
disp('选择信用卡2阈值为：');
disp(c_j);
disp('选择信用卡3阈值为：');
disp(c_k);
disp('银行收入最多为：');
disp(z);
```

(2)以单个阈值为主体的QUBO模型代码：

```
clear,clc
M=1000000;
q=0.08;
T=readtable('附件1：data_100.csv',"VariableNamingRule","preserve");
data=[T{:,:}];
Flag=[];
%设定惩罚系数
P1=15000;
P2=15000;
%计算Q矩阵
Q=zeros(130);
for i=1:30
for j=1:30
if i==j
```

```
Q(i,i)=P1;
else
if i<=10&&j<=10
Q(i,j)=-P1;
else
if i<10&&j<=20&&j>10
Q(i,j)=P2/2;
else
if j<10&&i<=20&&i>10
Q(i,j)=P2/2;
else
if i>10&&i<=20&&j>10&&j<=20
Q(i,j)=-P1;
else
if i>20&&j>20
Q(i,j)=-P1;
end
end
end
end
end
end
end
end

for i=1:20
for j=31:130
Q(i,j)=-P2;
Q(j,i)=-P2;
end
end

for i=31:130
Q(i,i)=3*P2;
end

for i=21:30
for l=31:130
Q(i,l)=1/2*M*data(fix((l-31)/10)+1,1)*data(l-10*fix((l-1)/10),3)*data(i-20,5)*(
    q-(1+q)*(data(fix((l-31)/10)+1,2)+data(l-10*fix((l-1)/10),4)+data(i-20,6))
    /3);
Q(l,i)=Q(i,l);
end
end
```

```
%设定约束
A = zeros(5,130);
for i=1:10
A(1,i)=1;
end
for i=11:20
A(2,i)=1;
A(5,i)=10*i;
end
for i=21:30
A(3,i)=1;
A(5,i)=i;
end
for i=31:130
A(4,i)=1;
A(5,i)=-i;
end
b = [1;1;1;1;100];
lb=zeros(1,130);
H = -Q;
f=zeros(1,130);
xtype = char(join( repmat("B",1,130),""));
Opt = opti('qp',H,f,'eq',A,b,'lb',lb,'xtype',xtype);
% Solve the MIQP problem
[x,fval,exitflag,info] = solve(Opt)
for i=1:30
    if(x(i)>0.99)
        Flag=[Flag,i];
    end
end
Flag
c_k=Flag(1);
c_i=Flag(2)-10;
c_j=Flag(3)-20;
z=M*q*data(c_i,1)*data(c_j,3)*data(c_k,5)*(1-1/3*(data(c_i,2)+data(c_j,4)+data(
    c_k,6)))-M*data(c_i,1)*data(c_j,3)*data(c_k,5)*(1/3*(data(c_i,2)+data(c_j
    ,4)+data(c_k,6)));
disp('选择信用卡1阈值为：');
disp(c_i);
disp('选择信用卡2阈值为：');
disp(c_j);
disp('选择信用卡3阈值为：');
disp(c_k);
disp('银行收入最多为：');
```

```
disp(z);
```

附录 3：问题 3 的 MATLAB 代码

```
clear,clc
M=1000000;
q=0.08;
T=readtable('附件1：data_100.csv',"VariableNamingRule","preserve");
data=[T{:,:}];
%依次遍历，也可设计固定值进行计算
for B_i=1:98
for B_j=B_i+1:99
for B_k=B_j+1:100
Flag=[];
Fval=[];
penalty=30000;
Q=penalty*ones(1000);
n=1000;
for i=1:10
    for j=1:10
        for k=1:10
            Q((i-1)*100+(j-1)*10+k,(i-1)*100+(j-1)*10+k)=-penalty-M*data(i,B_i
                *2-1)*data(j,B_j*2-1)*data(k,B_k*2-1)*(q-(1+q)*(data(i,B_i*2)+
                data(j,B_j*2)+data(k,B_k*2))/3);
        end
    end
end
for t=1:10
A = ones(1,n/10);
H=zeros(n/10);
b = 1;
lb=zeros(1,n/10);
for i=1:100
    for j=1:100
        H(i,j)=Q((t-1)*100+i,(t-1)*100+j);
    end
end
f=zeros(1,n/10);
xtype = char(join( repmat("B",1,n/10),""));
Opt = opti('qp',H,f,'ineq',A,b,'lb',lb,'xtype',xtype);
% Solve the MIQP problem
[x,fval,exitflag,info] = solve(Opt);
fval=-1*fval;
Fval=[Fval,fval];
```

```
flag=1;
for i=1:n
    if(x(i)==1)
        flag=i;
        break
    end
end
flag=t*100+flag+10;
Flag=[Flag,flag];
end
t=max(Fval);
index=find(max(Fval)==Fval);
c_i=index;
c_j=fix((Flag(index)-index*100)/10);
c_k=Flag(index)-c_i*100-c_j*10;
z=M*q*data(c_i,B_i*2-1)*data(c_j,B_j*2-1)*data(c_k,B_k*2-1)*(1-1/3*(data(c_i,
    B_i*2)+data(c_j,B_j*2)+data(c_k,B_k*2)))-M*data(c_i,B_i*2-1)*data(c_j,B_j
    *2-1)*data(c_k,B_k*2-1)*(1/3*(data(c_i,B_i*2)+data(c_j,B_j*2)+data(c_k,B_k
    *2)));
disp('信用卡1对应的卡号为:');
disp(B_i);
disp('信用卡2对应的卡号为:');
disp(B_j);
disp('信用卡3对应的卡号为:');
disp(B_k);
disp('选择信用卡1阈值为: ');
disp(c_i);
disp('选择信用卡2阈值为: ');
disp(c_j);
disp('选择信用卡3阈值为: ');
disp(c_k);
disp('银行收入最多为: ');
disp(z);
end
end
end
```