计算金融与仿真

课程论文

论文题目: 计算金融与仿真

学生姓名: 李晶晶　张璐　朱冯婧

指导老师: 邓智斌

1. 投资组合选择介绍

P2P 借贷 (peer-to-peer lending lending) 是一种个人之间直接借贷的方式, 通过在线平台进行, 无需传统银行中介. 这种创新的融资方式提供了一个市场, 借款人可以向多个贷方申请贷款, 贷方可以选择为这些贷款提供全部或部分资金. 这一模式利用技术手段简化了借贷过程, 通常为借款人带来更低的利率, 同时为投资者提供更高的回报.

目前, 全球几大平台主导着 P2P 借贷领域, 主要包括 LendingClub, Prosper 和 Funding Circle 等. 这类平台通过评估贷款项目和借款人的 FICO 分数, 贷款金额和期限, 借款人的资产, 债务状况, 就业类型等数据, 为每笔贷款提供风险评级. 其中, LendingClub 是美国最大的 P2P 借贷平台之一, 提供个人贷款和投资机会. 本文使用来自 LendingClub 的公共数据集（涵盖 2007–2018 年的贷款记录）, 将贷款属性转化为违约概率作为后续建模依据.

为了实现贷款投资组合的多元化, 投资者可依据自身的风险偏好从不同风险等级的贷款中进行选择. 与传统金融机构贷款不同, P2P 贷款中的每位个人投资者其可用于投资的资金较为有限. 因此, 如何在资金约束下, 基于借款人的风险特征评估潜在回报与风险, 选择合适的贷款项目, 并进行最优资金配置, 是 P2P 投资者亟需解决的问题.

大量学者针对 P2P 网络贷款的投资组合优化问题开展了深入研究. Wan 等人将贷款投资组合决策转化为一个在特定时点下实现收益最大化与风险最小化的优化问题, 并引入混合治愈模型 (mixture cure model,MCM) 以提升投资效果, 构建实例驱动的模型对投资者的组合决策进行优化 [1]. Guo 等人则提出一种基于实例的 P2P 投资组合决策模型, 从风险最小角度出发, 利用 LendingClub 和 Prosper 数据集实现投资组合配置优化 [2]. Ajay 等人则将该问题转化为多目标优化模型, 在计算贷款相似度时将期望值框架与传统核方法结合, 优化结果优于既有模型 [3].

贷款投资组合优化（Loan Portfolio Optimization）与传统投资组合优化具有相似性, 但更关注与贷款相关的特定风险, 例如借款人信用评分, 贷款金额, 借款用途等因素引发的借款人风险, 违约风险以及利率波动风险. 优化目标是通过合理配置不同借款人贷款项目, 在控制风险的同时实现收益最大化.

在风险量化方面, VaR（Value at Risk）与 CVaR（Conditional Value at Risk）被广泛用于衡量投资组合的风险水平. 其中, VaR 表示在置信水平 α 下, 金融资产或组合在未来特定持有期内的最大可能损失；而 CVaR 则刻画了超过 VaR 截止点的极端损失的期望, 即对尾部风险的加权平均, 是更稳健的风险衡量指标. 在贷款组合优化中, CVaR 能够帮助投资者更有效地控制整体尾部损失风险, 从而实现收益与风险的权衡优化.

当然可以。下面是润色并拓展后的内容，语言风格更加自然、学术性强，更符合人类撰写的论文习惯。内容涵盖了 VaR/CVaR 方法的引入动因、三类模型的演化关系与比较，增强了段落间的逻辑衔接性，并保留了完整的 LaTeX 结构。

—

2. 模型建立

2.1. 风险度量方法简介：VaR 与 CVaR

在实际投资决策中，单纯追求期望收益往往难以满足风险控制的需求。尤其是在 P2P 贷款等高风险资产配置中，投资者更为关注潜在损失的严重程度以及发生的概率。因此，在构建优化模型时，引入有效的风险度量工具是必要的。本文主要采用的两种风险度量方法为 **风险价值（Value at Risk, VaR）**和 **条件风险价值（Conditional Value at Risk, CVaR）**，它们在金融风险管理中具有广泛的应用。

VaR（Value at Risk） 是对未来某一给定置信水平 β 下，最大可能损失的估计。形式上，若将投资组合的损失视为随机变量 L，则 VaR 定义为：

$$\mathrm{VaR}_\beta(L) = \inf\{\eta \in \mathbb{R} \mid \mathbb{P}(L \le \eta) \ge \beta\},$$

即在置信水平为 β 时，损失超过 VaR_β 的概率不超过 $1-\beta$。

CVaR（Conditional Value at Risk） 则在 VaR 的基础上进一步刻画了尾部损失的严重程度。其定义为在损失超过 VaR 的条件下的期望损失，具体表达为：

$$\mathrm{CVaR}_\beta(L) = \mathbb{E}[L \mid L \ge \mathrm{VaR}_\beta(L)].$$

相比 VaR，CVaR 更加保守，能够提供关于极端风险更全面的视角，尤其适用于风险厌恶型投资者的决策模型中。

决策变量

$$x_i \in \{0,1\}, \quad i = 1,2,\ldots,N$$

其中：

- 若 $x_i = 1$，表示选择资助第 i 个贷款对象；
- 若 $x_i = 0$，表示不予资助。

参数说明

- A_i：第 i 个贷款的申请金额；
- r_i：第 i 个贷款的年利率；
- P_i：第 i 个贷款的违约概率；
- B：可用于投资的总预算；
- $R_{\max}$：可承受的最大违约风险值；
- G_k：第 k 类信用等级对应的贷款集合；
- α_k：第 k 类信用等级贷款可投资金额占比的上限；
- m：最多可选择的贷款项目数量（即 Top-m）。

模型一：期望收益最大化模型（P2P）

$$
\begin{aligned}
\max_{x_i\in\{0,1\}} \quad & \sum_{i=1}^{N} x_i A_i r_i (1-P_i) \\
\text{s.t.} \quad & \sum_{i=1}^{N} x_i A_i \le B, \\
& \sum_{i\in G_k} x_i A_i \le \alpha_k B, \quad \forall k, \\
& \sum_{i=1}^{N} x_i A_i P_i \le R_{\max}, \\
& \sum_{i=1}^{N} x_i \le m.
\end{aligned}
\tag{P2P}
$$

该模型较为基础，适用于对风险容忍度较高的投资者，其主要目标是最大化收益期望值，而非控制损失的分布或极值。

模型二：引入 VaR 约束的风险控制模型（P2P-VaR）

$$
\begin{aligned}
\max_{x_i\in\{0,1\}} \quad & \sum_{i=1}^{N} x_i A_i r_i (1-P_i) \\
\text{s.t.} \quad & \sum_{i=1}^{N} x_i A_i \le B, \\
& \sum_{i\in G_k} x_i A_i \le \alpha_k B, \quad \forall k, \\
& \sum_{i=1}^{N} x_i A_i \cdot \tilde{P}_i^{(s)} - \eta \le \mathcal{M} z_s, \quad \forall s = 1,\dots,S, \\
& \sum_{s=1}^{S} z_s \le (1-\beta) S, \\
& \sum_{i=1}^{N} x_i \le m, \\
& z_s \in \{0,1\}, \quad \eta \in \mathbb{R}.
\end{aligned}
\tag{P2P-VaR}
$$

VaR 约束的引入，使得模型能在整体期望收益不显著下降的前提下，对极端损失发生的频率进行限制，提高了组合的稳健性。

模型三：引入 CVaR 约束的稳健优化模型（P2P-CVaR）

$$
\begin{aligned}
\max_{x_i\in\{0,1\}} \quad & \sum_{i=1}^{N} x_i A_i r_i (1-P_i) \\
\text{s.t.} \quad & \sum_{i=1}^{N} x_i A_i \le B, \\
& \sum_{i\in G_k} x_i A_i \le \alpha_k B, \quad \forall k, \\
& \xi_s \ge \sum_{i=1}^{N} x_i A_i \cdot \tilde{P}_i^{(s)} - \eta, \quad \forall s = 1,\dots,S, \\
& \eta + \frac{1}{S(1-\beta)} \sum_{s=1}^{S} \xi_s \le \mathrm{CVaR}_{\max}, \\
& \sum_{i=1}^{N} x_i \le m, \\
& \xi_s \ge 0, \quad \eta \in \mathbb{R}.
\end{aligned}
\tag{P2P-CVaR}
$$

相比于模型二，该模型不仅关注极端损失发生的概率，更进一步衡量其可能带来的财务影响，特别适用于风险厌恶程度较高的投资者或高安全性需求的决策场景。

模型演化小结

综上所述，本文提出的三个模型构成了一个从基础到稳健、从收益导向到风险控制逐步增强的优化框架，具体特征如下：

- **模型一（P2P）**：仅考虑期望收益最大化，适用于高风险偏好场景；
- **模型二（P2P-VaR）**：在保障期望收益的同时，控制损失超过阈值的概率；
- **模型三（P2P-CVaR）**：进一步引入 CVaR 约束，控制尾部损失的平均程度，实现更高水平的风险稳健性。

投资者可根据自身的风险偏好与投资策略，选择相应的模型进行组合优化，以实现收益与风险的合理平衡。

3. 算法设计

3.1. 粒子群优化算法（PSO）

粒子群优化算法（Particle Swarm Optimization, PSO）是一种基于群体智能的随机优化方法，由 Kennedy 和 Eberhart 于 1995 年提出 [4]。该算法受鸟群觅食行为的启发，模拟粒子在解空间中基于个体经验与群体经验协同搜索最优解的过程。

在 PSO 中，每个粒子代表一个潜在解，其状态由位置与速度两个向量描述。粒子在搜索过程中根据个体历史最优位置（Personal Best, $pbest$）和群体历史最优位置（Global Best, $gbest$）共同指导其移动方向与速度调整。具体的更新公式如下：

$$v_i^{(t+1)} = w \cdot v_i^{(t)} + c_1 \cdot r_1 \cdot (pbest_i - x_i^{(t)}) + c_2 \cdot r_2 \cdot (gbest - x_i^{(t)}),$$
$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)},$$

其中，$x_i^{(t)}$ 与 $v_i^{(t)}$ 分别表示第 i 个粒子在第 t 代的位置信息与速度；w 为惯性权重，控制粒子搜索的稳定性；c_1 和 c_2 分别为个体学习因子与群体学习因子；r_1 和 r_2 为服从 $[0, 1]$ 区间均匀分布的随机数，用于增强算法的随机探索能力。

PSO 算法的优点在于结构简单、参数少、全局搜索能力强，尤其适用于复杂非凸、不可导甚至离散的组合优化问题。但其缺点是缺乏精确性与最优性保证，通常作为启发式方法用于生成可行解或局部最优解。

3.2. PSO 与 Gurobi 联合优化框架

在本研究中，我们结合粒子群优化算法的全局搜索能力与 Gurobi 求解器的精确优化能力，构建了一个两阶段优化框架，如图 1 所示。

1. **第一阶段：粒子群优化（PSO）生成可行解**。通过 PSO 对贷款选择向量进行初始化，利用其高效的全局搜索能力在可行域内探索期望收益较高的可行解，形成多个高质量的候选解。
2. **第二阶段：Gurobi 精确优化求解（MIP）**。将 PSO 所得到的最优解作为 warm-start 初始化输入至 Gurobi，进一步在严格的整数规划模型下进行精细求解，以获得满足全部约束条件的最优解或近似最优解。

该框架在实际运行中表现出良好的效率与稳定性：PSO 能在大规模组合空间中迅速收敛至高质量区域，Gurobi 则能够在此基础上进一步提升解的精度与稳定性。两者的协同配合弥补了启发式算法全局性与精确算法局部性之间的差距，实现了“粗定位 + 精优化”的一体化求解策略。

为提升整体求解速度，我们还设置了以下加速机制：

- 基于评分的粒子位置解码机制，将粒子实数编码映射为可行的 0-1 向量；
- 引入精英保留策略，避免 Gurobi 从质量较差的 PSO 解开始；
- 在 PSO 阶段提前剔除显著劣解，减少无效迭代与冗余计算；
- Gurobi 求解中采用时间上限、MIPGap 等参数控制器以提高求解效率。

总体而言，PSO+Gurobi 联合框架充分结合了群体智能与数学规划的优势，为解决复杂约束下的贷款组合优化问题提供了高效、稳健的算法支持。

4. 案例研究

4.1. 数据集描述

本研究使用的数据集来自 Lending Club 平台, 原始数据由 Kaggle 网站https://www.kaggle.com/datasets/wordsforthewise/lending-club公开提供. Lending Club 是美国最大的网

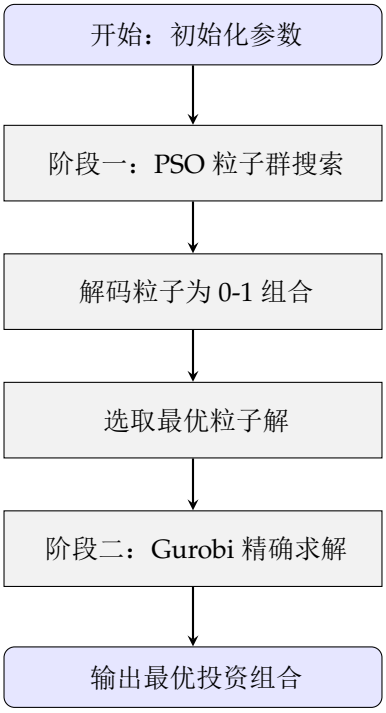

Figure 1　PSO 与 Gurobi 联合优化框架图（带表情提示）

络借贷平台之一, 提供了详尽的个人借款申请及其还款情况的数据, 广泛应用于学术界和工业界进行信贷风险评估, 违约预测及投资组合优化等研究.

该数据集包含了从 2007 年至 2018 年的借款记录, 共计数百万条样本. 每条记录对应一笔贷款申请, 涵盖了包括贷款金额, 利率, 借款人信用等级, 债务收入比, 贷款期限, 还款状态, 就业年限, 收入, 地址状态, 房屋所有权, FICO 评分区间等在内的多维度信息.

在本研究中, 我们主要筛选并保留以下变量用于建模分析：

- loan_amnt: 借款人申请的贷款金额, 作为 A_i；
- int_rate: 借款合同中约定的年利率, 用于计算收益率 r_i；
- grade: 借款人的信用等级（A 至 G）, 用于分组限制；
- loan_status: 实际贷款的还款状态（如 Fully Paid, Charged Off）, 用于推断违约情况；
- annual_inc: 借款人年收入；
- dti: 债务收入比, 用于辅助风险刻画；
- term: 贷款期限（如 36 months 或 60 months）；
- emp_length: 借款人工作年限；
- addr_state: 借款人所在州；
- fico_range_high, fico_range_low: 借款人 FICO 信用评分区间；

为了满足模型中对违约概率 P_i 的需求, 我们将 “loan_status” 字段中状态为 “Charged Off” 的贷款视为违约样本, 其余如 “Fully Paid”, “Current” 等状态作为非违约样本, 并基于历史频率法估算每一类贷款的违约概率.

此外, 为模拟贷款违约的风险场景, 我们以借款人的信用等级, FICO 评分和历史违约频率为依据, 构建了 S 个 Monte Carlo 风险场景, 用于后续 CVaR 优化模型的风险评估.

通过上述数据处理步骤, 最终形成了一个结构规范, 信息完备, 适用于组合优化问题的数据集, 为后续实证分析与建模提供了坚实的数据基础.

Table 1　三类投资优化模型的主要参数设置

参数符号	含义	适用模型
B	投资总预算，1×10^8 元	全部
m	最多选择的贷款数量（Top-m），5000	全部
P_i	贷款 i 的违约概率，$P_i\in[0,1]$	全部
$R_{\max}$	风险容忍值上限，1.5×10^7 元	原始模型
$\mathrm{CVaR}_{\max}$	期望风险容忍值上限，1.5×10^7 元	CVaR
β	置信水平（VaR、CVaR），0.95	VaR ，CVaR
η	VaR 临界值（上界），动态变量	VaR ，CVaR
S	蒙特卡洛模拟场景数量，1000	VaR ，CVaR
α_k	信用等级比例上限 A: 40%, B: 30%, C: 20%, D: 10%, E: 5%, F: 2%, G: 1%	原始模型
pop_size	粒子群算法种群规模，30	启发式辅助（VaR / CVaR）
max_iter	粒子群最大迭代次数，100	启发式辅助（VaR / CVaR）
(w,c_1,c_2)	PSO 参数设置：$w=0.7$, $c_1=c_2=1.5$	启发式辅助（VaR / CVaR）

4.2. 使用机器学习预测违约概率 P_i

尽管本研究使用的数据集中所有贷款均已获得实际资助, 但在资金有限, 需进行优选配置的情境下, 我们仍需要对这些已发放贷款的还款风险进行再评估. 为此, 我们引入机器学习方法, 对每笔贷款的未来违约概率 P_i 进行预测建模, 以作为后续优化模型中的风险输入参数.

建模目标 预测函数的目标是：对于每笔已发放贷款 i, 根据其已知特征向量 x_i, 估计其在未来发生违约的概率 $P_i=\mathbb{P}(y_i=1\mid \mathrm{x}_i)$. 其中, $y_i=1$ 表示贷款最终发生违约（如状态为 `Charged Off`）, $y_i=0$ 表示贷款最终还清（如状态为 `Fully Paid`）.

特征构造 我们基于借款人基本属性, 贷款合同信息以及信用评级等信息构建预测特征集, 涵盖但不限于以下变量：

- loan_amnt：贷款金额；
- term：贷款期限；
- int_rate：贷款利率；
- grade 和 sub_grade：Lending Club 信用评级；
- emp_length：工作年限；

- home_ownership：住房类型；
- annual_inc：年收入；
- dti：债务收入比；
- purpose：贷款用途；
- fico_range_high / low：信用评分；

建模方法 考虑到目标变量仍是二元状态（违约 / 未违约），我们采用监督学习的二分类方法进行建模. 尝试的模型包括逻辑回归（Logistic Regression），随机森林（Random Forest），梯度提升树（GBDT），极端梯度提升（XGBoost）等.

由于样本中违约样本占比相对较小, 我们在训练过程中采用类别加权, 欠采样等方式处理类别不平衡问题.

训练与评估 我们将全部已发放贷款随机划分为训练集（70%）与测试集（30%），使用交叉验证调优参数, 并基于测试集评估模型表现. 评价指标包括准确率（Accuracy），AUC 值（Area Under the ROC Curve），F1 分数等.

最终, 我们选择 AUC 表现最优的模型用于对所有贷款样本生成违约概率预测值 $\hat{P}_i$, 作为后续优化模型中的输入.

说明 虽然原始数据中每笔贷款都已实际放款, 但我们的建模任务是为现实中的“再选择”提供依据. 即在预算受限, 资源不足时, 如何在这些真实已放款的贷款中优先挑选违约概率低, 预期收益高的子集, 构建一个更稳健的投资组合. 因此, $\hat{P}_i$ 的预测并非用于决定放款与否, 而是作为组合优化的“风险估计量”, 用于构建期望收益与 CVaR 等风险指标.

4.3. 三类优化模型的构建与求解

本节分别构建并求解三类贷款筛选优化模型：原始模型、VaR（Value at Risk）模型和 CVaR（Conditional Value at Risk）模型。三种模型均以最大化投资收益为目标，在此基础上逐步引入风险控制机制，以刻画实际投资场景中对风险的多层次管控需求。

原始模型（期望损失约束） 为求解最大化期望回款的投资组合问题，本文构建了如算法 1 所示的混合整数规划模型。在原始贷款数据中，每条记录包含金额 A_i、年利率 r_i、还款概率 P_i 及其信用等级。模型目标是在预算限制、风险控制和信用结构约束下，筛选出收益最优的贷款子集。

模型主要特征如下：目标函数最大化组合的期望净收益 $\sum x_i A_i r_i (1 - P_i)$；约束条件涵盖总预算限制、信用等级比例控制、期望损失上限及投资笔数上限；决策变量 $x_i \in \{0, 1\}$ 表示是否选中贷款 i。

求解结果显示：Gurobi 优化器可在约 1.3 秒内收敛至最优解。最终选中 3293 笔贷款，总投资金额为 \$58,676,750，未触及预算上限，表明模型具备较强的风险控制能力与调节冗余。组合的期望净收益为 \$3,538,325.75。

按信用等级分类如下：

Algorithm 1: 最大化期望回款的投资组合优化

Input: 贷款数据集：每条数据包含 A_i（金额）、r_i（利率）、P_i（还款概率）、信用等级

Output: 最优贷款子集 $x_i \in \{0,1\}$ 及其组合统计

数据预处理：计算 $A_i = \texttt{loan\_amnt}_i$, $r_i = \texttt{int\_rate}_i/100$, $P_i = \texttt{default\_prob}_i$;

清洗缺失值与无效值；按等级分组为 G_k;

模型构建：;

定义决策变量 $x_i \in \{0,1\}$，表示是否选择贷款 i;

设定目标函数：最大化 $\sum x_i A_i r_i (1 - P_i)$;

添加约束：

- $\sum x_i A_i \le B$ // 预算约束
- $\sum_{i \in G_k} x_i A_i \le \alpha_k B$ // 信用等级比例约束
- $\sum x_i A_i P_i \le R_{\max}$ // 风险约束
- $\sum x_i \le m$ // Top-m 限制

调用 Gurobi 优化器求解模型;

if 模型可解 **then**

输出最优贷款组合 $x_i = 1$ 的子集;

统计并保存结果至 CSV 文件;

else

输出模型求解失败与错误类型（不可行/无界）;

Table 2 最优组合中各等级贷款统计

信用等级	贷款数量（笔）	金额合计（美元）	平均违约概率
A	2180	40,000,000	12.65%
B	1113	18,676,750	26.52%

结果显示，高评级（A 类）贷款在投资组合中占据主导地位，表明模型更偏好低风险、高稳定性的投资对象；B 类贷款作为收益补充。在严格的风险控制和结构约束下，该模型实现了收益与稳健性的平衡，具备现实投资指导价值。

VaR 模型（极端损失概率控制） VaR 模型通过限制极端损失出现的概率，提升投资组合在极端情形下的稳健性。结合粒子群算法（PSO）与混合整数规划（Gurobi）实现两阶段求解。实验表明，PSO 能在短时间内获得可行解，作为 warm-start 后续提升求解质量。最终组合期望收益从 701.66 万元提升至 838.91 万元，VaR 值亦略有上升，但仍在 95% 置信区间容许范围内，验证了模型的有效性与灵活性。

Algorithm 2: 基于 VaR 约束的贷款组合优化（PSO + Gurobi）

Input: 贷款数据 (A_i, r_i, P_i)，预算 B，个数上限 m，VaR 置信水平 β

Output: 最大期望收益的贷款子集 $x_i \in \{0, 1\}$

1. 数据处理与场景生成：

计算 $profit_i = A_i \cdot r_i \cdot (1 - P_i)$；

生成 S 个场景损失 $L_{i,s} \sim \text{Bernoulli}(P_i) \cdot A_i$；

2. 粒子群初始化：

启发式生成初始粒子，设置速度与参数；

3. PSO 迭代：

for $t = 1$ **to** T **do**

  计算每个粒子的 VaR 值 η；

  若 η 超过上限或约束不满足，则适应度为 $-\infty$；

  否则计算期望收益作为适应度；

  更新个体/全局最优，调整速度与位置；

4. Gurobi 精解（Warm-start）：

构建模型，最大化 $\sum x_i profit_i$，约束包括：

预算 $\sum x_i A_i \leq B$；个数 $\sum x_i \leq m$；

VaR 置信限制 $\sum_s z_s \leq (1 - \beta)S$；

损失 $> \eta$ 的场景由变量 z_s 表示；

使用 PSO 解 warm-start，进行精确求解；

Table 3　基于 VaR 约束的优化结果比较

方法	选中贷款数	投资组合期望收益（元）	VaR 上界 η（元）
PSO 粒子群算法	5000	7,016,573.73	36,698,075.00
Gurobi 精确求解	5000	8,389,092.16	47,556,800.00

CVaR 模型（尾部风险控制） CVaR 模型通过约束尾部风险期望（而非单一分位值），在提升稳健性的同时避免过度保守。实验结果表明：PSO 方法可快速生成收益较高的解，但可能不满足 CVaR 约束；Gurobi 则能精确满足风险限制，尽管收益略有下降。两者结合，可实现效率与稳健性的双重保障，为实际投资策略提供有力支持。

4.4. 结果分析

5. 结论

小组分工

Algorithm 3: 基于 CVaR 约束的贷款组合优化（PSO + Gurobi）

Input: 贷款数据 (A_i, r_i, P_i)，预算 B，个数上限 m，置信水平 β，CVaR 上限 $R_{\max}$

Output: 最大期望收益的贷款子集 $x_i \in \{0, 1\}$

1. 数据处理与场景生成：

计算 $profit_i = A_i \cdot r_i \cdot (1 - P_i)$；

生成 S 个损失场景 $L_{i,s}$；

2. 粒子群初始化与适应度：

初始化粒子 x；

适应度为 $\sum x_i profit_i$ 加风险利用率奖励；

其中 CVaR $= \eta + \frac{1}{S(1-\beta)} \sum_s \max(l_s - \eta, 0)$；

3. PSO 优化迭代：

更新粒子位置与速度，保存最优组合；

4. Gurobi 精解：

构建模型，最大化 $\sum x_i profit_i + \lambda \cdot \eta / R_{\max}$；

约束包括预算、项目个数、CVaR 构造与风险上限；

使用 PSO 解 warm-start，精确求解；

Table 4　PSO 与 Gurobi 优化结果比较（CVaR 约束）

指标	PSO 解	Gurobi 解
期望回款（元）	ě7,016,573.73	ě5,753,233.48
CVaR 风险值（元）	ě39,172,172.50	ě14,997,698.50
是否满足 CVaR 约束	否	是
选中项目数量	若干（≤ 5000）	若干（≤ 5000）

姓名	学号	分工
李晶晶	202428016443014	论文撰写, 模型设计
张璐	202428016446002	数据处理, 模型求解
朱冯婧	202428016443008	论文撰写, 算法设计

参考文献

[1] 万洁 and 郝俊. 基于混合治愈模型的网络借贷投资组合决策模型与实证研究. *系统科学与数学*, 43(5):1138–1156, 2023.

[2] Yiyang Guo, Wei Zhou, Chen Luo, and et al. Instance-based credit risk assessment for investment decisions in p2p lending. *European Journal of Operational Research*, 249(2):417–426, 2016.

[3] A Byanjankar, J Mezei, and M Heikkila. Data-driven optimization of peer-to-peer lending portfolios based on the expected value framework. *Intelligent Systems in Accounting, Finance and Management*, 28(2):119–129, 2021.

[4] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE, 1995.

6. 附录

6.1. VaR 模型求解结果

```
  PS E:\github\homeworkofcalculatefinance> python -u "e:\github\homeworkofcalculatefinance\5-25-模型求解\pso_gurobi_VaR.py"
  正在运行 PSO 寻优...

--- 粒子群优化 (PSO) 结果 ---
  选中借款人数 = 5000
  PSO 投资组合期望收益 = 7016573.73
  PSO 使用的 VaR 上界   = 36698075.00
Gurobi Optimizer version 11.0.3 build v11.0.3rc0 (win64 - Windows 11+.0 (26100.2))

CPU model: Intel(R) Core(TM) Ultra 9 185H, instruction set [SSE2|AVX|AVX2]
Thread count: 16 physical cores, 22 logical processors, using up to 22 threads

Optimize a model with 1003 rows, 30370 columns and 10302786 nonzeros
Model fingerprint: 0xa4fbee09
Variable types: 1 continuous, 30369 integer (30369 binary)
Coefficient statistics:
  Matrix range     [1e+00, 2e+08]
  Objective range  [1e+00, 3e+03]
  Bounds range     [1e+00, 1e+00]
  RHS range        [5e+01, 1e+08]
Warning: Model contains large matrix coefficients
         Consider reformulating model or setting NumericFocus parameter
         to avoid numerical issues.

User MIP start produced solution with objective 7.01657e+06 (2.25s)
Loaded user MIP start with objective 7.01657e+06
Processed MIP start in 2.23 seconds (0.95 work units)

Presolve removed 1001 rows and 1001 columns
Presolve time: 1.88s
Presolved: 2 rows, 29369 columns, 58738 nonzeros
Variable types: 0 continuous, 29369 integer (29369 binary)
Found heuristic solution: objective 7283784.2164

Starting sifting (using dual simplex for sub-problems)...

    Iter     Pivots    Primal Obj      Dual Obj        Time
       0          0     infinity     -8.5954234e+06      4s
       1          1  -1.0939240e+05  -8.5461998e+06      4s
       2          3  -1.3438183e+05  -8.5297172e+06      4s
       3          5  -1.8822504e+05  -8.5227263e+06      4s
       4          7  -2.1502998e+05  -8.5162448e+06      4s
       5          9  -2.6694422e+05  -8.5105148e+06      4s
       6         11  -3.1011424e+05  -8.5063219e+06      4s
       7         13  -3.4598473e+05  -8.5027436e+06      4s
       8         15  -3.7983946e+05  -8.4994864e+06      4s
       9         17  -4.1301854e+05  -8.4971532e+06      4s
      10         19  -4.4834645e+05  -8.4944963e+06      4s
      11         21  -4.8769815e+05  -8.4924081e+06      4s
      12         23  -5.2353089e+05  -8.4903439e+06      4s
      13         25  -5.6559651e+05  -8.4875771e+06      4s
      14         27  -6.2374219e+05  -8.4851169e+06      4s
      15         29  -6.9229155e+05  -8.4833902e+06      4s
      16         31  -7.4067974e+05  -8.4817202e+06      4s
      17         33  -7.9618193e+05  -8.4802313e+06      4s
      18         35  -8.3944338e+05  -8.4781641e+06      4s
      19         37  -9.0488706e+05  -8.4757665e+06      4s
      20         39  -9.6948627e+05  -8.4735416e+06      4s
```

```
      21         41  -1.0366669e+06  -8.4713711e+06      4s
      22         43  -1.1075983e+06  -8.4689240e+06      4s
      23         45  -1.1876166e+06  -8.4668946e+06      4s
      24         47  -1.2650869e+06  -8.4644744e+06      4s
      25         49  -1.3537151e+06  -8.4623352e+06      4s
      26         51  -1.4380471e+06  -8.4602487e+06      4s
      27         53  -1.5182690e+06  -8.4583143e+06      4s
      28         55  -1.6024285e+06  -8.4563366e+06      4s
      29         57  -1.6864300e+06  -8.4543667e+06      4s
      30         59  -1.7730984e+06  -8.4524107e+06      4s
      31         61  -1.8757160e+06  -8.4505185e+06      4s
      32         63  -1.9832589e+06  -8.4489934e+06      4s
      33         65  -2.0609758e+06  -8.4470407e+06      4s
      34         67  -2.1849239e+06  -8.4453954e+06      4s
      35         69  -2.2734543e+06  -8.4435278e+06      4s
      36         71  -2.3904212e+06  -8.4417966e+06      4s
      37         73  -2.5010673e+06  -8.4402299e+06      4s
      38         75  -2.6152965e+06  -8.4387511e+06      4s
      39         77  -2.7081939e+06  -8.4372583e+06      4s
      40         79  -2.8321013e+06  -8.4356705e+06      4s
      41         81  -2.9482273e+06  -8.4340800e+06      4s
      42         83  -3.0612090e+06  -8.4325333e+06      4s
      43         85  -3.1986347e+06  -8.4310381e+06      4s
      44         87  -3.3213768e+06  -8.4295375e+06      4s
      45         89  -3.4644936e+06  -8.4281239e+06      4s
      46         91  -3.6011453e+06  -8.4268796e+06      4s
      47         93  -3.7285862e+06  -8.4254399e+06      4s
      48         95  -3.8623207e+06  -8.4239676e+06      4s
      49         97  -3.9861846e+06  -8.4226384e+06      4s

Sifting complete


Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

      48         95  -3.8623207e+06  -8.4239676e+06      4s
      49         97  -3.9861846e+06  -8.4226384e+06      4s

Sifting complete


Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

      49         97  -3.9861846e+06  -8.4226384e+06      4s

Sifting complete




Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time


Sifting complete


Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

Sifting complete


Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

```

```
    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time


Root relaxation: objective 8.389622e+06, 106 iterations, 0.22 seconds (0.14 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time


    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0 8389621.93    0    2 7283784.22 8389621.93  15.2%     -    4s
H    0     0                     8389092.1600 8389621.93  0.01%     -    4s

     0     0 8389621.93    0    2 7283784.22 8389621.93  15.2%     -    4s
H    0     0                     8389092.1600 8389621.93  0.01%     -    4s

Explored 1 nodes (106 simplex iterations) in 4.55 seconds (2.12 work units)
Thread count was 22 (of 22 available processors)

Solution count 3: 8.38909e+06 7.28378e+06 7.01657e+06

Explored 1 nodes (106 simplex iterations) in 4.55 seconds (2.12 work units)
Thread count was 22 (of 22 available processors)

Solution count 3: 8.38909e+06 7.28378e+06 7.01657e+06
Explored 1 nodes (106 simplex iterations) in 4.55 seconds (2.12 work units)
Thread count was 22 (of 22 available processors)

Solution count 3: 8.38909e+06 7.28378e+06 7.01657e+06

Optimal solution found (tolerance 1.00e-04)

Solution count 3: 8.38909e+06 7.28378e+06 7.01657e+06

Optimal solution found (tolerance 1.00e-04)

Optimal solution found (tolerance 1.00e-04)
Best objective 8.389092160022e+06, best bound 8.389621934455e+06, gap 0.0063%

--- Gurobi 精确求解结果 ---
  Gurobi 选中借款人数 = 5000
  Gurobi 投资组合期望收益 = 8389092.16
  Gurobi 计算的 VaR 上界   = 47556800.00
```