**智能算法系统（IAS）**

**Intelligent Algorithm System**

**MRD**

|  |  |  |
| --- | --- | --- |
| **作者** | **版本** | **时间** |
| **曾锡金** | **V0.1** | **2017-06-06** |

# 1. 构建智能化匹配能力

## 产品匹配系统现状

1. 匹配逻辑过于简单粗放，没有充分利用产品和资产深层信息
2. 自适应性差，匹配规则的修改需要修改程序
3. 实时性差，通过跑批来完成匹配

## 现实场景和问题

暖薪贷产品中，比如每天的额度为50万，借款额有500和1000，借款期限有7天和14天，但是每天的符合借款资格的人往往会超过额度，就会出现部分申请通过的借款人当天借款时候出现额度用完的情况。目前的匹配规则是“先到先得”，即越早申请通过的人，在额度未占满的情况下，能够申请放款，一旦额度使用完成，其他的申请通过的人只能等下次有额度的时间申请放款。

很明显，这里匹配过程并没有充分利用每个资产深层次的信息，其实，在不明显影响用户体验的前提下，我们往往可以“优中选优”，在众多的资产中，申请的额度、期限以及客户本身的信息，不同的组合，存在带来不同的收益和风险，我们可以将这个匹配过程智能化，深层次的挖掘产品信息和客户信息，以最优的组合去完成额度匹配，这里指的最优组合，可以是带来最大收益，或者最低风险，又或者可以是收益和风险在某个比例水平的组合。

以上仅是我们遇到的产品的其中一个问题，其他产品同样存在类似的问题。

## 问题数学模型

不失一般性，我们将以上场景归约化为更一般的场景数学模型：

某类产品P，有以下约束条件：



比如，这里的可以表示每日的额度，表示额度，表示期限。

以表示一个申请借款信息，那么一个借款信息，将包含产品本身信息以及用户本身的信息，不适一般化，我们使用表示产品信息，表示客户本身信息（客户自身信息主要用以对客户进行某种规则量化后的信息，比如评分或者归一化的某种权重系数，这类信息主要用于评估客户的风险水平），那么就可以表示为：



我们用集合表示当前所有申请借款信息的集合，则可以表示为：



这里的函数可以是线性的或者非线性的，非线性的甚至可以是一种组合。

那么整个模型的智能匹配过程也就是对于给定的约束条件



和表达式



的条件下，在集合空间中，求得一个解空间，使得满足或接近目标函数的取值规则，由前面所述，目标函数的根据实际使用场景，比如：

最大收益的目标函数：



最小风险的目标函数：



在收益个风险比例在某个水平区间的目标函数：



根据上面的描述，显然，解空间是在集合空间的一个子集，满足约束条件和目标函数，这个子集就是我们实现匹配的目标借款信息集合。

## 产品

根据上文描述的问题场景可知，我们可以充分利用产品信息和客户刻画信息，来进一步增强匹配的能力，来提升我们的产品运营效能，因此，我们将规划和建设**智能算法系统**来完成这些目标。

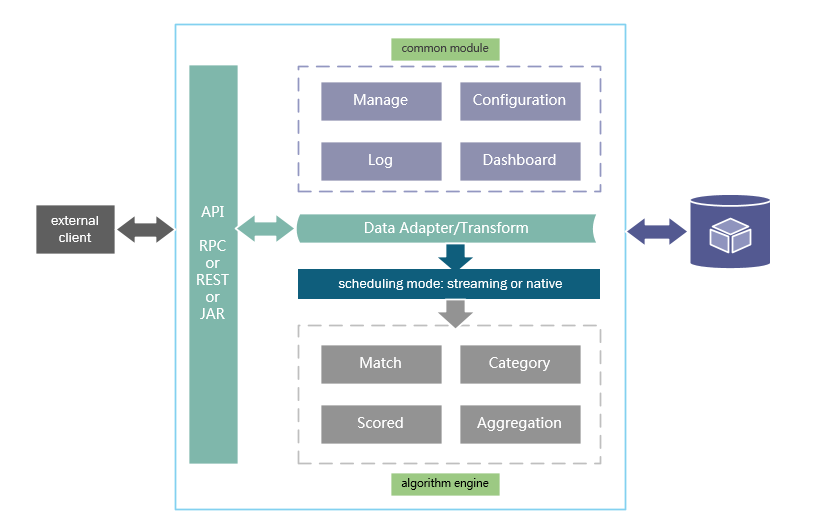
### 产品带来的优势

1. 实现智能的匹配，以提升收益或减少风险
2. 提供匹配算法的自适应性，支持动态的约束条件和目标函数的配置
3. 增强现有RiskEngine
4. 提供实时或准实时的匹配服务

### 产品目标客户群

|  |  |  |  |
| --- | --- | --- | --- |
| **最终用户** | **范围** | **功能** | **说明** |
| IT现有贷中设计匹配流程应用系统 | 标的，DB | 使用IAS完成匹配 | 可能需要这部分系统做改造配合 |
|  |  |  |  |

### 产品架构



说明：

**Algorithm Engine**

整个架构最核心是algorithm engine的算法引擎，包括匹配、分类、评分和聚合等算法，并以module模块化的形式封装，作为算法引擎包统一提供服务。

**API Module**

算法引擎的服务通过封装为平台化的API（REST或RPC）提供服务，或以嵌入jar包形式提供服务。

**Scheduling**

计算调度模式，所有的算法模块基于封装的统一Service向外暴露服务，计算的调度有上层的Scheduling模块来决定，可以引入Streaming流式计算模式，或者native本地直接计算模式，算法引擎需要支持上层调度的设计框架。

**Data Adapter/Transform**

数据适配和转换层，考虑到上层应用数据源的复杂性，以及各类算法在进入计算之前可能需要根据配置中心中约束条件或规则，需要在进入计算调度之前做数据的适配或转换处理，比如某些算法需要对数据的归一化处理、权重奖励或处罚处理等。

**Common Module**

通用模块，主要包括算法引擎的管理、配置、日志和控台展示等，提供包括算法的约束条件、目标函数、算法参数等的配置，提供算法性能监控，计算结果分析等数据。

### 匹配算法解决方案

从上文的数学模型抽象我们可以得知，在一定约束条件和目标函数下，在有限集合空间下，寻求最优解的组合，这是一个NP问题，如果使用常规的穷举排列组合的解法，算法复杂度将高达O(n!)，在大规模的数据中，对于这类NP问题，业界目前还没有找到精确的多项式解法，算法性能几乎无法工程化实用，需要采用很好的平衡运行时间和解质量的算法，因此解决方案的目标将寻找当前在组合优化最优解的可工程化和高效的解法。

解决NP问题的组合优化最优解的智能算法主要有：

|  |  |  |
| --- | --- | --- |
| **算法名称** | **优点** | **缺点** |
| 爬山法 | 能快速向局部搜索更好解，使解能够得到迅速优化 | 全局搜索能力差 |
| 模拟退火 | 优点是局部搜索能力强，运行时间较短 | 缺点是全局搜索能力差，容易受参数的影响 |
| 粒子群算法 | 适合求解实数问题，算法简单，计算方便，求解速度快 | 存在着陷入局部最优等问题 |
| 蚁群算法 | 适合在图上搜索路径问题 | 计算开销会大 |
| 捕食搜索算法 | 在局部能够很快使解得到改善，同时具备全局搜索能力，速度快 | 约束条件设计不合理的时候，会有重复访问候选解的情况 |

基于对组合优化的智能算法的解质量，运行时间，工程化复杂性，综合比较后，我们将主要基于捕食搜索算法研究和工程化增强，解决我们上文提到数学模型问题，并实现智能匹配的过程。捕食搜索算法已在解决TSP问题和超大规模集成电路设计证明了它的高效性和实用性。

当然，我们在实践的过程中，除了可以尽量工程优化此算法外，也需要关注和学习学界在解决组合优化智能算法上的进展，以增强我们算法引擎模块。

#### 捕食算法原理

思想原理：



核心算法过程：



# 2. 构建智能化分类能力

## 分类能力现状

目前没有针对客户进行快速分类的算法，主要使用了logistic逻辑回归算法作评分卡，基于评分卡做用户的准入判断。

目前还没有针对用户进行快速，自适应分类的算法应用。

## 现实场景和问题

在基于评分卡的模型准入之前或之后，可以进一步增强客户的刻画和快速分类，比如，在评分卡之前，可以根据分类模型，快速对客户进行分类，哪些是欺诈客户，哪些是正常的客户，又比如在满足评分准入里面，哪些客户是优质的，哪些客户比较一般的，如此我们可以针对特性的客户，做精确的营销或产品匹配，提高运营效率。

## 问题数学模型

对于结构化的客户属性，我们可以通过一定的规则进行量化进行刻画。

比如客户信息，具有如3个比较重要的评估属性。

假定我们根据历史信息，总结出具有属性值：



的客户信息是优质的客户信息，而具有



的属性值的客户是欺诈的客户。

那么对于任意一个新来的具有属性值客户



如何快速的知道，他究竟是优质的客户还是欺诈的客户？

那么问题就转为，在给定任意离散组合，如何计算与或与与的相似度的问题。

以上的例子是3个属性的情况，不失一般性，根据历史数据或者经验，我们假定客户具有权重较高的几个属性，问题就归约化为：

根据已知的离散有序组合

，

求给定任意的有序组合



的相似度，并根据相似度的阈值比较，进行快速的分类。

## 产品

参考构建匹配能力的章节描述

### 产品带来的优势

1. 提供基于相似度比较的快速分类能力
2. 提供自适应的属性配置
3. 基于分类进行精确营销或匹配

### 产品目标客户群

|  |  |  |  |
| --- | --- | --- | --- |
| **最终用户** | **范围** | **功能** | **说明** |
| IT现有贷前或贷中流程中需要对客户进行快速分类的应用 | 标的，DB，决策 | 使用IMS完成匹配 | 可能需要这部分系统做改造配合 |
|  |  |  |  |

### 产品架构

参考构建匹配能力的章节描述

### 产品解决方案

基于对离散有序矩阵的相似性计算，或者分类算法目前学界有非常多，尤其余弦相似性算法（cosine similiarity）使用非常普遍，而且效果不错，实现简单，具有自适应性。

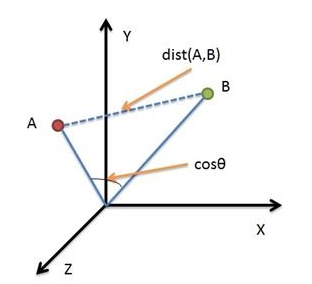
因此，智能化分类能力的构建基于余弦定理相似性的和用户属性合理建模的基础上，通过适当的参数量化规则和工程化，形成算法引擎的分类模块。

#### 余弦相似性算法

几何中夹角余弦可用来衡量两个向量方向的差异，机器学习中借用这一概念来衡量样本向量之间的差异。余弦距离更多的是从方向上区分差异，而对绝对的数值不敏感。

下图所示，就是的就是3维空间向量的和的夹角距离，通过以下公式可以计算A向量和B向量的相似性，当A与B相似时，夹角会非常小，不相似则反之。





在扩展到多维空间的有序离散向量时，余弦相似性定理依然成立，不失一般性，我们假设n维向量空间下，计算2个向量之间的相似性，公式如下：

这里写图片描述

#### 基于余弦相似性算法的分类算法

根据上文的数学模型和余弦相似性算法的原理，我们在实际工程化的时候，需要优先处理这3个问题：

1. 用于计算客户相似性的属性建模
2. 基于历史数据分析或经验规则目标特征向量值（比如优质客户的特征向量，比如欺诈用户的特征向量）
3. 基于余弦相似性算法计算时的属性量化规则(权重、归一化、奖惩系数等)

这以上3个问题的抽象，跟应用场景息息相关，那么基于余弦相似性的算法的设计将解耦这部分的依赖，这部分信息将支持算法使用方通过配置的形式传递，算法引擎通过配置信息，解析和执行这部分规则，完成特定模型和特定约束条件下的快速分类计算。