

研究报告

2020 年第 58 期

2020.12.18

执笔人：陈垣桥

邮箱：

yuanqiao.chen@icbc.com.cn

基于监督学习技术的智能金融交易监测方法

摘要：

- 本文针对金融机构风险防控工作中的重要工具——金融交易监测模型，构建了一套基于人工智能领域中监督学习技术的智能金融交易监测方法论，旨在提升金融交易监测质量和效率，从而加强金融机构风险防控水平。这套方法论包含四个模型构建环节：场景搭建、特征提取、特征工程和模型训练。本文对每一个环节的开展方式和注意事项进行了讨论。

关键词：

- 金融交易监测 人工智能 监督学习

重要声明：本报告中的原始数据来源于官方统计机构和市场研究机构已公开的资料，但不保证所载信息的准确性和完整性。本报告不代表研究人员所在机构的观点和意见，不构成对阅读者的任何投资建议。本报告（含标识和宣传语）的版权为中国工商银行现代金融研究院所有，仅供内部参阅，未经作者书面许可，任何机构和个人不得以任何形式翻版、复制、刊登、上网、引用或向其他人分发。

金融交易监测是金融机构反洗钱、反欺诈、操作风险防控等工作的重要环节，它主要指：金融机构根据特定风险防控场景，构建基于交易数据的统计模型，用来对客户及其交易进行实时监控，并评估客户及其交易的风险大小，进而筛选出风险较大的客户及其交易。在交易监测之后，专业人员或模型对被筛选出来的客户及其交易进行风险分析，并对确实可疑的交易及其背后的客户采取风险防控措施。

金融交易监测模型的有效性直接关系到风险防控工作的质量，如果这张过滤网漏抓了很多高风险交易，那么风险防控工作就会存在严重的覆盖面不足；如果这张过滤网错抓了过多的正常交易，那么后续的风险分析人员还需耗费大量时间和精力来做进一步筛选，不仅导致风险防控工作效率低下，而且带来了较大的操作风险隐患。

传统的金融交易监测模型是由一些规则指标所构成的线性二分类模型，其主要存在两方面局限性：第一，线性二分类模型只能拟合出变量空间中的线性分类边界，但实际情况中的分类边界往往是非线性的；第二，模型构建主要基于专家经验和传统统计方法，如线性回归、证据权重（weight of evidence, WOE）、朴素贝叶斯分析等，其数据挖掘能力有限，难以充分利用数据中所包含的风险信息。

为了提升金融交易监测模型的效果，一些金融机构和金融科技公司通过引入人工智能领域中的机器学习技术，以突破传统金融交易



监测模型的局限性，从而构建出监测效果更好的智能金融交易监测模型。笔者通过初步的探索，构建了一套智能金融交易监测方法论，下面对其框架进行介绍。

一、智能模型原理

在特定风险场景（例如地下钱庄类洗钱风险防控场景）下，智能模型以一组可衡量场景特征的变量作为输入，通过一个基于智能算法的函数输出一个介于 0 到 1 之间的数值，用于衡量该场景之下客户及其交易的风险大小。当函数值大于或等于某一预设阈值时，模型触发警报。

与传统模型相比，智能模型的主要优势在于：

(1) 既可以是线性模型，也可以是非线性模型，能够更加精准地拟合出实际场景中可疑类和正常类客户之间的分类边界；

(2) 模型构建主要依赖智能算法和高级统计方法，对专家经验的依赖度大幅降低，能够更加充分、精准、科学地挖掘和利用数据信息。

根据机器学习理论，智能金融交易监测模型构建路径主要包括“场景搭建、特征提取、特征工程、模型训练”这四步，下面给出具体解释。

二、场景搭建与特征提取

金融交易监测场景的搭建往往需要结合具体的风险状况以及相

关金融监管要求，例如：在反洗钱领域，金融监管一般是按照上游犯罪类型（赌博、零包贩毒等等）或异常交易模式（异常还贷、分拆购汇等等）来构建金融交易监测场景。

在特定场景下，金融机构应结合场景内在属性和历史典型案例，提取出该场景的特征。例如：在“零包贩毒类洗钱交易监测场景”之下，可提取出“ATM 交易频率较高”“多笔交易金额为某毒品价格倍数”等特征。笔者根据实际建模经验，归纳出了 9 个金融交易监测模型特征提取维度：客户身份、开户情况、账户使用、交易金额、交易结构、交易时间、交易地点、交易渠道、交易对手。

需要说明的是，场景搭建和特征提取工作仍主要基于专家经验，也可通过一些统计分析模型和基于大数据分析技术的智能辅助分析模型来进行特征探查，但这方面的方法论有待进一步探索。

三、特征工程

在特征提取完成后，建模者需构建出用于衡量特征的变量，并兼顾变量与人工智能算法的适配性。为了提升特征工程效率，可从统计角度将特征划分为四类，它们分别对应不同的加工方法，如表 1 所示：

表 1 四类统计特征及其加工方法

类型	定义	特征加工方法
无序特征	取值不可比大小的离散特征，包括二分类和多分类特征。	<ul style="list-style-type: none"> ➤ 0-1 变量（适用于二分类特征） ➤ One-Hot 编码（适用于多分类特征）



类型	定义	特征加工方法
有序特征	取值可比大小的连续或离散特征。	<ul style="list-style-type: none"> ➢ 函数变换 ➢ 离散化
特殊特征	较复杂或抽象的特征，很难用常规方法来加工。	<ul style="list-style-type: none"> ➢ 复合变换 ➢ 特殊算法
衍生特征	通过交互、分拆等方式，由以上特征衍生出的特征。	<ul style="list-style-type: none"> ➢ 变量交互 ➢ 变量分拆

补充说明：

(1) 一个特征的加工方式可能有很多种，特征加工方式差异可能导致显著的模型预测效果差异，特征加工方式需和智能算法匹配；

(2) 因为金融交易监测通常是按日对全量客户进行监测，所以交易监测模型变量所对应的数据粒度通常为“客户+日期”或“客户”，前者对应随时间变化的特征，后者对应不随时间变化的特征。

四、模型训练

在这一环节中，建模者基于特征变量数据和智能算法，将智能模型“训练”出来。这一环节可分为两个子环节：(1) 数据采样与分组、(2) 算法选用与调参。

(一) 数据采样与分组

首先通过数据采样和数据分组，将数据准备好。数据采样是以客户和交易数据为基础，计算出各个特征变量在每一个样本中所对应的数值（可接受一定程度的缺失）。数据样本为带标签样本，以支持有监督学习，标签为“黑”（对应异常客户及其交易）或“白”（对应

正常客户及其交易)。

很多金融交易监测场景存在非常显著的黑白样本不均衡问题，白样本量往往远大于黑样本量。在这样的非平衡样本集上训练模型会使模型倾向于学习白样本中的信息，而忽略了黑样本中的信息，从而使模型抓取黑客户的能力明显不足。而测试集中的样本不均衡性会导致一些测试指标失真，例如一个将所有样本都预测为白的模型，其预测误差很小，因为它对占绝大部分的白样本都能给出正确的预测结果，但实际上该模型完全不具备抓取异常客户和交易的能力。表 2 给出了一些应对样本不平衡问题的路径和方法。

表 2 样本不平衡性处理路径和方法

路径	定义	方法
欠采样	在尽可能保持样本分布的前提下，降低大样本集的样本量。	<ul style="list-style-type: none"> ➤ 原型生成 (prototype generation) ➤ 原型选择 (prototype selection)
过采样	在尽可能保持样本分布的前提下，提升小样本集的样本量。	➤ SMOTE (synthetic minority over-sampling technique)
		SMOTE 升级版 <ul style="list-style-type: none"> ➤ ADASYN (Adaptive Synthetic) ➤ Borderline-SMOTE ➤ K-means-SMOTE
算法优化	通过引入有倚重的模型算法，针对少量样本着重拟合，以提升对少量样本特征的学习。	<ul style="list-style-type: none"> ➤ 代价敏感方法 ➤ Meta Cost ➤ Easy Ensemble ➤ Balance Cascade ➤ Focal Loss
评估指标优化	构建新的测试效果指标，以反映模型对少量样本的预测能力。	<ul style="list-style-type: none"> ➤ G-mean 指标



在完成数据采样后，需将数据集划分成训练集、验证集和测试集，训练集用来训练模型，验证集用来调试算法参数，测试集用来测试模型效果；如果最优算法参数较易于寻找，则可以只划分训练集和测试集。通常训练集需要占较大比重，例如 50%—75%，以使模型能从数据中尽可能充足地学习有用信息。

（二）算法选用与调参

在准备好数据集后，建模者在“训练数据集”上运行智能算法，从而将模型“训练”出来。表 3 给出了一些适用于二分类监督学习模型构建的智能算法，并分析了它们的各自优劣势。

表 3 适用于二分类模型构建的智能算法

算法名称	优点	缺陷
k 邻近 (KNN)	<ul style="list-style-type: none"> ➤ 原理简单，建模难度低 ➤ 可解释性很强 	<ul style="list-style-type: none"> ➤ 运行效率较低，不适用于数据量较大的情况 ➤ 不适合处理不均衡样本集
朴素贝叶斯	<ul style="list-style-type: none"> ➤ 原理简单，建模难度低 ➤ 可解释性很强 	<ul style="list-style-type: none"> ➤ 假设特征之间相互独立，往往不符合实际情况
逻辑回归	<ul style="list-style-type: none"> ➤ 训练和预测效率较高； ➤ 可解释性较强； ➤ 超参数设置较为简便； ➤ 适合处理高维稀疏数据。 	<ul style="list-style-type: none"> ➤ 本质上为线性模型，无法拟合非线性分类边界； ➤ 易受到变量之间多重共线性的干扰，虽可通过正则化来缓解该问题； ➤ 需进行特征标准化。
支持向量机 (SVM)	<ul style="list-style-type: none"> ➤ 可拟合非线性分类边界； ➤ 适用于高维和低维数据； ➤ 适用于稀疏和稠密数据。 	<ul style="list-style-type: none"> ➤ 对超参数较敏感，调参难度大； ➤ 需进行特征标准化； ➤ 训练和预测效率一般； ➤ 可解释性较弱。

算法名称		优点	缺陷
决策树		<ul style="list-style-type: none"> ➤ 原理简单，建模难度低 ➤ 可解释性很强 ➤ 为决策树集成算法提供支撑 	<ul style="list-style-type: none"> ➤ 属于基础性算法，往往会造成严重的过拟合（在新数据中的泛化能力较差）
决策树集成	随机森林（RF）	<ul style="list-style-type: none"> ➤ 易达到较好的预测效果，对超参数的敏感度较低； ➤ 无需进行特征标准化； ➤ 在稠密数据集上表现较好； ➤ 受变量间多重共线性的干扰较小； ➤ 善于发掘特征间的关联； ➤ 可拟合非线性分类边界。 	<ul style="list-style-type: none"> ➤ 不适合处理高维稀疏数据； ➤ 训练和预测效率一般。
	梯度提升决策树（GBDT）	<ul style="list-style-type: none"> ➤ 在超参数设定恰当时可达到很好效果（上限高于 RF）； ➤ 无需进行特征标准化； ➤ 在稠密数据集上表现较好； ➤ 受变量间多重共线性的干扰较小； ➤ 善于发掘特征间的高阶关联； ➤ 可拟合非线性分类边界。 	<ul style="list-style-type: none"> ➤ 对超参数有点敏感； ➤ 不适合处理高维稀疏数据； ➤ 训练和预测效率一般。
	GBDT 升级版	<div>XGBoost</div> <div>LightGBM</div>	<ul style="list-style-type: none"> ➤ 它们都具备 GBDT 的优点，且能够达到更高的预测精准度、泛化能力和运行效率；LightGBM 作为 XGBoost 的改进版，可达到更好的预测效果和运行效率。但这两种算法在处理高维稀疏数据方面的能力仍存在一定不足。
神经网络		<ul style="list-style-type: none"> ➤ 在超参数设定恰当的情况下可达到极好效果（上限高于决策树集成）； ➤ 可拟合非线性分类边界； ➤ 适用于高维和低维数据； ➤ 适用于稀疏和稠密数据。 	<ul style="list-style-type: none"> ➤ 超参数较多，寻找最优超参数的难度较大； ➤ 可解释性较弱； ➤ 训练和运行效率较低。

上表所示的这些算法中， k 邻近、朴素贝叶斯和决策树算法的局限性使其很难在金融交易监测场景下达到较好预测效果，因此建模者一般在逻辑回归、SVM、决策树集成和神经网络算法中进行选择。至于最终选择哪一种算法，需从算法的理论优劣势和实际表现上来综



合考虑，在训练模型之前选择几种待定算法，进而对这几种算法之下的模型分别进行训练和参数调试，最终选择效果最好的一个模型。分类模型的预测效果可以通过“召回率”（Recall）、“准确率”（Precision）、“F-分数”（F-score）等指标来衡量，这些指标的构建是以表4所示的“混淆矩阵”（Confusion matrix）为基础：

表4 二分类问题中的混淆矩阵

实际正例	真正例（True positive, TP）	假负例（False negative, FN）
	假正例（False positive, FP）	真负例（True negative, TN）
实际负例		
	预测正例	预测负例

基于混淆矩阵，构建以下二分类模型效果度量指标：

(1) 可通过“召回率”（Recall）这一指标来衡量模型预测结果对洗钱交易的覆盖率：

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

(2) 可通过“准确率”（Precision）这一指标来衡量模型预测结果对洗钱交易的命中率：

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

(3) 可通过调和平均数来对准确率和召回率进行汇总，从而得到“F-分数”（F-score），以提供一个完整的模型效果图景：

$$F = 2 \cdot \text{Recall} \cdot \text{Precision} / (\text{Recall} + \text{Precision})$$

通常，召回率与准确率之间存在着负相关关系，如果追求大召

回率，则往往要牺牲一定的准确率；如果要追求高准确率，则一般要牺牲一定的召回率。而 **F 分数** 作为这两个指标的调和平均数，能够给出一个综合性的评判标准。在实际场景中，如果某一算法能使 F 分数显著高于其他算法，而且召回率和准确率能够达到可接受水平，那么可以将此算法视为该场景下的最优算法。

当然，算法比较的合理性需要建立在一个重要的前提之上——每一种待选算法都在特定模型和数据之下挥出了最佳表现。事实上，在模型和数据固定的情况下，一种算法的表现只会受到**算法“超参数”**的影响——超参数是在开始机器学习过程之前设置值的参数，而不是通过训练得到的模型参数数据（例如线性模型中的各项系数）。每一种机器学习算法都包含需要预设的超参数，在 SVM、GBDT 和神经网络等算法之下，模型效果对超参数的敏感度较大。相对而言，神经网络的超参数较为复杂，调参难度最大，但在找到最优超参数的情况下，该算法的预测能力上限也是最高的。

为了对算法超参数进行调试，建模者应将数据集拆分为训练集、验证集和测试集。在一种待测算法之下，验证集用于来寻找最优超参数，训练集用于训练出最优超参数之下模型，测试集用于测试模型在新数据集中的预测能力。需要说明，仅基于训练集和测试集来调试参数是不合理的，因为如果这样的话，那么算法在测试集上的运行结果就只能显示最优超参数组相对于其他超参数组的优势，而不能



显示出最优超参数组之下的模型在新数据集中的表现。

寻找最优超参数常采用“网格搜索”(Grid Search)或“随机搜索”(Random Search)方法,前者本质是一种穷举法,一般用于优化三个或者更少数量的超参数,它指的是:对于每个超参数,建模者选择若干个可选数值,构成一个有限集;于是各个超参数所对应集合的笛卡尔乘积构成若干“超参数组”,形成一个“网格”;随后建模者使用这一网格中的每一组超参数组训练模型,并给出验证集上模型效果,最终选定验证效果最优的超参数组。然而,超参数个数的增加会使训练次数指数级增长,因此在超参数较多时不宜采用网格搜索法,而更适合采用随机搜索:在超参数空间中随机生成一些超参数组,在每一个超参数组之下训练模型,并给出验证集上模型效果,最终选定验证效果最优的超参数组。在高维超参数空间中,随机搜索不仅能达到比网格搜索更高的调参效率,而且随机搜索对最优超参数组的定位能力也稍强于网格搜索。

综上所述,智能模型算法调试主要包括两个层面的选择,一是对每一种待选算法,利用训练集和验证集,通过网格搜索或随机搜索找到最优参数组;二是比较各类机器学习算法在测试集上的预测效果,选择效果最好的一种算法。具体流程为:

- (1) 结合智能算法优劣势以及实际场景、数据,给出待选算法;
- (2) 列出或随机生成每一种待选算法之下的待选超参数组;

(3) 在每一种待选算法所对应的每一组超参数组之下，在训练集上运行算法以训练出模型，随后在验证集上运行相应模型并利用召回率、准确率、F 分数等指标度量模型效果，再**选择出每一种待选算法之下的最优超参数组**；

(4) 在每一种待选算法所对应的最优超参数组之下，在测试集中运行相应模型并利用召回率、准确率、F 分数等指标度量模型效果，最后**选择效果最优的算法及其所对应的模型**。