金融机器学习算法

第八讲

金融机器学习回测算法

本讲主要内容

- 回测的常见陷阱
- 对选择偏差的评估算法
- 历史型单路径回测算法
- 场景型交叉验证回测算法
- 组合清除交叉验证回测算法

回测的常见陷阱：回测的目的

- 回测 (Back test) 的目标: 通过金融机器学习模型生成的投资策略在过去的表现来推断策略在未来的表现
- 回测要包括完整的评估在特定场景下各种变量的效果，包括投资规模、投资周期、成本变化等
- 回测不是实验，不能用来挖掘因果关系

回测的常见陷阱：量化投资七宗罪

- 幸存者偏差（Survivorship bias）：回测数据仅包含当前活跃资产，忽略了随时间推移由于破产、摘牌或被并购的资产
- 前视偏差（Look-ahead bias）：使用了在历史上看还尚未公开的信息进行决策
- 事后诸葛亮（Storytelling）：事后错误的寻找因果关系来证实随机（不自知）的模式
- 数据窥探 (Data snooping)：在测试集上训练模型
- 交易成本（Transaction costs）：设定不切实际的交易成本进行模拟
- 异常值控制（Outliers）：对极端情况情况筛选或裁剪
- 做空 (Shorting)：做空的可行性不进行正确的评估

回测的常见陷阱：马尔科斯回测定律

- 马尔科斯回测第一定律：回测不是研究工具，特征重要性才是
- 特征重要性有助于人们理解机器学习算法获得的模式的本质，但并不涉及如何使用它们盈利。回测是基于研究结果的基础上评价盈利的可能性
- 马尔科斯回测第二定律：回测时研究就像开车时喝酒，不要在回测中去训练模型
- 回测的意义是剔除不好的模型，而不是去通过不断回测改进他们。通过回测去调模型会产生选择偏差，从而浪费时间
- 在所有的研究流程结束后才能进行回测，回测结果不好必须重新开始研究

回测的常见陷阱：选择偏差风险

- 选择性偏差（selection bias）通过反复回测，调整模型参数，最后筛选出一个策略表现最好的回测结果。
- 风险 1：无法保证最好的回测结果不是来自于运气
- 风险 2：无法保证是否模型足够复杂以至于总有一套参数可以完美的拟合历史数据（拟合了噪声）
- 回测过拟合 (Backtest Overfitting: BO)： 选择性偏差造成的效应

对选择偏差的评估算法：CSCV 算法

- 组合对称交叉验证（Combinatorially Symmetric Cross-Validation: CSCV）通过交叉验证方法来评估回测过拟合的概率
- 仅适用于评估模型的选择性偏差的风险，并不是完整的回测
- 与开发模型时评估使用的 CV 不同，在开发完模型之后再使用
- 需要对模型的所有超参数组合进行总体评估
- 对称划分训练集和验证集，保证了样本内和样本外数据的平衡性

对选择偏差的评估算法：CSCV 算法

Algorithm 1: 组合对称交叉验证算法 CSCV

Result: 产生回测过拟合概率 PBO

step1：确定性能矩阵 $M_{T\times N}$：N 为可选模型数量（超参组合总量）；

step2：将性能矩阵 M 按照行分割为 S 个子矩阵 $M_s, s = 1, 2, ...S$；

step3：构造 M_s 的可能组合，每组的大小为 $S/2$，一共有 $C_s = C_S^{S/2}$ 个可能组合;

step4：$\lambda = \{\}$;

for $c \in C_s$ **do**

对比 c 号性能的 IS 的表现和 OOS 表现差异，计算对数几率比 λ_c;

$\lambda = \lambda \cup \{\lambda_c\}$

end

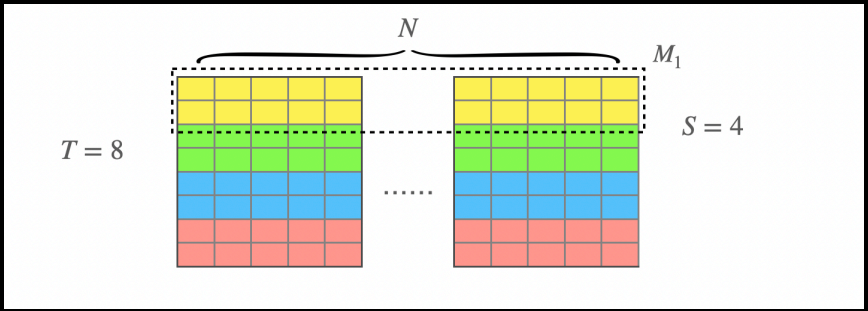
step5：$PBO = \int_{-\infty}^{0} f(\lambda)\mathrm{d}\lambda$

对选择偏差的评估算法：CSCV 算法 Step1

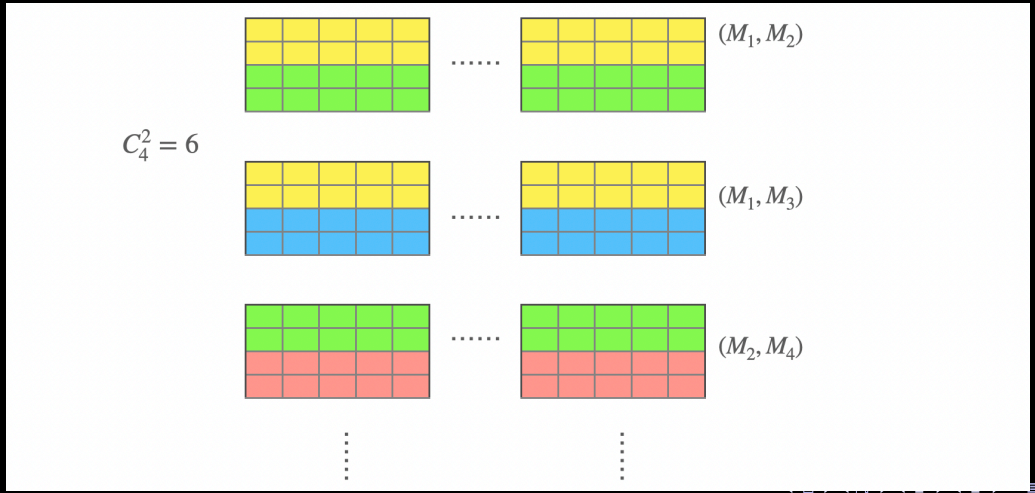
- $M_{T\times N}$ 的第 n 列对应从 1 到 T 的时间段内，第 n 号策略在每一个时间 t 上的损益情况
- 第 n 号策略由第 n 组模型超参组合所唯一确定
- $M_{T\times N}$ 的同一行对应了所有策略在时间 t 各自执行的同步结果
- 要求所有策略在 t 上都要用明确的结果，因此 $[t,\ t+1]$ 是所有策略执行频率的最小公倍数

对选择偏差的评估算法：CSCV 算法 Step2

- S 必须是一个偶数
- 子矩阵 M_s 的维度为 $\frac{T}{S} \times N$



对选择偏差的评估算法：CSCV 算法 Step3



对选择偏差的评估算法：CSCV 算法 Step4 I

- (1) 对于每一个 c 将其对应的子矩阵组合设定为训练集 J，J 的维度应为是 $\frac{T}{2} \times N$
- (2) 将没有选入训练集的子矩阵组合在一起设定为测试集 $\overline{J}$，$\overline{J}$ 的维度应为是 $\frac{T}{2} \times N$
- (3) 构造一个长度为 N 的向量 R，$R[n] = R_n$ 给出 J 中第 n 列计算得到的策略损益的回测统计量
- (4) 确定训练集内的最优表现策略 $n^* = \arg\max\{R_n\}$

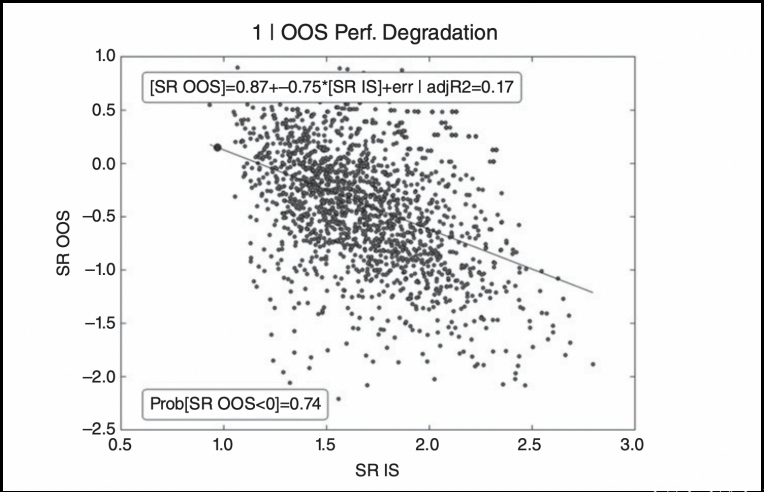
对选择偏差的评估算法：CSCV 算法 Step4 II

- (5) 构造一个长度为 N 的向量 $\overline{R}$，$\overline{R}[n] = \overline{R}_n$ 给出测试集 $\overline{J}$ 中第 n 列计算得到的策略损益的回测统计量
- (6) 确定训练集中最优策略的回测统计量 $\overline{R}_{n^*}$ 的在测试集 $\overline{J}$ 中的排序相对位置 $\omega_c \in [0, 1]$。ω_c 越大，则说明 IS 中最优策略在 OOS 中表现依然越好。
- (7) 计算对数几率比 $\lambda_c = \log\left(\frac{\omega_c}{1-\omega_c}\right)$

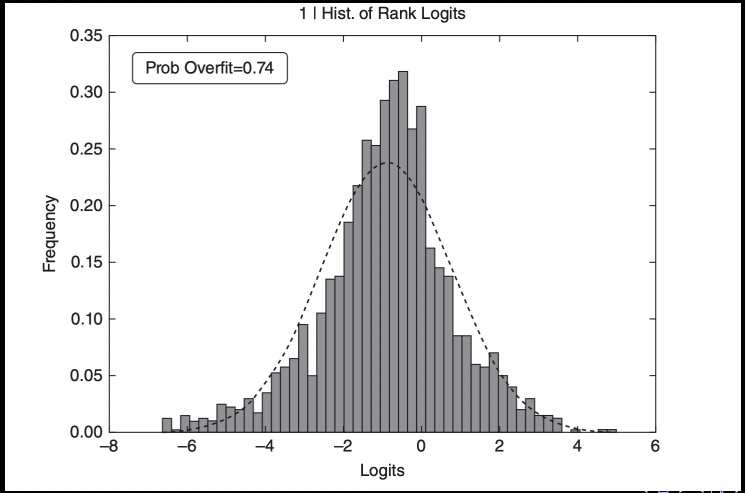
对选择偏差的评估算法：CSCV 算法 Step5

- 当训练集的最优策略表现比样本外一半的策略好时，$\omega_c = 0.5$，$\lambda_c = 0$
- 对于 λ_c 构成的分布，有 $\int_{-\infty}^{+\infty} f(\lambda)\mathrm{d}\lambda = 1$
- ：$PBO = \int_{-\infty}^{0} f(\lambda)\mathrm{d}\lambda$ 计算出的是 IS 中最优策略表现劣于 OOS 所有策略表现中位数的概率。
- 如果选择性偏差越大，则计算出的 PBO 也会越大。

对选择偏差的评估算法：CSCV 算法实例 OOS 策略退步



对选择偏差的评估算法：CSCV 算法实例 OOS 策略退步



历史型单路径回测算法

- 历史模拟 (Historical simulation)：假设策略在曾经的历史数据执行一遍
- 合理性：只要避免使用后视数据，历史模拟表现可以看作是假设策略在历史中执行的实际表现
- 作为当未来历史重演时策略的真实表现
- 历史只有一次，所以模拟出也是一条路径上策略的表现，故称之为历史型单路径回测 (Historical Simple Path: HSP)

历史型单路径回测算法的缺点

- 因为仅在一条历史路径上进行回测，很容易陷入选择性偏差，导致策略回测过拟合
- 一段回测历史中往往包含了多个明显不同的市场环境，比如包含快牛市、股灾、灾后反弹，HSP 只能是这些环境按历史顺序演化后策略的总体结果
- 未来环境按不同顺序不同时间长度出现时，HSP 无法给出评估结果

场景型交叉验证回测算法

- 动机：克服 HSP 的选择性偏差问题，数据通过 CV 算法拆分为不同的环境，然后推断特定环境下策略未来的表现
- 场景型交叉验证（Non-historical CV: NCV）对于每个回测场景，使用除该场景时间窗口内的所有数据训练模型，生成策略。在该场景中模拟策略效果
- 首先模拟在各种不同的环境下未来的表现，然后再按这些环境出现的历史顺序重新拼装起来产生路径上的模拟效果

场景型交叉验证回测算法：优点

- 使用 CV 的思想进行回测，可以支持多个不同的场景（k 个测试集）
- 生成策略的训练集样本大小一致，利用的信息量一致，后期有可比性
- 每个场景对应一个唯一的测试集，不像 HSP 需要设定预热窗口。这样是的每个数据都能参与回测。

场景型交叉验证回测算法：缺点

- 没有明确的历史型解释，它的结果不是完整的模拟策略在过去的表现以反映未来
- NCV 结果经过拼装后还是只在一条路径 (代表历史路径)，只形成一次推断。当需要以历史为基础建立多种场景下的策略执行效果的统计分布时，NCV 无法实现。
- 在回测时可能出现信息泄露问题

组合清除交叉验证回测算法

- 组合清除交叉验证回测 (Combinatorial Purged Cross Validation: CPCV) 是 NCV 的推广，将一条历史路径扩展到多条
- CPCV 法基于 Purged-CV 来构建场景型交叉验证，避免了 NCV 的信息泄露问题

Algorithm 2: 组合清除交叉验证回测算法: CPCV 算法

Result: 产生在 φ 条模拟的回测路径上，策略执行效果评价指标

step1：将 T 个样本分为 N 组，前 $N-1$ 个样本容量为 T/N，最后一个为 $T-\lfloor T/N \rfloor (N-1)$;

step2：从 N 个组中取 k 个作为测试集，一共有 $S=C_N^k$ 个组合;

step3：对每一个组合使用 Purged 和 Embargo 算法进行清除和隔离;

step4：执行 CV;

for $s \in 1:S$ **do**

| 在第 s 组的训练集中训练策略模型，并在测试集中进行预测

end

step5：执行 Back test。拼接构造 $\varphi[N,k]=\frac{C_N^k k}{N}$ 条模拟路径;

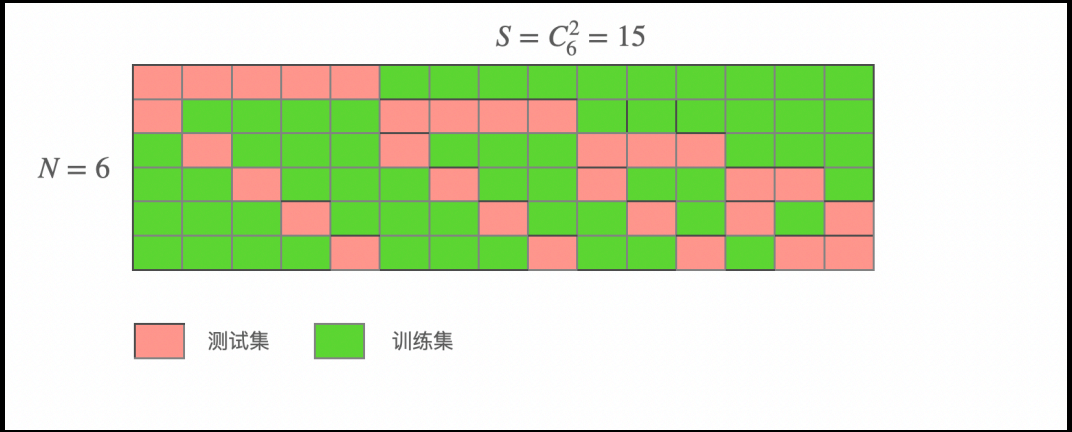
for $i \in 1:\varphi$ **do**

| 在第 i 条路径上完整的计算执行策略的效果（夏普率、胜率）

end

计算 φ 条模拟路径上的效果分布

组合清除交叉验证回测算法：步骤 1-步骤 4



组合清除交叉验证回测算法：步骤 5

$$S = C_6^2 = 15$$

$N = 6$

测试集　训练集

$$\varphi = \frac{2 \times 15}{6} = 5$$

组合清除交叉验证回测算法：特殊情况

- $k=1$：$\varphi=\dfrac{C_N^1\times 1}{N}=1$ CPCV 退化为 Purged-Embargo CV
- $k=2$: $\varphi=\dfrac{C_N^2\times 2}{N}=N-1\approx N$ 所以要生成 φ 条路径，那么就划分为 $\varphi+1$ 组
- $k=2$：选 $N=T$，一共将有 $T-1$ 条回测路径，策略将会在 $1-\dfrac{2}{T}$ 比例上进行训练。
- $k\to\dfrac{N}{2}$：路径为最多。